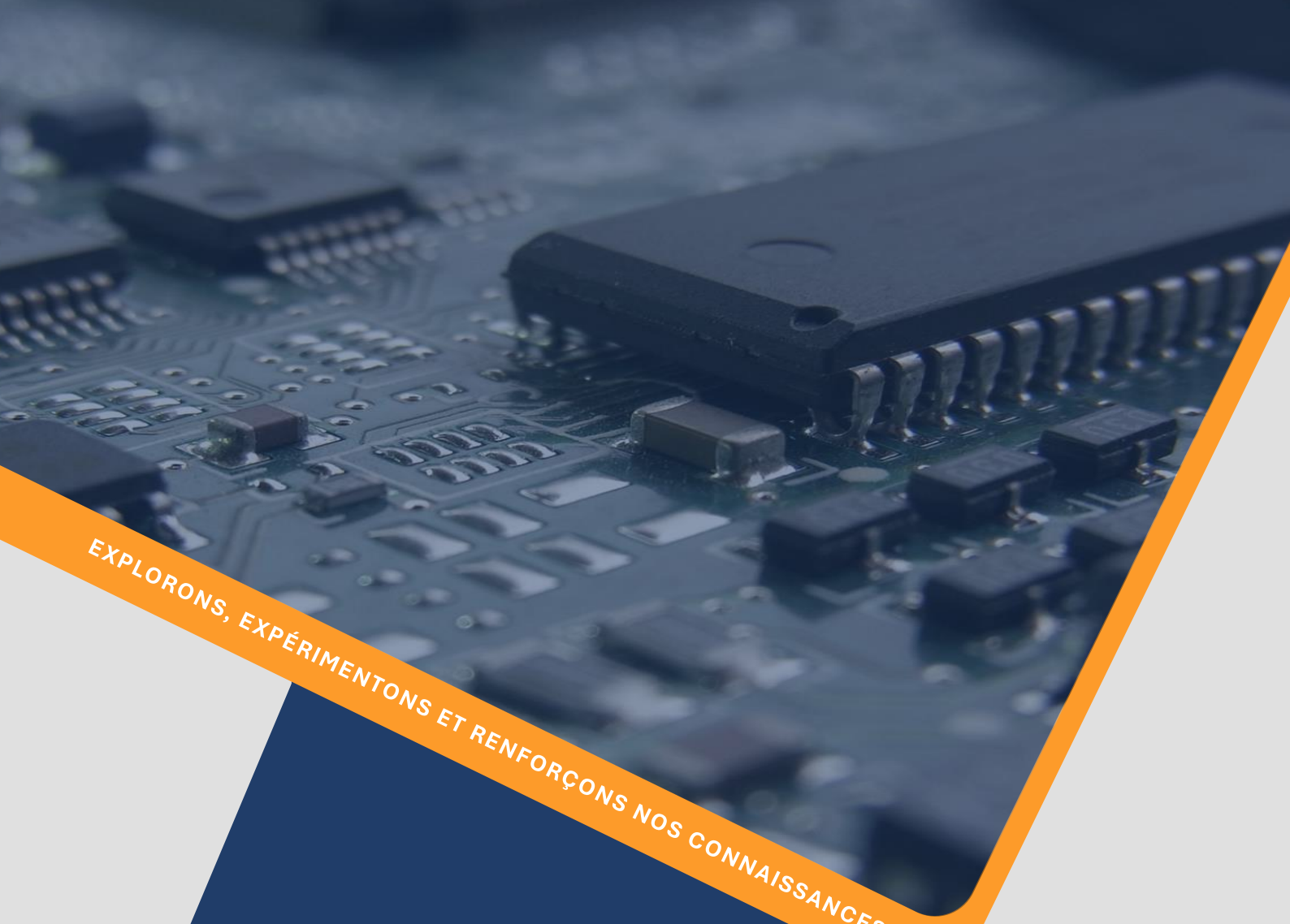


EXPLORONS, EXPÉRIMENTONS ET RENFORÇONS NOS CONNAISSANCES

INTRODUCTION À LA TECHNOLOGIE DES ORDINATEURS

DOMINIQUE LIEFFRIG

www.clicparclic.eu

dlieffrig@hotmail.com

EAFC
CENTRE ARDENNE

Sommaire

1 COMMENÇONS PAR QUELQUES DÉFINITIONS ET NOTIONS FONDAMENTALES 1

1.1 CALCULATEUR 1
1.2 ORDINATEUR 1
1.3 INFORMATIQUE 1
1.4 NOTION D'INFORMATION 1
1.4.1 Définition 1
1.4.2 Caractéristiques de l'information 1
1.4.3 Différences entre donnée et information 2
1.4.4 Rôle de l'information en informatique 2
1.4.5 Traitement de l'information 2

2 DU BOULIER À L'ORDINATEUR QUANTIQUE 3

2.1 INTRODUCTION 3
2.2 DES ROMAINS AU XVII^E SIÈCLE 4
2.3 PROGRÈS AUX XVIII^E ET XIX^E SIÈCLES 6
2.4 XX^E SIÈCLE 8
2.5 LES ORDINATEURS ÉLECTRONIQUES UNIVERSELS : LES DIFFÉRENTES GÉNÉRATIONS 10
2.5.1 Première génération : utilisation de tubes à vide 10
2.5.2 Deuxième génération : l'ère du transistor 12
2.5.3 Troisième génération : l'ère des circuits intégrés 13
2.5.4 Quatrième génération : l'ère des microprocesseurs 14
2.5.5 Et le futur... 18

3 NOTIONS DE SYSTÈME, SYSTÈME DE PILOTAGE, SYSTÈME D'INFORMATION, SYSTÈME OPÉRANT, SYSTÈME INFORMATIQUE... 21

3.1 DÉFINITIONS 21
3.2 LE SYSTÈME D'INFORMATION 21
3.3 SCHÉMA RÉCAPITULATIF 22
3.4 NE PAS CONFONDRE « SYSTÈME D'INFORMATION » ET « SYSTÈME INFORMATIQUE » 22
3.5 LES COUCHES D'UN SYSTÈME INFORMATIQUE 23
3.6 LE SYSTÈME D'EXPLOITATION 23
3.6.1 Fonctions essentielles d'un système d'exploitation (SE) 23
3.6.2 Liens entre les fonctions du système d'exploitation et le schéma en couches 24
3.7 LES DIFFÉRENTES CATÉGORIES DE SYSTÈMES INFORMATIQUES 25
3.7.1 Introduction 25
3.7.2 Classification selon la taille et la capacité de traitement 25
3.7.3 Selon le mode d'utilisation 31
3.7.4 Points clés à retenir 32
3.7.5 Conclusion 32
3.8 COMPATIBILITÉ DES SYSTÈMES INFORMATIQUES 32
3.8.1 Définition 32
3.8.2 Systèmes compatibles 32
3.8.3 Systèmes non compatibles 32
3.8.4 Facteurs favorisant la compatibilité 33
3.8.5 Points clés à retenir 33

4 LA NUMÉRATION 34

4.1 BIT, BYTE, OCTET 34
4.2 NUMÉRATIONS DE POSITION - ILLUSTRATION AVEC LA NUMÉRATION DÉCIMALE 35
4.3 LE POIDS DES CHIFFRES - ILLUSTRATION AVEC LA NUMÉRATION DÉCIMALE 36
4.4 NUMÉRATION BINAIRE 36
4.4.1 Conversion de la base binaire vers la base décimale 36
4.4.2 Conversion de la base décimale vers la base binaire 37
4.5 NUMÉRATION HEXADÉCIMALE 37
4.5.1 Conversion de la base hexadécimale vers la base décimale 38

4.5.2 Conversion de la base décimale vers la base hexadécimale 38
4.6 Opérations en binaire 39
4.6.1 L'addition binaire 39
4.6.2 La soustraction binaire par la méthode du complément à 2 40
4.6.3 La multiplication binaire 41
4.6.4 La division binaire 41
4.7 Très bien pour les nombres naturels mais comment gérer des nombres entiers négatifs ? 41
4.7.1 Manipuler des nombres entiers négatifs en binaire 41
4.7.2 Manipuler des nombres entiers négatifs en hexadécimal 43

5 NUMÉRIQUE / ANALOGIQUE 45

5.1 Deux modes de représentation des grandeurs 45
5.2 Quelques notions théoriques 46
5.3 Numérisation du son 48
5.3.1 Les caractéristiques du son 49
5.3.2 Conversion d'un signal sonore analogique en numérique 52

6 ARCHITECTURE DES MACHINES ET ASSEMBLEUR 53

6.1 Le modèle de John von Neumann 53
6.1.1 Présentation générale 53
6.1.2 Unité de commande 54
6.1.3 Unité arithmétique et logique 54
6.1.4 Mémoire 54
6.1.5 Qu'en est-il aujourd'hui ? 54
6.2 Initiation à l'assembleur 55
6.2.1 Introduction 55
6.2.2 Familles des instructions machine 56
6.2.3 Programme en langage machine 57
6.2.4 Présentation d'un langage assembleur par l'exemple 58
6.3 Simulation d'un CPU : l'assembleur AQA 60
6.3.1 Présentation 60
6.3.2 Les registres dans le simulateur AQA 61
6.3.3 La RAM dans le simulateur AQA 61
6.3.4 Le CPU dans le simulateur AQA 62
6.3.5 Bon à savoir 62
6.3.6 Exercices « Découverte guidée » 63
6.3.7 Autres simulateurs 72

7 LES ORDINATEURS 74

7.1 Introduction à la notion d'ordinateur 74
7.1.1 Présentation de l'ordinateur 74
7.1.2 Types d'ordinateurs 74
7.2 Architecture matérielle – Vue globale 74
7.3 La carte mère 76
7.3.1 Présentation de la carte-mère 76
7.3.2 Facteur d'encombrement ou facteur de forme d'une carte-mère 77
7.3.3 Le chipset 78
7.3.4 L'horloge et la pile du CMOS 79
7.3.5 Le BIOS et l'interface de configuration 80
7.3.6 Le processeur : cœur de l'ordinateur 81
7.3.7 La mémoire-cache : une mémoire ultra-rapide intégrée au processeur 81
7.3.8 La mémoire vive 82
7.3.9 Le bus système et ses évolutions 83
7.3.10 Les connecteurs d'extension (slots) 84
7.4 Le processeur : le cerveau de l'ordinateur 85
7.4.1 Introduction 85
7.4.2 Principe de fonctionnement du CPU 85
7.4.3 Parallélisation dans les processeurs modernes 86
7.4.4 L'architecture CISC 87
7.4.5 L'architecture RISC 87

7.4.6 CISC ou RISC? 88
7.5 Les mémoires 88
7.5.1 Rôle des mémoires 88
7.5.2 La ROM 88
7.5.3 Les registres 89
7.5.4 La RAM 90
7.5.5 Les temps d'accès et cycle mémoire 91
7.5.6 Hiérarchie des mémoires 92
7.5.7 Fonctionnement de la mémoire vive 93
7.5.8 La correction d'erreurs 93
7.6 Les périphériques externes 93
7.6.1 Les ports USB 94
7.6.2 HDMI / DisplayPort 94
7.6.3 Ethernet / Wi-Fi 94
7.6.4 Bluetooth 95
7.6.5 Thunderbolt 95
7.6.6 Autres interfaces 95
7.7 Les dispositifs d'entrée et de sortie 96
7.7.1 Dispositifs d'entrée 96
7.7.2 Dispositifs de sortie 96
7.7.3 Dispositifs d'entrée/sortie (mixtes) 96
7.8 Le système d'affichage 97
7.8.1 Le moniteur 97
7.8.2 La carte graphique 97
7.9 Les périphériques de stockage 98
7.9.1 Définition 98
7.9.2 Classification des périphériques de stockage 98
7.9.3 Principes physiques utilisés 99
7.9.4 Comparaison des périphériques de stockage 99
7.10 Comment vérifier la compatibilité d'une carte mère avec d'autres composants ? 100
7.10.1 Identification des besoins spécifiques de notre configuration 101
7.10.2 Compatibilité du processeur (CPU) 101
7.10.3 Compatibilité de la mémoire vive (RAM) 101
7.10.4 Compatibilité de la carte graphique (GPU) 102
7.10.5 Autres considérations importantes 102
7.10.6 Utilisation des outils de vérification en ligne 102

8 INFORMATISATION D'UN PROBLÈME POSÉ 103

8.1 Étapes de l'informatisation d'un problème posé 103
8.1.1 Analyse et compréhension du problème 103
8.1.2 Définition des objectifs et des besoins 103
8.1.3 Évaluation des ressources et élaboration d'un cahier des charges 103
8.1.4 Planification et organisation du projet 103
8.1.5 Choix des outils, matériels et logiciels 103
8.1.6 Développement spécifique (si nécessaire) 103
8.1.7 Installation, paramétrage et tests 104
8.1.8 Saisie et migration des données 104
8.1.9 Formation des utilisateurs et accompagnement au changement 104
8.1.10 Suivi, maintenance et amélioration continue 104
8.2 L'algorithmique 104
8.2.1 Qu'est-ce qu'un algorithme ? 104
8.2.2 Caractéristiques d'un algorithme 104
8.2.3 Représentations d'un algorithme 105
8.2.4 Exemple simple : calculer la somme de deux nombres 105
8.2.5 Importance de l'algorithmique 105
8.3 Les langages de programmation 105
8.3.1 Pourquoi des langages de programmation ? 105
8.3.2 Types de langages 105
8.3.3 Familles de langages 105
8.3.4 Exemple simple en Python 106

8.3.5 Importance des langages de programmation 106

9 DOMAINES D'INTERVENTION DANS UN SYSTÈME INFORMATIQUE ET MÉTIERS S'Y RAPPORTANT 107

9.1 MATÉRIEL ET INFRASTRUCTURE 107
9.2 RÉSEAUX ET TÉLÉCOMMUNICATIONS 107
9.3 LOGICIELS ET DÉVELOPPEMENT 107
9.4 SYSTÈMES D'INFORMATION ET GESTION DE DONNÉES 107
9.5 CYBERSÉCURITÉ 108
9.6 SUPPORT, ASSISTANCE ET FORMATION 108
9.7 GOUVERNANCE, GESTION DE PROJETS ET CONSEIL 108

10 BIBLIOGRAPHIE 109

1 Commençons par quelques définitions et notions fondamentales

1.1 Calculateur

Un calculateur est un dispositif, mécanique ou électronique, conçu pour exécuter automatiquement des opérations arithmétiques ou logiques sur des données. Historiquement, le terme désignait des machines limitées à des tâches spécifiques, sans réelle capacité de traitement général ni souplesse d'utilisation. Les premiers calculateurs étaient mécaniques (comme la Pascaline ou la machine analytique de Babbage), puis électroniques. Dans les usages modernes, le terme peut aussi désigner un système spécialisé embarqué, dédié à une tâche précise.

1.2 Ordinateur

À l'heure actuelle, un ordinateur est défini comme étant un dispositif électronique programmable, capable de traiter automatiquement des données selon des instructions codées dans un programme. Il se distingue par sa polyvalence : il peut exécuter de nombreuses tâches (calcul, gestion de fichiers, navigation, etc.). Il est composé de divers éléments (processeur, mémoire, stockage, périphériques) et fonctionne selon le cycle entrée-traitement-sortie-stockage.

1.3 Informatique

L'informatique est **la science du traitement automatique de l'information.**
Elle regroupe l'ensemble des théories, des méthodes, des techniques et des outils qui permettent de représenter, stocker, traiter, transmettre et exploiter l'information, généralement à l'aide de systèmes électroniques programmables (ordinateurs, serveurs, systèmes embarqués, etc.).

Le terme « informatique » est formé par la fusion des mots « information » et « automatique ». Il a été introduit en 1962 par Philippe Dreyfus pour désigner l'automatisation du traitement de l'information. Pour information, le terme « Informatik » avait déjà été utilisé en 1957 par l'ingénieur allemand Karl Steinbuch.

1.4 Notion d'information

1.4.1 Définition

L'information est un concept fondamental en informatique et dans de nombreux autres domaines. Elle peut être définie comme l'ensemble des données qui ont été organisées ou traitées de manière à **avoir une signification** pour celui qui les reçoit. Autrement dit, l'information est une donnée interprétée qui permet de réduire l'incertitude ou d'apporter une connaissance nouvelle à un individu ou à un système.

« L'information, c'est une donnée à laquelle on donne du sens dans un contexte donné. »

1.4.2 Caractéristiques de l'information

Voici les principales caractéristiques de l'information :

- **Signification :** Une information a du sens pour la personne ou le système qui la reçoit. Par exemple, le nombre « 42 » seul est une donnée, mais « 42°C, température du processeur » devient une information.
- **Pertinence :** Une information doit répondre à un besoin précis ou à une question posée.
- **Exactitude :** Elle doit être correcte et refléter la réalité pour être utile.
- **Actualité :** L'information doit être à jour pour être pertinente et fiable.
- **Accessibilité :** Elle doit être disponible et compréhensible pour l'utilisateur ou le système qui en a besoin.
- **Confidentialité :** Dans certains contextes, l'information doit être protégée contre les accès non autorisés.

1.4.3 Différences entre donnée et information

Élément	Définition	Exemples
Donnée	Élément brut, non interprété, sans signification propre	19, 2025, « bleu »
Information	Donnée interprétée, qui prend du sens dans un contexte donné	19°C : température extérieure Année de naissance : 2025 Couleur du ciel : bleu

1.4.4 Rôle de l'information en informatique

En informatique, l'information est à la base du fonctionnement des systèmes. Elle est collectée, traitée, stockée et transmise pour permettre la prise de décision, la communication, la gestion et l'automatisation des opérations. Les systèmes d'information, qui comprennent à la fois les aspects technologiques et humains, sont essentiels pour la gestion des activités d'une organisation et pour son efficacité.

1.4.5 Traitement de l'information

1.4.5.1 Définition

Le traitement de l'information désigne l'ensemble des opérations qui permettent de transformer des données brutes en informations utiles, exploitables et compréhensibles par l'humain ou par une machine.

Ce traitement est généralement effectué automatiquement par un système informatique, selon une suite d'instructions codées dans un programme.

1.4.5.2 Étapes typiques du traitement de l'information

Le traitement de l'information suit le cycle de traitement suivant :

1) Entrée (Input) : acquisition des données (clavier, capteurs, fichiers...).
2) Traitement (Processing) et analyse des données : transformation des données (calcul, tri, filtrage, comparaison...).
3) Sortie (Output) : restitution du résultat (écran, imprimante, sortie audio...).
4) Stockage (Storage) : mémorisation temporaire ou permanente des données et des résultats.

D'autres étapes peuvent venir s'ajouter : suppression des données...

1.4.5.3 Exemples de traitement de l'information

Voici quelques exemples de traitement de l'information :

- Transformer une température saisie en °F en °C.
- Trier une liste de noms par ordre alphabétique.
- Calculer un bulletin de notes à partir de points.
- Convertir un signal sonore en fichier audio numérique.
- Calculer les salaires, les cotisations sociales.
- Calculer un itinéraire.

2 Du boulier à l'ordinateur quantique

Source de ce point :
Certaines informations ont été reprises de : Zanella, P., Ligier, Y., & Lazard, E. (s. d.). *Architecture et technologie des ordinateurs - 6e éd. : Cours et exercices corrigés*. Dunod

2.1 Introduction

L'homme a toujours eu besoin de compter. Au cours de la Préhistoire (période se terminant vers 3300 avant J.-C.), il ne savait calculer essentiellement à l'aide de cailloux (latin : calculi) ou de ses mains, qui furent sans doute les premières calculatrices de poche. Des os gravés de séries de traits regroupés de façon organisée ont également été trouvés. Ils dateraient de 20000 av. J.-C.

On trouve des traces de symboles et de chiffres dans certaines civilisations de l'Antiquité (-3300 à 476 ou 395 suivant les historiens), quelques millénaires avant notre ère. Chinois, Égyptiens, Sumériens, Babyloniens, Grecs ou Romains, tous avaient des symboles numériques et des méthodes pour compter et calculer.

Ces systèmes de numération s'inspiraient naturellement du nombre de doigts; c'est ainsi que les Romains, par exemple, établirent des symboles spéciaux pour indiquer 5 et 10 unités (V et X). Dans certaines civilisations pieds-nus utilisant les mains et les pieds pour compter, le nombre 20 était parfois choisi comme base de numération. Dans certaines régions asiatiques, on comptait en se servant des articulations des doigts ou des phalanges, d'où des numérations en base 12, 14, 15, 24, 30, 60, etc.

Curieusement, on utilise, en langue anglaise, le terme de calcul digital (Digital calculation) en science informatique, le mot digit ayant le sens de chiffre. En français, on parlera plutôt de calcul numérique. Dans les ordinateurs, on utilise des bits (le terme bit est la contraction de l'expression anglaise binary digit), l'écriture binaire des nombres ne comportant que les deux chiffres 0 et 1.

La plus naturelle et la plus répandue des numérations était celle qui comptait en base 10 et elle nous est parvenue à travers les siècles avec ses symboles introduits par les Indiens, modifiés et complétés par les Arabes. Notre système décimal actuel est le résultat de cette évolution et des moyens mis en œuvre pour lui donner des formes adaptées à l'expression écrite et orale et aux méthodes de calcul.

Si le système décimal est celui de l'immense majorité des humains, il ne faut pas oublier que d'autres sont toujours utilisés, tel le système sexagésimal (base 60) pour exprimer les mesures du temps, tout comme celles des arcs et des angles. L'origine du système sexagésimal remonte aux Sumériens. Au cours de l'histoire , on trouve aussi souvent le nombre 12 à la base de nombreux systèmes de comptage et de mesure, par exemple dans la division du jour en heures.

Parallèlement à cette évolution des signes, chiffres, calculs mentaux et manuels, on assistait au développement d'outils, de systèmes, de machines pour simplifier et accélérer les calculs nécessaires, par exemple pour garder la trace des transactions commerciales ou des cycles astraux et pour faire face aux besoins croissants des paysans, de l'armée et d'une société en pleine évolution.

Des exemples de numérations utilisées au cours de l'histoire sont résumés ci-après.

BASE 1 : comptage avec les doigts, cailloux, entailles
BASE 2 : système binaire : logique symbolique, ordinateurs
BASE 5 : système quinaire : Aztèques
BASE 7 : notes musicales, jours de la semaine
BASE 8 : système octal : premiers ordinateurs
BASE 10 : système décimal
BASE 12 : gamme des notes et demi-tons ; monnaie et mesures anglaises ; mois ; heures (2 fois 12)
BASE 16 : système hexadécimal : ordinateurs
BASE 20 : comptage sur les doigts des mains et des pieds : Mayas
BASE 24 : heures du jour
BASE 60 : degrés, minutes et secondes; angles; savants de Babylone

2.2 Des Romains au XVII[e] siècle

Il y a 2000 ans, les civilisations méditerranéennes utilisaient l'abaque pour leurs calculs. Bien avant l'ère chrétienne, les Chinois comptaient à l'aide de bouliers. Mais il fallut attendre le XVII[e] siècle, époque de grandes effervescences intellectuelles, pour voir apparaître des systèmes de calcul plus rapides et plus automatiques. Les débuts furent lents et difficiles.

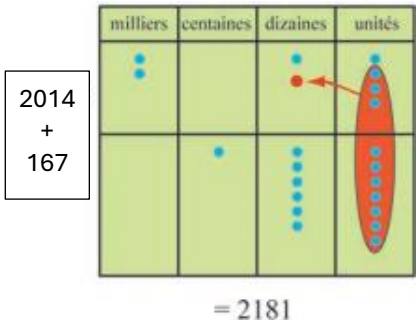


Figure 2 : Abaque

La numération romaine, utilisée en Europe pendant le premier millénaire de notre ère, n'était pas une numération positionnelle; c'est-à-dire que la position des chiffres dans la représentation d'un nombre n'était pas associée à des poids implicites (unités, dizaines, centaines, etc.) permettant une écriture des nombres plus compacte (MDCCCLXXIII = 1873) et une grande simplification des calculs. Les Romains ne connaissaient pas le chiffre zéro ! L'étonnante idée du zéro vint à l'esprit des Indiens et des Arabes quelques siècles après Jésus-Christ. Le chiffre zéro fit son apparition en Europe dans un manuscrit célèbre sur les chiffres indiens, écrit par le mathématicien AlKhwarizmi vers l'an 820 après J.-C. (les savants de Babylone connaissaient apparemment une numération en base 60, positionnelle, avec le chiffre zéro, déjà au IIIe siècle avant J.-C.). Les chiffres arabes sont adoptés en Europe au cours du XIe siècle, mais il faut attendre le milieu du XVIe siècle pour voir des ouvrages traitant de méthodes arithmétiques.


Figure 1 : Boulier

Revenons sur la conversion du nombre romain MDCCCLXXIII en décimal

Le nombre romain MDCCCLXXIII se convertit en décimal en additionnant la valeur de chaque lettre en respectant les règles des chiffres romains.

- Étape 1 : Identifier la valeur de chaque lettre

Voici les valeurs :

M = 1000
D = 500
C = 100
L = 50
X = 10
I = 1

- Étape 2 : Décomposer le nombre

MDCCCLXXIII peut être décomposé ainsi :

M = 1000
D = 500
C + C + C = 100 + 100 + 100 = 300
L = 50
X + X = 10 + 10 = 20
I + I + I = 1 + 1 + 1 = 3

- Étape 3 : Additionner

  1000 (M)
  500 (D)
  300 (CCC)
  50 (L)
  20 (XX)
  3 (III)
  = 1873

MDCCCLXXIII = 1873 en chiffres arabes et en base 10.

En 1614, le mathématicien écossais John Neper (ou Napier), qui transformait des multiplications compliquées en de simples additions, utilisa la règle à calcul, un outil pratique et efficace. Neper inventa un système non logarithmique (pour simplifier les multiplications) basé sur le simple déplacement de tiges (Bâtons ou Os de Neper). En 1623, Wilhelm Schickard construit à Tuebingen en Allemagne, la première machine à calculer en appliquant le principe du déplacement de tiges développé par Neper. Sa machine se perd au cours de la guerre de Trente Ans; de ce fait, on ne sait pas exactement si, et éventuellement de quelle manière elle fonctionnait. Les quelques dessins qui nous sont parvenus semblent prouver que Schickard avait utilisé des roues chiffrées et s'était attaqué au problème de la retenue. Bien que le principe des roues dentées et autres engrenages fût connu depuis des siècles (astrolabes[1], horloges des églises, etc.), les techniques de construction étaient primitives et la fiabilité résultante assez modeste. Schickard se plaint d'ailleurs des problèmes de mécanique dans ses lettres à Kepler, où l'on trouve de précieuses indications sur la conception de sa machine. À partir de ses dessins, une réplique de la machine, améliorée pour être fonctionnelle, fut construite en 1971.

En 1642, à Paris, Blaise Pascal présente une machine qui peut additionner et même soustraire des nombres de six chiffres. En dix ans, il en construit plus de cinquante versions dont certaines peuvent calculer avec huit chiffres. Des exemplaires sont conservés à Paris. Son système est basé sur une série de roues dentées figurant les colonnes décimales. Le problème de la retenue est résolu de la manière suivante : chaque roue peut dépasser le chiffre 9 en effectuant une rotation complète et en décalant d'un cran la roue immédiatement supérieure. Pascal a réalisé sa première machine, la Pascaline, alors qu'il n'avait que 19 ans.

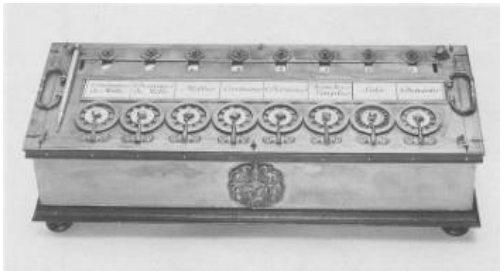


Figure 3 : La Pascaline

Quelques vidéos :
Additionner avec la réplique de la Pascaline 1645 : https://www.youtube.com/watch?v=GX4RQK__fQc
Le mécanisme interne de la réplique de la Pascaline : https://www.youtube.com/watch?v=hSl2WFfCTD8
How the Pascaline Works : https://www.youtube.com/watch?v=3h71HAJWnVU

La machine de Pascal pouvait en principe exécuter des opérations plus complexes, telle la multiplication, par des méthodes compliquées d'additions répétitives. Mais il faudra attendre **1673** pour voir apparaître une calculatrice capable d'exécuter automatiquement les quatre opérations arithmétiques. Ce sera l'œuvre d'un génie allemand, Leibniz , qui ajoutera aux mécanismes de la Pascaline un chariot mobile et une manivelle permettant d'accélérer et d'automatiser l'exécution des additions et des soustractions répétitives exigées par les multiplications et les divisions. Les principes des machines de Pascal et de Leibniz seront adoptés dans la conception des machines à calculer pendant près de trois siècles !

1697 : Gottfried Leibniz introduit le calcul binaire en Europe. Au lieu d'utiliser 10 chiffres, on n'en utilise plus que 2 : le 0 et le 1. Par exemple, 9 s'écrit 1001 et 4 s'écrit 100. Compter en binaire revient à énoncer 0, 1, 10, 11, 100, 101, 110, 111. Les opérations arithmétiques simples telles que l'addition, la soustraction et la multiplication sont faciles à effectuer en binaire. L'addition en binaire se fait avec les mêmes règles qu'en décimal : on commence à additionner les bits de poids faibles (les bits de droite) puis on a

[1] Astrolabe : instrument qui fournissait une représentation du ciel à un instant donné, en projection sur un plan.

des retenues lorsque la somme de deux bits de mêmes poids dépasse la valeur de l'unité la plus grande (dans le cas du binaire : 1), cette retenue est reportée sur le bit de poids plus fort suivant.

Leibniz, qui avec Newton est à l'origine du calcul différentiel et intégral, inventa aussi le système binaire sous sa forme moderne (des numérations base 2 existaient déjà en Chine dans l'Antiquité) avec ses deux chiffres 0 et 1, et souligna la puissance et la simplicité de l'arithmétique binaire, qui sera finalement adoptée par la plupart des ordinateurs contemporains.

On peut à juste titre considérer le XVII^{e} siècle comme un tournant dans le développement de la connaissance scientifique. Des géants tels Galilée, Newton et Leibniz sont à l'origine d'une véritable révolution intellectuelle qui propulsa l'Europe au premier plan dans le développement des mathématiques et dans leur application aux sciences naturelles. C'est au XVII^{e} siècle que l'on a conceptualisé les bases de la science moderne et c'est là que l'on trouve les racines de ce grand développement d'idées qui conduira à l'ordinateur.

2.3 Progrès aux XVIII^{e} et XIX^{e} siècles

En 1728, le mécanicien français Falcon construit une commande pour métier à tisser à l'aide d'une planchette en bois munie de trous. C'est la première machine capable d'exécuter un programme externe. **En 1805**, Joseph Jacquard perfectionne le système de Falcon en remplaçant les planches en bois par des cartons troués articulés (les premières cartes perforées), que l'on peut encore voir de nos jours dans certains orgues de manège. Le système à bande de programmation de Jacquard permet de produire les dessins les plus compliqués en grande quantité et de qualité toujours égale. Il s'agit du premier pas de la révolution industrielle, mais aussi d'une étape très importante vers l'exécution automatique des calculs les plus complexes.



Figure 4 : Métier à tisser de Joseph-Marie Jacquard

Le métier à tisser de Jacquard :
https://www.youtube.com/watch?v=Ygj6l_p_h58
https://www.youtube.com/watch?v=K6NgMNvK52A
Les cartes perforées : https://www.youtube.com/watch?v=TNiOv4iTyrQ

Au milieu du XIX^{e} siècle, on s'approche conceptuellement et matériellement de l'ordinateur grâce aux idées et au travail exceptionnel d'un mathématicien anglais : Charles Babbage, considéré comme le grand-père de l'ordinateur pour avoir fait le rapprochement entre les machines à calculer et les systèmes de commande automatique de Jacquard.

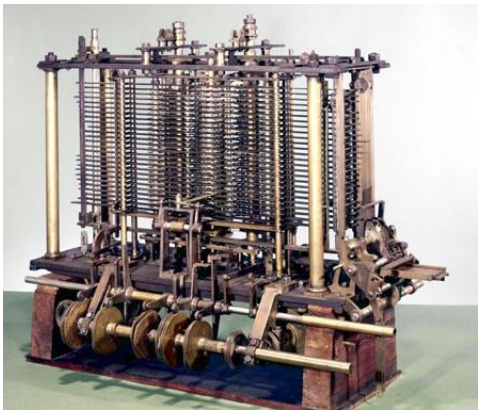


Figure 5 : Première machine de Babbage

De 1822 à 1832, Babbage est totalement absorbé par la conception d'une machine capable de calculer et d'imprimer des tables numériques selon la méthode des différences. Il construit un prototype, basé sur des roues dentées glissant sur des arbres actionnés par une manivelle. Avec l'aide de l'État et le soutien de la Royal Society, il entreprend la construction d'un modèle utilisable. Il se heurte alors à des problèmes techniques et son projet prend du retard. Après dix ans de travail acharné, les subventions sont suspendues et il doit abandonner sa machine à différences.

Les tables numériques que Babbage souhaitait générer à partir de sa machine devaient contenir :

- des valeurs de logarithmes,
- des fonctions trigonométriques (sinus, cosinus…),
- des racines carrées ou cubiques,

- ou encore des coefficients pour l'artillerie ou la navigation (angle à donner au canon pour atteindre une certaine portée, correction latérale à appliquer en fonction du vent…).

En 1833, Babbage se lance dans la réalisation d'une machine encore plus ambitieuse, la machine analytique. Elle était conçue pour faire des séquences d'opérations arithmétiques en fonction d'instructions données par l'utilisateur. Ce serait donc une machine aux applications les plus variées : le premier calculateur programmable ! On trouve dans sa machine analytique des idées très avancées pour l'époque, qui seront adoptées ou réinventées par les constructeurs d'ordinateurs une centaine d'années plus tard, comme la notion de processeur, de mémoire, de programme ou de techniques d'entrées/sorties par cartes perforées.

Babbage pouvait compter sur l'aide et le soutien d'un autre personnage de son temps, Ada Augusta comtesse de Lovelace (fille du poète anglais Lord Byron) qui laissa à la postérité des dessins et des descriptions de la machine analytique. Elle laissa aussi des programmes qui constituent une véritable première dans l'histoire de l'ordinateur. Malheureusement ces programmes ne seront jamais exécutés.

La machine de Babbage était probablement irréalisable avec les techniques et les outils de son temps. Babbage envisageait que sa machine soit alimentée par une machine à vapeur et était beaucoup trop complexe et ambitieuse. Elle ne sera jamais complétée. Babbage passa sa vie sur ce projet et y dépensa toute sa fortune. Il ne nous reste que des centaines de dessins et plans ainsi que quelques parties nous montrant quelles difficultés Babbage avait rencontrées dans la réalisation de son invention. Sa machine à différences devait connaître un meilleur sort. Reprise en main par un inventeur suédois, P.G. Scheutz, une version réduite sera réalisée avec l'aide de Babbage et présentée à Londres en 1854.

Babbage's Difference Engine No. 2, Part 1: The method of finite differences
https://www.youtube.com/watch?v=PFMBU17eo_4

Babbage's Difference Engine No. 2, Part 2: The Calculation Section
https://www.youtube.com/watch?v=vdra5Ms__9s

Babbage's Difference Engine No. 2, Part 3: Interlocks
https://www.youtube.com/watch?v=sklp9WutBhs

Babbage's Difference Engine No. 2, Part 4: The Control Section
https://www.youtube.com/watch?v=9wId_ZePJVs

Babbage's Difference Engine No. 2, Part 5: Initialization
https://www.youtube.com/watch?v=vgPLEhqFifg

1854 : George Boole (1815-1864), professeur au Queen's College en Irlande, définit une nouvelle forme de logique, fondée sur trois opérateurs ET, OU, NON, dans un ouvrage intitulé « An Investigation into the Laws of Thought » (Une étude des lois de la pensée). Nous arrivons à l'algèbre booléenne. Il fait ainsi passer la logique de la philosophie aux mathématiques. Une proposition vraie se voit affecter la valeur 1 tandis qu'une proposition fausse, la valeur 0. On peut ainsi effectuer des opérations sur des propositions unies par l'un des trois opérateurs logiques. Cette forme de logique s'accorde bien avec le calcul binaire et elle sera au centre des futurs ordinateurs.

1879 : L'algèbre booléenne prend sa forme actuelle, grâce aux travaux de Gottlob Frege.
C'est le philosophe anglais Bertrand Russell qui a principalement fait connaître les travaux de Frege.

Avant la fin du XIX^e siècle, l'Américain Hermann Hollerith construit un calculateur de statistiques fonctionnant avec des cartes perforées pour accélérer le traitement des données du recensement américain de **1890**. Inspiré par les travaux de Babbage et par les applications des cartons perforés dans la commande des métiers à tisser, il perfectionne la carte perforée et un système de codage des informations qui porte son nom. La carte de Hollerith comprenait douze rangées de vingt positions à perforer pour figurer les données du recensement de la population des États-Unis (âge, profession, situation de famille, etc.).

Une fois perforées, les cartes étaient placées dans des lecteurs qui détectaient les trous. Ceci à l'aide d'aiguilles qui traversaient les trous et établissaient un circuit électrique en trempant dans des petits pots de mercure placés de l'autre côté de la carte. Le système de Hollerith était si rapide qu'un premier décompte fut établi en quelques semaines et une analyse statistique complète en deux ans et demi. La population était passée en dix ans de 50 à 63 millions d'individus, mais le recensement de 1890 avait été fait en trois fois moins de temps qu'en 1880 ! Hollerith fonda la Tabulating Machines Company pour produire ses systèmes à cartes perforées. Sa compagnie rencontra un succès de longue durée. En 1924, cinq ans avant la mort de son fondateur, elle devint l'International Business Machines Corporation, ou IBM.

Dans le même temps, les premières machines analogiques utilisées pour effectuer des calculs mathématiques sont inventées comme le calculateur de marées par Lord Kelvin en **1872**. Sa machine utilise des roues avec des chevilles décentrées pour produire un mouvement sinusoïdal et une corde glissant sur des poulies pour additionner les différentes harmoniques.

2.4 XX^{e} siècle

Les machines à cartes perforées continuent à se développer pendant la première moitié du siècle. Durant cette période , on assiste aussi à la naissance d'une véritable industrie des calculatrices de table.

1900 : David Hilbert (1862-1943) est un mathématicien allemand qui a publié en 1899, un ouvrage sur les fondements de la géométrie. Il présente à la Conférence internationale des mathématiques de Paris de l'année 1900, une proposition d'agenda de recherche qui comprend une liste des 23 problèmes les plus urgents à résoudre en mathématiques pour le prochain siècle.

1903 : Bertrand Russell est un jeune mathématicien et philosophe anglais du début du XX^{e} siècle. Il publie en 1903 l'ouvrage majeur The Principles of Mathematics qui propose de nouveaux fondements à la logique et la théorie des ensembles : les lois de base de l'arithmétique sont réductibles à des propositions de logique élémentaire. Puis, en collaboration avec son tuteur de l'époque, North Whitehead (1861-1947), il entreprend d'appliquer ses théories aux mathématiques.

1904 : John Flemming invente le premier tube à vide. Il s'agit d'une diode pouvant prendre deux états ON/OFF.

1907 : À partir de la diode de John Flemming, L. De Forest met au point la « triode », un amplificateur d'intensité électrique. La première « lampe » réellement utilisable a été mise au point en France par H. Abraham et utilisée par l'ensemble des Alliés pendant la première guerre mondiale, d'où son nom : Lampe TM (pour Télégraphie Militaire).

1919 : Invention du basculeur d'Eccles et Jordan à partir de deux triodes. Plus connu maintenant sous le nom de flip-flop ou circuit bistable.

1930 : Vannevar Bush du M.I.T. et ses collègues conçoivent, en 1925, un prototype de machine analogique pour calculer des équations différentielles. Elle sera construite en 1930, sous le nom de The Differential Analyser. Elle doit servir à calculer les trajectoires de l'artillerie durant la Deuxième guerre mondiale.

1937 : Alan Turing (1912-1954) propose le modèle théorique d'une machine universelle capable de réaliser n'importe quel calcul mathématique. Son modèle est fondé sur l'idée que tout problème humain peut être résolu par une suite d'algorithmes comme ceux définis par Ada Lovelace. Il en fait la démonstration avec sa machine théorique, dite « Machine de Turing ». Celle-ci serait dotée d'une unité logique de traitement et d'une unité de contrôle qui puisse entraîner plusieurs rubans. Chacun de ces rubans contient des informations codées comme une suite de 0 et de 1. Le premier ruban représente les données, et le second représente le programme. Si on change le programme, on change la manière dont le système de contrôle modifie les données, ce qui lui donne son caractère



Figure 6 : Alan Turing

universel. La machine de Turing démontre également que certains problèmes de mathématiques et de logique sont insolubles : bien qu'ils soient clairement formulés, il n'est pas possible de trouver des solutions avec sa machine. Mais ces problèmes ne sont pas plus solubles par l'esprit humain. Il rejoint ainsi le théorème d'incomplétude de Kurt Gödel (1931). Néanmoins, la machine de Turing est si simple mais si puissante qu'elle va inspirer fortement les constructeurs des premiers ordinateurs.

Vers la fin des années trente, d'autres chercheurs parvinrent à la conclusion que la logique booléenne pouvait être employée efficacement dans la conception des calculateurs. L'adoption du système binaire au lieu du décimal était aussi dans l'esprit de certains pionniers. Konrad Zuse à Berlin, John Atanasoff à l'université de l'État de Iowa et George Stibitz aux Laboratoires Bell, travaillant indépendamment, construisirent des prototypes de machines binaires. **Dès 1936**, Zuse fabrique avec des moyens très modestes des machines électromécaniques, qu'il appelle Z1 et Z2, fonctionnant selon le système binaire. **En 1938**, il propose la construction d'un calculateur électronique comportant 1500 tubes à vide, mais le gouvernement allemand juge le projet irréalisable. Zuse construit alors un calculateur binaire universel avec 2600 relais de téléphone, le Z3. Les programmes sont introduits au moyen d'un film vidéo perforé. Une multiplication dure environ 5 secondes. Le Z3 sera terminé en **1941** et, comme d'ailleurs son successeur le Z4, sera effectivement utilisé pour résoudre des problèmes de conception aéronautique et de missiles.

1939 : À cette époque, Claude Shannon est étudiant de maîtrise en génie (électricité) au M.I.T. à Boston. Dans son mémoire, il propose une idée géniale : on peut décrire les états d'un relais électromagnétique en utilisant la logique booléenne.

1940 : C'est aux Anglais que l'on doit ce qui pourrait être le premier ordinateur opérationnel du monde, le Robinson. Il est fait de relais électromagnétiques et de tubes à vide. Il n'est pas programmable. Il est issu d'un effort de guerre du gouvernement britannique impliquant 10 000 personnes, pour construire un ordinateur; cet effort est connu sous le nom de projet Ultra. Le Robinson sert à décoder les messages secrets des Allemands au cours de la Deuxième guerre mondiale, messages encodés par une machine appelée Enigma inventée par les Nazis.

1943 : Le Colossus Mark I est mis au point en Angleterre, durant la Deuxième guerre mondiale, dans le cadre du projet Ultra. Les Allemands ayant perfectionné leur système d'encodage de messages, le Robinson ne suffit plus à la tâche. Le Colossus doit prendre la relève. On doit sa mise au point à Max M. A. Newman, professeur de mathématiques à la Cambridge University, à T. H. Flowers, ingénieur au Post Office Research Station et à Alan Turing, un Hongrois installé en Angleterre depuis 1936. Ce dernier inventa une machine universelle sur papier appelée la machine de Turing qui jette les bases de ce que sera l'ordinateur moderne. C'est également Turing qui adopte l'algorithme pour le calcul de certaines fonctions. Le Colossus, qui comporte 2 000 tubes à vide, peut lire des rubans perforés à la vitesse de 5 000 caractères à la seconde. Un seul ruban à la fois est donné à la machine. Les comparaisons sont effectuées électroniquement, ce qui est une innovation majeure pour l'époque, et les résultats sont conservés dans une mémoire de triodes thyratron remplies de gaz. Pour éliminer les erreurs, une pulsation d'horloge générée par une cellule photosensible permet de marquer la cadence des opérations. La programmation se fait à l'aide de cartes enfichables. Le Colossus Mark II, complété en 1944, aura 2 400 lampes et sera cinq fois plus rapide que le Colossus Mark I.
Plusieurs autres versions du Colossus verront le jour jusqu'à la fin de la guerre.

Le Colossus est généralement considéré comme l'un des premiers ordinateurs électroniques programmables, mais il ne répond pas entièrement à la définition d'un ordinateur universel au sens strict du terme (comme celle donnée plus tard par le modèle de Turing).

Voici un résumé nuancé :

Ce qui fait du Colossus un ordinateur (partiellement) :

- Électronique : Il utilisait des tubes à vide (environ 2 500) pour réaliser des calculs très rapides, ce qui en fait le premier ordinateur électronique fonctionnel.
- Programmable : Bien qu'il ne soit pas programmable au sens moderne (il n'avait pas de mémoire pour stocker des instructions ni un langage de programmation), il pouvait être configuré pour

effectuer différentes tâches de déchiffrement via des interrupteurs et des connexions manuelles.
- Automatisé : Il traitait automatiquement des données transmises à grande vitesse (bandes perforées), sans intervention humaine entre les étapes.

Ce qui lui manque pour être un véritable ordinateur au sens moderne :
- Pas universel : Il ne pouvait pas exécuter n'importe quel programme ; il était spécialisé dans la cryptanalyse des messages chiffrés allemands (notamment le chiffrement de Lorenz).
- Pas de mémoire d'instruction : Il ne stockait pas de programmes comme le feront les machines de type von Neumann.
- Pas de traitement généraliste : Il ne faisait pas de calculs arithmétiques généraux ou de traitement de texte.

Parmi les derniers précurseurs de l'ordinateur il faut citer le Harvard Mark I, une énorme machine électromécanique construite entre **1939 et 1944** par IBM et l'université de Harvard, sous la direction de Howard Aiken. Ce calculateur, capable de multiplier deux nombres de 23 chiffres décimaux en 6 secondes et d'effectuer des additions ou soustractions en trois dixièmes de seconde, était conçu sur la base du système décimal. Avec ses milliers de roulements à billes et ses 760 000 pièces électromécaniques, le Harvard Mark I était une sorte de machine analytique du XX^{e} siècle; en fait, il se révéla obsolète avant même d'être terminé.

En résumé :
- Le Colossus Mark I (1943-1944) est le premier ordinateur électronique partiellement programmable, utilisé pendant la Seconde Guerre mondiale par les Britanniques à Bletchley Park.
- Comme cela sera vu dans la suite des notes, l'ENIAC (1946), bien que programmable seulement en reconfigurant manuellement les câbles, est souvent considéré comme le premier ordinateur électronique généraliste.
- L'EDSAC (1949) et le Manchester Mark I (à ne pas confondre avec le Harvard Mark I) sont les premiers à respecter complètement le modèle de von Neumann (programme stocké en mémoire). NB : Mark = version, modèle.

2.5 Les ordinateurs électroniques universels : les différentes générations

2.5.1 Première génération : utilisation de tubes à vide

C'est en **1945** que John Presper Eckert et John Mauchly terminent à l'université de Pennsylvanie la construction de l'ENIAC (Electronic Numerical Integrator And Computer), machine universelle, programmable, numérique, basée sur le système décimal et entièrement électronique. Avec ses 18000 tubes et ses 30 tonnes, l'ENIAC pouvait multiplier deux nombres de 10 chiffres en 3 millisecondes ! C'était la fin des calculatrices électromécaniques dépassées par une technologie mille fois plus rapide.

L'ENIAC fonctionnait convenablement, traitant des millions de cartes perforées au cours de ses premiers essais. Mais l'inconvénient résidait dans la difficulté de modifier ses programmes. La mémoire interne était très petite et les programmes étaient câblés sur des fiches électriques interchangeables; il n'était pas possible de les enregistrer sur un support extérieur. Pour passer d'un calcul à l'autre, il fallait brancher et débrancher des centaines de câbles (il sera même modifié en 1948 pour intégrer son programme dans une mémoire non-modifiable).
Avant la fin de 1945, John von Neumann, un mathématicien d'origine hongroise, associé comme

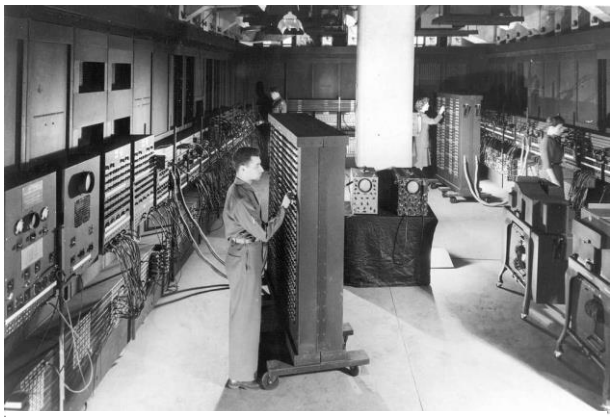


Figure 7 : L'ENIAC

consultant au projet ENIAC, franchit le dernier obstacle et proposa la construction de l'EDVAC (Electronic Discrete Variable Automatic Computer), machine modèle de l'ordinateur tel qu'on le conçoit à présent. Il accomplit une abstraction géniale du système de commande de la machine, en proposant d'enregistrer le programme en mémoire. La machine gagne ainsi en souplesse et en vitesse. Instructions et données sont stockées dans la mémoire même de la machine. Le programme peut décider quels calculs exécuter, quel chemin choisir pour la suite des opérations, sur la base des résultats intermédiaires. Le déroulement du programme peut être commandé par des décisions logiques, ce qui permet des sauts et des branchements conditionnels dans le programme. L'ordinateur est né.

On peut ainsi résumer les caractéristiques de l'ordinateur selon von Neumann :

- une machine universelle contrôlée par programme;
- les instructions du programme sont codées sous forme numérique (binaire) et enregistrées en mémoire avec les données;
- le programme peut modifier ses propres instructions, qui sont normalement exécutées en séquence, en effet, il peut accéder à une zone mémoire contenant une instruction, la modifier et l'exécuter ensuite;
- des instructions existent permettant les ruptures de séquence.

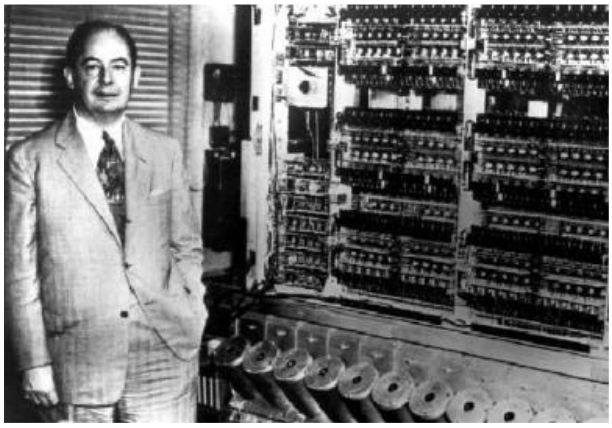


Figure 8 : John von Neumann

Dans une communication qui fera date, von Neumann décrit en 1945 les cinq composants essentiels de ce qui allait désormais être appelé **l'architecture de von Neumann :**

- **l'unité arithmétique et logique (UAL);**
- **l'unité de commande;**
- **la mémoire centrale;**
- **l'unité d'entrée;**
- **l'unité de sortie.**

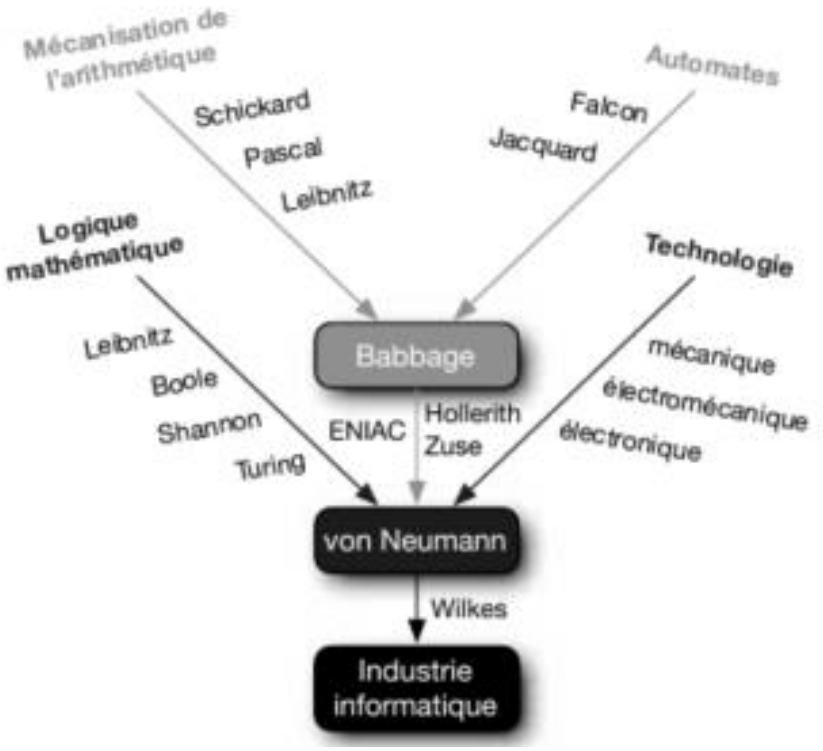
John von Neumann fait aussi remarquer qu'un tel système, pour être vraiment efficace et performant, doit fonctionner électroniquement et selon la numération binaire. Les principes de von Neumann guideront la conception des ordinateurs jusqu'à nos jours.

Un jour, en 1947, l'ENIAC tomba en panne. Après exploration, on constata qu'un insecte s'était logé dans un relais; le technicien qui a fait la découverte s'est écrié : « There is a bug in the machine » (bug = insecte). Le nom « bug » est resté pour désigner une erreur de matériel ou de programmation.

1946 : Le BINAC (Binary Automatic Computer) est l'oeuvre de Mauchly et de Eckert. Il s'agit en réalité de deux ordinateurs réunis qui calculent simultanément, leurs résultats étant ensuite comparés. On disait que le BINAC travaillait en temps réel car il fournissait des résultats dans l'immédiat. Le BINAC était un ordinateur extrêmement fiable.

1947 : Après l'ENIAC, il y a eu l'EDVAC auquel John von Neumann a travaillé. Il s'agit d'une machine binaire sérielle qui nécessite beaucoup moins de lampes que l'ENIAC. Elle possède aussi une mémoire interne beaucoup plus grande (100 fois plus grande). Avec l'EDVAC, l'invention de l'ordinateur numérique de première génération est à toute fin complétée. Ses plans, largement diffusés, ont donné lieu à plusieurs autres projets d'ordinateurs vers la fin des années 40 jusqu'en 1953. Mentionnons le SEAC, le SWAC, les Harvard Mark III et IV, le IAS Machine, les UNIVAC I et 1103, le WHIRLWIND et les IBM 701, 702. De 1953 à 1959, période reconnue comme étant celle de la première génération avancée d'ordinateurs, les IBM 650, 704, 705, 709, les UNIVAC II, 1103A, SS80, les Burroughs 205, 220, les NCR 120, 200, le Datamatic 1000, le RCA Bizmac ont vu le jour.

1947 : Parmi les projets d'ordinateurs de la fin des années 1940, il faut mentionner le SSEC (Selective Sequence Electronic Calculator), première machine d'IBM à programme enregistré. Depuis 1944, John von Neumann travaille avec l'équipe qui a mis au point l'ENIAC. Il imagine cet ordinateur à programme enregistré. La mémoire de l'ordinateur ne contient pas seulement des données mais également des instructions que l'ordinateur peut modifier pour choisir sa séquence de calcul, d'où son nom. Le SSEC était composé de 13 500 tubes à vide, 21 000 relais et il pouvait additionner 3 500 nombres de 14 décimales par seconde. Il multipliait 100 fois plus rapidement que le Mark 1.



2.5.2 Deuxième génération : l'ère du transistor



Figure 9 : Transistor

1947 : John Bardeen, Walter Brattain et William Shockley inventent le transistor aux Bell Laboratories du New Jersey, en 1947. Le transistor ressemble à un petit sandwich qui contient de la matière capable de conduire l'électricité à un voltage donné. Le transistor fait le même travail que la lampe à vide des premiers ordinateurs mais il est beaucoup plus petit, moins cher à fabriquer et beaucoup plus fiable.

Cependant, en raison de problèmes de production, ce n'est qu'à la fin des années 1950 que le transistor s'est répandu.

1956 : Le premier ordinateur à base de transistors, le Leprechaun, a été construit dans les laboratoires Bell. Plus de 5000 transistors sont utilisés dans cet ordinateur.

1959 : Le 1401 d'IBM est un ordinateur à transistors très populaire. IBM en a livré plus de 10 000 aux petites et moyennes entreprises.

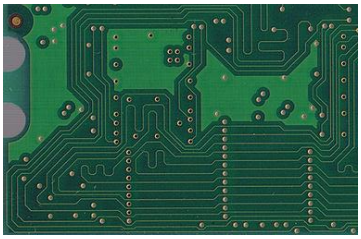
1960 : Le STRETCH d'IBM, comporte 150 000 transistors et a une vitesse de traitement de 200 instructions à la seconde. C'est le plus rapide de son époque, affirme IBM. C'est à l'occasion de la mise au point du STRETCH que le mot octet est employé pour la première fois. Un octet égale 8 unités binaires. Un octet désigne communément un caractère alphanumérique car chaque lettre ou chaque nombre est représenté par une série de 8 zéro ou 1.

1961 : Le premier système de travail à temps partagé sur ordinateur, le CTSS (Compatible Time Sharing System) a été mis au point au M.I.T. (Massachusetts Institute of Technology) à Boston, sous la direction du professeur F. Corbato. Il réunissait des ordinateurs IBM 709 et 7090. C'est un concept majeur pour l'époque, qui aura des retombées importantes : il s'agit de permettre à plusieurs usagers en même temps mais chacun à leur tour, sans qu'ils s'en aperçoivent, d'avoir accès à la puissance de calcul d'un gros ordinateur à partir de terminaux à distance.

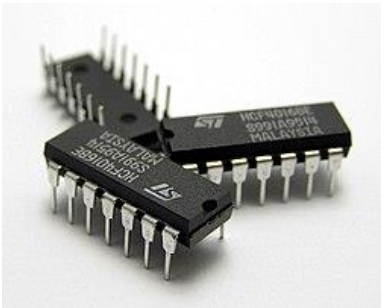
1959-1964 : La multiplication des transistors sur le marché a fait qu'ils ont complètement remplacé les lampes à vide dans les ordinateurs, à partir de 1959. Les ordinateurs de deuxième génération possédaient des mémoires principales magnétiques à tores (le tore est un anneau magnétique).
Quelques-uns d'entre eux possédaient comme mémoire auxiliaire des tambours magnétiques et des rubans magnétiques. Nommons le Philco 2000, les CDC 1604, 3600, les IBM 7000, 1400, les RCA 302, 501, le Honeywell 800, les UNIVAC III, 1107, les Ferranti Atlas.

2.5.3 Troisième génération : l'ère des circuits intégrés

L'invention du transistor va vite appeler le développement d'une technologie qui permettra de miniaturiser les autres composants de l'ordinateur et surtout de les intégrer, car les liaisons électriques multiples qui doivent se faire entre chaque transistor sont complexes, coûteuses à réaliser, pas assez rapides et peu fiables, en tout cas pas assez pour les militaires qui intègrent les composants électroniques dans le guidage des avions. Le circuit imprimé va résoudre ce problème puis le circuit intégré. Dans les premiers ordinateurs, les éléments des circuits, tels les lampes, étaient reliés entre eux par des réseaux extrêmement complexes de fils.
Le premier développement important avec l'apparition du transistor a été de monter les transistors sur une même plaque de circuits et de graver les fils qui les reliaient dans la plaque; on a appelé cela des circuits imprimés.


Figure 10 : Circuits imprimés

Par la suite, grâce d'abord aux travaux d'un ingénieur de la Texas Instruments, Jack Kilby, et ensuite, d'un ingénieur de Fairchild Semiconductors, Robert Noyce, en 1959, on a été capable de relier entre eux tous les éléments du circuit, transistors, diodes, condensateurs, fils, etc. dans des circuits dits complètement intégrés, manufacturés en une seule opération. Si, à ce moment-là, cette percée technologique n'a pas connu tout le succès qu'elle méritait, on prétendait que le coût en était trop élevé, le lancement du Spoutnik par les Russes, l'inquiétude qui s'en est suivie chez les Américains d'être surpassés et le désir du président d'alors, John Kennedy, d'envoyer un homme sur la Lune,


Figure 11 : Circuits intégrés

acheva de convaincre, car il fallait pour réaliser cette conquête de la Lune, des ordinateurs assez petits pour tenir dans une capsule spatiale. Par exemple, l'ordinateur de la cabine Apollo ne pesait que 54 livres, soit le centième du poids de certains gros ordinateurs de l'époque.

Évidemment, ces efforts de la NASA et de l'armée américaine également, qui avait besoin de petits ordinateurs pour guider ses fusées Minuteman, ont permis de financer la recherche et développement sur les circuits intégrés et d'en réduire sensiblement les coûts de fabrication, si bien que les circuits intégrés se sont retrouvés non seulement dans les nouveaux ordinateurs mais également dans toutes sortes d'appareils électroniques y compris les appareils domestiques, radio, télévision, horloges, etc. C'est donc grâce aux circuits intégrés qu'on a pu construire une nouvelle génération d'ordinateurs appelée mini-ordinateurs dont les coûts étaient très inférieurs aux gros mais encore trop chers pour le commun des mortels.

Les circuits intégrés, appelés aussi puces (ou chips en anglais), sont des composants électroniques miniaturisés gravés sur une pastille de silicium, un matériau semi-conducteur. Ces puces contiennent de nombreux transistors et autres composants interconnectés, organisés pour exécuter des fonctions logiques, arithmétiques, de mémoire ou de gestion.

L'évolution des circuits intégrés s'est traduite par une augmentation continue du nombre de composants qu'ils peuvent contenir. On parle alors de niveaux d'intégration :

Niveau d'intégration	Abréviation	Nombre de transistors sur une puce (surface de 1 à 800 mm^2 (multi-chiplets))
Intégration à petite échelle	SSI (Small Scale Integration)	Quelques dizaines
Intégration à moyenne échelle	MSI (Medium Scale Integration)	Jusqu'à quelques centaines
Intégration à grande échelle	LSI (Large Scale Integration)	Plusieurs milliers
Intégration à très grande échelle	VLSI (Very Large Scale Integration)	De plusieurs millions à quelques milliards
Intégration ultra-large	ULSI (Ultra Large Scale Integration)	Plusieurs dizaines de milliards de transistors

Aujourd'hui, les circuits ULSI sont la norme : par exemple, un processeur Apple M1 contient environ 16 milliards de transistors, et certaines puces récentes dépassent les 25 milliards.

Cette densité de composants a été rendue possible grâce aux progrès des technologies de gravure (passant de 10 µm à 3 nm, voire moins). Plus la gravure est fine, plus il est possible de placer de transistors sur une surface réduite, ce qui augmente les performances tout en réduisant la consommation énergétique.

Réduction des coûts

L'industrialisation des circuits intégrés a permis une chute spectaculaire du coût par transistor. Alors qu'un transistor coûtait plusieurs dollars dans les années 1960, il revient aujourd'hui à moins d'un millionième de centime. Cela a permis une démocratisation de l'informatique, en rendant accessibles des appareils toujours plus puissants pour un coût raisonnable.

2.5.4 Quatrième génération : l'ère des microprocesseurs

La mise au point des microprocesseurs va entraîner la miniaturisation des composants d'ordinateurs et l'apparition de deux nouveaux types d'ordinateurs : le super ordinateur et le micro-ordinateur ou ordinateur personnel. La miniaturisation va aussi permettre l'invention des calculatrices de poches, des montres à affichage numérique, des contrôles numériques d'appareils domestiques comme le four à micro-ondes, la machine à laver, etc...

1968 : Robert Noyce et Gordon Moore quittent la Fairchild pour fonder Intel avec chacun un investissement de 250 000 $. Intel Corp. va devenir le plus gros fabricant de puces et de microprocesseurs au monde.

1969 : La société AMD est créée par W.J. Sanders III de Fairchild et sept autres personnes pour entrer en compétition avec Intel sur le marché des microprocesseurs.

1970 : C'est en 1970, que Bob Abbott, sous la direction de Les Vadasz, met au point chez Intel, la première puce à mémoire dynamique, la DRAM (Dynamic Read Access Memory), Intel 1103, qui deviendra à compter de 1972, la plus vendue à travers le monde. Elle n'avait qu'un seul kilo-octet. Sa vitesse était de 300 nanosecondes. Il faudrait aujourd'hui 65 000 de ces puces 1103 pour faire 8 Mo de mémoire. Les puces ne sont pas fabriquées à l'unité. Elles sont à peine grandes comme des timbres-poste. Après l'étape du design, on fabrique des masques par processus photographique répétés de telle sorte à obtenir jusqu'à 200 fois l'image réduite du circuit de la puce sur une même gaufre. Par conséquent, plus la puce est réduite, plus on peut en mettre sur une gaufre (wafer) et plus le coût de revient est bas.

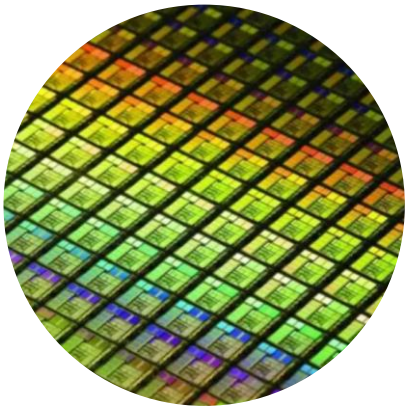


Figure 12 : Le wafer

1971 : Intel crée la première puce à mémoire morte qu'on peut programmer électroniquement, la EPROM (Electronic Programmed Read Only Memory) Intel 1702.

1971 : La compagnie Intel, fondée en 1968, progresse très rapidement, en particulier grâce au travail de Ted Hoff, jeune ingénieur et de son équipe formée de Stan Mazor, Federico Faggin et Masatoshi Shima. Cette équipe est la première à concevoir le design d'un ordinateur sur une puce, autrement dit un microprocesseur. Le premier microprocesseur a été fabriqué pour répondre à une commande spéciale d'une compagnie japonaise, la Busicom, qui voulait intégrer une série de 6 puces très complexes à son nouveau modèle de calculatrice. Ted Hoff leur fait savoir que leur design est beaucoup trop complexe à réaliser; il se met alors au travail devant son miniordinateur PDP-8 et il se demande pourquoi une simple calculatrice devait être si complexe. Il fait alors le design du microprocesseur qui n'a pas réussi à impressionner les Japonais. Il en rachètera les droits à Busicom au coût de 60 000$ US. Mais Bob Noyce, le fondateur d'Intel avec Gordon Moore, continue à supporter le projet de Hoff. Un microprocesseur à 4 bits voit le jour, le 4004 : 2 300 transistors de 10 microns sur une puce de 12 mm carré. Le 15 novembre 1971, le 4004 est lancé sous le slogan « A micro-programmable computer on a chip ! » dans une publicité parue dans le Electronic News de New York. Le Intel 4004 tournait à la vitesse de 108 KHz, soit 0,06 MIPS (million d'instructions par seconde). Dans le cas du 4004, il fallait en moyenne 1,8 cycles d'horloge pour exécuter une instruction. Sa mémoire adressable était de 640 octets. Il fallait 16 griffes pour le fixer.

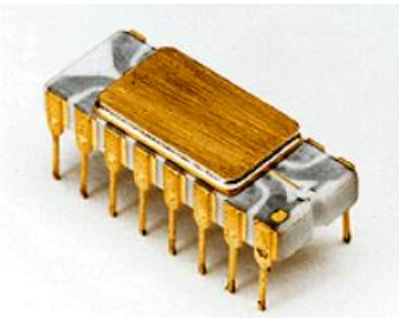


Figure 13 : Microprocesseur Intel 4004

1973 : A. Truong, fondateur de R2E, une petite société Française, crée le premier microordinateur du monde : le Micral-N.

1973 : Le prototype Alto, de XEROX, pensé pour devenir le bureau du futur, est un condensé des idées proposées par les chercheurs réunis par XEROX au Palo-Alto Research Center (PARC). Il est le premier à introduire l'idée de fenêtres et d'icônes que l'on peut gérer grâce à une souris.
Principalement, en raison de son coût, cet ordinateur ne connaîtra qu'un succès d'estime.

1975 : Ed Roberts fonde son entreprise, la MITS, qui met sur le marché l'Altair 8800, premier micro-ordinateur en kit.



Figure 14 : Altair 8800

1977 : Apple, fondée par Steve Jobs et Steve Wozniak avec l'aide d'un ex-ingénieur millionnaire d'Intel, Mike Markula, lance en grande pompe en avril 1977 (les livraisons commenceront en juin de la même année) le Apple II, un ordinateur à 4 Ko de mémoire vive (RAM) extensible à 16 Ko, 16 Ko de mémoire morte (ROM) et à processeur MOS 6502 de 1 MHz, le même que le Pet de Commodore et un clavier intégré de 52 touches.



Figure 15 : Apple II

Au départ, il comportait un lecteur-enregistreur de cassettes puis à compter de mars 1978 d'un lecteur de disquettes 5,25 po. On le branchait à un téléviseur noir et blanc ou couleur ou à un écran monochrome séparé qui affiche en 40 colonnes et 24 lignes, bien que le Apple II soit déjà capable d'afficher des graphiques haute résolution en couleur, ce qui constitue une innovation. Il coûtait 1 295 $ US initialement mais deux ans plus tard, son prix baissera à 970 $ pour un modèle à 2 Ko de mémoire alors qu'un modèle à 48 Ko coûtait 1 795 $ US.

1978 : C'est le 8 juin 1978, qu'Intel lance le premier microprocesseur à 16 bits, le 8086 à 4,77 MHz (0,33 MIPS[2]) , et 8MHz (0,66 MIPS) ou 10 MHz (0,75 MIPS), qui deviendra, à l'époque, un standard de l'industrie. Il comprenait 29 000 transistors de 3 microns soit six fois plus que le 8080. Il permettait la division et la multiplication accélérant ainsi les calculs de précision. La performance du 8086 est dix fois celle de son ancêtre à 8 bits, le 8080. Son bus est de 16 bits et sa mémoire adressable est de 1 Mo.

[2] MIPS : nombre de millions d'instructions machine par seconde

1980 : Motorola introduit un microprocesseur 32 bits à bus interne de 16 bits et bus de mémoire à 24 bits qui équipera les premiers LISA de Apple et les premiers Macintosh mis sur le marché en 1984. Il équipera également les Amiga de Commodore et les ST de Atari et quelques stations de travail UNIX.

1981 : Basé sur le microprocesseur Z80A, le ZX81 est le premier micro-ordinateur à moins de 100 $. Il utilise l'écran du poste de télévision.

1981 : IBM, sentant Apple devant elle dans un nouveau marché, annonce en août 1981, qu'il se lance sur le marché du micro-ordinateur avec le concept d'ordinateur personnel qui prend la forme du IBM PC construit autour du microprocesseur Intel 8088 à 8-16 bits et 4,77 MHz. Il avait 16 Ko de mémoire vive, extensible à 64 Ko , un seul lecteur de disquettes simple face de 5,25 po d'une capacité de stockage de 160 Ko. Il était doté d'un écran monochrome vert de 12 po. Il était vendu à 2 880 $.
Un compétiteur géant pour Apple venait de voir le jour. Le PC a été créé par une équipe de jeunes informaticiens dirigée par Philip D. Estridge; cette équipe comprenait Bill Sydnes, ingénieur chef, Dan Wilkie, responsable de la fabrication et H.L. Sparks, responsable des ventes. Il fonctionne sous un système d'exploitation mis au point par Microsoft sous contrat avec IBM, le MS DOS et le PC DOS. Ce système est un héritage de cerveaux d'informaticiens qui ne se soucient pas de convivialité. C'est une machine qui nécessite la formation de l'utilisateur. Malgré cette absence de convivialité, le PC prend une bonne part du marché; on en vend 35 000 en 1981 soit cinq fois plus que ce qui avait été prévu au départ, aidée en cela par une campagne de publicité bien orchestrée autour du personnage de Charlie Chaplin des Temps modernes (https://www.youtube.com/watch?v=NFmHn6DJOZs).

1982 : Compaq lance le premier clone de l'IBM PC.

1984 : Apple lance le Macintosh. Basé sur le projet LISA , c'est l'ordinateur convivial par excellence : Son utilisation est très simple grâce à la souris et à la qualité de ses graphismes. Il devient au fil des années et des versions, l'autre grand standard (avec le PC d'IBM) du monde de la micro-informatique.

1985 : Microsoft lance Windows 1.0, une interface graphique rudimentaire venant se greffer sur MS-DOS. L'expérience utilisateur reste limitée mais marque le début de la domination de Microsoft dans l'informatique personnelle.

1989 : Introduction du processeur Intel 80486, premier à intégrer un coprocesseur mathématique et une architecture pipeline, avec plus d'un million de transistors.

1991 : Lancement du système Linux par Linus Torvalds, système libre inspiré de UNIX. Il connaîtra un succès grandissant dans les serveurs et systèmes embarqués.

1993 : Intel introduit le Pentium, remplaçant la série 80486. Il utilise une architecture superscalaire[3] permettant l'exécution de plusieurs instructions par cycle.

1995 : Windows 95 est lancé, avec une interface utilisateur modernisée, le menu « démarrer » et le support du plug-and-play. Il est couplé aux processeurs Pentium.

2000 : Lancement de Windows 2000, puis XP en 2001, très stables et largement utilisés en entreprise comme à domicile.

2006 : Apple abandonne les processeurs PowerPC pour adopter l'architecture Intel x86, un changement majeur dans la stratégie matérielle de la firme.

2008 : Introduction du premier processeur Intel Core i7, marquant l'avènement des architectures multicœurs grand public.

2010 : Prolifération des SSD (Solid State Drives), qui remplacent progressivement les disques durs mécaniques pour le stockage système, grâce à leur rapidité.

[3] Une architecture superscalaire est une architecture de processeur capable d'exécuter plusieurs instructions par cycle d'horloge grâce à la présence de plusieurs unités de traitement travaillant en parallèle.

2012 : Introduction de Windows 8, avec une interface tactile controversée. Microsoft vise la convergence PC/tablette.

2015 : Windows 10 succède à Windows 8, avec un retour à une interface plus classique et des mises à jour continues.

2020 : Apple annonce sa transition vers ses propres processeurs ARM, les Apple Silicon M1, mettant fin à 15 ans de collaboration avec Intel. Ces puces offrent une excellente performance par watt.

2020–2024 : montée en puissance des processeurs hybrides (x86 big.LITTLE) chez Intel (séries Alder Lake et Raptor Lake), mêlant cœurs haute performance et cœurs basse consommation.

2022–2025 : démocratisation des GPU pour l'intelligence artificielle, avec des cartes graphiques NVIDIA RTX ou AMD Radeon utilisées pour le calcul parallèle, notamment dans l'IA générative.

2.5.5 Et le futur...

L'évolution des ordinateurs dans les 10 prochaines années (2025–2035) s'annonce profondément transformante, non seulement sur le plan matériel, mais aussi dans la manière dont nous interagirons avec les machines. Voici un aperçu structuré des grandes tendances probables, fondé sur les avancées actuelles et les recherches prospectives.

2.5.5.1 Puissance de calcul : vers une spécialisation accrue

Aspect	Évolution attendue
Processeurs	Architecture hétérogène (big.LITTLE), gravure inférieure à 2 nm, multi-chiplets, SoC[4] plus intégrés
GPU & IA	Accélérateurs spécialisés IA (NPU : Neural Processing Unit), GPU massivement parallèles pour le calcul scientifique et graphique
Mémoire	Unification RAM/stockage (type CXL, HMC, MRAM), montée en capacité et rapidité

Traditionnellement, un ordinateur utilise deux types de mémoire bien distincts :

- RAM (mémoire vive) : rapide, mais volatile (elle perd les données à l'arrêt).
- Stockage (HDD, SSD) : non volatile, mais plus lent.

Cette séparation pose des problèmes de performance et de gestion de la mémoire à grande échelle.

Que signifie unification RAM / stockage ?

L'unification vise à réduire ou abolir la frontière entre RAM et stockage pour :

- accélérer les échanges entre mémoire et disque ;
- simplifier l'architecture informatique ;
- réduire les besoins en transferts (ce qui est coûteux en énergie et temps).

C'est un objectif majeur pour les centres de données, l'IA et les architectures hautes performances.

Technologies associées

1. CXL (Compute Express Link)

- CXL est une nouvelle interface à très haute vitesse (basée sur PCIe) permettant aux processeurs d'accéder de manière unifiée à différents types de mémoire (RAM classique, mémoire persistante, mémoire partagée entre CPU et GPU, etc.).
- Elle permet de partager la mémoire entre plusieurs composants (CPU, GPU, FPGA) sans copie redondante.

[4] Un SoC est une puce unique qui peut contenir le processeur (CPU), la mémoire, la carte graphique (GPU), les interfaces d'entrée/sortie, et parfois même le modem réseau, le contrôleur USB, etc.

2. HMC (Hybrid Memory Cube)

- Architecture mémoire 3D empilée, plus rapide que la DRAM traditionnelle.
- Combine haut débit et faible consommation.
- Permettrait d'unifier RAM et cache hautes performances.

3. MRAM (Magnetoresistive RAM)

- Mémoire non volatile, mais presque aussi rapide que la RAM.
- Idéal pour remplacer à la fois la RAM et le stockage.
- Peut conserver les données même sans alimentation électrique.

Montée en capacité et en rapidité

Ces technologies permettent d'(de) :

- augmenter la capacité mémoire disponible (RAM + stockage réunis) ;
- accélérer le traitement des données (moins de goulots d'étranglement) ;
- réduire la latence d'accès aux données ;
- optimiser l'usage de l'énergie.

2.5.5.2 Ordinateurs quantiques (et post-quantiques)

- Des prototypes fonctionnels (IBM, Google, D-Wave) existent déjà.
- À horizon 10 ans :
  - Calculs spécialisés en chimie, cryptanalyse, optimisation.
  - Pas encore un remplacement du calcul classique, mais complémentarité.

En parallèle, émergence de la cryptographie post-quantique pour sécuriser les systèmes.

2.5.5.3 Ordinateur distribué, edge et cloud

Évolution	Détail
Cloud computing	Davantage de calcul effectué sur des serveurs distants (intégration IA, traitement de données massives).
Edge computing	Traitement local sur des objets connectés, avec faible latence (voiture, domotique, santé).
Ordinateurs personnels	Moins isolés, plus interconnectés aux plateformes distantes.

La frontière entre local et distant devient floue.

2.5.5.4 Interaction homme-machine repensée

Domaine	Avancées prévues
Interface vocale	Généralisée, contextuelle, multilingue (grâce aux IA génératives).
Réalité augmentée/virtuelle	Intégration dans l'environnement de travail (écrans virtuels, interfaces immersives).
Neural interfaces	Débuts d'expérimentation (ex. : Neuralink), applications médicales d'abord.

2.5.5.5 Systèmes plus intelligents, plus autonomes

- IA embarquée : assistants personnels contextuels, correction automatique, automatisation de tâches complexes.
- Personnalisation adaptative : les systèmes apprendront ton style, tes besoins, tes usages.

Les ordinateurs s'ajusteront à l'utilisateur, plutôt que l'inverse.

2.5.5.6 Sécurité et souveraineté numérique

- Enjeux critiques : protection des données, cybersécurité, confidentialité.
- Vers des OS plus compartimentés, des matériels avec sécurité intégrée (TPM, enclaves sécurisées).
- Augmentation des tensions géopolitiques autour de la souveraineté technologique (USA, Chine, Europe).

Le TPM est une puce (ou module logiciel) intégrée à la carte mère d'un ordinateur.

Elle est conçue pour assurer des fonctions de sécurité matérielle, comme :

- le chiffrement des données,
- la protection des clés cryptographiques,
- l'authentification du système,
- la vérification de l'intégrité au démarrage (boot sécurisé).

Dans des OS plus sécurisés et compartimentés, comme Windows 11, Linux avec Secure Boot, ou les systèmes de virtualisation :

- Le TPM garantit que le système d'exploitation démarre dans un état sûr.
- Il peut sceller des données à une configuration matérielle précise (elles deviennent inaccessibles si la machine est modifiée).
- Il sert à activer des fonctions comme BitLocker (chiffrement de disque sous Windows).
- Il protège l'accès aux enclaves sécurisées ou environnements d'exécution isolés.

Une enclave sécurisée (ou environnement d'exécution sécurisé) est une zone protégée à l'intérieur d'un processeur ou d'un système informatique, conçue pour exécuter du code et stocker des données de manière isolée, même en cas de compromission du reste du système.

Objectif principal

Protéger les données sensibles (mots de passe, clés cryptographiques, algorithmes confidentiels…) contre toute tentative d'accès non autorisé, même si l'attaquant a un contrôle partiel sur le système (comme un OS compromis).

Fonctionnement général

- Le processeur crée une zone mémoire isolée (l'enclave) avec des protections matérielles.
- Seul le code autorisé à l'intérieur de cette enclave peut accéder aux données qu'elle contient.
- Même le système d'exploitation, l'hyperviseur ou un administrateur ne peuvent y accéder.
- Lorsqu'un programme a besoin de traiter des données sensibles, il les envoie temporairement dans l'enclave.

3 Notions de système, système de pilotage, système d'information, système opérant, système informatique...

3.1 Définitions

Un système est un ensemble d'éléments en interaction dynamique et organisés en fonction d'un but.

Si on se réfère à cette définition, une entreprise est bien un système, étant donné qu'elle est composée d'un ensemble d'éléments (services, départements...) organisés en fonction d'un but (produire, vendre, analyser des résultats...).
D'autre part, ces éléments sont en interaction dynamique les uns avec les autres mais aussi avec le monde extérieur (banques, clients, fournisseurs...).

Un système est constitué notamment des trois parties essentielles suivantes :

- le système de pilotage
- le système d'information
- le système opérant

Système de pilotage : Il correspond à la direction et au management, qui fixent les objectifs, prennent les décisions et contrôlent les résultats. Le système d'information fournit des indicateurs, des rapports et des analyses pour aider à la prise de décision et reçoit des instructions ou des stratégies à appliquer.

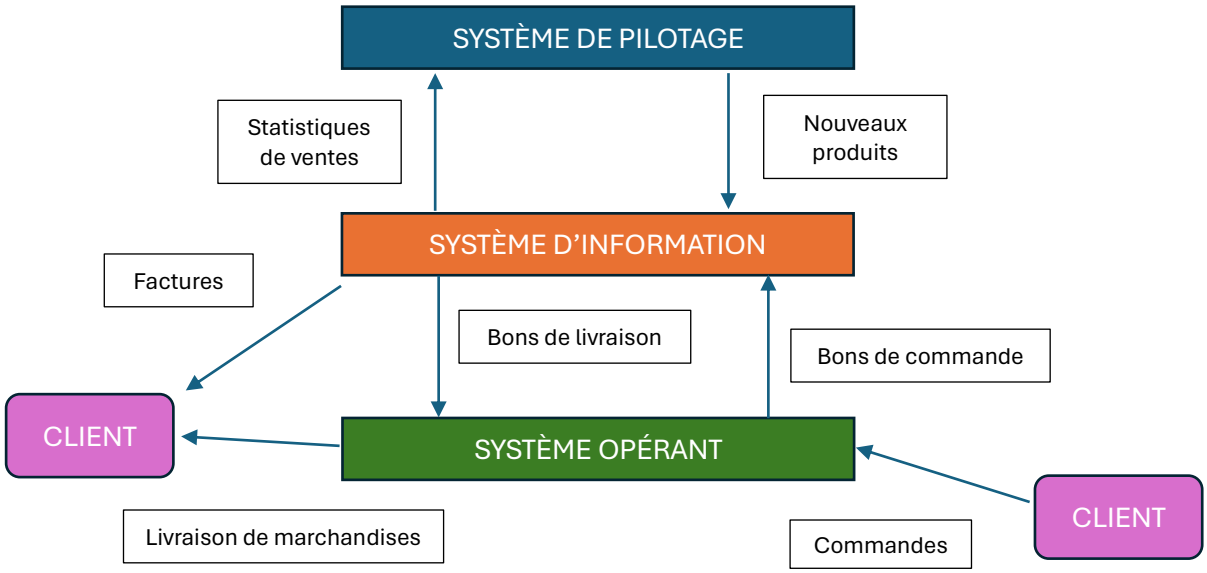
Système d'information : C'est ce qui intervient entre les deux autres systèmes. Ce système s'occupe de collecter, stocker, transformer et diffuser des données et informations dans le système opérant et le système de pilotage.

Système opérant : Il regroupe les activités de production, de vente, de logistique, etc. Le système d'information collecte et transmet les informations nécessaires à ces opérations et reçoit en retour des données issues de l'activité réelle (commandes, stocks, livraisons, etc.).

3.2 Le système d'information

Un système d'information est composé de différents éléments (ordinateurs, employé(e)s, règles de gestion...) dont le rôle est de collecter, stocker, traiter, gérer et diffuser des données et informations dans le système opérant et de pilotage.

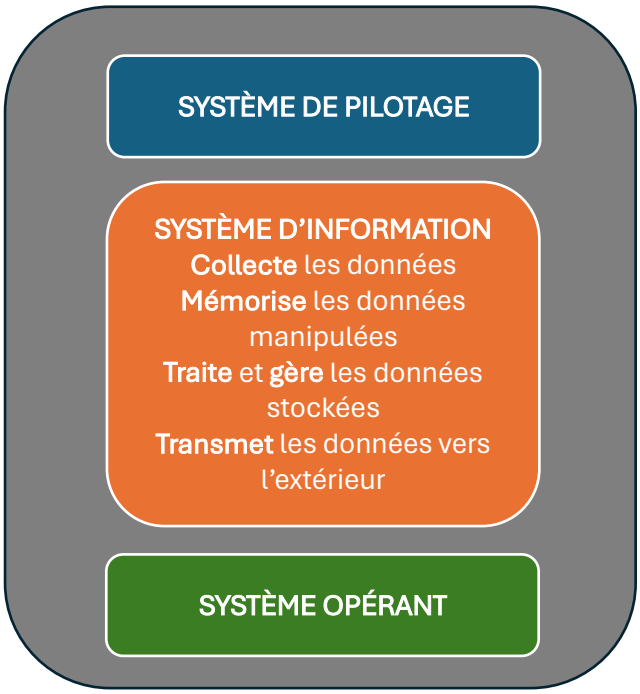
Exemples de flux entre le système d'information et les autres éléments :



Le système d'information a donc 4 fonctions principales :

- **Collecter** les informations provenant d'autres éléments du système ou de l'environnement extérieur.
- **Stocker** les informations.
- **Traiter** et **gérer** les données stockées.
- **Transmettre** les informations vers d'autres composants du système ou vers l'environnement extérieur.

3.3 Schéma récapitulatif



3.4 Ne pas confondre « Système d'information » et « Système informatique »

Les notions de système d'information et de système informatique ne sont pas strictement synonymes, même si elles sont souvent confondues dans le langage courant. Elles désignent deux réalités différentes mais complémentaires dans le domaine de l'informatique et de la gestion de l'information.

- **Système d'information**

Petit rappel : Un système d'information est un ensemble organisé de ressources (personnes, procédures, données, matériels, logiciels) permettant de collecter, stocker, traiter, gérer et diffuser l'information au sein d'une organisation.

Composants :

- Ressources humaines (utilisateurs, décideurs)
- Procédures et règles de gestion
- Données et informations
- Moyens techniques (matériel, logiciels)

Objectif : Fournir l'information utile à la bonne marche de l'organisation, soutenir la prise de décision, coordonner les activités.

- **Système informatique**

Définition : Sous-ensemble **technique** du système d'information, constitué uniquement des équipements matériels (ordinateurs, serveurs, réseaux, etc.) et des logiciels (systèmes d'exploitation, applications) qui permettent l'automatisation du traitement de l'information.

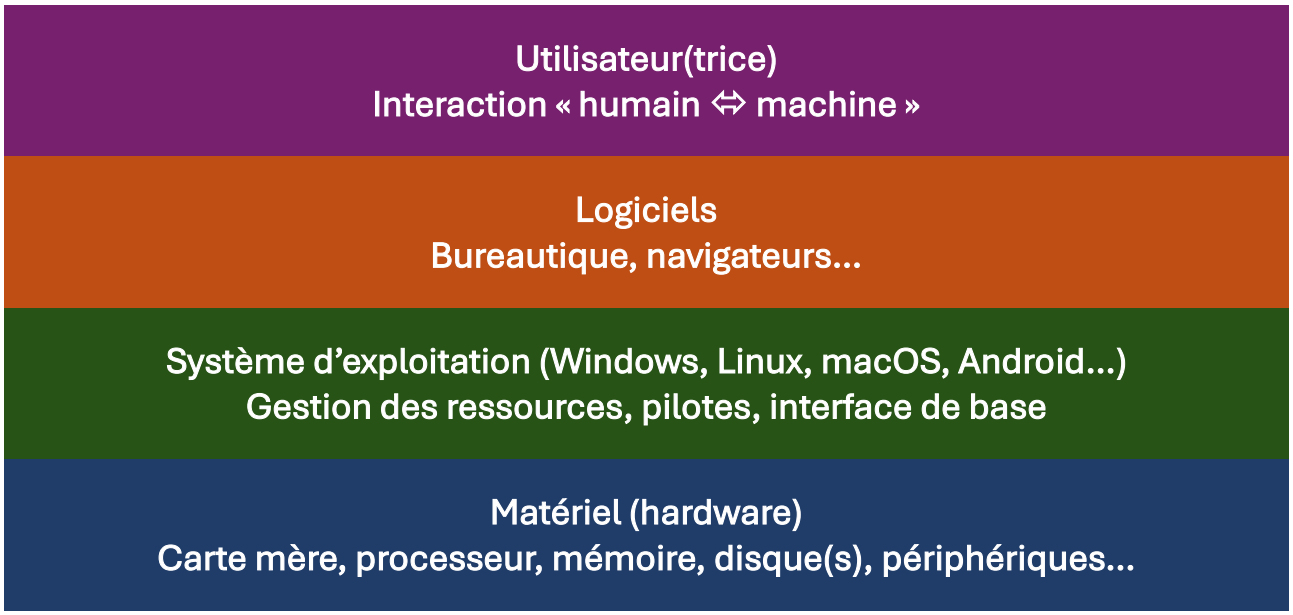
Composants :
- ✓ Matériel informatique (hardware)
- ✓ Logiciels (software)
- ✓ Réseaux et infrastructures techniques

Objectif : Assurer le traitement automatisé des données et le fonctionnement technique des applications.

3.5 Les couches d'un système informatique

Voici un schéma en couches d'un système informatique :
- Matériel (Hardware) : processeur, mémoire, périphériques.
- Système d'exploitation : gère les ressources, les pilotes et fournit une interface de base.
- Logiciels applicatifs : bureautique, navigateurs, logiciels métiers.
- Utilisateur : interagit avec les applications et, indirectement, avec tout le système.

Chaque couche s'appuie sur la précédente et offre des services à la suivante.



3.6 Le système d'exploitation

3.6.1 Fonctions essentielles d'un système d'exploitation (SE)

Un système d'exploitation joue le rôle de chef d'orchestre entre le matériel et les utilisateurs. Ses fonctions principales peuvent être décrites en 5 grands domaines :

- **Gestion du processeur (CPU)**
  - Ordonnancement des processus : décider quel programme s'exécute et quand (time-sharing, multitâche).
  - Concurrence : gérer l'exécution simultanée de plusieurs processus sans interférence.
  - Exemple concret : lancer Word et un navigateur en parallèle → le SE répartit le temps processeur.

- **Gestion de la mémoire**
  - Allocation : réserver des zones mémoire aux programmes.
  - Protection : empêcher qu'un programme accède à la mémoire d'un autre.
  - Mémoire virtuelle : utiliser le disque dur comme extension de la RAM (fichier d'échange).
  - Exemple concret : ouvrir une grande image → si la RAM est insuffisante, le SE bascule vers la mémoire virtuelle.

- **Gestion des entrées/sorties (E/S) et des périphériques**
  - Pilotes (drivers) : traduire les commandes du SE en instructions compréhensibles par chaque périphérique.
  - Buffering et spooling : gérer l'attente et l'envoi de données vers les périphériques (ex. files d'impression).
  - Exemple concret : imprimer un document → le SE place la tâche dans une file d'attente gérée par le pilote de l'imprimante.

- **Gestion du système de fichiers**
  - Organisation hiérarchique (répertoires, sous-répertoires).
  - Opérations fondamentales : création, lecture, écriture, suppression, copie, déplacement.
  - Sécurité : gestion des droits d'accès (lecture, écriture, exécution).
  - Exemple concret : enregistrer un document Word dans « Mes documents » → le SE gère son emplacement, son nom et ses droits d'accès.

- **Gestion de la sécurité et des utilisateurs**
  - Comptes utilisateurs : identifiant, mot de passe, profils.
  - Permissions et privilèges : définir qui peut lire/écrire/exécuter un fichier.
  - Protection du système : antivirus, mises à jour de sécurité.
  - Exemple concret : sous Windows, un utilisateur « invité » ne peut pas installer un logiciel sans droits administrateur.

- **Interface utilisateur**
  - Interface en ligne de commande : commandes textuelles (ex. Linux/Unix).
  - Interface graphique (GUI) : fenêtres, menus, icônes (Windows, macOS).
  - Exemple concret : cliquer sur l'icône d'un logiciel → le SE interprète cette action et lance le programme.

3.6.2 Liens entre les fonctions du système d'exploitation et le schéma en couches

Comme cela a été vu précédemment, on peut représenter un système informatique en couches hiérarchisées. Chaque couche s'appuie sur la précédente et fournit des services à la suivante. Voyons les liens entre les fonctions du système d'exploitation et le schéma en couches.

- **Couche « matérielle » (Hardware)**
  - Processeur, mémoire, périphériques.
  - Le système d'exploitation (SE), via ses fonctions de gestion CPU, mémoire et E/S, contrôle directement cette couche.

- **Couche « système d'exploitation »**
  - Fait le lien entre le matériel et les applications.
  - Intègre la gestion des fichiers, la sécurité, l'interface utilisateur.
  - Exemple : l'utilisateur demande « ouvrir un fichier » → le SE traduit cette action en opérations sur disque et envoie les données à la RAM.

- **Couche « logiciels »**
  - Programmes utilisés par l'utilisateur : traitement de texte, navigateur, jeu vidéo, logiciel métier.
  - Ces logiciels ne parlent pas directement au matériel, mais passent par le SE (API, bibliothèques système).
  - Exemple : un navigateur demande au SE d'accéder à la carte réseau pour charger une page.

- **Couche utilisateur**
  - Interagit avec les applications à travers l'interface fournie par le SE (de manière directe ou indirecte).
  - Exemple : cliquer sur « Imprimer » → l'application envoie la commande au SE, qui gère la file d'impression et communique avec l'imprimante.

3.7 Les différentes catégories de systèmes informatiques

3.7.1 Introduction

Les systèmes informatiques sont omniprésents dans notre quotidien et se déclinent en plusieurs catégories, chacune adaptée à des besoins spécifiques. Ce chapitre présente et caractérise les principales familles de systèmes informatiques : supercalculateurs, ordinateurs centraux, serveurs, stations de travail, micro-ordinateurs et systèmes embarqués.

3.7.2 Classification selon la taille et la capacité de traitement

3.7.2.1 Supercalculateurs

Un supercalculateur est un système informatique extrêmement puissant, capable d'exécuter jusqu'à des milliards de milliards d'opérations par seconde, souvent utilisé pour résoudre des problèmes qui nécessitent d'énormes ressources de calcul.

Caractéristiques principales :

- Capacité de calcul très élevée.
  - Mesurée en pétaflops (10^{15} opérations en virgule flottante par seconde) voire exaflops (10^{18}). NB : FLOP => **FL**oating point **OP**eration, c'est-à-dire une opération en virgule flottante. Cela désigne une opération mathématique impliquant des nombres réels (décimaux), comme une addition, une multiplication, une division, etc.
  - Composés de dizaines de milliers à millions de cœurs de processeur.
- Architecture massivement parallèle.
  - De nombreux processeurs travaillent simultanément sur différentes parties d'un même problème.
  - Reposent sur des architectures distribuées, GPU et CPU, souvent hybrides.
- Mémoire et stockage très importants.
  - Grandes capacités de RAM (plusieurs téraoctets ou pétaoctets).
  - Systèmes de stockage distribués à haut débit.
- Refroidissement avancé.
  - Refroidissement à air forcé, liquide ou immersion, en raison de la dissipation thermique importante.
- Coût et consommation énergétique élevés.
  - Coût de millions d'euros à l'achat et à l'exploitation.
  - Consommation électrique pouvant atteindre plusieurs mégawatts.
- Fonctionnement en environnement contrôlé.
  - Installés dans des centres de calcul dédiés, avec sécurité, ventilation, alimentation de secours, etc.

Utilisations typiques :

Domaine	Utilisation concrète
Météorologie	Modélisation du climat, prévisions météorologiques précises
Recherche scientifique	Simulation physique, chimique ou biologique Exemple : repliement des protéines Lorsqu'une protéine est synthétisée dans une cellule (dans un ribosome), elle est d'abord sous la forme d'une chaîne linéaire d'acides aminés appelée structure primaire.

Domaine	Utilisation concrète
	Pour devenir fonctionnelle, cette chaîne doit se replier de manière précise en une forme tridimensionnelle complexe. C'est ce qu'on appelle le repliement. La forme d'une protéine détermine sa fonction. Si la protéine ne se replie pas correctement, elle ne fonctionne pas, ou pire : elle devient toxique.
Aéronautique	Simulation de flux d'air sur des modèles 3D, essais virtuels de prototypes
Astrophysique	Simulation de l'univers, analyse de données d'observations astronomiques
Intelligence artificielle	Entraînement de modèles de grande taille (deep learning, langage, image)
Sécurité nationale	Cryptanalyse, simulation d'armes nucléaires (sans essai réel)
Industrie pharmaceutique	Criblage virtuel de médicaments, modélisation de molécules. Le criblage virtuel de médicaments (virtual screening en anglais) est une technique informatique utilisée en découverte de médicaments pour identifier rapidement des molécules potentiellement actives sans passer par des tests en laboratoire dans un premier temps.
Énergie	Simulation de réacteurs nucléaires, recherche sur la fusion

Exemples de supercalculateurs :

1. Frontier (États-Unis)
   - Lieu : Oak Ridge National Laboratory (Tennessee, USA)
   - Puissance : +1,1 exaflops (1 100 pétaflops) — premier supercalculateur exascale opérationnel (exascale signifie que le supercalculateur est capable de réaliser au moins 10^{18} opérations en virgule flottante par seconde)
   - Architecture : AMD EPYC + GPU AMD Instinct
   - Utilisations : recherche sur le climat, fusion nucléaire, IA, physique des matériaux
   - Classement : Classé n°1 mondial dans le classement TOP500 (2024)

2. Fugaku (Japon)
   - Lieu : RIKEN Center for Computational Science, Kobe
   - Puissance : ~442 pétaflops en performance LINPACK (jusqu'à 1 exaflop en théorie)
   - Architecture : Processeurs ARM (Fujitsu A64FX) — une première à ce niveau
   - Utilisations : simulations sismiques, pharmacologie, IA
   - Classement : a été n°1 mondial de 2020 à 2022

3. LUMI (Finlande)
   - Lieu : CSC – IT Center for Science, Kajaani (Finlande)
   - Particularité : Supercalculateur européen du programme EuroHPC
   - Énergie verte : alimenté à 100 % par une énergie hydroélectrique durable
   - Puissance : ~380 pétaflops (pics à 550 pétaflops)
   - Utilisations : modélisation climatique, simulations médicales, IA, innovation industrielle

4. Leonardo (Italie)
   - Lieu : CINECA, Bologne
   - Puissance : 250 pétaflops
   - Architecture : Intel + GPU NVIDIA
   - Utilisations : intelligence artificielle, climat, santé, physique des particules
   - Partenaire : Projet EuroHPC, financé par l'UE

5. Juwels Booster Module (Allemagne)
   - Lieu : Forschungszentrum Jülich
   - Puissance : +70 pétaflops
   - Utilisations : climat, physique, neurosciences
   - Modularité : Conçu pour s'intégrer dans une architecture de calcul évolutive (précurseur de l'exascale européen)

3.7.2.2 Ordinateurs centraux (Mainframes)

Définition : Machines de très grande puissance, conçues pour traiter d'énormes volumes de données et gérer simultanément des milliers d'utilisateurs ou de transactions.

Caractéristiques principales :

- Traitement massif des données
  - Capacité à gérer plusieurs millions de transactions par seconde.
  - Architecture orientée multiprogrammation et multitraitement.
- Haute disponibilité
  - Fonctionnement continu (24/7/365) avec redondance intégrée (alimentation, disque, processeur…).
  - Taux de disponibilité souvent supérieur à 99,999 % (« five nines »).
- Sécurité renforcée
  - Systèmes hautement sécurisés contre les intrusions, avec contrôles d'accès, cryptage, audit et conformité réglementaire (ex. RGPD, normes bancaires…).
- Capacité multi-utilisateurs
  - Peut accueillir des milliers d'utilisateurs connectés en simultané, chacun avec son propre environnement.
- Virtualisation avancée
  - Un seul mainframe peut faire fonctionner des centaines de systèmes d'exploitation ou serveurs virtuels.
- Systèmes d'exploitation spécialisés
  - Utilise des OS spécifiques comme IBM z/OS, z/VM, ou Linux on Z.
- Longévité exceptionnelle
  - Durée de vie souvent bien supérieure à celle des PC ou serveurs classiques.
- Très grande évolutivité
  - Capacité à ajouter des ressources (CPU, mémoire, disques…) sans interruption du service.

Utilisation typique des mainframes aujourd'hui :

Domaine	Utilisation concrète	Exemple
Secteur bancaire	Traitement des virements, gestion des comptes, autorisations de cartes	BNP Paribas, Belfius…
Assurances	Gestion des contrats, sinistres, calculs actuariels	AXA, AG Insurance
Fonction publique	Calcul des pensions, impôts, données de la sécurité sociale	SPF Finances, ONSS
Transport aérien	Réservations en temps réel, gestion de la billetterie	Systèmes de type SABRE, Amadeus
Grande distribution	Gestion de l'approvisionnement, facturation massive	Carrefour, Colruyt, Delhaize…
Télécommunications	Facturation, gestion des abonnements, routage des appels	Proximus, Orange
Recherche scientifique	Analyse massive de données (ex. génomique (étude de l'ensemble du matériel génétique d'un organisme), statistiques)	Organismes publics et universitaires

Exemples : IBM z16, IBM z17, Fujitsu GS21 Series, BullSequana M (M7000/M9600)
Un peu de lecture : https://www.channelnews.fr/z17-ibm-annonce-un-mainframe-pense-pour-lia-144522

3.7.2.3 Serveurs

Un serveur est un ordinateur ou un système informatique qui fournit des services ou des ressources à d'autres ordinateurs appelés clients, via un réseau. Il peut être physique (machine dédiée) ou virtuel (machine virtuelle sur une infrastructure partagée).

Caractéristiques principales :

- Fonction de service
  - Le serveur est conçu pour répondre à des requêtes provenant de clients (utilisateurs, applications ou appareils).
  - Exemples de services : fichiers, pages web, bases de données, impression, authentification, etc.
- Disponibilité élevée
  - Fonctionne 24h/24 – 7j/7.
  - Intègre des dispositifs de redondance, de sauvegarde automatique et de supervision.
- Matériel renforcé
  - Matériel souvent plus puissant et plus fiable qu'un poste client : processeurs multi-cœurs, mémoire ECC, disques en RAID, alimentation redondante, etc.
    NB: La mémoire ECC (Error-Correcting Code memory) est un type de mémoire vive (RAM) capable de détecter et de corriger automatiquement certaines erreurs de données qui peuvent survenir lors du stockage ou du transfert d'informations.
- Capacité de gestion multi-utilisateurs
  - Peut gérer simultanément des dizaines, centaines voire milliers de connexions client.
- Systèmes d'exploitation spécialisés
  - Systèmes optimisés pour les services réseau : Windows Server, Linux (Debian, CentOS, Ubuntu Server, etc.).
- Sécurité renforcée
  - Configuration axée sur la sécurisation des accès, la gestion des droits, les protocoles cryptés, les pare-feux et la détection des intrusions.
- Peut être virtualisé
  - Plusieurs serveurs virtuels peuvent coexister sur une même machine physique (virtualisation via VMware, Hyper-V, Proxmox...).

Utilisations typiques :

Type de serveur	Description et usage courant	Exemple concret
Serveur de fichiers	Stockage et partage centralisé de documents	NAS d'entreprise, Samba
Serveur web	Hébergement de sites et d'applications web	Apache, Nginx, Microsoft IIS
Serveur de messagerie	Gestion de la messagerie électronique	Microsoft Exchange, Postfix
Serveur de base de données	Hébergement et traitement de bases de données	MySQL, PostgreSQL, Oracle
Serveur d'impression	Partage d'imprimantes en réseau	Serveur CUPS (Linux), Windows Server
Serveur DNS	Résolution des noms de domaine en adresses IP	BIND, Windows DNS
Serveur DHCP	Attribution automatique d'adresses IP	ISC DHCP, Windows Server DHCP
Serveur d'authentification	Gestion des identifiants et des droits d'accès	Active Directory, LDAP

Type de serveur	Description et usage courant	Exemple concret
Serveur de virtualisation	Héberge des machines virtuelles	VMware ESXi, Proxmox, Hyper-V
Serveur de sauvegarde	Centralisation des copies de sécurité	Veeam, Bacula, UrBackup

3.7.2.4 Stations de travail

Définition : Une station de travail est un ordinateur individuel haut de gamme, conçu pour des tâches professionnelles exigeantes en puissance de calcul, de mémoire et de graphisme. Elle est plus performante qu'un PC standard, mais moins puissante qu'un serveur.

Caractéristiques principales :

- Matériel haute performance
  - Processeurs multicœurs (Intel Xeon, AMD Ryzen Threadripper…).
  - Quantité de RAM importante, souvent avec mémoire ECC (Error-Correcting Code).
  - Cartes graphiques professionnelles (NVIDIA Quadro, AMD Radeon Pro) pour le calcul ou le rendu 3D.
- Fiabilité accrue
  - Conçue pour fonctionner de manière continue avec un haut niveau de stabilité.
  - Meilleure tolérance aux pannes que les PC classiques.
- Stockage rapide et capacitaire
  - Disques SSD NVMe pour les performances.
  - RAID possible pour la redondance.
- Systèmes d'exploitation optimisés
  - Windows 10/11 Pro ou Enterprise, Linux, voire UNIX.
  - Certifiés pour des logiciels professionnels spécifiques (ex. : AutoCAD, SolidWorks, Adobe CC).
- Support matériel et logiciel professionnel
  - Stations souvent certifiées par les éditeurs de logiciels (ISV : Independent Software Vendors).
  - Maintenance facilitée et garanties étendues.
- Connectivité étendue
  - Ports multiples (USB, Thunderbolt, DisplayPort…).
  - Connexion à des réseaux professionnels, voire à des baies de stockage ou clusters de calcul.

Utilisations typiques :

Domaine	Utilisations spécifiques	Logiciels ou outils typiques
Conception assistée par ordinateur (CAO/DAO)	Modélisation 3D, dessins techniques	AutoCAD, CATIA, SolidWorks
Infographie et animation	Rendu 3D, animation, effets spéciaux	Blender, Maya, Adobe After Effects
Architecture	Modélisation de bâtiments, maquettes numériques	Revit, ArchiCAD
Sciences et ingénierie	Simulation numérique, calcul scientifique	MATLAB, ANSYS, COMSOL
Post-production audiovisuelle	Montage vidéo, colorimétrie, effets spéciaux	DaVinci Resolve, Adobe Premiere Pro
Développement logiciel	Compilation, virtualisation, émulation	Visual Studio, Docker, VirtualBox
Analyse de données / IA	Traitement de grands volumes de données	Python, TensorFlow, Jupyter, R

Exemples : Workstations HP Z, Dell Precision, Lenovo ThinkStation.

3.7.2.5 Micro-ordinateurs

Un micro-ordinateur est un ordinateur de petite taille, destiné à un usage individuel, équipé d'un micro-processeur unique. Il regroupe les composants essentiels pour assurer les tâches générales de traitement de l'information.

Le terme « micro-ordinateur » inclut les ordinateurs de bureau (desktop), les ordinateurs portables (laptops) et parfois les netbooks ou mini-PC.

Caractéristiques principales :

- Processeur unique
  - Utilise un microprocesseur (CPU) intégré, généralement de type Intel Core, AMD Ryzen, Apple M1/M2…
  - Capacité suffisante pour les tâches quotidiennes (bureautique, internet, multimédia).
- Mémoire modérée
  - RAM généralement entre 4 et 32 Go.
  - Mémoire non-ECC (contrairement aux stations de travail ou serveurs).
- Stockage interne
  - Disque dur (HDD) ou SSD selon la génération.
  - Stockage secondaire possible via périphériques externes ou cloud.
- Système d'exploitation courant
  - Windows (majoritaire), macOS (Apple), Linux (Ubuntu, Mint…), Chrome OS.
- Accessibilité et polyvalence
  - Utilisation conviviale avec interfaces graphiques (GUI).
  - Adapté à un large public : particuliers, professionnels, étudiants.
- Périphériques standards
  - Clavier, souris, écran, webcam, imprimante, etc.
  - Connectivité : USB, Wi-Fi, Bluetooth, HDMI, etc.
- Coût relativement abordable
  - Prix variable selon les performances, généralement entre 400 € et 1500 € mais le prix peut monter à plusieurs milliers d'euros.

Utilisations typiques :

Domaine	Utilisation concrète	Exemples de logiciels
Bureautique	Traitement de texte, tableur, présentations	Microsoft Office, LibreOffice
Navigation Internet	Recherche, consultation de sites, services en ligne	Navigateurs web, messagerie
Communication	Visio, e-mail, messagerie	Zoom, Teams, Outlook, WhatsApp
Multimédia	Lecture de vidéos, musique, gestion de photos	VLC, Spotify, Photos
Apprentissage / Études	Environnement d'apprentissage numérique	Moodle, Teams, Google Classroom
Jeux vidéo	Jeux occasionnels à intensifs selon configuration	Steam, Epic Games
Programmation	Développement d'applications ou de sites	VS Code, Python, XAMPP
Petite gestion	Comptabilité, gestion de stock, facturation légère	Excel, Odoo (version locale), EBP

Exemples : PC, Mac, ordinateurs portables, tablettes, smartphones
Et oui, les tablettes et les smartphones peuvent être classés dans la famille des micro-ordinateurs, selon la définition technique du terme.

Un micro-ordinateur est un ordinateur :

- destiné à un usage individuel,
- basé sur un microprocesseur (CPU intégré sur une seule puce),
- intégrant des composants de base : processeur, mémoire, entrées/sorties, stockage.

À ce titre, tout appareil informatique personnel basé sur un microprocesseur peut être considéré comme un micro-ordinateur.

3.7.2.6 Systèmes embarqués

Définition : systèmes informatiques autonomes, intégrés dans un appareil pour remplir une tâche spécifique, souvent en temps réel.

Caractéristiques principales :

- Dédiés à une fonction précise (contrôle, mesure, automatisation).
- Faible encombrement, consommation d'énergie optimisée.
- Fiabilité, réactivité et autonomie.
- Peuvent fonctionner sans interface utilisateur classique (clavier, écran).
- Utilisation dans l'automobile, l'aéronautique, l'électroménager, la santé, les objets connectés.

Exemples : Systèmes de navigation GPS, microcontrôleurs d'électroménager, robots, cartes Arduino, Raspberry Pi dédiées

3.7.2.7 Tableau récapitulatif

Type de système	Caractéristiques principales	Utilisations typiques
Micro-ordinateur	• Usage individuel • Processeur unique • RAM et stockage modérés • Coût abordable	Bureautique, navigation internet, multimédia, apprentissage
Station de travail	• Composants haut de gamme • Calculs intensifs • Graphisme professionnel	CAO/DAO, modélisation 3D, montage vidéo, simulations techniques
Serveur	• Services à plusieurs clients • Fonctionnement 24/7 • Sécurité et redondance	Hébergement web, messagerie, fichiers partagés, bases de données
Mainframe	• Traitement massif de transactions • Multi-utilisateurs • Haute disponibilité	Banques, sécurité sociale, gestion de grandes bases de données
Supercalculateur	• Puissance extrême • Calcul parallèle massif • Infrastructure spécialisée	Recherche scientifique, simulation climatique, IA, modélisation complexe

3.7.3 Selon le mode d'utilisation

Voici une autre manière de catégoriser les systèmes informatiques.

Type	Caractéristiques	Exemple
Système personnel	Utilisé par un seul utilisateur à la fois.	PC de bureau
Système centralisé	Un seul ordinateur principal traite toutes les données.	Mainframe avec terminaux

Type	Caractéristiques	Exemple
	Les terminaux sont « passifs » : pas d'exécution de programmes…	
Système client-serveur	Les clients demandent des services à un serveur central. Les terminaux sont actifs, exécutent des programmes…	Applications web
Système embarqué	Intégré dans un appareil pour une fonction spécifique.	GPS, lave-linge, voiture
Système distribué	Plusieurs ordinateurs coopèrent sur un réseau pour une tâche commune.	Cloud computing, clusters

3.7.4 Points clés à retenir

Chaque catégorie de système informatique répond à des besoins spécifiques en termes de puissance, de capacité, de coût et d'usage.

L'évolution technologique tend à réduire la taille et le coût tout en augmentant la puissance, rendant les frontières entre catégories parfois floues.

Le choix d'un système dépend du contexte d'utilisation : volume de données, nombre d'utilisateurs, contraintes de temps réel, portabilité, etc.

3.7.5 Conclusion

La diversité des systèmes informatiques permet de couvrir l'ensemble des besoins, du traitement massif de données à l'automatisation de tâches simples dans des objets du quotidien. Comprendre leurs caractéristiques est essentiel pour choisir la solution la mieux adaptée à chaque situation professionnelle ou personnelle.

3.8 Compatibilité des systèmes informatiques

3.8.1 Définition

Compatibilité : Capacité de deux systèmes à fonctionner ensemble, à échanger des données ou à exécuter les mêmes programmes sans problèmes.

3.8.2 Systèmes compatibles

Les systèmes compatibles partagent des normes, des formats ou des interfaces qui leur permettent de communiquer ou de fonctionner ensemble.

- Compatibilité logicielle : Un logiciel conçu pour Windows pourra fonctionner sur toutes les machines Windows, mais pas forcément sur macOS ou Linux sans adaptation.
- Compatibilité matérielle : Certains composants (ex : cartes graphiques, disques durs) sont compatibles avec plusieurs systèmes, d'autres non.
- Compatibilité des systèmes de fichiers : Par exemple, FAT32 est lisible et modifiable par Windows, macOS et Linux, alors que NTFS est en lecture/écriture sur Windows, mais en lecture seule sur macOS sans logiciel tiers. exFAT est compatible Windows et supporté nativement dans les dernières versions de macOS et Linux.

3.8.3 Systèmes non compatibles

Les systèmes non compatibles ne peuvent pas échanger de données, exécuter les mêmes applications ou fonctionner ensemble sans adaptation :

- Formats propriétaires : Certains logiciels ou matériels utilisent des formats fermés, empêchant l'échange de données ou la compatibilité avec d'autres systèmes.

- Systèmes d'exploitation différents : Un programme Windows ne fonctionne pas nativement sous macOS ou Linux (et inversement), sauf via des émulateurs ou des couches de compatibilité (ex : Wine pour Linux).
- Problèmes de versions : Deux versions d'un même système peuvent devenir incompatibles si les évolutions ne respectent pas la rétrocompatibilité.

Exemples de systèmes non compatibles

Système	Incompatibilité	Conséquence
macOS & certains logiciels	Logiciels Windows	Nécessité d'un émulateur ou d'une VM
Systèmes de fichiers ext4 (Linux)	Non lus par Windows	Impossible de lire un disque Linux sous Windows sans outil spécialisé
Matériel Apple & OS tiers	Verrouillage matériel	Impossible d'installer d'autres OS sans modification

3.8.4 Facteurs favorisant la compatibilité

- Normes ouvertes : HTML, XML, SQL, etc., facilitent l'échange de données et l'intégration.
- API et protocoles standardisés : Permettent à des systèmes différents de communiquer facilement.
- Adoption de standards industriels : Favorise la compatibilité entre matériels et logiciels de différents fournisseurs. Exemples : connecteurs USB, normes wi-fi (IEEE 802.11), formats de fichiers PDF, JPG, MP3...

3.8.5 Points clés à retenir

- La compatibilité permet l'intégration, la flexibilité et la pérennité des systèmes informatiques.
- L'incompatibilité peut engendrer des silos de données, des surcoûts et des difficultés d'évolution. La notion de silos de données (data silos en anglais) désigne une situation dans laquelle les données d'une organisation sont fragmentées, stockées séparément dans différents services, équipes, outils ou systèmes, sans communication ni partage efficace entre eux.
- Le choix de systèmes compatibles ou interopérables est crucial pour l'efficacité et la sécurité des infrastructures informatiques.

En résumé, les systèmes informatiques compatibles facilitent la communication, l'échange de données et l'évolution technologique, tandis que les systèmes non compatibles limitent l'intégration et nécessitent souvent des solutions de contournement ou des adaptations spécifiques.

4 La numération

Bien que ce sujet soit principalement l'objet du cours de mathématiques appliquées à l'informatique, voici le minimum de ce qu'il faut savoir à propos de la numération décimale, binaire et hexadécimale.

4.1 Bit, byte, octet

L'informatique repose sur un système binaire, c'est-à-dire un système à deux états correspondant à la présence ou l'absence d'un signal (électrique, magnétique, optique, etc.). Ces deux états sont représentés par les valeurs 0 et 1.

Le bit
Le bit (contraction de binary digit) est la plus petite unité d'information manipulable par une machine numérique. Il ne peut prendre que deux valeurs : 0 ou 1. Il peut être représenté :

- par un **signal électrique** (présence ou absence de tension),
- par une **polarisation magnétique** (orientation d'un champ magnétique),
- par des **marques optiques** (zones réfléchissantes ou non sur un disque optique),
- ou par des **composants électroniques bistables**, capables de maintenir l'un de deux états stables.

Avec :

- 1 bit : 2 valeurs possibles (0 ou 1),
- 2 bits : 4 combinaisons possibles (00, 01, 10, 11),
- 3 bits : 8 combinaisons possibles, etc.

En général, n bits permettent de représenter 2^n valeurs différentes.

L'octet
Un octet (ou byte en anglais) regroupe 8 bits. C'est l'unité de base de stockage d'un caractère ou d'un chiffre. Un octet permet de représenter 256 valeurs différentes (de 0 à 255).

Le mot
Un mot (word) est une unité de données plus large. La taille d'un mot dépend de l'architecture du processeur. C'est la taille que le processeur traite en une seule opération. Pour une architecture 64 bits, un mot correspond à 8 octets (64 bits).

Les multiples de l'octet
Les capacités de stockage et de mémoire sont exprimées en puissances de 2 :

Unité (norme binaire IEC)	Valeur exacte en octets
1 Kio (kibioctet)	1 024 octets (2^{10})
1 Mio (mébioctet)	1 048 576 octets (2^{20})
1 Gio (gibioctet)	1 073 741 824 octets (2^{30})
1 Tio (tébioctet)	1 099 511 627 776 octets (2^{40})

NB : Il ne faut pas confondre ces unités binaires avec les unités décimales souvent utilisées dans le domaine commercial (1 Ko = 1 000 octets, 1 Mo = 1 000 000 octets, etc.). Cette différence peut expliquer pourquoi un disque dur de 500 Go affiche une capacité inférieure dans le système d'exploitation. En effet, dans les propriétés affichées, bien que le système indique Go, en fait il calcule des Gio.

Le codage des informations
Les ordinateurs ne manipulent que des nombres. Il est donc nécessaire de coder toutes les informations (textes, sons, images, etc.) en données numériques.

Exemple : le code ASCII
Le code ASCII (American Standard Code for Information Interchange) permet de représenter les caractères alphanumériques sous forme binaire :

- ASCII de base : sur 7 bits, soit 128 caractères (valeurs de 0 à 127), sans caractères accentués.
- ASCII étendu : sur 8 bits (1 octet), pour un total de 256 symboles, incluant ponctuations, symboles graphiques et caractères accentués selon les variantes (ISO Latin-1, Windows-1252, etc.).

Vers une représentation universelle : Unicode

Les variantes d'ASCII ne couvrant que peu de langues, un système plus universel a été créé :
Unicode permet de représenter tous les caractères des langues du monde, y compris les symboles techniques, mathématiques, et emojis.

Il attribue un numéro unique à chaque caractère (appelé point de code).

Unicode est compatible avec ASCII pour les 128 premiers caractères.

Il peut être encodé sous différentes formes, dont UTF-8, UTF-16, etc.

Aujourd'hui, UTF-8 est l'encodage le plus répandu sur le Web. Il permet de stocker les caractères Unicode tout en restant efficace pour les textes contenant majoritairement des caractères ASCII.

NB : Les caractères accentués (é è à…) peuvent être codés en ISO-8859-1 ou en Windows-1252. Dans ce cas, cela ne prend que 1 octet.

Détail du codage en UTF-8 :

Octets utilisés	Intervalle Unicode	Type de caractères
1 octet (8 bits)	U+0000 à U+007F	ASCII standard (lettres, chiffres, ponctuation)
2 octets	U+0080 à U+07FF	Caractères accentués, grec, hébreu, arabe, etc.
3 octets	U+0800 à U+FFFF	Langues asiatiques, symboles divers
4 octets	U+10000 à U+10FFFF	Emojis, anciens alphabets, caractères rares

Exemples :
'A' → Unicode U+0041 → codé sur 1 octet : 01000001
'é' → Unicode U+00E9 → codé sur 2 octets : 11000011 10101001
'€' → Unicode U+20AC → codé sur 3 octets : 11100010 10000010 10101100
'😊' → Unicode U+1F60A → codé sur 4 octets : 11110000 10011111 10011000 10001010

À retenir :

- ASCII ⟶ codé sur 1 octet en UTF-8.
- UTF-8 est rétrocompatible avec ASCII.
- Avantage : UTF-8 économise de la mémoire pour les textes contenant surtout des caractères latins.
- Inconvénient : certains caractères peuvent occuper jusqu'à 4 octets.

4.2 Numérations de position - Illustration avec la numération décimale

Considérons le nombre décimal 1975_{10}. Ce nombre est un « mot » dont les caractères sont les chiffres.
La valeur que l'on attribue à chaque chiffre dépend du chiffre en lui-même et de sa position.

- le 1 vaut 1 x 1000.
- le 9 qui suit représente des centaines, il vaut 9 x 100.
- le 7 représente des dizaines, il vaut 7 x 10.
- le 5 vaut 5 x 1.

Nous utilisons un système de numération de **position** et la **base 10**.

La base 10 ou base décimale est celle que nous utilisons tous les jours.

Nous disposons de 10 chiffres (de 0 à 9) pour compter.

Après avoir compté 10 unités, nous utilisons deux chiffres pour écrire le nombre 10 :

- le 1 signifie que nous avons fait un paquet de dix unités que nous appelons dizaine ;
- le 0 qui suit signifie que nous comptons les nouvelles unités dans un paquet qui pour l'instant est vide.

Le nombre de dizaines ne peut pas dépasser 9. Il nous faut donc un troisième chiffre pour compter dizaines par dizaines, ce « report » indique les centaines.
De droite à gauche on a donc les unités puis les dizaines, les centaines, les milliers et toutes les puissances successives de 10.

NB : Comme cela a été vu précédemment, il existe d'autres bases. Les Babyloniens utilisaient par exemple la base 60. C'est ce qui a marqué notre façon de compter les minutes et les secondes, celles pour mesurer le temps qui passe ou pour mesurer les angles.

4.3 Le poids des chiffres - Illustration avec la numération décimale

Reprenons notre exemple 1975_{10}. Numérotons les positions des chiffres de droite à gauche en commençant par 0 pour la position des unités.

Position →

3	2	1	0
1	9	7	5

Le **poids** d'un chiffre dépend de sa **position** et de la **base** => **Poids = base** $^{\textbf{position}}$

NB : Ne pas confondre le poids et la valeur.

Voici ce que cela donne dans notre exemple :

Le **poids** du chiffre 1 est 10^3 sa **valeur** est $1 \times 10^3 = 1 \times 1000 = 1000$
Le **poids** du chiffre 9 est 10^2 sa **valeur** est $9 \times 10^2 = 9 \times 100 = 900$
Le **poids** du chiffre 7 est 10^1 sa **valeur** est $7 \times 10 = 70$
Le **poids** du chiffre 5 est 10^0 sa **valeur** est 5×1 (car $10^0 = 1$) $= 5$

Positions	3	2	1	0
Chiffres décimaux	1	9	7	5
Valeurs de chaque chiffre	1×10^3 **1000**	9×10^2 **900**	7×10^1 **70**	5×10^0 **5**

4.4 Numération binaire

La base est 2. Nous n'utilisons que deux chiffres 0 et 1.

Il faut remarquer qu'en base 2, le chiffre 2 n'existe pas; tout comme le chiffre 10 n'existe pas en base 10.

Il s'agit toujours d'une numération de position. De droite à gauche nous avons donc les unités puis les « deuzaines », les « quatraines », les « huitaines », les « seizaines », les « trentedeuzaines » etc.

4.4.1 Conversion de la base binaire vers la base décimale

Exemple : que vaut le nombre binaire 10110_2 en base 10 ?

Le poids d'un chiffre dépend de sa position et de la base.

Poids = base ^{position} ici en binaire le poids = 2 ^{position}

Positions	4	3	2	1	0
Chiffres binaires	1	0	1	1	0
Valeurs de chaque chiffre	1 x 2^4 **16**	0 x 2^3 **0**	1 x 2^2 **4**	1 x 2^1 **2**	0 x 2^0 **0**

On a donc ici une « seizaine », une « quatraine » et une « deuzaine » soit 16 + 4 + 2 = 22

4.4.2 Conversion de la base décimale vers la base binaire

Étapes de la méthode classique :

1) Diviser le nombre décimal par 2.
2) Noter le reste : il sera 0 ou 1 (ceci représente un chiffre binaire). Ceci correspond au BMS (bit le moins significatif).
3) Diviser le quotient obtenu à l'étape précédente par 2, noter à nouveau le reste. On obtient un bit de poids supérieur au précédent.
4) Répéter jusqu'à obtenir un quotient de 0. On obtient à chaque fois un bit de poids supérieur au précédent.
5) Le reste de la dernière division correspond au BPS (bit le plus significatif).

Le binaire est obtenu en lisant les restes **à l'envers** (du dernier au premier).

Exemple de conversion de 13_{10} en binaire

Étape	Quotient de la division par 2	Reste
1	13 ÷ 2 → 6	1
2	6 ÷ 2 → 3	0
3	3 ÷ 2 → 1	1
4	1 ÷ 2 → 0	1

En lisant les restes de bas en haut : 1101_2 (c'est donc 13_{10} en binaire).

4.5 Numération hexadécimale

La base est 16. Il nous faut donc 16 chiffres, nous avons déjà les chiffres 0 à 9, ajoutons-y les caractères A, B, C, D, E et F pour représenter les « chiffres » de 10 à 15.

Il faut remarquer qu'en base 16, le chiffre 16 n'existe pas; tout comme le chiffre 10 n'existe pas en décimal ni le chiffre 2 en binaire.

Nous appliquons toujours les mêmes principes de la numération de position.

Le poids d'un chiffre dépend de sa position et de la base.

Poids = base ^{position} ici en hexadécimal le poids = 16 ^{position}

De droite à gauche nous avons donc les unités puis les « seizaines », les « 256zaines » etc.

À quoi sert l'hexadécimal ?

Les codes hexadécimaux sont bien pratiques en informatique. Ils permettent de lire les codes binaires de manière compacte et nous évitent de devoir lire de longues enfilades de 0 et de 1 qui conviennent mieux aux ordinateurs qu'aux humains.

Un groupe de quatre bits permet de former 16 combinaisons différentes. Une valeur hexadécimale correspond à chaque combinaison. L'hexadécimal c'est du binaire condensé.

Le code hexadécimal 1A2F est bien plus lisible que 0001 1010 0010 1111 en binaire.

Il suffit de savoir compter jusqu'à 16 en binaire et en hexadécimal pour savoir passer facilement du binaire à l'hexadécimal ou inversement.

Décimal	Binaire	Hexa
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

4.5.1 Conversion de la base hexadécimale vers la base décimale

Exemple : que vaut le nombre hexadécimal $1A2F_{16}$ en base 10 ?

Positions	3	2	1	0
Chiffres hexadécimaux	1	A	2	F
Valeurs de chaque chiffre	1×16^3 **4096**	10×16^2 **2560**	2×16^1 **32**	15×16^0 **15**

L'addition des valeurs de ces 4 chiffres donne : 4096 + 2560 + 32 + 15 = 6703

4.5.2 Conversion de la base décimale vers la base hexadécimale

Étapes de la conversion de la base décimale vers la base hexadécimale :

1) Diviser le nombre décimal par 16.
2) Noter le reste : ce sera un chiffre hexadécimal (de 0 à 9 ou de A à F, où A=10, B=11… F=15).
3) Prendre le quotient de la division et recommencer à l'étape 1 avec ce nouveau nombre.
4) Répéter jusqu'à ce que le quotient soit zéro.
5) Lire les restes du dernier au premier : cela donne le nombre en hexadécimal.

Exemple : convertir 300_{10} en hexadécimal

300 ÷ 16 = 18, reste 12 (soit C en hexadécimal)
18 ÷ 16 = 1, reste 2
1 ÷ 16 = 0, reste 1
On lit les restes en remontant : 1, 2, C

Résultat : $12C_{16}$

Rappel concernant la notation :

On écrit généralement les nombres hexadécimaux avec un indice, par exemple : $12C_{16}$ ou 0x12C (notation utilisée en informatique).

La notation 0x... provient du langage de programmation C, où elle a été introduite comme convention pour indiquer très clairement qu'un nombre littéral est exprimé en hexadécimal (base 16). Ainsi, lorsque l'on écrit 0x7B, cela signifie spécifiquement « 7B en base 16 », ce qui vaut 123 en décimal.

Le choix du préfixe 0x répond à un besoin : sans indication de base, il serait impossible de distinguer rapidement un nombre décimal, hexadécimal ou encore octal lors de la lecture du code source. Le « 0 » en début permet d'indiquer qu'il s'agit d'un nombre et la lettre x (« hex ») signale l'hexadécimal ; d'autres bases ont leurs propres conventions, par exemple 0b pour le binaire ou 0o pour l'octal selon les langages.

De nombreux langages informatiques ont repris cette notation du C (C++, Java, Python, etc.) du fait de sa clarté et de son succès. On trouve également d'autres conventions dans certains langages ou contextes : $FF, **F, 16#FF#, FFh ou #..... mais 0x est la plus répandue dans les environnements liés au C et à l'informatique moderne.

Langage / contexte	Binaire	Octal	Décimal	Hexadécimal
Mathématique (académique)	1011_2	77_8	42_{10} ou 42	$1A_{16}$
Python / C++ / Java	0b1011	0o77	42	0x1A
Assembleur x86	1011b	77o	42	1Ah

4.6 Opérations en binaire

4.6.1 L'addition binaire

Règles de base :

0 + 0 = 0
0 + 1 = 1
1 + 1 = 0 => report de 1
1 +1 + 1 = 1 => report de 1

Exemples

```
    1011
+   0110
    ----
   10001
```

```
   10101
+   1101
    ----
  100010
```

4.6.2 La soustraction binaire par la méthode du complément à 2

Un microprocesseur ne sait pas soustraire directement deux nombres binaires.
Pour effectuer une soustraction $A - B$, il utilise la méthode du complément à deux (C_2) :

$$A - B = A + C_2(B)$$

Autrement dit, il suffit d'additionner à A le complément à deux de B pour obtenir le résultat de la soustraction.

Pour obtenir le complément à 2 (C_2) d'un nombre binaire :

1) rechercher son complément à 1 (C_1) en inversant tous les bits.
2) additionner 1 au C_1 trouvé dans le point précédent

- **Exemple du calcul du complément à 2 avec le nombre binaire 00101101**

Étape 1
Complément à 1 : 11010010

Étape 2
Complément à 2 : 11010010 + 1 = 11010011

- **Exemple de soustraction de deux nombres binaires**

Soit A = 00010111
Soit B = 00001111

Objectif : calculer A - B

Étape 1 : Calcul du complément à 2 de B
C_2(B) = 11110001

Étape 2 : additionner A et C_2(B)

```
   00010111
+  11110001
   --------
  100001000
```

Étape 3 : gérer l'overflow

On est en 8 bits, on ne garde que les 8 bits de droite. On obtient donc : 00001000

Note
En interne, l'ALU (Arithmetic Logic Unit • unité arithmétique et logique) n'effectue pas une soustraction directe : elle transforme toujours la soustraction en une addition avec le complément à deux. Cependant, pour le programmeur, la plupart des processeurs modernes proposent une instruction de soustraction (SUB), qui masque ce détail. L'ALU transforme en réalité cette opération en addition avec le complément à deux et c'est totalement transparent pour l'utilisateur.

4.6.3 La multiplication binaire

Historiquement, la multiplication binaire était réalisée par additions répétées. Aujourd'hui, les microprocesseurs modernes intègrent des unités matérielles spécialisées qui permettent d'effectuer des multiplications beaucoup plus rapidement.

Elles utilisent des algorithmes optimisés (Booth, Wallace tree, etc.) et des circuits parallèles.
Cela permet d'obtenir des multiplications très rapides, ce qui est indispensable en traitement d'images, en cryptographie ou en IA.

La FPU (Floating Point Unit) est un circuit spécialisé du processeur, conçu pour effectuer rapidement des calculs sur les nombres réels en virgule flottante.

4.6.4 La division binaire

Historiquement, la division binaire était réalisée par soustractions successives (comme pour la multiplication avec additions répétées). Aujourd'hui, comme pour la multiplication, il existe des circuits spécialisés (circuit spécialisé pour la division, Newton-Raphson, Goldschmidt, etc.) permettant des divisions rapides.

4.7 Très bien pour les nombres naturels mais comment gérer des nombres entiers négatifs ?

4.7.1 Manipuler des nombres entiers négatifs en binaire

Manipuler des nombres négatifs en binaire demande d'utiliser des conventions particulières, car dans le système binaire simple, il n'existe pas de « signe » évident comme en décimal.

4.7.1.1 Représentation des nombres négatifs en binaire

Voici les principales méthodes pour représenter les nombres négatifs en binaire :

1. Bit de signe (représentation signe-magnitude)

- Principe : On réserve le bit le plus à gauche (bit de poids fort) pour indiquer le signe :
  - 0 = positif
  - 1 = négatif
- Exemple sur 8 bits :
  - +5 : 00000101
  - -5 : 10000101
- Limites : Cette méthode complique les opérations arithmétiques.

2. Complément à 2 (méthode la plus courante)

- Principe :
  1) Écrire la valeur absolue en binaire.
  2) Inverser chaque bit (complément à 1).
  3) Ajouter 1 au résultat.
- Exemple pour -5 sur 8 bits :
  1) +5 = 00000101
  2) inversion des bits : 11111010
  3) ajouter 1 : 11111011 (c'est la représentation de -5 en complément à 2)
- Avantages : facile pour le matériel, permet des additions et soustractions sans traitement particulier du signe.

3. Complément à 1 (plus rare)

- Principe : Idem que le complément à 2, mais on ne rajoute pas 1 à la fin.
- Limites : Deux représentations du zéro.

4. Décimal signé (moins utilisé en informatique)

Certains systèmes peuvent conserver une colonne de signe ou des formats particuliers, mais ce n'est pas la base du travail binaire classique.

4.7.1.2 Extension du signe

Soit le nombre binaire (complément à 2) suivant : 11111011

Le bit de poids fort est 1 → c'est un nombre négatif en complément à deux.

Extension du signe sur 32 bits :

Pour passer de 8 bits à 32 bits en complément à deux, on étend le signe (bit de poids fort) :

Comme le bit de poids fort est 1, on étend avec des 1 à gauche.

Le nombre 8 bits 11111011 devient sur 32 bits : 11111111 11111111 11111111 11111011

4.7.1.3 Convertir un nombre binaire négatif (complément à 2) en décimal

Première technique :

Pour convertir un nombre binaire négatif (complément à 2) en décimal :

- Si le bit de poids fort (bit le plus à gauche) est 0, c'est un nombre positif → conversion normale.
- Si le bit de gauche est 1, c'est négatif.

Pour retrouver la valeur :

1) Inverser tous les bits
2) Ajouter 1
3) Convertir le résultat en décimal, rajouter le signe négatif.

Exemple avec 11101110 (nombre binaire complément à 2) :

a. Vérifier le bit de signe :
Le bit de gauche est 1, donc le nombre est négatif.

b. Inverser les bits :
11101110 => 00010001

c. Ajouter 1 :
00010001 + 1= 00010010

d. Convertir 00010010 en décimal :
1×16 + 0×8 + 0×4 + 1×2 + 0×1 = 16 + 2 = 18

e. Remettre le signe :
Donc, c'est **-18**

Deuxième technique :

Pour convertir un nombre binaire négatif (complément à 2) en décimal :

- Si le bit de poids fort (bit le plus à gauche) est 0, c'est un nombre positif → conversion normale.
- Si le bit de gauche est 1, c'est négatif.

Pour retrouver la valeur :

1) Soustraire 1
2) Inverser tous les bits
3) Convertir le résultat en décimal, rajouter le signe négatif.

Exemple avec 11101110 (nombre binaire complément à 2) :

a. Vérifier le bit de signe :
Le bit de gauche est 1, donc le nombre est négatif.

b. Soustraire 1 :
11101110 – 1= 11101101

c. Inverser les bits :
11101101 => 00010010

d. Convertir 00010010 en décimal :
1×16 + 0×8 + 0×4 + 1×2 + 0×1 = 16 + 2 = 18

e. Remettre le signe :
Donc, c'est **-18**

Si je vois un nombre binaire, comment puis-je savoir qu'il est codé en complément à deux ?

Si le contexte ne précise rien, il est impossible de distinguer le complément à deux d'un autre codage juste à partir de la séquence de bits. Tout dépend du contexte : protocole, matériel… et/ou des spécifications données dans un programme informatique par exemple (type de variable…)

4.7.2 Manipuler des nombres entiers négatifs en hexadécimal

Pour coder des nombres négatifs en hexadécimal, on utilise la même méthode que pour le binaire négatif, c'est-à-dire la représentation en complément à deux, mais exprimée en base 16. En effet, l'hexadécimal n'est qu'une représentation plus lisible du binaire, donc le principe reste identique.

Voici comment cela fonctionne :

- Un nombre négatif est d'abord représenté en binaire en complément à deux sur une taille fixe (par exemple 8, 16, 32 bits).
- Ensuite, ce nombre binaire est converti en hexadécimal en regroupant les bits par groupes de 4 (un chiffre hexadécimal = 4 bits).
- Le résultat hexadécimal correspond donc au complément à deux du nombre négatif.

Par exemple, pour représenter le nombre -42 en hexadécimal sur 32 bits :

1) On calcule le complément à deux de -42 en binaire sur 32 bits.
2) La représentation binaire négative correspondante sera convertie en hexadécimal.
3) Le résultat est alors une valeur hexadécimale comme « FFFF FFD6 ».

Représentation décimale	Binaire (complément à deux 32 bits pour la valeur négative)	Hexadécimal (32 bits)
+42	00000000 00000000 00000000 00101010	0000 002A
-42	11111111 11111111 11111111 11010110	FFFF FFD6

Ce qui signifie que « FFFFFFD6 » est bien la représentation hexadécimale du nombre signé -42 en complément à deux sur 32 bits.

Important à noter :

- Ce n'est pas une notation hexadécimale avec un signe devant (comme -2A), mais une représentation hexadécimale du binaire signé en complément à deux.
- Afficher un nombre négatif en hexadécimal avec un signe (par exemple -2A) n'est pas la pratique standard en informatique car la valeur est stockée et manipulée sous forme binaire (complément à deux) et convertie directement en hexadécimal.
- Donc, un nombre négatif en hexadécimal est généralement vu comme une valeur hexadécimale dont le bit de poids fort est 1, indiquant un nombre négatif selon le complément à deux.

5 Numérique / analogique

5.1 Deux modes de représentation des grandeurs

Distinguons deux modes de représentation des grandeurs : l'analogique et le numérique.

Le codage des informations et leur transport sous forme numérique est une technique assez récente. Jusque dans les années 70, les informations étaient surtout codées et transportées sous forme analogique.

Les systèmes analogiques utilisent des grandeurs continues, comme une tension électrique, pour représenter les valeurs à traiter. On parle d'analogique car il y a « analogie » entre la donnée et sa représentation.

Les systèmes analogiques paraissent maintenant plus rudimentaires et manquent de précision (approximation, tolérance, etc.) dès que le support qui sert à son écriture ou à sa transmission est de qualité insuffisante.

Lors de chaque manipulation des signaux analogiques, des erreurs d'imprécision viennent dégrader l'information (exemples : photocopies, transmissions radio à l'ancienne etc.).

Les systèmes numériques ne manipulent que des nombres. C'est à la fois plus simple et plus compliqué !

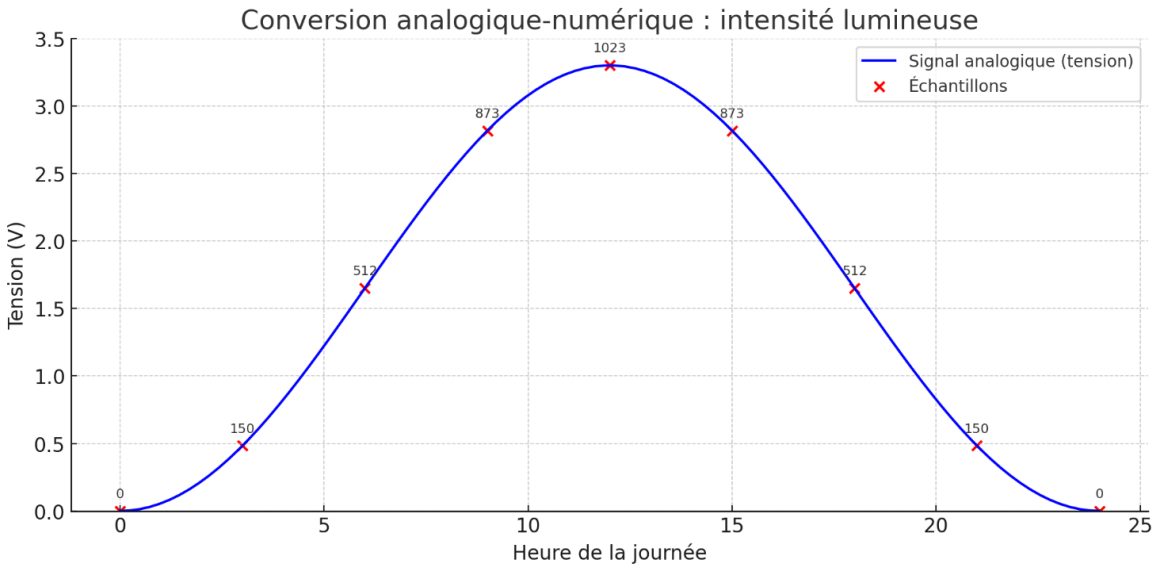
Plus simple parce que :

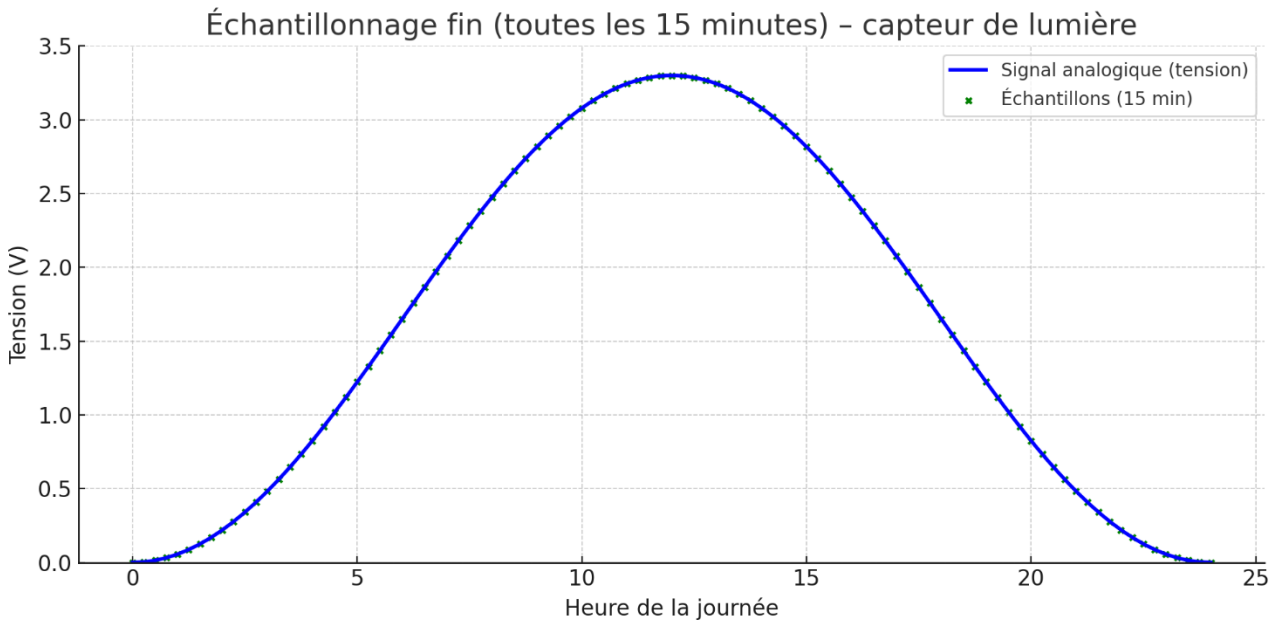
- La mémorisation de données numériques est plus facile.
- Les données peuvent être recopiées avec exactitude => l'information chiffrée reste intacte.
- On dispose de nombreuses méthodes arithmétiques et logiques pour traiter ces codes : les vérifier, les crypter et les compresser.

Plus compliqué parce que :

- pour avoir ces signaux sous forme digitale il faut d'abord les numériser ;
- et à l'inverse il faut parfois les reconvertir en analogique quand la reproduction sous cette forme est nécessaire.

Exemple : capteur d'intensité lumineuse





On part d'un phénomène continu (lumière) que l'on mesure à intervalles réguliers, arrondit, puis code en binaire. Cela permet à un système numérique de comprendre et traiter des données physiques.

5.2 Quelques notions théoriques

Le quantum, dans le contexte de la conversion analogique-numérique, est la plus petite variation de la grandeur analogique (généralement une tension) que le convertisseur est capable de détecter et de transformer en un changement d'une unité dans la valeur numérique de sortie.

Synonymes de quantum :

- pas de quantification ;
- marche de quantification ;
- **résolution** élémentaire du convertisseur.

Plus concrètement :

Le quantum correspond à la différence de tension (ou autre grandeur) entre deux codes numériques consécutifs à la sortie du CAN.

Mathématiquement, pour un convertisseur sur n bits et une tension de référence $Uref$, le quantum se calcule ainsi :

$$Quantum\ (ou\ Pas) = \frac{Uref}{2^n}$$

Exemple 1

Un convertisseur analogique-numérique (CAN) travaille sur 3 bits et permet de mesurer des tensions comprises entre 0V et 8V $(Uref = 8V)$.

Pour rappel, $Uref$ (ou Vref) est la tension de référence d'un convertisseur analogique-numérique (CAN). Il s'agit de la tension maximale que le convertisseur peut mesurer et coder en valeur numérique. Autrement dit, c'est la valeur « pleine échelle » utilisée pour diviser la plage de mesure en intervalles (quantums).

1) Combien de valeurs différentes le CAN (convertisseur analogique => numérique) peut-il représenter ?

   Le CAN travaille sur 3 bits, donc 2^3 = 8 valeurs différentes : 0 à 7

2) Quel est le pas de quantification (quantum, résolution) de la conversion ?

$$Quantum\ (ou\ Pas) = \frac{Uref}{2^n} = \frac{8V}{8} = 1V$$

   La plus petite variation détectable est de 1V.

3) Pour chacune des tensions suivantes, donner la valeur numérique correspondante (en décimal et en binaire) fournie par le CAN : 0V, 2V, 4V, 6V, 7V.

Tension entrée (V)	Code décimal	Code binaire
0V à <1V	0	000
1V à <2V	1	001
2V à <3V	2	010
3V à <4V	3	011
4V à <5V	4	100
5V à <6V	5	101
6V à <7V	6	110
7V à <8V	7	111

   Pour les valeurs demandées :

   0V : code 0 (000)
   2V : code 2 (010)
   4V : code 4 (100)
   6V : code 6 (110)
   7V : code 7 (111)

   En pratique, la tension maximale codée est : $Uref$ – le quantum, donc 8V – 1V soit 7V.

4) Quelle est la tension réelle correspondante au code binaire 101 ?

   101 en binaire = 5 en décimal.

   Tension associée :5 X 1V = 5V, le code 101 représente les tensions comprises de 5V à <6V

Exemple 2 : sonde de température et conversion analogique-numérique

Une sonde de température délivre une tension proportionnelle à la température, soit 10mV/°C (ex : capteur LM35). On souhaite mesurer des températures de 0°C à 100°C à l'aide d'un convertisseur analogique-numérique (CAN) 10 bits, alimenté en 5V.

1) Quelle est la tension délivrée par la sonde pour 0°C et 100°C ?

   À 0°C : 0 X 10 mV = 0V
   À 100°C : 100 X 10mV = 1V

2) Quel est le quantum du CAN (résolution en volts) ?

   Un convertisseur 10 bits code 2^{10} = 1024 niveaux (de 0 à 1023)

   Plage de 0 à 5 V, donc :

$$Quantum\ (ou\ Pas) = \frac{Uref}{2^n} = \frac{5V}{1024} \approx 4{,}88mV$$

3) Quelle sera la valeur numérique fournie pour 20°C ? Donner la méthode de calcul et le résultat arrondi à l'entier.

   Tension délivrée : 20 X 10mV = 200 mV = 0,2V

4) À combien de degrés Celsius près peut-on distinguer deux mesures consécutives (précision maximale) ?

   Chaque quantum = 4,88 mV

   Conversion en degrés :

$$\frac{4{,}88mV}{10mV/°C} = 0{,}49°C$$

   La température peut donc être mesurée tous les ± 0,5°C

5.3 Numérisation du son

Le son est un phénomène de vibration de l'air qui nous parvient à l'oreille par des variations de pression sur nos tympans. Ces variations sont continues et donc de nature analogique. Le son pourrait être représenté graphiquement comme une courbe qui oscille autour d'une valeur moyenne en variant en amplitude et en fréquence.

Cette valeur moyenne correspond à la pression atmosphérique normale (pas de son : silence). Les valeurs situées au-dessus de cette moyenne correspondent à des compressions de l'air et les valeurs situées en-dessous de cette moyenne correspondent à des dépressions.

Ces fluctuations peuvent être enregistrées de manière analogique. C'est ce que l'on faisait en retranscrivant ces oscillations dans les sillons des disques vinyles.

À l'heure du son numérique, les signaux analogiques captés par le micro sont numérisés (on dit aussi « digitalisés ») en échantillonnant à intervalles réguliers les mesures de ces fluctuations.

La numérisation, comme toutes retranscriptions d'ailleurs, n'est jamais réalisée avec une précision absolue. La figure ci-après illustre l'échantillonnage d'un son avec des valeurs pouvant aller de -7 à +7 ce qui pourrait être réalisé avec un code de 4 bits. On voit sur ce graphique que les valeurs des échantillons sont très approximatives et la restitution de la courbe à partir des échantillons risque d'être bien moins nuancée que la courbe initiale.

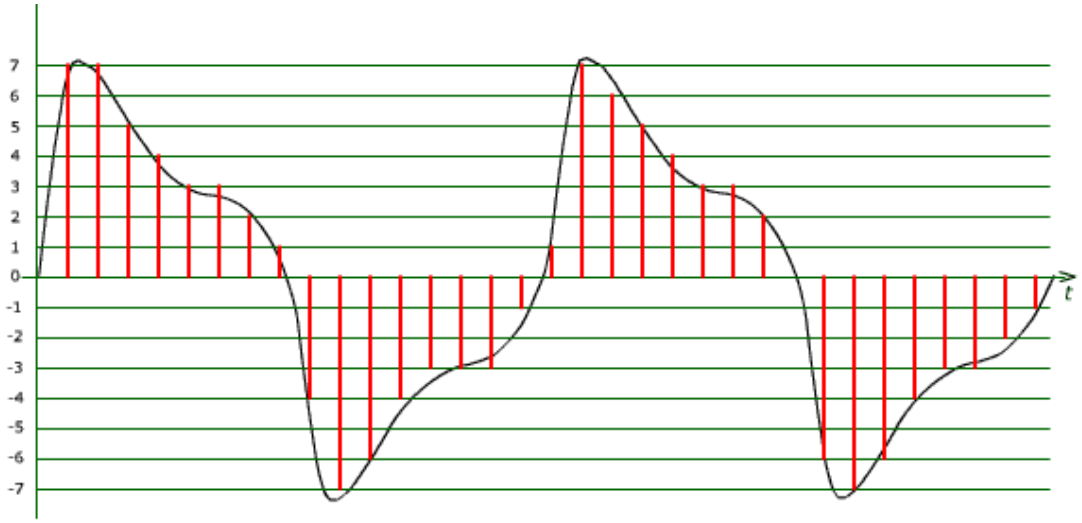


Figure 16 : Echantillonnage avec échantillons codés sur 4 bits

Il est possible pour améliorer la qualité de cette numérisation d'agir sur deux facteurs :

- Le taux d'échantillonnage
- La précision des mesures des échantillons

Le taux ou fréquence d'échantillonnage est fonction de la qualité attendue. L'oreille humaine est capable dans les meilleures conditions d'entendre les fréquences allant de 20 Hz à 20.000 Hz. On comprendra que pour avoir un signal de haute fidélité HiFi on devra avoir une fréquence d'échantillonnage qui soit au moins égale au double de la fréquence à reproduire la plus haute. Dans ce cas 2 x 20kHz = 40 kHz. Ainsi les CD sont enregistrés avec une fréquence de 44,1kHz.

La numérisation représentée par la figure ci-après est meilleure que la précédente car les échantillons y sont deux fois plus nombreux.

Les valeurs échantillonnées sont de plus codées de manière beaucoup plus fine puisqu'elles sont reprises avec des valeurs discrètes qui vont de -15 à +15 au lieu de -7 à +7. Il faudrait dans ce cas coder ces nombres sur 5 bits.
Dans le cas des CD audio, les échantillons sont codés sur 16 bits et peuvent donc alors prendre 2^{16} = 65.536 valeurs distinctes.

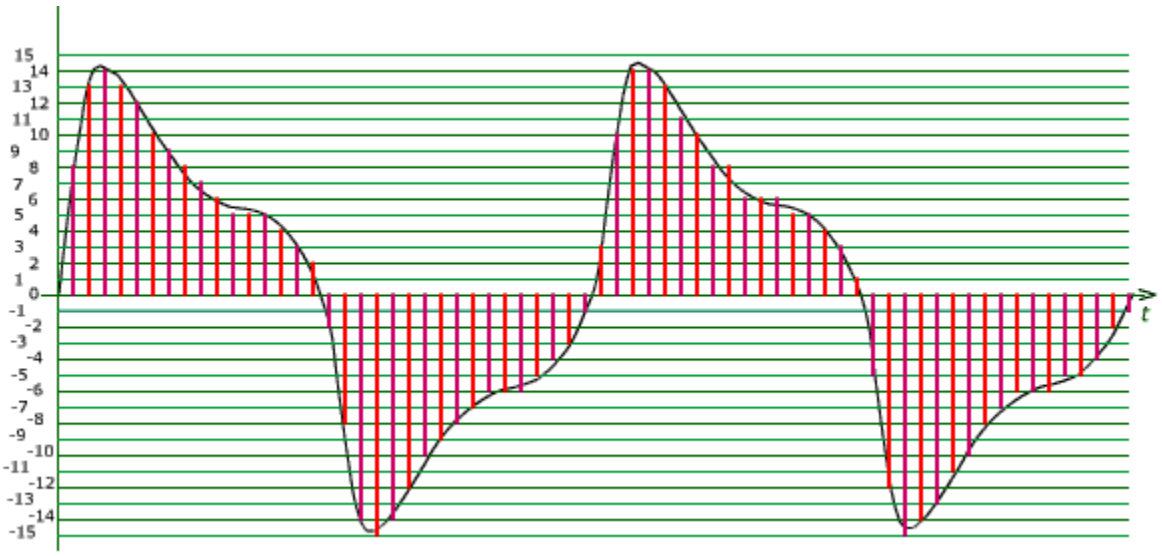


Figure 17 : Echantillonnage à fréquence double avec échantillons codés sur 5 bits

5.3.1 Les caractéristiques du son

Tout le son (intensité, hauteur, timbre, etc.) est contenu dans un seul élément physique : la variation de pression de l'air au cours du temps.

C'est un peu comme si toute la « richesse » de la musique ou de la voix n'était qu'une courbe qui monte et descend (pression en fonction du temps). La magie, c'est que cette courbe est très complexe et que l'on peut en tirer toutes les caractéristiques du son.

5.3.1.1 Intensité (volume)

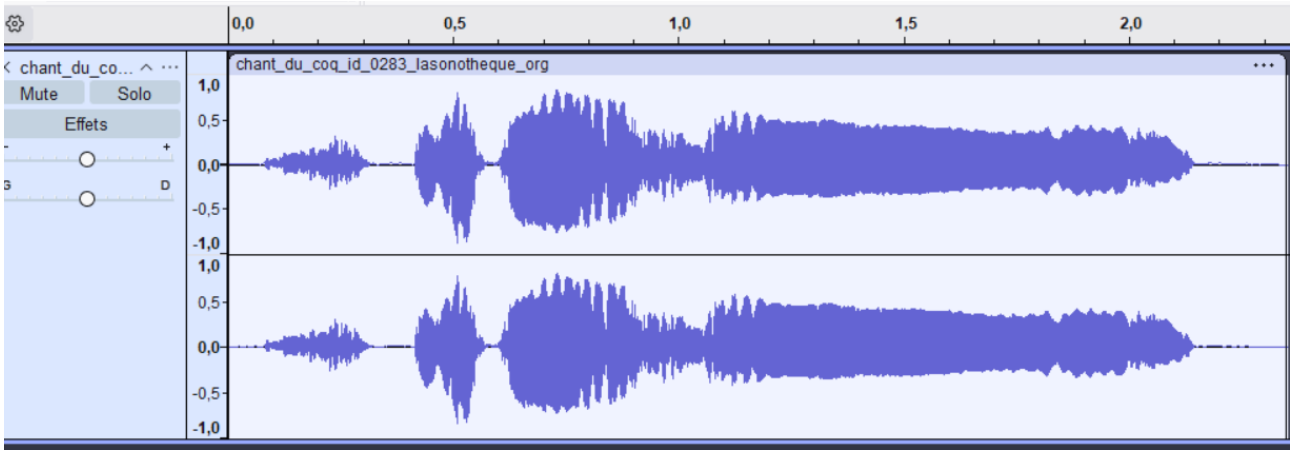
L'**intensité** sonore est liée à l'**amplitude des variations de pression**.

Grandes variations de pression => amplitude élevée => son fort.

Petites variations => son faible.

Exemple :
Chuchoter = oscillations de pression toutes petites.
Crier = oscillations de pression beaucoup plus grandes.

Exemple de spectre pour un son fort (chant d'un coq) :



Exemple de spectre pour un son faible (chuchotement) :



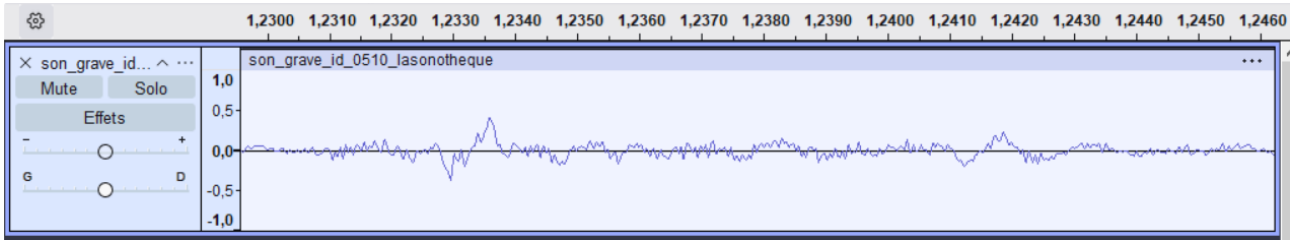
5.3.1.2 Hauteur (grave / aigu)

La **hauteur** du son dépend de la **fréquence des oscillations** (nombre de cycles par seconde).

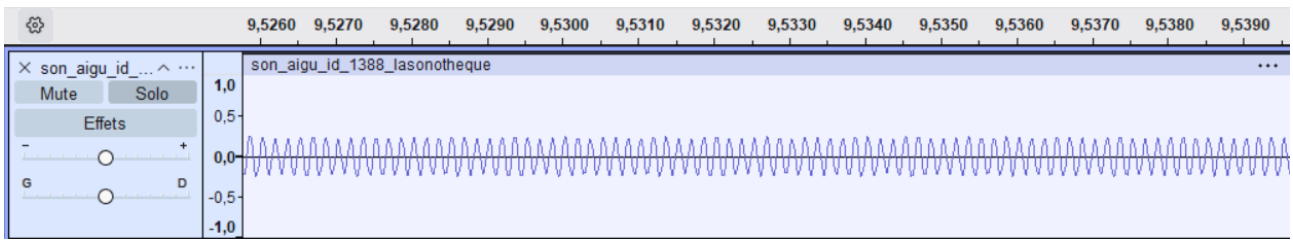
Fréquence basse (100 Hz) → son grave.
Fréquence élevée (1000 Hz) → son aigu.

Autrement dit : c'est la vitesse des variations de pression qui détermine la hauteur.

Exemple de spectre pour un son grave () :



Exemple de spectre pour un son aigu () :



5.3.1.3 Timbre (couleur du son)

Le **timbre** dépend de la **forme de la variation de pression**.

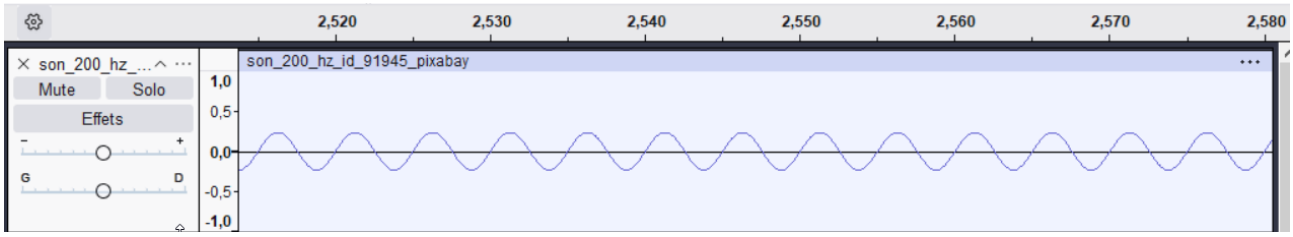
Un son « pur » (un diapason) => oscillation quasi sinusoïdale.

Une voix ou une guitare → oscillation plus complexe, mélange de fréquences (fondamentale + harmoniques).

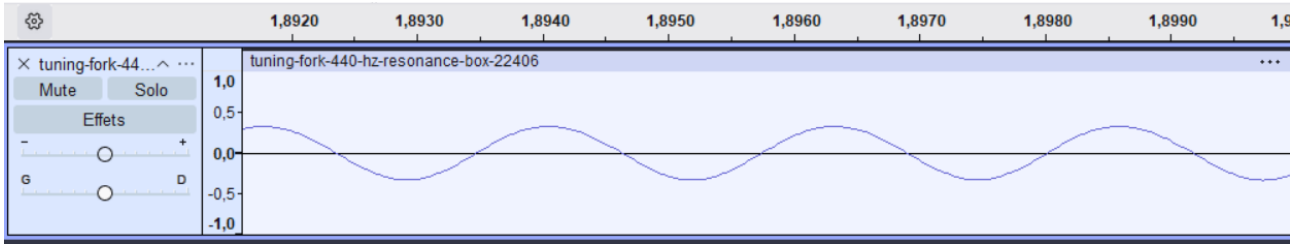
Ce mélange est visible dans le spectre : c'est la « signature fréquentielle » du son.

Deux instruments jouant la même note « La » à 440 Hz produisent la même fréquence fondamentale, mais pas les mêmes harmoniques → leur timbre est différent.

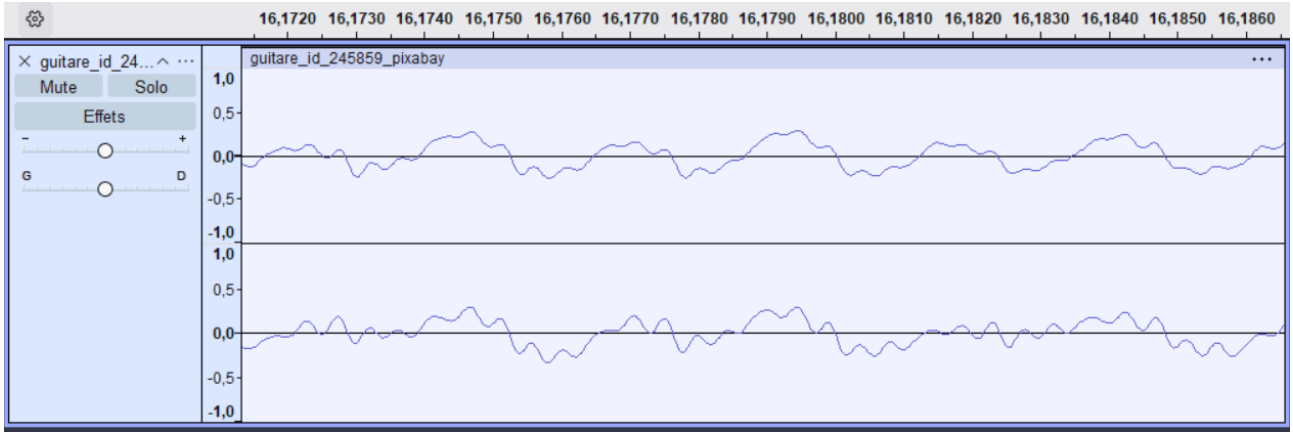
Exemple de spectre pour un son pur à 200 Hz :



Exemple de spectre pour un son pur à 440 Hz :



Exemple de spectre pour un son de guitare :



5.3.1.4 En résumé

La variation de pression dans l'air contient toutes les informations parce que :

- L'amplitude → intensité perçue.
- La fréquence → hauteur.
- La forme de la courbe (donc la composition en fréquences) → timbre.

En mathématiques (via la transformée de Fourier), toute onde sonore, aussi compliquée soit-elle, peut être décomposée en une somme de sinusoïdes de différentes fréquences et amplitudes.
C'est ce qui permet de retrouver hauteur, intensité et timbre.

5.3.2 Conversion d'un signal sonore analogique en numérique

Un capteur de son délivre une tension continue variable entre 0V et 3V, proportionnelle à l'intensité du son.

On utilise un convertisseur analogique-numérique (CAN) sur 4 bits, avec une tension de référence Uref=3V

1) Combien y a-t-il de niveaux numériques possibles avec ce CAN ?

   2^4 = 16 niveaux (0 à 15)

2) Quel est le quantum (pas) de quantification ?

$$Quantum\ (ou\ Pas) = \frac{Uref}{2^n} = \frac{3V}{16} = 0{,}1875V$$

3) Pour les tensions suivantes, donner leur code numérique binaire et décimal :
   0V ; 0,75V ; 1,5V ; 2,25V ; 3V.

Tension (V)	Code décimal	Code binaire
0	0	0000
0,75	$\frac{0{,}75}{0{,}1875} = 4$	0100
1,5	$\frac{1{,}5}{0{,}1875} = 8$	1000
2,25	$\frac{2{,}25}{0{,}1875} = 12$	1100
3	15	1111

6 Architecture des machines et assembleur

Source : (*Architecture*, s. d.)

6.1 Le modèle de John von Neumann

Nous allons revenir sur le modèle de John von Neumann déjà évoqué précédemment. Ce modèle a été développé au milieu du XXe siècle et est toujours d'actualité. Toutes les machines modernes sont basées sur ce modèle.

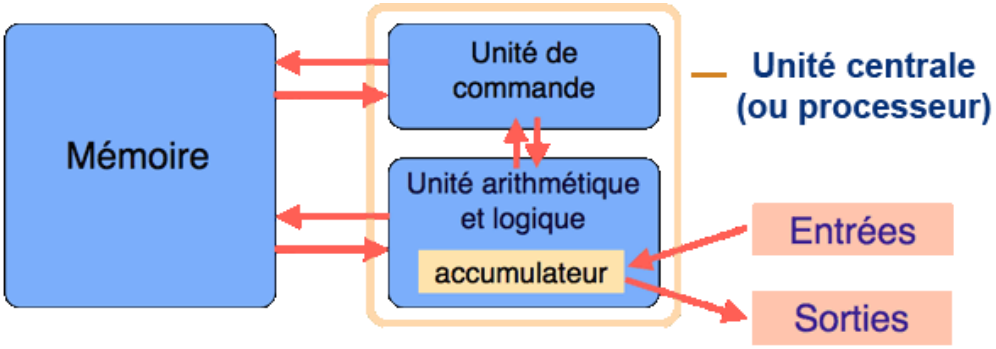
Nous travaillerons aussi dans un langage particulier à chaque machine : l'assembleur.

6.1.1 Présentation générale

En 1944, John von Neumann propose un modèle d'architecture novateur qui sert toujours de base à l'architecture des ordinateurs actuels.

La **première innovation** est la **séparation** nette entre l'**unité de commande**, qui organise le flot de séquencement des instructions et l'**unité arithmétique**, chargée de l'exécution proprement dite de ces instructions. La **seconde innovation**, la plus fondamentale, est l'**idée du programme enregistré** : les **instructions**, au lieu d'être codées sur un support externe (ruban, cartes, tableau de connexions), **sont enregistrées dans la mémoire** selon un codage conventionnel. Un compteur ordinal contient l'adresse de l'instruction en cours d'exécution; il est automatiquement incrémenté après exécution de l'instruction, et explicitement modifié par les instructions de branchement.

Un emplacement de mémoire peut contenir indifféremment des instructions et des données, et une conséquence majeure (dont toute la portée n'avait probablement pas été perçue à l'époque) est qu'un programme peut être traité comme une donnée par d'autres programmes. Cette idée, présente en germe dans la machine de Turing, trouvait ici sa concrétisation.



- Processeur : composé de deux unités
  - Unité de commande : contrôle la séquence d'instructions
  - Unité arithmétique et logique : exécute les instructions
- Mémoire : contient les données et les programmes. mémoire volatile (qui s'efface après redémarrage, mais rapide) et de masse (stockage)
- Entrées : clavier, cartes perforées, etc.
- Sorties : imprimante, écran

Comme indiqué précédemment, **une des nouveautés principales est de regrouper les programmes et les données dans une seule mémoire**. Les programmes deviennent des données que l'on peut manipuler.

L'ordinateur effectue des instructions en séquence.

Chaque instruction se découpe en trois parties :

- fetch (récupérer)
- read (lire)
- execute (exécuter)

6.1.2 Unité de commande

Elle contrôle les instructions réalisées par la machine.

C'est elle qui récupère les instructions et les décode.

Elle s'occupe donc des parties « fetch » et « read »

6.1.3 Unité arithmétique et logique

Elle s'occupe de réaliser les calculs à effectuer.

6.1.4 Mémoire

Avec le modèle de von Neumann, dans la mémoire on trouve à la fois les programmes et les données. C'est la nouveauté majeure de ce modèle. Voir ainsi les programmes permet de les modifier comme le reste des données.

6.1.5 Qu'en est-il aujourd'hui ?

Plus de quatre-vingts ans après son invention, le modèle d'architecture de von Neumann régit toujours l'architecture des ordinateurs. Par rapport au schéma initial, on peut noter certaines évolutions.

Les ordinateurs comportent maintenant des processeurs multiples, qu'il s'agisse d'unités séparées ou de « cœurs » multiples à l'intérieur d'une même puce. Cette organisation permet d'atteindre une puissance globale de calcul élevée sans augmenter la vitesse des processeurs individuels, limitée par les capacités d'évacuation de la chaleur dans des circuits de plus en plus denses.

Malgré sa prévalence, l'architecture de von Neumann présente des limites, notamment le « goulot d'étranglement de von Neumann » : le bus unique ralentit le transfert entre mémoire et processeur, ce qui limite les performances, surtout pour des tâches nécessitant un fort parallélisme ou de grands volumes de données.

Pour pallier ces limites, des alternatives ou des adaptations sont apparues :

- **Architecture Harvard** : sépare la mémoire des instructions et des données, utilisée dans certains microcontrôleurs et systèmes embarqués.
- **Traitement parallèle** : processeurs multicœurs, grappes de calcul, superordinateurs adaptent le modèle pour augmenter le parallélisme.
- **Nouvelles architectures** : l'informatique quantique, les architectures neuromorphiques[5] ou orientées mémoire proposent des paradigmes[6] différents, mais restent encore marginaux dans l'usage courant.

[5] Une architecture neuromorphique est un type d'architecture informatique qui cherche à imiter le fonctionnement du cerveau humain en reproduisant la structure et le comportement des neurones et des synapses.
[6] Paradigme : Modèle théorique de pensée qui oriente la recherche et la réflexion scientifiques.

6.2 Initiation à l'assembleur

6.2.1 Introduction

6.2.1.1 Instructions machines

Voici quelques notions de base.

- Les instructions machines sont gérées par l'unité de commande qui est chargée de :
  - lire les instructions en mémoire,
  - les décoder,
  - orchestrer leur exécution (par les autres unités du CPU, comme l'ALU (unité arithmétique et logique)).

- Un programme est une suite d'instructions.

- Le CPU est incapable d'exécuter directement des programmes écrits en Python (ou C, java etc.). Les langages de haut niveau sont :
  - compilés (ex. C → assembleur → binaire),
  - ou interprétés (ex. Python → interpréteur → exécution indirecte).

  Programme compilé
  - Le code source est traduit entièrement avant l'exécution en code machine spécifique à la plateforme (fichier exécutable).
  - Ce processus s'appelle la compilation, réalisée par un compilateur.
  - Le programme compilé s'exécute donc directement par le processeur, ce qui est généralement plus rapide à l'exécution.
  - Exemples de langages compilés : C, C++...
  - Avantages : rapidité d'exécution, optimisation possible.
  - Inconvénients : il faut recompiler pour chaque architecture différente, étape de compilation pouvant être longue.

  Programme interprété
  - Le code source est traduit et exécuté ligne par ligne en temps réel par un programme appelé interpréteur.
  - Il n'y a pas de traduction complète préalable en code machine.
  - Cela facilite la portabilité et le débogage, car on peut modifier le code et le tester rapidement.
  - Exemples de langages interprétés : Python, JavaScript.
  - Avantages : développement rapide, flexibilité.
  - Inconvénients : généralement plus lent à l'exécution.

- Les instructions exécutées au niveau du CPU sont donc codées en binaire.

6.2.1.2 Langage machine

Le langage machine désigne l'ensemble des instructions exécutables par le microprocesseur.

Chaque architecture de processeur (ex. x86, ARM, RISC-V) a son propre jeu d'instructions.

Tous les processeurs d'une même architecture utilisent le même jeu d'instructions de base, avec parfois des extensions. Voici quelques grandes familles d'architecture :
- x86 / x86-64 : utilisés dans la plupart des ordinateurs de bureau et portables.
- ARM : très répandus dans les smartphones, tablettes, et de plus en plus dans certains ordinateurs (ex. MacBook M1/M2).
- RISC-V : architecture libre, en plein développement.

6.2.1.3 Précisions

Une **instruction machine** est une **suite de bits** interprétée directement par le processeur. Elle est généralement composée de deux parties principales :

- **le champ « code opération » (ou opcode), qui indique au processeur quelle opération effectuer (par exemple : addition, soustraction, copie de données, etc.) ;**
- **le champ « opérandes », qui précise les données concernées par cette opération, ou les emplacements mémoire (ou registres) où se trouvent ces données.**

Par exemple, une instruction peut ordonner d'additionner le contenu de deux registres, ou de copier une valeur en mémoire.

La structure exacte d'une instruction peut varier selon l'architecture du processeur.

6.2.2 Familles des instructions machine

On trouve trois grandes familles d'instructions machine :

- arithmétiques et logiques ;
- transfert de données ;
- contrôle de séquence (rupture de séquence).

6.2.2.1 Instructions arithmétiques et logiques

Ces instructions réalisent des opérations de calcul ou de traitement logique sur des données :

- Addition, soustraction, multiplication, etc.
- Opérations logiques : ET, OU, NON, etc.

Exemple :
Additionner la valeur contenue dans le registre R1 avec le nombre 789 et placer le résultat dans le registre R0.

6.2.2.2 Instructions de transfert de données

Ces instructions permettent de déplacer des données d'un emplacement à un autre, par exemple :

- d'un registre à un autre ;
- de la mémoire vers un registre ;
- d'un registre vers la mémoire.

Exemple :
Copier la valeur située à l'adresse mémoire 487 dans le registre R2.

Registres et mémoire

- Un **registre** est une **petite unité de stockage rapide** à l'intérieur du processeur. Les registres sont très rapides. Dans les processeurs modernes, des dizaines de registres par cœur sont visibles par le programmeur mais plusieurs centaines existent en interne et sont utilisés par le système.
- **La mémoire vive (RAM)** est beaucoup plus vaste mais plus lente. Chaque cellule de mémoire est repérée par une adresse.

Les opérations les plus rapides s'effectuent entre registres.

Les processeurs modernes sont conçus pour effectuer leurs opérations arithmétiques, logiques et de contrôle uniquement sur des registres internes. Voici les points clés :

Temps d'accès :

- Accéder à un registre prend 1 cycle d'horloge ou moins.
- Accéder à la mémoire (même RAM) prend plusieurs cycles.
- Accéder au disque (stockage) prend des millions de fois plus de temps.

Architecture classique (Von Neumann, Harvard) :
- Les instructions manipulent des données stockées temporairement dans les registres.
- Les unités de calcul (ALU, FPU…) prennent leurs opérandes (données manipulées par l'opérateur) dans les registres.

Taille d'un mot

Un mot est un groupe de bits correspondant à la largeur de données manipulée par le processeur :
- Un processeur 8 bits utilise des mots de 8 bits (1 octet),
- Un processeur 32 bits, des mots de 32 bits (4 octets),
- Un processeur 64 bits, des mots de 64 bits (8 octets), etc.

6.2.2.3 Instructions de contrôle de séquence (rupture de séquence)

Par défaut, les instructions sont exécutées séquentiellement, les unes après les autres, depuis la mémoire vive.
Le processeur lit chaque instruction en mémoire à l'adresse indiquée par le compteur ordinal (registre PC – Program Counter), la charge dans un registre interne (registre d'instruction ou Instruction Register), puis l'exécute.

Exemple :
- Instruction 1 à l'adresse 343
- Instruction 2 à l'adresse 344
- etc.

Mais certaines instructions modifient ce déroulement. Ce sont les instructions de rupture de séquence, on parle aussi de sauts (jumps) ou de branches.

Exemple :

Si la valeur dans R1 est > 0, alors sauter à l'adresse 4521,
sinon, continuer à l'adresse suivante (ou sauter à 355).

Deux types de sauts :
- Saut inconditionnel : le saut a lieu dans tous les cas.
- Saut conditionnel : le saut dépend d'une comparaison préalable (égalité, inégalité, supériorité…).

6.2.3 Programme en langage machine

Un programme en langage machine est une suite d'instructions codées en binaire (une très longue chaîne de 0 et de 1).

Exemple :

11100010 10000010 00010000 01111101

Ce type de représentation est très difficile à lire, à écrire et à comprendre pour un humain.
Pour faciliter la programmation, on utilise à la place des codes symboliques appelés mnémoniques, plus proches du langage humain.

Par exemple, la ligne précédente peut être représentée en assembleur comme suit :

ADD R1, R2, #125

Décomposition d'une instruction machine :

Champ binaire	Signification
11100010	Code opération (ex. : ADD)
10000010	Registre 1 (R1)
00010000	Registre 2 (R2)
01111101	Valeur immédiate (#125)

Assembleur
Le langage assembleur (ou « assembly ») utilise ces mnémoniques (ADD, MOV, SUB, etc.) pour représenter les instructions machines de façon plus lisible.

Un programme écrit en assembleur doit ensuite être traduit en langage machine par un programme appelé **assembleur**.

Le **programme assembleur** est **un logiciel externe au processeur** qui se trouve sur l'ordinateur (ou dans l'outil utilisé). Il transforme les instructions symboliques en binaire pour produire un programme exécutable par le matériel. Avec le simulateur AQA, le programme assembleur est intégré dans l'environnement du simulateur (souvent en ligne ou en local).

Exemple :

ADD R1, R2, #125

est traduit par l'assembleur en :

11100010 10000010 00010000 01111101

Pourquoi apprendre l'assembleur ?

Bien que la plupart des programmes soient aujourd'hui écrits en langages de haut niveau (Python, C, Java…), l'assembleur reste utilisé pour :

- la programmation bas niveau (firmware[7], pilotes, microcontrôleurs) ;
- les optimisations de performance critiques[8] ;
- la sécurité informatique (analyse de code malveillant, rétro-ingénierie) ;
- l'enseignement de l'architecture des ordinateurs.

6.2.4 Présentation d'un langage assembleur par l'exemple

6.2.4.1 Déplacements de données

Mnémonique	Exemple	Description
LDR	LDR R1, 78	Charge dans le registre R1 la valeur située à l'adresse mémoire 78.
STR	STR R3, 125	Stocke dans la mémoire à l'adresse 125 la valeur contenue dans le registre R3.
MOV	MOV R1, #23	Place la valeur 23 dans le registre R1.

[7] Un firmware est un programme stocké en mémoire non-volatile (souvent de type ROM, EEPROM ou flash) qui contrôle le comportement basique d'un matériel électronique.
[8] Optimisations de performance critiques : Ce sont des situations où chaque cycle processeur compte, et où les langages de haut niveau (C, Python, Java…) ou même un compilateur C optimisé ne suffisent pas à produire du code assez rapide ou efficace. Dans ces cas-là, on écrit directement une partie du code en assembleur pour exploiter au maximum le matériel.

6.2.4.2 Opérations arithmétiques

Mnémonique	Exemple	Description
ADD	ADD R1, R0, #128	Additionne 128 à R0 et stocke le résultat dans R1.
ADD	ADD R0, R1, R2	Additionne les contenus de R1 et R2, place le résultat dans R0.
SUB	SUB R1, R0, #128	Soustrait 128 de R0, place le résultat dans R1.
SUB	SUB R0, R1, R2	Soustrait R2 de R1, place le résultat dans R0.

6.2.4.3 Instructions de contrôle de flux (ruptures de séquence)

Mnémonique	Exemple	Description
HALT	HALT	Arrête immédiatement l'exécution du programme.
B	B 45	Saut inconditionnel vers l'instruction située à l'adresse mémoire 45.
CMP	CMP R0, #23	Compare le contenu de R0 avec 23. Met à jour les indicateurs de condition (utilisés pour les sauts conditionnels).
BEQ	BEQ 78	Saute à 78 si les deux valeurs comparées par CMP sont égales.
BNE	BNE 78	Saute à 78 si les deux valeurs comparées par CMP sont différentes.
BGT	BGT 78	Saute à 78 si le registre est supérieur à la valeur comparée.
BLT	BLT 78	Saute à 78 si le registre est inférieur à la valeur comparée.

6.2.4.4 Labels

Les instructions de branchement en assembleur, comme B, BEQ, BNE, BGT ou BLT, ne font pas appel directement à une adresse mémoire chiffrée, mais utilisent des labels (étiquettes).

Un label est un nom symbolique placé avant une instruction. Lors de l'assemblage, ce nom est automatiquement remplacé par l'adresse mémoire réelle de l'instruction ciblée.

Cela permet une lecture et une modification plus faciles du code.

Exemple commenté :

```
CMP R4, #18     ; Compare le contenu de R4 avec 18
BGT monLabel    ; Si R4 > 18, saute à l'instruction appelée « monLabel »
MOV R0, #14     ; Sinon, place 14 dans R0
HALT            ; Arrêt du programme

monLabel:       ; Début du bloc étiqueté « monLabel »
MOV R0, #18     ; Place 18 dans R0
HALT            ; Arrêt du programme
```

Résultat attendu :

Si R4 > 18 : saut vers monLabel → R0 = 18

Sinon : pas de saut → R0 = 14

6.3 Simulation d'un CPU : l'assembleur AQA

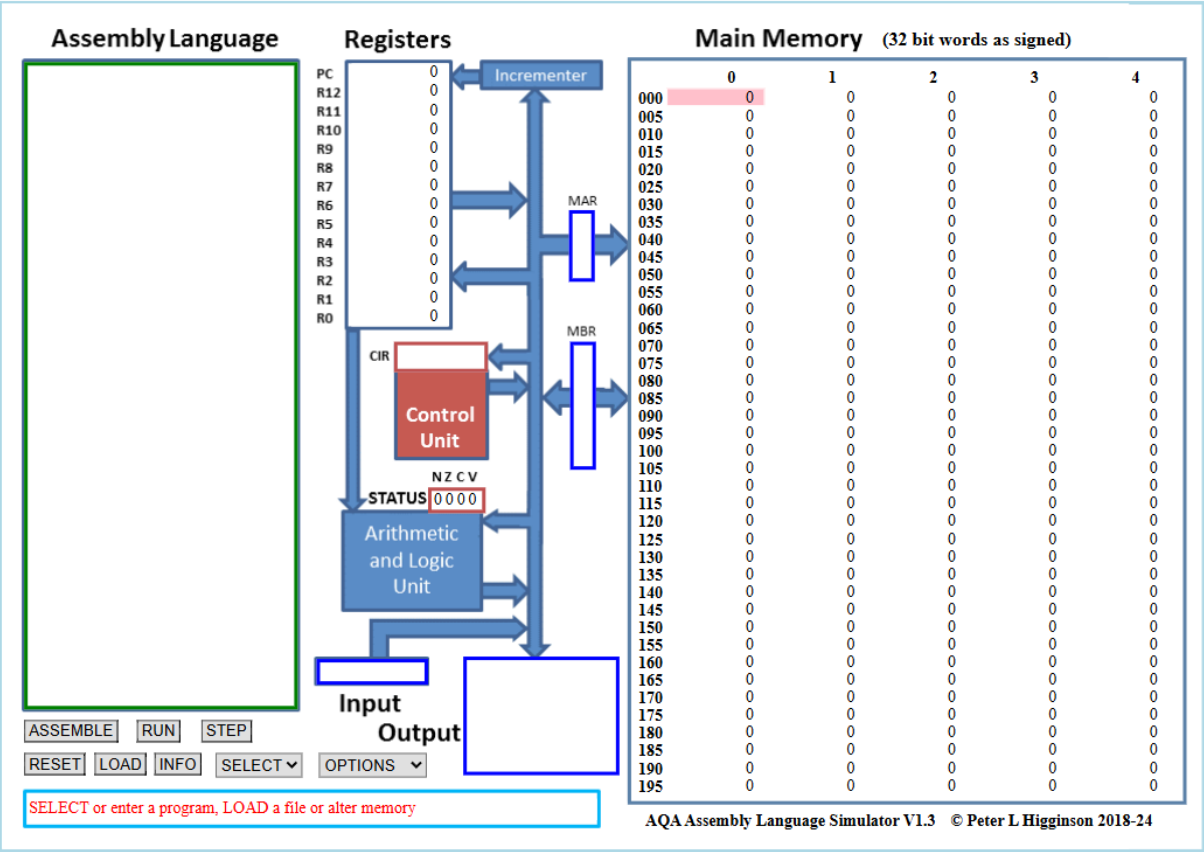
L'assembleur AQA (Assessment and Qualifications Alliance[9]) est un simulateur de processeur mettant en oeuvre le modèle de Von Neumann.

6.3.1 Présentation

Afin de pouvoir mettre en pratique ce qui a été vu dans le volet consacré au modèle d'architecture de von Neumann, nous allons utiliser un simulateur développé par Peter L Higginson.

Une version en ligne de ce simulateur est disponible ici : https://www.peterhigginson.co.uk/AQA/

Voici ce que vous devriez obtenir en vous rendant à l'adresse indiquée ci-avant :



Il est relativement facile de distinguer les différentes parties du simulateur :

- à droite, on trouve la mémoire vive (« main memory ») ;
- au centre, on trouve le microprocesseur ;
- à gauche on trouve la zone d'édition (« Assembly Language »), c'est dans cette zone que nous allons saisir nos programmes en assembleur.

Reprenons les bases du fonctionnement de la RAM et du CPU dans le simulateur de Peter Higginson, voici quelques principes fondamentaux :

- **Pas de variables nommées :**
  Le simulateur ne permet pas de déclarer des variables comme dans un langage de haut niveau. Les données sont manipulées soit à des adresses mémoire précises, soit dans l'un des registres disponibles (R0 à R12).

[9] AQA (Assessment and Qualifications Alliance) : C'est un organisme britannique d'examen et de certification, l'un des principaux au Royaume-Uni. AQA propose des qualifications pour les élèves du secondaire, dont le GCSE (General Certificate of Secondary Education) et le A-Level (Advanced Level).

- **Pas de structures de contrôle de haut niveau :**
  Il n'existe aucune construction de type if … then … else, while, ou for.
  Les conditions et les boucles doivent être mises en œuvre à l'aide d'instructions de saut conditionnel ou inconditionnel, en combinaison avec des instructions de comparaison (CMP).
  Les cibles de saut sont repérées grâce à des étiquettes (labels) que l'on place dans le code.

- **Travail indirect avec la mémoire :**
  Pour effectuer des calculs avec des données stockées en mémoire, **il faut d'abord les charger dans un registre** (LDR) avant de les manipuler.
  Les opérations arithmétiques et logiques s'effectuent uniquement entre registres ou entre un registre et une valeur immédiate.

6.3.2 Les registres dans le simulateur AQA

Dans le simulateur AQA, les registres sont des petites zones de mémoire ultra-rapides situées au cœur du processeur (CPU). Ils servent à stocker temporairement des données ou des adresses utilisées pendant l'exécution des instructions.

Types et nombre de registres

- **R0 à R12 :**
  Treize registres généraux, numérotés de R0 à R12, utilisés pour stocker des valeurs numériques, effectuer des calculs, gérer des adresses mémoire, etc.
- **PC (Program Counter)**
  Registre spécial qui contient l'adresse de la prochaine instruction à exécuter.

Utilisation typique des registres

- Charger une donnée depuis la mémoire vers un registre (ex. LDR R1, 100)
- Effectuer une opération (ex. ADD R0, R1, #5)
- Stocker une donnée d'un registre vers la mémoire (ex. STR R0, 120)

Pourquoi les registres ?

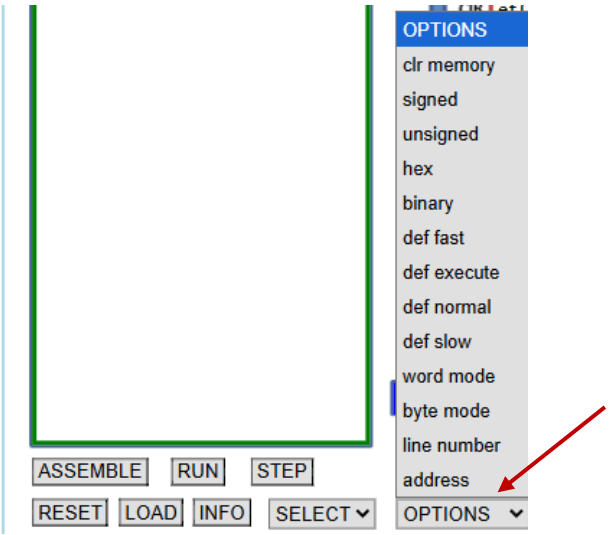
Les registres permettent au processeur de travailler beaucoup plus rapidement que s'il devait manipuler directement les données en RAM.

6.3.3 La RAM dans le simulateur AQA

Par défaut, le contenu des cellules de la mémoire est affiché en **base décimale signée** (entiers pouvant être positifs ou négatifs).
Il est toutefois possible de modifier le format d'affichage :

- Décimal non signé (« unsigned »)
- Hexadécimal (« hex »)
- Binaire (« binary »)

Ces options sont accessibles via le bouton « OPTIONS », situé en bas à gauche de l'interface du simulateur.



6.3.4 Le CPU dans le simulateur AQA

Dans la partie centrale du simulateur, on retrouve — de haut en bas — les éléments suivants :

- Le bloc « Registers » (Registres) : il contient 13 registres généraux (de R0 à R12) ainsi qu'un registre spécial nommé PC (Program Counter), qui stocke l'adresse mémoire de l'instruction en cours d'exécution.
- Le bloc « Control Unit » (Unité de commande) : il affiche l'instruction machine actuellement en cours d'exécution, généralement au format hexadécimal.
- Le bloc « Arithmetic and Logic Unit » (ALU – Unité arithmétique et logique) : cette unité réalise les opérations de calculs (addition, soustraction, etc.) ainsi que les comparaisons logiques (égalité, infériorité, etc.).

6.3.5 Bon à savoir

Peut-on écrire directement en mémoire sans passer par le registre ?
Non, dans le simulateur AQA de Peter Higginson (et plus généralement en assembleur sur une architecture de type Von Neumann), **il n'est pas possible d'écrire directement une valeur en mémoire sans passer par un registre**.

Pourquoi ?

Le jeu d'instructions du simulateur AQA impose que toute écriture en mémoire s'effectue à l'aide de l'instruction STR (Store Register), qui copie la valeur d'un registre vers une adresse mémoire.

Étapes nécessaires pour modifier une case mémoire :

1) Charger une valeur dans un registre avec MOV (valeur immédiate) ou LDR (depuis la mémoire).
2) Utiliser STR pour copier cette valeur du registre vers une adresse mémoire spécifique.

Il n'existe pas d'instruction permettant d'écrire directement une valeur en mémoire sans utiliser un registre intermédiaire.

Exemple valide :

```
MOV R0, #10     // Place la valeur 10 dans le registre R0
STR R0, 100     // Stocke la valeur de R0 à l'adresse mémoire 100
```

Exception (hors exécution) :

On peut **initialiser statiquement la mémoire** à l'aide de la directive DAT (data). Cette directive **n'est pas exécutée par le processeur,** mais interprétée par l'assembleur pour **pré-remplir la mémoire** avant l'exécution du programme.

En résumé :

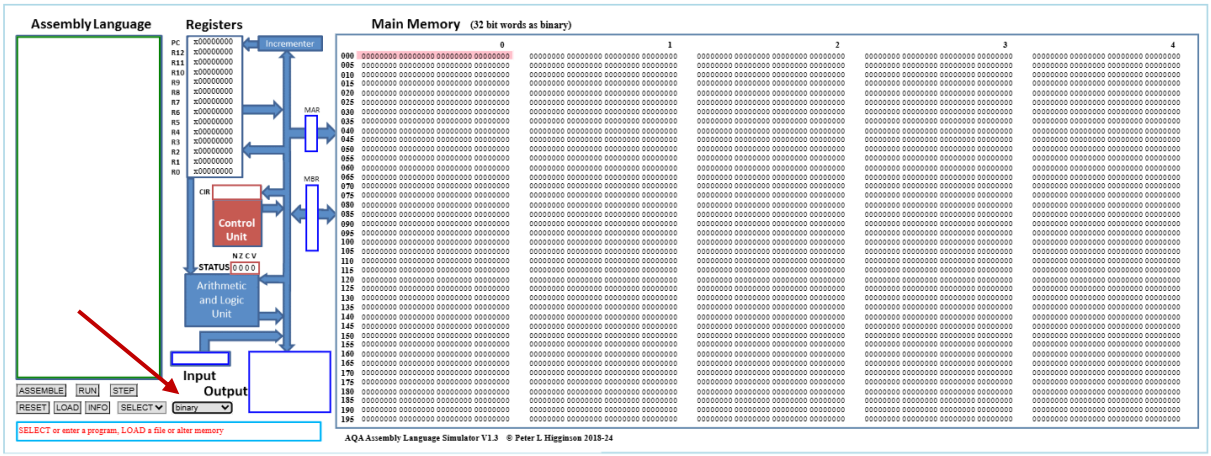
Toute écriture en mémoire durant l'exécution du programme doit impérativement passer par un registre.
Il est impossible d'écrire directement un nombre en mémoire sans passer par une instruction comme STR utilisant un registre comme source.

6.3.6 Exercices « Découverte guidée »

6.3.6.1 Exercice DGSAQA 1 (manipulation des registres et transfert mémoire)

À l'aide de la liste « OPTIONS », passer à un affichage en binaire.

Comme on peut le constater, chaque cellule de la mémoire comporte 32 bits. Chaque cellule de la mémoire possède une adresse (de 000 à 199), ces adresses sont codées en base 10.



Revenir à un affichage en base 10 (liste « OPTIONS » => « signed »).

Dans la partie « éditeur » (Assembly Language) saisir les lignes de codes suivantes :

```
MOV R0,#42
STR R0,150
HALT
```

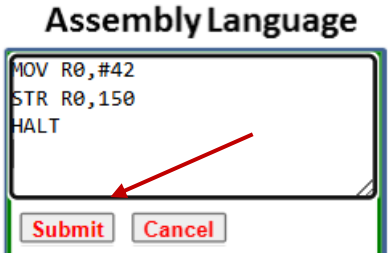
Explications de code :

- MOV : abréviation de « move » (déplacer ou charger).
- R0 : registre de destination. C'est là que la valeur sera envoyée.
- #42 : valeur immédiate (le symbole # indique qu'on utilise directement la valeur 42, et non le contenu d'une adresse mémoire ou d'un autre registre).

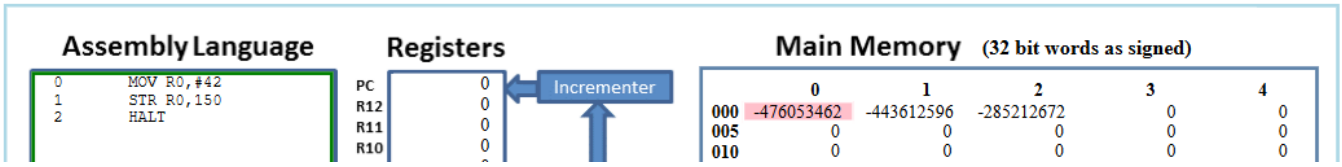
- STR : abréviation de « store » (stocker).
- R0 : registre source, dont la valeur sera copiée.
- 150 : adresse mémoire de destination (ici, la case mémoire numéro 150).

- HALT signifie littéralement « arrêter ».
- Lorsque cette instruction est exécutée, le processeur arrête immédiatement l'exécution du programme.
- Plus aucune instruction n'est lue ni exécutée après un HALT.
- C'est la façon standard de signaler la fin d'un programme.

Une fois la saisie terminée, cliquer sur le bouton « Submit ». De cette manière, nous soumettons le code et l'assembleur traduit les instructions en langage machine.

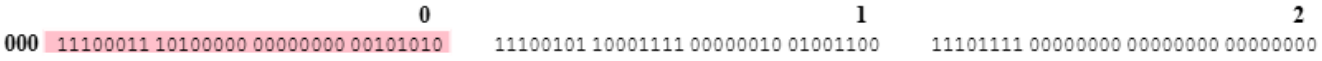


On devrait voir apparaitre des nombres « étranges » dans les cellules mémoires d'adresses 000, 001 et 002 :



L'assembleur a fait son travail, il a converti les 3 lignes de notre programme en instructions machines, la première instruction machine est stockée à l'adresse mémoire 000 (elle correspond à « MOV R0,#42 » en assembleur), la deuxième à l'adresse 001 (elle correspond à « STR R0,150 » en assembleur) et la troisième à l'adresse 002 (elle correspond à « HALT » en assembleur) Pour avoir une idée des véritables instructions machines, il faut repasser à un affichage en binaire ((liste « OPTIONS » => « binary »)).

On devrait obtenir ceci :



Nous pouvons donc maintenant affirmer que :

- l'instruction machine « 11100011 10100000 00000000 00101010 » correspond au code assembleur « MOV R0,#42 »
- l'instruction machine « 11100101 10001111 00000010 01001100 » correspond au code assembleur « STR R0,150 »
- l'instruction machine « 11101111 00000000 00000000 00000000 » correspond au code assembleur « HALT »

Au passage, pour l'instruction machine « 11100011 10100000 00000000 00101010 », on peut remarquer que l'octet le plus à droite, 00101010_2 correspond bien à 42_{10}

Repasser à un affichage en base 10 afin de faciliter la lecture des données présentes en mémoire.

Pour exécuter notre programme, il suffit maintenant de cliquer sur le bouton « RUN ». Nous allons voir le CPU « travailler » en direct grâce à de petites animations. Si cela va trop vite (ou trop doucement), on peut régler la vitesse de simulation à l'aide des boutons "<<" et ">>". Un appui sur le bouton « STOP » met en pause la simulation, si on appuie une deuxième fois sur ce même bouton « STOP », la simulation reprend là où elle s'était arrêtée.

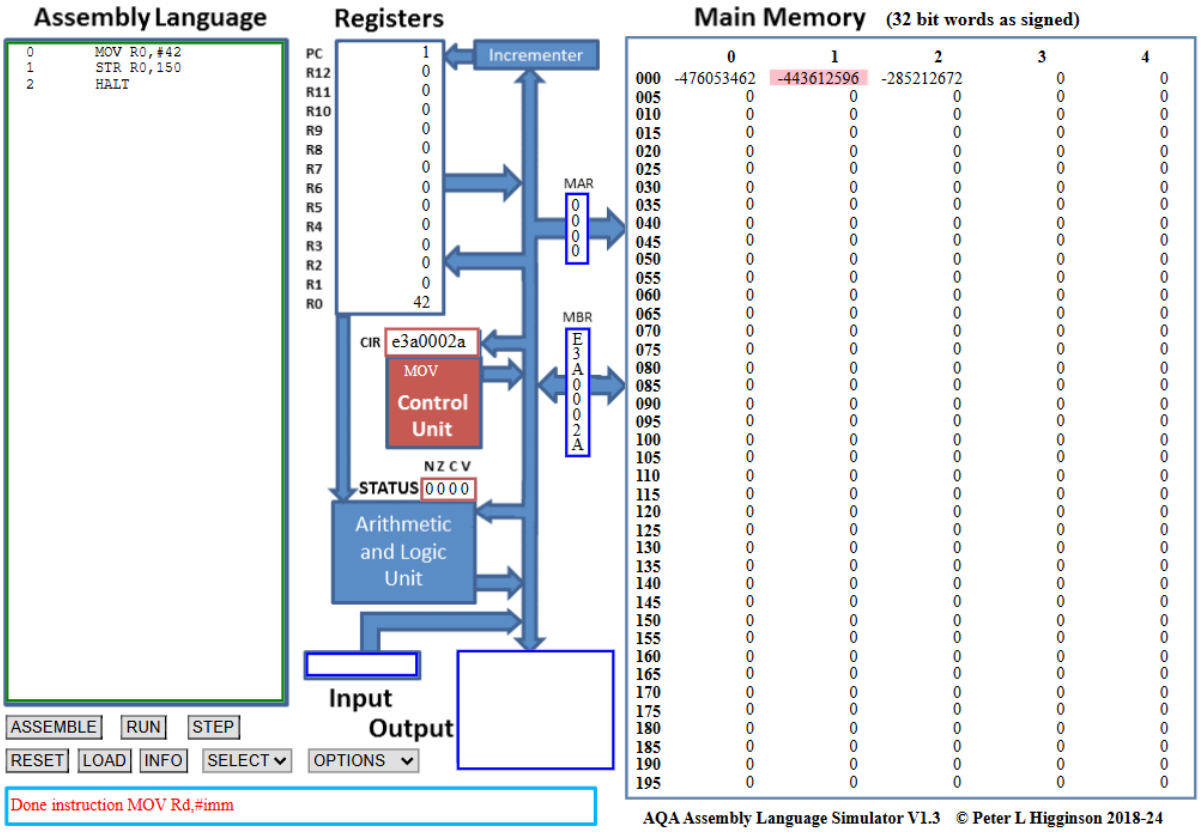
Si on clique sur « STEP », le système exécute la première instruction puis s'arrête. On clique sur « STEP », il exécute la deuxième instruction...

Registres :

- PC (Programme Counter) : compteur de programme, contient l'adresse de la prochaine instruction à exécuter
- MAR (Memory Address Register) : adresse d'écriture/lecture en mémoire
- MBR (Memory Buffer Register) : donnée à lire / écrire. **La valeur est exprimée en hexadécimal (car représentation compacte, claire).**
- CIR (Counter Instruction Register) : registre qui a pour rôle principal de stocker l'instruction en cours d'exécution dans le processeur , permettant ainsi au CPU de gérer et d'exécuter séquentiellement les instructions du programme ; ce registre est aussi appelé registre d'instructions.

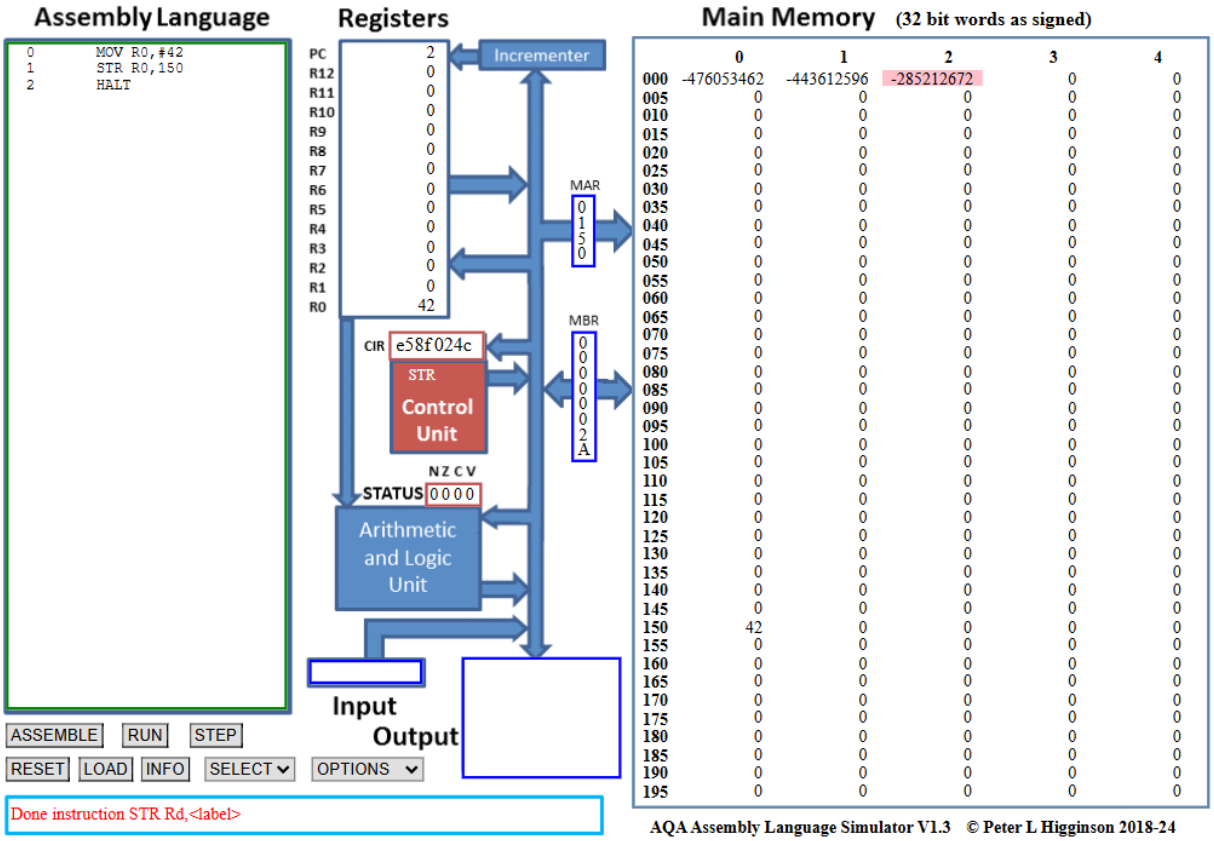
Voici l'illustration d'une exécution **pas à pas (bouton « STEP »)** :

1re étape



PC contient 1. Il s'agit bien de l'adresse de la prochaine instruction à exécuter.

2me étape



- PC contient 2. Il s'agit bien de l'adresse de la prochaine instruction à exécuter.
- MAR contient 0150, il s'agit de l'adresse d'écriture en mémoire (STR R0, 150).
- MBR contient 0000002A correspond à la donnée écrite. La valeur est exprimée en hexadécimal et correspond à 42_{10}
- CIR contient l'instruction qui est en cours d'exécution. La valeur est exprimée en hexadécimal. Le mnémonique est affiché en dessous.

Une fois la simulation terminée, on peut constater que la cellule mémoire d'adresse 150, contient bien le nombre 42 (en base 10). On peut aussi constater que le registre R0 a bien stocké le nombre 42.



ATTENTION : Pour relancer la simulation, il est nécessaire d'appuyer sur le bouton « RESET » afin de remettre les registres R0 à R12 à 0, ainsi que le registre PC (il faut que l'unité de commande pointe de nouveau sur l'instruction située à l'adresse mémoire 000). La mémoire n'est pas modifiée par un appui sur le bouton « RESET », pour remettre la mémoire à 0, il faut cliquer sur le bouton « OPTIONS » et choisir « clr memory ». Si on remet la mémoire à 0, il faudra cliquer sur le bouton « ASSEMBLE » avant de pouvoir exécuter de nouveau le programme puisque les instructions devront être à nouveau assemblées (traduites en langage machine).

6.3.6.2 Exercice DGSAQA 2 (absence de variables et structures de haut niveau)

But : Illustrer l'absence de variables et la gestion « manuelle » de la mémoire.

```
MOV R0, #25
STR R0, 180     // Stocke 25 à l'adresse 180
LDR R1, 180     // Charge la valeur de l'adresse 180 dans R1
HALT
```

Explications :
On utilise uniquement des adresses mémoire et des registres, sans variables nommées.
La valeur 25 est stockée puis relue en mémoire.

6.3.6.3 Exercice DGSAQA 3 (transfert mémoire)

But : Illustrer comment lire et écrire des valeurs en mémoire.

Code à expérimenter :

```
MOV R0, #10    // R0 = 10
STR R0, 50     // Stocke la valeur contenue en R0 (10) à l'adresse 50
LDR R1, 50     // Charge la valeur de l'adresse 50 dans R1 (R1 = 10)
HALT
```

Pour créer des commentaires avec le simulateur AQA, il est possible d'utiliser / ou //
Ce qui suit le caractère / est ignoré et est considéré comme étant un commentaire. En utilisant //, nous sommes simplement plus proches de ce que l'on utilise avec certains langages de haut niveau.

Voici quelques explications bien que l'essentiel soit indiqué dans les commentaires.

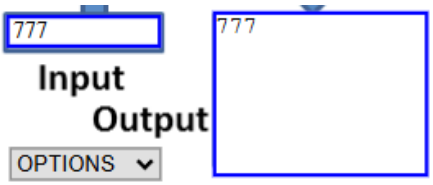
- MOV R0, #10 : initialise R0 à 10.
- STR R0, 50 : stocke la valeur de R0 (10) à l'adresse mémoire 50.
- LDR R1, 50 : charge la valeur de la mémoire à l'adresse 50 dans R1 (donc R1 = 10).
- HALT : fin du programme.

6.3.6.4 Exercice DGSAQA 4 (entrées, sorties)

But : Expérimenter des opérations d'entrée et de sortie.

Dans la partie « éditeur » (Assembly Language) saisir les lignes de codes suivantes :

```
INP R0, 2
OUT R0, 4
HALT
```



Explications du code :

- INP R0, 2

  INP : C'est l'instruction d'entrée (input) qui permet de lire une valeur depuis un périphérique d'entrée.
  R0 : C'est le registre de destination où la valeur lue sera stockée. Dans ce cas, la valeur sera placée dans le registre R0.
  2 : C'est le numéro du périphérique d'entrée. Dans le simulateur AQA, le périphérique 2 est utilisé pour lire un nombre depuis l'entrée standard (généralement le clavier). Il n'y a qu'un périphérique d'entrée.

  En résumé :
  Cette instruction lit un nombre encodé par l'utilisateur et le stocke dans le registre R0. Quand le programme atteint cette instruction, il s'arrête et attend que l'utilisateur saisisse une valeur numérique, puis cette valeur est placée dans le registre R0 et le programme continue.

- OUT R0, 4

  OUT : C'est l'instruction de sortie qui permet d'afficher une valeur.
  R0 : C'est le registre source dont le contenu sera affiché. La valeur stockée dans le registre R0 sera envoyée vers le périphérique de sortie.
  4 : correspond à une demande d'affichage des nombres en format signé (c'est le format par défaut).

  En résumé :
  Cette instruction prend la valeur stockée dans le registre R0 et l'affiche à l'écran sous forme de nombre signé (avec le signe + ou - si applicable).

  Autres valeurs pour le deuxième paramètre :

  OUT R0, 5 : Affiche le nombre en format non signé (unsigned)
  Affiche les nombres sans signe, toujours positifs
  Exemple : 42, 15, 0

  OUT R0, 6 : Affiche le nombre en hexadécimal
  Affiche en base 16 (0-9, A-F)
  Exemple : 2A, FF, 0

  OUT R0, 7 : Affiche le nombre comme un caractère ASCII
  Convertit la valeur numérique en caractère correspondant
  Exemple : 65 → 'A', 48 → '0', 32 → ' ' (espace)

6.3.6.5 Exercice DGSAQA 5 (comparaisons et branchements)

But : Expérimenter des opérations de comparaison et les branchements.

Le système va comparer deux nombres et afficher le plus grand.

Dans la partie « éditeur » (Assembly Language) saisir les lignes de codes suivantes puis soumettre et exécuter :

```
MOV R0, #8
MOV R1, #5
```

```
CMP R0, R1    // Compare R0 et R1
BGT PLUS_GRAND // Si R0 > R1, saute à PLUS_GRAND
OUT R1, 4     // Sinon, affiche R1
B FIN

PLUS_GRAND:
OUT R0, 4     // Affiche R0 si R0 > R1

FIN:
HALT
```

Explications :

- CMP R0, R1 : compare R0 et R1.
- BGT PLUS_GRAND : saute à l'étiquette PLUS_GRAND si R0 > R1.
- Si la condition n'est pas remplie, affiche R1.
- Sinon, si la condition est remplie, affiche R0.
- HALT termine le programme.

6.3.6.6 Exercice DGSAQA 6 (exemple d'accès indirect en assembleur AQA)

But : Expérimenter les accès indirects.

Dans la partie « éditeur » (Assembly Language) saisir les lignes de codes suivantes puis soumettre et exécuter :

```
MOV R5, #196 // R5 reçoit l'adresse mémoire 196
MOV R0, #42 // On place la valeur 42 dans R0
STR R0, [R5] // Stocke le contenu du R0 (42) à l'adresse contenue dans R5 (donc à l'adresse 196)
LDR R2, [R5] // Charge dans R2 la valeur située à l'adresse contenue dans R5 (donc 42)
OUT R2, 4 // Affiche la valeur de R2 (42)
HALT // Fin du programme
```

L'utilisation de crochets [R5] indique explicitement que l'on travaille en mode indirect : le registre contient une adresse, et non une valeur à traiter directement.

Explications étape par étape :

- MOV R5, #196 : R5 contient désormais l'adresse mémoire 196.
- MOV R0, #42 : R0 prend la valeur 42.
- STR R0, [R5] : la valeur de R0 (42) est stockée à l'adresse mémoire pointée par R5 (soit 196).
- LDR R2, [R5] : la valeur située à l'adresse 196 est chargée dans R2 (donc R2 = 42).
- OUT R2,4 : affiche la valeur de R2 (42).
- HALT : arrêt du programme.

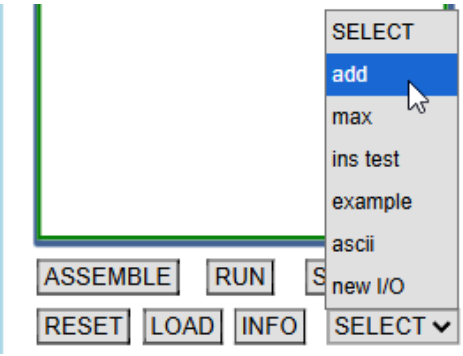
Ce type d'accès est très utile pour :

- Parcourir des tableaux en mémoire (en incrémentant le registre-adresse).
- Manipuler dynamiquement des adresses (pointeurs).
- Écrire des algorithmes plus généraux et puissants.

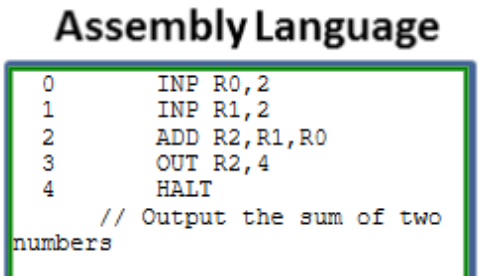
6.3.6.7 Exercice DGSAQA 7 (somme de deux nombres et affichage du résultat)

But : Faire appel à un des exemples de code proposés afin de faire la somme de deux nombres.

Dérouler la liste « SELECT » et choisir « add ».



Voici le code obtenu.



Explication :

- INP R0, 2 et INP R1, 2 : l'utilisateur saisit deux valeurs qui sont placées respectivement dans R0 et dans R1.
- ADD : C'est l'instruction d'addition qui additionne deux valeurs.
  - R2 : C'est le registre de destination où le résultat de l'addition sera stocké.
  - R1 : C'est le premier registre source (premier opérande de l'addition).
  - R0 : C'est le second registre source (second opérande de l'addition).
- OUT : permet d'afficher le contenu de R2 dans la fenêtre de sortie. Pour rappel, le paramètre 4 permet d'afficher des nombres en format signé (c'est le format par défaut).

Fonctionnement :

Cette instruction calcule : R2 = R1 + R0

Elle prend la valeur contenue dans le registre R1, l'additionne avec la valeur contenue dans le registre R0, et stocke le résultat dans le registre R2.

Exécuter étape par étape.

Observer les valeurs dans les différents registres.

Note : l'instruction SUB permet d'effectuer des soustractions. Exemple : SUB R1, R0, #128

6.3.6.8 Exercice DGSAQA 8 (gestion d'un tableau en mémoire)

But : Expérimenter la gestion d'un tableau en mémoire.

```
MOV R0, #100    // L'adresse mémoire de début du tableau est copiée dans R0
MOV R1, #7    // La première valeur du tableau est copiée dans R1
STR R1, [R0]   // Stocke 7 à l'adresse 100
MOV R2, #8    // La deuxième valeur du tableau est copiée dans R2
ADD R0, R0, #1    // Ajoute 1 au contenu du registre R0 et place le résultat dans R0.
STR R2, [R0]   // Stocke 8 à l'adresse 101

MOV R3, #100
LDR R4, [R3]   // R4 reçoit la valeur (7) contenue à l'adresse 100
ADD R3, R3, #1
LDR R5, [R3]   // R5 reçoit la valeur (8) contenue à l'adresse 101
HALT
```

Explications :

- Initialise un « tableau » en mémoire à partir de l'adresse 100.
- Stocke 7 à l'adresse 100, puis 8 à l'adresse 101.
- Lit ensuite ces valeurs en mémoire pour les placer dans les registres R4 et R5

6.3.6.9 Exercice DGSAQA 9 (utilisation de la notation hexadécimale)

But : Montrer l'utilisation de la notation hexadécimale avec le simulateur AQA.

```
MOV R0, #0xA    // 0xA = 10 en décimal
MOV R1, #0xB    // 0xB = 11 en décimal
ADD R2, R0, R1   // R2 = 21
HALT
```

Explications :

- 0xA est la notation hexadécimale pour 10.
- 0xB est la notation hexadécimale pour 11.
- Le programme additionne ces deux valeurs et le résultat (21) est placé dans le registre R2.

Note : Malheureusement la notation 0b1001 pour le binaire ne fonctionne pas avec ce simulateur.

6.3.6.10 Exercice DGSAQA 10 (incrémentation d'une valeur)

But : Montrer l'incrémentation d'une valeur.

```
MOV R0, #25    // Place la valeur 25 dans R0
STR R0, 100    // Stocke la valeur de R0 (25) à l'adresse mémoire 100

LDR R1, 100    // Charge la valeur de l'adresse mémoire 100 dans R1
ADD R1, R1, #1  // Incrémente R1 (R1 = R1 + 1)

STR R1, 101    // Stocke la nouvelle valeur (26) à l'adresse mémoire 101

HALT        // Fin du programme
```

Explications étape par étape

- MOV R0, #25 : R0 reçoit la valeur 25.
- STR R0, 100 : On sauvegarde 25 à l'adresse mémoire 100.
- LDR R1, 100 : On lit la valeur stockée à l'adresse 100 (soit 25) et on la place dans R1.
- ADD R1, R1, #1 : On ajoute 1 à R1, qui vaut maintenant 26.
- STR R1, 101 : On sauvegarde 26 à l'adresse mémoire 101.
- HALT : Arrêt du programme.

Résumé

- STR permet de sauvegarder la valeur d'un registre dans la mémoire.
- LDR permet de charger une valeur de la mémoire dans un registre.
- Ces instructions sont essentielles pour manipuler des données persistantes entre différentes étapes d'un programme assembleur.

6.3.6.11 Exercice DGSAQA 11 (structure conditionnelle)

But : réaliser un test conditionnel équivalent à :
SI la valeur saisie par l'utilisateur est strictement supérieure à 10, alors le programme affiche 10.
SINON, il affiche la valeur saisie.

```
INP R0,2       // Demande à l'utilisateur un nombre (stocké dans R0)
CMP R0, #10    // Compare R0 à 10
BGT SUP_10     // Si R0 > 10, saute à SUP_10
OUT R0,4       // Sinon, affiche R0
B FIN          // Termine le programme

SUP_10:
MOV R1, #10    // Prépare la valeur 10 dans R1
OUT R1,4       // Affiche 10

FIN:
HALT           // Fin du programme
```

Explications étape par étape :

- INP R0,2 : l'utilisateur saisit un nombre, stocké dans R0.
- CMP R0, #10 : compare la valeur saisie à 10.
- BGT SUP_10 : si la valeur saisie est strictement supérieure à 10, le programme saute à l'étiquette SUP_10.
- OUT R0, 4 : si la valeur n'est pas supérieure à 10, on affiche la valeur saisie.
- B FIN : évite d'exécuter la suite inutilement.
- SUP_10: : si la condition est vraie, on affiche 10 à la place.
- FIN: : arrêt du programme.

Résultat attendu :

- Si l'utilisateur saisit un nombre supérieur à 10, le programme affiche 10.
- Sinon, il affiche le nombre saisi.

6.3.6.12 Exercice DGSAQA 12 (boucle)

But : Générer une boucle qui s'exécute tant que la condition R1 != 4 est vraie.

```
    MOV R1, #0    // Initialise le compteur à 0
    MOV R2, #1    // Valeur d'incrément
    MOV R3, #4    // Limite supérieure (4)

Boucle:
    OUT R1,4
    CMP R1, R3    // Compare R1 à 4
    BEQ Fin    // Si R1 == 4, sortir de la boucle
    ADD R1, R1, R2    // Incrémente R1 de 1 (valeur contenue en R2)
    B Boucle    // Retour au début de la boucle

Fin:
    HALT // Fin du programme
```

Explications étape par étape :

- Initialisation :
  R1 = 0 (compteur)
  R2 = 1 (incrément)
  R3 = 4 (valeur d'arrêt)

- Boucle :
  On compare R1 à R3.
  Si R1 atteint 4, le programme saute à l'étiquette Fin.
  Sinon, on incrémente R1 et on recommence la boucle.

Fin :
Le programme s'arrête avec HALT.

Cet exemple illustre parfaitement la notion de boucle en assembleur AQA, où l'on utilise des instructions de comparaison et de saut pour répéter un bloc d'instructions tant qu'une condition n'est pas remplie.

6.3.7 Autres simulateurs

- **ARMlite Simulator**

Également développé par Peter Higginson, il propose une interface différente et un jeu d'instructions ARM simplifié, adapté à l'apprentissage et à la visualisation du fonctionnement des instructions.

- **GIF-1001 Simulateur ARM**

Utilisé dans des contextes universitaires (notamment à l'Université Laval), ce simulateur permet de manipuler les registres, la mémoire et de visualiser l'exécution d'instructions ARM. Il est particulièrement adapté aux exercices pratiques et à la découverte de l'architecture ARM.

- **CPUlator**

Ce simulateur en ligne supporte plusieurs architectures : ARMv7, MIPS, Nios II, et RISC-V. Il offre un environnement complet avec débogueur, visualisation de la mémoire, des registres, et des périphériques. Il est très utilisé dans l'enseignement pour la programmation en assembleur et l'organisation des ordinateurs.

- **VisUAL**

Un émulateur ARM très visuel, conçu pour rendre l'apprentissage de l'assembleur ARM plus intuitif. Il permet d'écrire, d'exécuter et de visualiser le comportement du code assembleur ARM dans une interface graphique moderne.

- **Microchip Studio (ex-Atmel Studio)**

Pour ceux qui souhaitent aller plus loin, cet environnement gratuit permet de programmer et de simuler du code assembleur sur des microcontrôleurs[10] AVR (Arduino Uno…) et ARM de Microchip. Il intègre un simulateur et un débogueur puissants, adaptés à des projets plus avancés.

[10] Microcontrôleur = système complet (microprocesseur + mémoire RAM + mémoire flash ou EEPROM + ports…)

7 Les ordinateurs

7.1 Introduction à la notion d'ordinateur

7.1.1 Présentation de l'ordinateur

Un ordinateur est un ensemble de circuits électroniques permettant de manipuler des données sous forme binaire, c'est-à-dire sous forme de bits.

Le mot ordinateur date de 1955. Il fut inventé par Jacques Perret à la demande du responsable publicité d'IBM qui voulait un nom pour communiquer sur le « calculateur », traduction littérale du mot « computer ». En effet, IBM souhaitait commercialiser une machine destinée au traitement de l'information. Or l'appellation « calculatrice électronique type 650 » ne lui paraissait pas très « accrocheur » au sein d'un message publicitaire et trop réducteur.

Il eut l'idée de faire appel à son professeur de philologie latine à la Sorbonne auquel il confia une description de la nouvelle machine. Après avoir exploré plusieurs pistes, Jacques Perret remet sa copie le 16 avril 1955, proposant le mot « ordinateur ». Le professeur a ainsi retenu une notion de « mise en ordre » pour définir la machine.

Ayant remarqué que le vocabulaire désignant d'autres machines, comme tabulatrice ou trieuse, étaient du genre féminin, il suggéra aussi le mot « ordinatrice ».

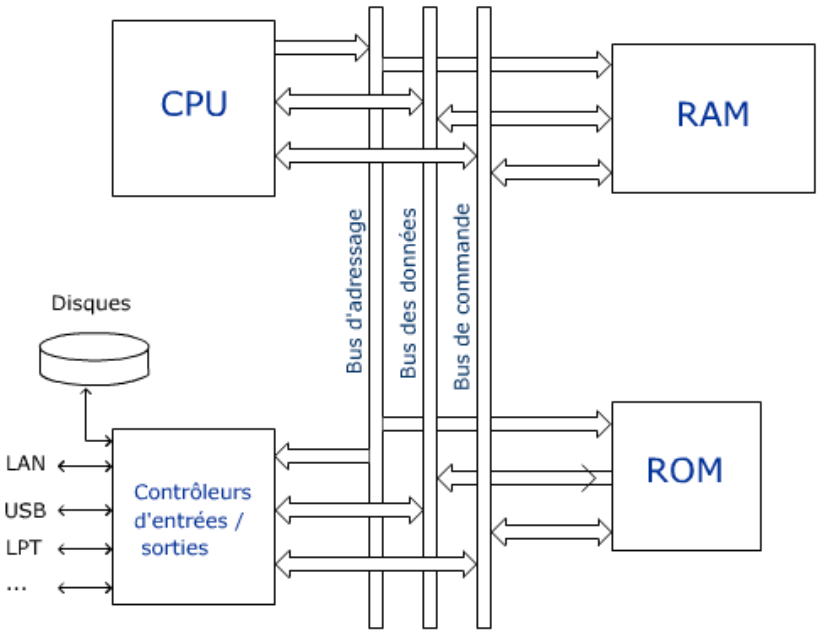
Ce fut le mot « ordinateur » qui fut retenu par IBM pour sortir son « ordinateur IBM 650 ». Le nom fut déposé par la marque, mais passa très vite dans le langage courant. De fait, IBM abandonna alors les droits sur l'appellation.

7.1.2 Types d'ordinateurs

Toute machine capable de manipuler des informations binaires peut être qualifiée d'ordinateur.
Toutefois, la plupart des personnes pensent à un ordinateur personnel (PC, abréviation de personal computer), le type d'ordinateur le plus présent sur le marché, toutefois il existe beaucoup d'autres types d'ordinateurs comme cela a été vu dans le point 3.7.2 (page 25).

7.2 Architecture matérielle – Vue globale

Source : De Mey (s. d.-b)



Un ordinateur est un système électronique complexe et modulaire, composé de nombreux éléments interconnectés. Ces composants, dont certains peuvent être remplacés ou mis à jour, sont organisés autour d'un élément central : la carte mère.

La carte mère est un grand circuit imprimé qui relie et coordonne tous les composants de l'ordinateur. Elle contient :

- des connecteurs pour le processeur, la mémoire, les disques et les cartes d'extension,
- des circuits intégrés pour assurer la communication entre les éléments,
- des contrôleurs pour la gestion des entrées/sorties, de l'audio, du réseau, etc.

La carte mère est installée dans un boîtier (ou châssis) qui contient également :

- des emplacements pour les disques durs ou SSD (stockage) ;
- une alimentation électrique qui fournit un courant stable à tous les composants ;
- des interfaces d'entrée-sortie (USB, HDMI, audio, Ethernet, etc.) ;
- des systèmes de refroidissement (ventilateurs, caloducs[11], ou watercooling selon les modèles).

L'ensemble constitué du boîtier et de ses composants internes (CPU, mémoire, stockage, carte mère, etc.) forme ce que l'on appelle l'unité centrale.



Figure 18 : Exemple de caloduc

Les composants essentiels d'un ordinateur

- **Le processeur (CPU – Central Processing Unit)**

C'est le centre de traitement de l'ordinateur. Il exécute les instructions des programmes et traite les données. Ses performances dépendent de plusieurs facteurs :

- fréquence (GHz),
- nombre de cœurs (multicœurs ou threads),
- taille de la mémoire cache (rapide),
- architecture (ex. : x86, ARM, RISC-V).

- **La mémoire**

RAM (Random Access Memory) : mémoire vive temporaire, utilisée pour stocker les données en cours d'utilisation. Elle est volatile (son contenu est perdu à l'arrêt).

ROM (Read Only Memory) : contient les données essentielles au démarrage, comme le BIOS ou UEFI. Elle est aujourd'hui intégrée dans des puces de type flash reprogrammables.
L'acronyme UEFI signifie « Unified Extensible Firmware Interface », en français « Interface Micrologicielle Extensible Unifiée ». C'est une spécification qui décrit une interface entre le système d'exploitation et le microprogramme (firmware) d'un ordinateur, remplaçant le BIOS.
En d'autres termes, UEFI est un système plus moderne qui succède au BIOS pour gérer le démarrage et la configuration du matériel d'un ordinateur. Il offre des fonctionnalités améliorées telles qu'un

[11] Un caloduc est un dispositif de transfert thermique très efficace qui transporte la chaleur d'une zone chaude vers une zone froide grâce au principe de l'évaporation-condensation d'un fluide contenu à l'intérieur d'un tube hermétique.

démarrage plus rapide, une meilleure sécurité et une compatibilité avec les systèmes d'exploitation plus récents.

Mémoire virtuelle : extension logicielle de la RAM, utilisant le disque dur (HDD) ou SSD comme espace temporaire (fichier d'échange, ou swap). Elle est plus lente que la RAM.

- **Les unités de stockage**

Disques durs (HDD) : supports mécaniques à plateaux magnétiques. Ils offrent un grand volume à bas coût, mais sont plus lents et sensibles aux chocs.

Disques SSD (Solid State Drive) : sans pièces mobiles, rapides et silencieux, à base de mémoire flash. Devenus la norme sur la majorité des PC.
Capacité : de quelques centaines de Go à plusieurs To (1 To = 1 000 Go).

Supports optiques (CD, DVD, Blu-ray) : de plus en plus rares, réservés à des usages spécifiques (archivage, ancien matériel).

- **Les circuits d'entrée/sortie (I/O)**

Ils permettent les échanges de données entre l'ordinateur et ses périphériques :
- clavier, souris, écran,
- imprimante, scanner, disque dur externe, clé USB,
- casque audio, webcam, manette de jeu, smartphone, etc.

- **Les périphériques et les formes d'ordinateurs**

Un ordinateur a besoin de périphériques externes pour permettre l'interaction avec l'utilisateur :
- écran, clavier, souris ou pavé tactile,
- imprimantes, casques, stockage externe, etc.

Aujourd'hui, les ordinateurs fixes (tour + écran) cohabitent avec :
- les ordinateurs portables (laptops),
- les PC tout-en-un (moniteur avec composants intégrés),
- les tablettes et smartphones, qui sont également des ordinateurs miniaturisés, largement utilisés dans de nombreux usages courants.

7.3 La carte mère

7.3.1 Présentation de la carte-mère

La carte mère est le composant principal de l'ordinateur. C'est sur elle que sont connectés ou soudés l'ensemble des éléments essentiels du système. Elle assure la liaison et la coordination entre le processeur, la mémoire, les périphériques et les autres composants.



Figure 19 : Carte mère

Elle intègre plusieurs éléments embarqués, c'est-à-dire directement intégrés à la carte :
- le chipset, un circuit qui gère la majorité des ressources de l'ordinateur (interface du processeur, mémoire cache et vive, slots d'extension, etc.) ;
- l'horloge système et la pile CMOS, qui maintient les paramètres matériels même lorsque l'ordinateur est éteint ;
- le BIOS (ou UEFI sur les systèmes récents), programme de démarrage qui initialise les composants.

Une carte mère peut être définie selon plusieurs critères :

- son format physique (ou facteur de forme) : ATX, Micro-ATX, Mini-ITX, etc. ;
- son chipset, qui détermine les fonctionnalités et la compatibilité avec les processeurs, la RAM, les ports… ;
- le type de socket processeur qu'elle supporte, qui doit être compatible avec le modèle de processeur utilisé.

7.3.2 Facteur d'encombrement ou facteur de forme d'une carte-mère

On entend généralement par facteur d'encombrement, la géométrie et les dimensions de la carte-mère. Afin de fournir des cartes-mères pouvant s'adapter dans différents boîtiers de marques différentes, des standards ont été mis au point.

Format	Dimensions (mm)	Taille relative	Usage typique
HPTX	345 X 381	Exceptionnelle	Stations de calcul ou serveurs haute densité
XL-ATX	345 X 262	Très grande	Gaming extrême, multi-GPU
E-ATX	305 X 272 à 330	Très grande	Stations de travail, serveurs, PC haut de gamme
ATX	305 X 244	Grande	PC standard, gaming, bureautique
Mini-ATX	284 X 208	Moyenne / compacte	Mini-PC, systèmes embarqués
Micro-ATX	244 X 244	Moyenne	PC compacts, bureautique, gaming modéré
FlexATX	229 X 191	Petite	PC très compacts, usage spécifique
Mini-ITX	170 X 170	Très compacte	Mini-PC, Home Cinéma, systèmes compacts
Nano-ITX	120 X 120	Ultra-compacte	Systèmes embarqués, IoT (cf. ci-après)
Pico-ITX	100 X 72	Ultra-compacte	IoT, robotique, applications spécifiques

L'Internet des objets (ou IoT pour Internet of Things) désigne un réseau constitué d'objets physiques connectés à Internet, équipés de capteurs et de technologies leur permettant d'échanger des données entre eux et avec d'autres systèmes. Ces objets peuvent être des appareils du quotidien (comme des montres connectées, des électroménagers, des voitures intelligentes) ou des dispositifs industriels. L'objectif est que ces objets collectent des informations, les transmettent, et parfois agissent automatiquement selon les données reçues, souvent avec peu ou pas d'intervention humaine. Cette interconnexion permet notamment d'automatiser des processus, d'améliorer la gestion et de générer des analyses utiles dans divers domaines comme la maison intelligente, la santé, ou l'industrie.

7.3.3 Le chipset

7.3.3.1 Le chipset : chef d'orchestre de la carte mère

Le chipset (ou jeu de composants) est un ensemble de circuits intégrés, généralement conçu par Intel ou AMD, chargé de coordonner les échanges de données entre les principaux composants de l'ordinateur : processeur, mémoire, cartes d'extension, périphériques de stockage, ports USB, etc.

Il est intégré à la carte mère et joue un rôle central dans le fonctionnement du système.
C'est lui qui gère :

- les communications entre le CPU, la RAM, les cartes PCIe (graphique, son, réseau…),
- les interfaces de stockage (SATA (Serial Advanced Technology Attachment), NVMe),
- les ports USB, Ethernet, audio, etc.

Le stockage NVMe (Non-Volatile Memory Express) est un protocole et une interface de communication spécialement conçus pour les disques SSD utilisant la mémoire flash. NVMe tire parti du bus PCI Express (PCIe), qui offre des voies de transfert très rapides entre le stockage et le processeur, permettant un accès aux données beaucoup plus rapide qu'avec les anciens protocoles comme SATA ou SAS.

Certains chipsets intègrent également des fonctions graphiques ou audio. Ces composants intégrés peuvent suffire pour des usages courants (bureautique, vidéo, navigation), mais restent limités pour les applications exigeantes (jeux, montage vidéo, DAO). Si besoin, on peut désactiver ces fonctions dans le BIOS/UEFI pour utiliser des cartes d'extension dédiées.

7.3.3.2 Pourquoi le chipset est-il un critère de choix ?

Le chipset détermine les capacités et les limites de la carte mère :

- le nombre et le type de ports disponibles (USB 3.x, SATA, etc.) ;
- la compatibilité avec certains modèles de processeurs et de RAM ;
- les fonctions avancées (par ex. : support de l'overclocking) ;
- la version des lignes PCI Express (3.0, 4.0, 5.0).

Il est donc essentiel de choisir une carte mère avec un chipset adapté à ses besoins actuels et futurs, notamment pour garantir une bonne évolutivité du système.

Pour les processeurs Intel :

Chipset	Gamme	Overclocking	Fonctions principales
H610	Entrée de gamme	Non	Fonctionnalités minimales, peu de ports, RAM limitée, 1 slot M.2 max[12], PCIe 3/4
B660	Milieu de gamme	Non	PCIe 4.0, plus de ports, support RAM plus rapide, bon rapport qualité/prix
B760	Milieu de gamme	Non	Similaire au B660, meilleure gestion USB, compatibilité élargie Raptor Lake
H670	Milieu/haut	Non	Plus de lignes PCIe, plus de ports SATA/USB, mais toujours sans overclocking
Z690	Haut de gamme	Oui	Overclocking CPU et RAM, PCIe 5.0 (x16), connectique riche, jusqu'à 28 lignes PCIe
Z790	Haut de gamme	Oui	Évolution du Z690, plus de ports USB 3.2/4.0, meilleure gestion du PCIe 5.0

[12] Branchement d'un seul SSD VMe.

Pour les processeurs AMD :

Chipset	Plateforme (socket…)	Gamme	Over-clocking	PCIe max	RAM	Fonctions principales
A520	AM4	Entrée de gamme	Non	PCIe 3.0 / 4.0 (CPU)	DDR4	Fonctionnalités minimales, pas d'OC, peu de ports
B550	AM4	Milieu de gamme	Oui	PCIe 4.0 (CPU)	DDR4	Overclocking CPU/RAM, support GPU et SSD PCIe 4.0, bon compromis
X570	AM4	Haut de gamme	Oui	PCIe 4.0	DDR4	PCIe 4.0 généralisé, nombreuses lignes, ports USB/SATA, refroidissement actif
A620	AM5	Entrée de gamme	Non	PCIe 4.0	DDR5	Pas d'OC CPU, PCIe 4.0 seulement, peu d'options d'extension
B650	AM5	Milieu de gamme	Oui	PCIe 5.0 (SSD)	DDR5	Overclocking CPU/RAM, PCIe 5.0 pour SSD, support Ryzen 7000
X670	AM5	Haut de gamme	Oui	PCIe 5.0 (SSD)	DDR5	Plus de connectique, PCIe 5.0 sur SSD, options avancées
X670E	AM5	Haut de gamme	Oui	PCIe 5.0 (GPU + SSD)	DDR5	PCIe 5.0 généralisé, overclocking avancé, très riche connectique
X870	AM5	Haut de gamme	Oui	PCIe 5.0 + USB4	DDR5	Prévu pour Ryzen 9000, PCIe 5.0 généralisé, support USB4 natif, DDR5 optimisée

7.3.4 L'horloge et la pile du CMOS

L'horloge temps réel (RTC – Real-Time Clock) est un petit circuit intégré situé sur la carte mère. Il est chargé de conserver l'heure et la date, même lorsque l'ordinateur est éteint ou déconnecté de l'alimentation électrique.

Ce circuit est alimenté en permanence par une pile bouton (souvent de type CR2032), également placée sur la carte mère. Si cette pile est déchargée, l'ordinateur perdra l'heure entre deux redémarrages, ou affichera une date incorrecte.

L'ancienne mémoire qui stockait les paramètres du BIOS, comme l'ordre de démarrage ou les informations sur les disques, était appelée à tort « mémoire CMOS », car elle utilisait la technologie CMOS (Complementary Metal-Oxide Semiconductor). Aujourd'hui, ces paramètres sont intégrés dans la mémoire du firmware UEFI, mais le terme « pile CMOS » reste couramment utilisé.

À ne pas confondre :

- La RTC (Real-Time Clock) conserve uniquement l'heure et la date.
- La fréquence du processeur (exprimée en MHz ou GHz) est liée à une autre horloge, qui cadence l'exécution des instructions via un cristal différent. Cet oscillateur est alimenté par la carte mère, pas par la pile bouton.

Symptômes de pile déchargée
Si l'ordinateur perd l'heure ou affiche une erreur au démarrage (comme « CMOS checksum error »), il est probablement temps de remplacer la pile de la carte mère.

7.3.5 Le BIOS et l'interface de configuration

Le BIOS (Basic Input/Output System) est un programme de bas niveau qui assure la liaison entre le matériel de l'ordinateur et le système d'exploitation. Il est stocké dans une mémoire ROM (mémoire morte, c'est-à-dire non effacée à l'extinction).

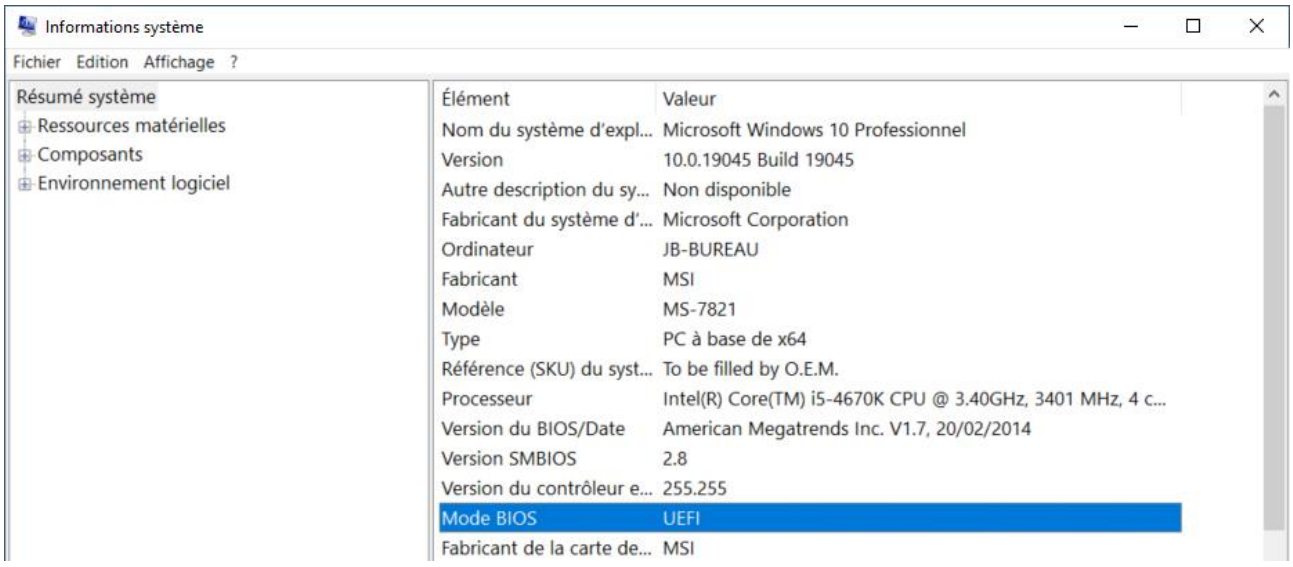
Lors du démarrage de l'ordinateur, le BIOS :

- effectue un test du matériel (POST — Power-On Self Test),
- initialise les composants essentiels (processeur, mémoire, clavier…),
- charge le système d'exploitation à partir du support de démarrage.

Les paramètres de configuration du BIOS (ordre de démarrage, gestion de l'horloge, ports activés, etc.) sont stockés dans une petite mémoire complémentaire, historiquement appelée mémoire CMOS, alimentée par une pile.

Il est possible de modifier ces paramètres en accédant à une interface appelée BIOS Setup (ou UEFI Setup sur les systèmes récents). Cette interface s'ouvre au démarrage de l'ordinateur, généralement en appuyant sur la touche Suppr (Del), F2, ou une autre touche spécifique selon le fabricant.

À noter : Sur les ordinateurs modernes, le BIOS classique est souvent remplacé par un firmware UEFI, plus puissant, avec une interface graphique et des fonctions avancées (support des disques GPT, démarrage sécurisé…).
Un disque GPT est un disque dur ou un SSD qui utilise le GUID Partition Table (GPT) comme table de partitionnement. Il s'agit d'un format moderne, plus puissant que l'ancien format MBR (Master Boot Record), et utilisé aujourd'hui sur la plupart des ordinateurs récents, notamment ceux équipés de firmware UEFI.



Pour plus d'informations sur les options de votre BIOS/UEFI, référez-vous au manuel de votre carte mère.

7.3.6 Le processeur : cœur de l'ordinateur

Le processeur (ou microprocesseur) est le composant central de l'ordinateur. Il exécute les instructions des programmes à partir d'un jeu d'instructions propre à son architecture.

Il est caractérisé par :

- sa fréquence d'horloge, exprimée en GHz, qui correspond au nombre de cycles qu'il effectue chaque seconde. Par exemple, un processeur cadencé à 3,5 GHz peut effectuer 3,5 milliards de cycles par seconde (même si cela ne correspond pas nécessairement à 3,5 milliards d'instructions).
- son nombre de cœurs, qui permet de traiter plusieurs tâches en parallèle.

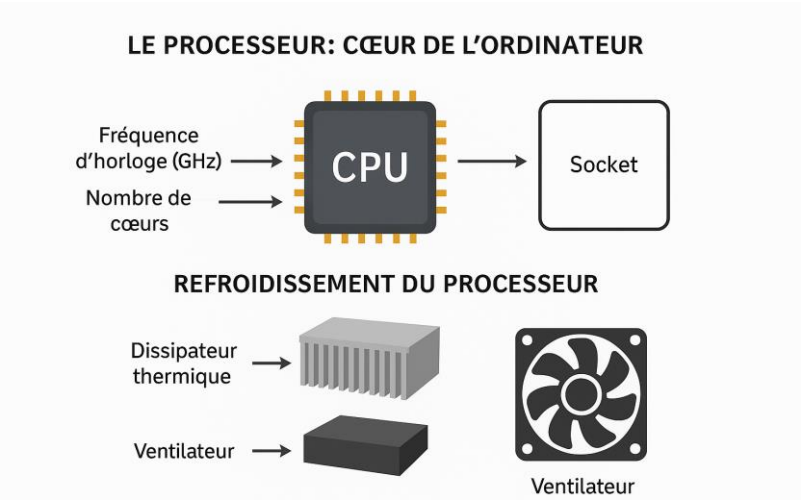
Le processeur est installé sur la carte mère, dans un socket : un connecteur spécifique adapté à chaque type de processeur (par exemple, AM5 chez AMD ou LGA1700 chez Intel).

Remarque : les anciens connecteurs de type slot (où le processeur était inséré verticalement) ne sont plus utilisés aujourd'hui.

Refroidissement du processeur

Le processeur génère de la chaleur lorsqu'il fonctionne. Pour éviter toute surchauffe, il est nécessaire de dissiper cette chaleur :

- Un dissipateur thermique est placé directement sur le processeur. Il est composé d'une base métallique (généralement en cuivre ou aluminium) et d'ailettes qui augmentent la surface d'échange thermique.
- Un ventilateur est souvent fixé au-dessus du dissipateur pour améliorer l'évacuation de la chaleur.
- Des ventilateurs de boîtier assurent la circulation de l'air : l'un extrait l'air chaud, l'autre fait entrer de l'air frais. Un bon flux d'air dans le boîtier est essentiel à la stabilité de l'ensemble du système.



Des explications plus détaillées concernant le processeur figurent dans le point 7.4 (page 85).

7.3.7 La mémoire-cache : une mémoire ultra-rapide intégrée au processeur

La mémoire cache est une mémoire très rapide intégrée directement dans le processeur. Elle sert à accélérer l'accès aux données et instructions que le processeur utilise fréquemment, afin de ne pas devoir les recharger à chaque fois depuis la mémoire RAM (plus lente).

Contrairement à une idée répandue, la mémoire cache ne stocke pas les opérations déjà effectuées, mais plutôt :

- des instructions récemment exécutées,
- des données récemment traitées ou souvent demandées.

Elle fonctionne un peu comme un bloc-notes temporaire du processeur, où ce dernier garde ce dont il pourrait avoir immédiatement besoin.

Organisation en niveaux :

La mémoire cache est souvent divisée en plusieurs niveaux :

- L1 : très rapide, mais de petite taille (quelques dizaines de Ko par cœur),
- L2 : plus grande (512 Ko à 1 Mo), un peu plus lente,
- L3 : partagée entre plusieurs cœurs, avec une capacité allant de 4 Mo à plus de 64 Mo selon les modèles.

Exemple : un processeur moderne comme un AMD Ryzen ou un Intel Core peut intégrer plusieurs dizaines de Mo de cache L3.

Grâce à cette hiérarchie, le processeur peut accéder aux informations plus rapidement qu'en passant par la RAM, ce qui améliore considérablement les performances globales du système.

7.3.8 La mémoire vive

7.3.8.1 Rôle de la mémoire vive (RAM)

La mémoire vive, ou RAM (Random Access Memory), est une mémoire temporaire utilisée pour stocker les données et instructions en cours d'utilisation par le processeur.

Elle est composée de cellules électroniques capables de stocker des bits (0 ou 1), grâce à des circuits logiques.

Son contenu est effacé dès que l'ordinateur est éteint ou redémarré, contrairement à la mémoire de masse (comme les disques durs ou SSD) qui conserve les données à long terme.

Malgré cela, la RAM est indispensable, car elle est :

- beaucoup plus rapide que les disques de stockage ;
- directement accessible par le processeur, avec des temps d'accès de l'ordre de quelques dizaines de nanosecondes, contre plusieurs microsecondes pour un SSD NVMe, et plusieurs millisecondes pour un disque dur classique (HDD).

Aujourd'hui, la majorité des ordinateurs utilisent de la RAM DDR4 ou DDR5, offrant une bande passante élevée et de faibles latences, idéales pour assurer la fluidité du système.

Génération	Année approx.	Fréquence (MHz)	Bande passante (Go/s)	Tension	Usage courant
DDR (DDR1)	2000	200–400	~1,6–3,2	2,5 V	Obsolète
DDR2	2004	400–1066	~3,2–8,5	1,8 V	Obsolète
DDR3	2007	800–2133	~6,4–17	1,5 V	Encore présente dans des anciens PC
DDR4	2014	2133–3200 (jusqu'à 4800+)	~17–25+	1,2 V	Standard dans de nombreux PC
DDR5	2021	4800–8000+	~38–60+	1,1 V	PC récents, gaming, stations pro
DDR6	2026 ou 2027				

7.3.8.2 Organisation : par mots et adresses

La mémoire ne manipule pas les bits un par un, mais par mots.

Un mot est un ensemble de bits, dont la taille dépend de l'architecture du processeur :
8 bits (1 octet), 16 bits, 32 bits, jusqu'à 64 bits.

Exemple : un processeur 64 bits lit et écrit généralement 64 bits à la fois, soit 8 octets.

La mémoire est organisée comme une suite de cellules, chaque cellule :

- stocke un mot,
- possède une adresse unique,
  est accessible directement, pas de manière séquentielle : c'est pourquoi on parle de mémoire à accès aléatoire (Random Access Memory, ou RAM).

7.3.8.3 Comment fonctionne une cellule mémoire ?

Ancien modèle (DRAM - Dynamic RAM)

Chaque bit est stocké dans une cellule composée :

- d'un transistor,
- d'un condensateur chargé (1) ou déchargé (0).

Le condensateur doit être régulièrement rafraîchi pour ne pas perdre sa charge : c'est pourquoi on parle de mémoire dynamique.

Autres modèles

- SRAM (Static RAM) : utilisée dans les caches, elle repose sur des bascule logiques (flip-flop), plus rapides, mais plus coûteuses.
- Mémoire flash : utilisée dans les SSD, elle conserve les données sans alimentation. Des électrons sont piégés dans la grille flottante d'un transistor. La présence ou l'absence de ces électrons, détectée par le passage d'un courant, représente un bit binaire (0 ou 1).

Des explications plus détaillées concernant les mémoires figurent dans le point 7.5 (page 88).

7.3.9 Le bus système et ses évolutions

Un bus est un ensemble de lignes de conducteurs (fils ou pistes sur une carte mère) qui permettent à plusieurs composants électroniques de communiquer entre eux. Il s'agit d'un canal de transmission partagé, utilisé pour transférer des données, des adresses et des signaux de contrôle entre le processeur, la mémoire et les périphériques.

Historiquement, on parlait de bus système ou de Front Side Bus (FSB). Ce terme tend à disparaître avec les architectures récentes, qui privilégient désormais des liaisons point-à-point[13] (comme Intel QuickPath Interconnect - QPI ou AMD Infinity Fabric), offrant des débits plus élevés.

Le bus système est généralement composé de trois sous-ensembles :

Le bus d'adressage (Address Bus)

Il permet au processeur de spécifier l'adresse mémoire ou l'adresse d'un périphérique d'entrée/sortie à laquelle il souhaite accéder.

Il est unidirectionnel, du processeur vers les autres composants.

Chaque ligne transmet un bit. Un bus de n lignes permet d'adresser 2^n cellules mémoire.

Par exemple, un bus d'adressage de 32 bits permet d'adresser 4 294 967 296 emplacements (4 Go).

Le bus de données (Data Bus)

Il assure le transfert des données entre le processeur, la mémoire et les périphériques.

[13] Une liaison point à point, par opposition à un bus multiplexé, est une communication directe et dédiée entre deux composants ou dispositifs seulement, créant une connexion privée sans partager la ligne avec d'autres appareils. Contrairement à un bus où un émetteur envoie des données à plusieurs récepteurs potentiels, une liaison point à point relie deux entités spécifiques pour une communication ciblée et souvent plus performante.

Il est bidirectionnel : le CPU peut lire ou écrire.

Sa largeur (nombre de bits transmis simultanément) détermine la quantité de données échangées à chaque cycle.

Sa fréquence détermine la vitesse du transfert.

Analogie : comme une autoroute, plus il y a de voies (bits), plus le flux est rapide. Un bus de 64 bits transfère 8 octets à la fois, contre 1 seul pour un bus de 8 bits.

Le bus de contrôle (Control Bus)

Il transporte des signaux de commande et de synchronisation.

Ces signaux permettent de :

- préciser l'opération souhaitée (lecture, écriture),
- gérer les interruptions (demande d'attention par un périphérique),
- signaler des états (prêt, erreur, validation, etc.).

Évolution des bus

- Les premiers microprocesseurs (ex. Intel 8080) utilisaient des bus de 8 bits.
- Les processeurs modernes (x86-64, ARMv8, etc.) exploitent des architectures 64 bits et utilisent des bus de données et d'adressage plus larges.
- Le FSB (Front Side Bus) a été progressivement remplacé par des interfaces dédiées haut débit (ex. DMI, QPI, Infinity Fabric), favorisant des communications point-à-point plutôt que des bus partagés.

7.3.10 Les connecteurs d'extension (slots)

Les connecteurs d'extension (ou slots) sont des emplacements sur la carte mère permettant d'installer des cartes d'extension, c'est-à-dire des composants qui ajoutent ou améliorent les fonctionnalités de l'ordinateur.

Le standard actuel est le PCI Express (PCIe). Il remplace tous les anciens connecteurs comme ISA, AGP ou PCI classique.

Il existe plusieurs formats de slots PCIe, en fonction du nombre de lignes utilisées :

- PCIe x1 : petites cartes (réseau, son, USB, Wi-Fi) ;
- PCIe x4 ou x8 : cartes spécialisées (RAID, capture vidéo, etc.) ;
- PCIe x16 : utilisé principalement pour les cartes graphiques.

Les versions les plus courantes sont :

- PCIe 3.0 : encore largement utilisé,
- PCIe 4.0 et 5.0 : dans les PC récents, avec des vitesses plus élevées,
- PCIe 6.0 : en cours d'adoption pour les stations professionnelles.

Ces slots permettent une évolution facile du PC : ajout de ports, amélioration graphique, accélération réseau, etc.

Version PCIe	Débit par ligne (x1)	Débit total (x16)	Année approxima-tive	Usage typique
PCIe 1.0	250 Mo/s	~4 Go/s	2003	Anciennes cartes
PCIe 2.0	500 Mo/s	~8 Go/s	2007	Débit doublé, cartes graphiques PCIe 2
PCIe 3.0	~1 Go/s	~16 Go/s	2010	Standard courant (Intel 10e gen…)

PCIe 4.0	~2 Go/s	~32 Go/s	2017	SSD NVMe rapides, GPU récents
PCIe 5.0	~4 Go/s	~64 Go/s	2021	Cartes graphiques haut de gamme
PCIe 6.0	~8 Go/s	~128 Go/s	2023–2024	Serveurs, stations IA, futur PC haut de gamme

7.4 Le processeur : le cerveau de l'ordinateur

7.4.1 Introduction

Le processeur, aussi appelé CPU (Central Processing Unit) ou microprocesseur, est l'un des composants les plus importants — et souvent les plus coûteux — d'un ordinateur.

Sa puissance de traitement dépend de plusieurs facteurs :

- **Le nombre de cœurs (cores) :** un cœur exécute une tâche à la fois. Plus il y a de cœurs, plus le processeur peut traiter de tâches en parallèle ;
- **La taille de la mémoire cache :** elle permet d'accéder rapidement aux données fréquemment utilisées ;
- **L'architecture :** chaque génération de processeur (ex. : Intel Alder Lake, AMD Zen 4) apporte des optimisations internes ;
- **La fréquence d'horloge**, exprimée en GHz, qui indique combien de cycles d'instructions le processeur peut effectuer par seconde.

Si la fréquence est un indicateur de rapidité, elle ne suffit pas à juger la performance : un processeur récent à 3,5 GHz est souvent plus performant qu'un ancien modèle à 4 GHz.

Pour rappel, la loi de Moore énoncée en 1965 par Gordon Moore (PDG d'Intel) n'est plus d'actualité depuis environ 2005. Celle-ci indiquait que tous les 18 mois, le nombre de transistors par processeur doublait.

7.4.2 Principe de fonctionnement du CPU

Rappelons quelques notions importantes.

Le fonctionnement du processeur repose sur un cycle répétitif : chercher, décoder et exécuter les instructions.

1) Le compteur ordinal (Program Counter, PC) contient l'adresse de la prochaine instruction à exécuter. Cette adresse est placée sur le bus d'adressage.
2) Le bus de contrôle émet alors un signal de lecture vers la mémoire.
3) La mémoire, sélectionnée par l'adresse envoyée, envoie le code binaire de l'instruction via le bus de données.
4) Ce code est chargé dans le registre d'instruction, puis transmis à l'unité de commande.
5) L'unité de commande décodifie l'instruction et génère une séquence de signaux de commande, internes et externes, qui permettent l'exécution coordonnée de l'instruction.
6) Chaque instruction est composée de plusieurs micro-opérations, cadencées par les tops d'horloge du système.

Ce processus est appelé cycle d'instruction. Il se répète indéfiniment tant que le processeur est alimenté.

7.4.3 Parallélisation dans les processeurs modernes

Dans un processeur classique, chaque instruction suit un cycle d'exécution en plusieurs étapes : lecture, décodage, exécution, accès mémoire, etc. Chaque étape prend généralement un cycle d'horloge, si bien qu'une instruction complète peut nécessiter plusieurs cycles pour être totalement exécutée.

Pour améliorer les performances sans augmenter indéfiniment la fréquence d'horloge (ce qui pose des limites en matière de dissipation thermique et de consommation énergétique), les fabricants de processeurs ont mis en œuvre différentes techniques de parallélisation.

Comme détaillé ci-après, on distingue plusieurs formes de parallélisme.

7.4.3.1 Pipeline et architecture superscalaire

Le pipeline est un mécanisme qui divise l'exécution d'une instruction en étapes successives (lecture, décodage, exécution, etc.), permettant de traiter plusieurs instructions simultanément, chacune se trouvant à une étape différente. Les processeurs modernes vont plus loin avec l'architecture superscalaire, qui intègre plusieurs pipelines parallèles : ils peuvent ainsi décoder et exécuter plusieurs instructions indépendantes au cours d'un même cycle, à condition que les dépendances soient correctement gérées. Par exemple, un processeur « 4-wide » peut exécuter jusqu'à quatre instructions par cycle si les conditions sont réunies.

7.4.3.2 Architecture multi-cœurs

Un même processeur peut contenir plusieurs cœurs physiques indépendants. Chaque cœur exécute sa propre séquence d'instructions, ce qui permet un véritable traitement parallèle de plusieurs programmes ou tâches.

7.4.3.3 Multithreading (ex. : Hyper-Threading)

Chaque cœur de processeur peut gérer plusieurs flux d'exécution (threads) simultanément. Cela améliore l'utilisation des ressources internes en masquant les temps d'attente (par exemple lors d'un accès mémoire). Plusieurs threads partagent alors les mêmes unités d'exécution.

7.4.3.4 Parallélisme de données (SIMD, AVX, NEON...)

Une même instruction peut être appliquée simultanément à plusieurs données. Ce mode de traitement est particulièrement efficace pour des calculs répétitifs sur de grands ensembles de données (images, sons, simulations scientifiques, etc.).

Exemple : additionner en parallèle 4 paires de nombres avec une seule instruction SIMD.
Une instruction SIMD (Single Instruction, Multiple Data) est une instruction qui permet d'exécuter la même opération sur plusieurs données en même temps. Cela signifie qu'une seule commande du processeur agit simultanément sur un ensemble de données regroupées dans ce qu'on appelle un vecteur. Cette approche permet une forme de parallélisme au niveau des données, améliorant significativement les performances pour certains types de calculs.

La mémoire cache : un rôle stratégique

La mémoire cache est essentielle pour combler l'écart de vitesse entre le processeur et la mémoire RAM. Ce décalage reste important, même avec les mémoires DDR5 modernes.

- Plus la fréquence du CPU est élevée, plus le recours à une cache rapide et bien dimensionnée est crucial.
- Les processeurs modernes embarquent plusieurs niveaux de cache (L1, L2, L3), avec des tailles allant jusqu'à plusieurs dizaines de Mo.

La taille du cache explique souvent la différence de prix entre deux processeurs d'une même gamme.

Fréquences et limite thermique

L'augmentation des fréquences d'horloge a atteint une limite physique autour de 4 à 5 GHz, en raison :

- de la dissipation thermique,
- de la consommation énergétique croissante.

7.4.4 L'architecture CISC

L'architecture CISC (Complex Instruction Set Computer) est historiquement celle des processeurs x86 et x86-64 (Intel, AMD, etc.). Elle se caractérise par un jeu d'instructions riche et complexe, permettant d'effectuer des opérations sophistiquées en une seule instruction.

Dans les processeurs modernes, ces instructions sont souvent décodées en micro-instructions plus simples, exécutées par une architecture interne proche d'un cœur RISC, combinée à des mécanismes superscalaires.

Cela permet au processeur d'exécuter plusieurs instructions simultanément, en optimisant l'utilisation des pipelines internes.

L'architecture CISC facilite la programmation à bas niveau, mais au prix d'un décodage plus complexe, d'où la nécessité de mécanismes sophistiqués pour maintenir de hautes performances.

Aujourd'hui, l'architecture CISC reste dominante sur les PC de bureau et portables, mais elle est concurrencée dans certains domaines (notamment les appareils mobiles et embarqués) par les architectures RISC, comme ARM.

7.4.5 L'architecture RISC

L'architecture RISC (Reduced Instruction Set Computer) repose sur l'utilisation d'un jeu d'instructions limité, composé d'opérations simples, exécutées en un seul cycle d'horloge.

Contrairement aux architectures CISC, les instructions RISC sont toutes de longueur fixe, ce qui simplifie leur décodage et leur traitement dans les pipelines. Ce modèle favorise une architecture interne plus régulière, et permet une exécution plus prévisible.

Cette simplicité se traduit par une conception moins complexe, une meilleure efficacité énergétique et une excellente évolutivité, ce qui explique l'adoption massive des architectures RISC dans les systèmes embarqués, les smartphones, les tablettes, et plus récemment, les ordinateurs personnels (ex. : Apple M1/M2/M3 basés sur ARM).

Les performances sont optimisées grâce à des compilateurs puissants, capables de générer efficacement les séquences d'instructions nécessaires, même pour des opérations complexes.

Enfin, les architectures RISC modernes intègrent également des mécanismes avancés de traitement parallèle (pipelining, exécution out-of-order, prédiction de branchement, etc.), comme les architectures CISC.

NB :
L'exécution out-of-order (dans le désordre) est une technique avancée utilisée par les processeurs modernes pour optimiser leur performance. Au lieu d'exécuter les instructions dans l'ordre strict où elles apparaissent dans le code du programme (exécution in-order), le processeur les réorganise de manière dynamique.

Le principe est le suivant : si une instruction ne peut pas être exécutée immédiatement (par exemple, parce qu'elle attend les données d'une autre instruction qui n'a pas encore terminé son travail), le processeur ne reste pas inactif. Il va chercher dans les instructions suivantes celles qui sont prêtes à être exécutées, c'est-à-dire qui n'ont aucune dépendance en attente.

7.4.6 CISC ou RISC?

Pendant longtemps, les architectures CISC (comme x86) dominaient le marché des ordinateurs personnels, notamment grâce à leur compatibilité historique et à leur écosystème logiciel établi.

Toutefois, les architectures RISC, longtemps cantonnées aux systèmes embarqués, sont aujourd'hui largement répandues et connaissent une forte croissance, notamment grâce à l'essor de l'architecture ARM. Celle-ci équipe la plupart des smartphones, des tablettes, et de nombreux ordinateurs récents (notamment ceux d'Apple).

L'argument selon lequel un processeur RISC nécessiterait plus de mémoire pour exécuter un programme n'a plus vraiment de pertinence aujourd'hui. Les avancées technologiques en matière de mémoire, d'optimisation logicielle et de compilateurs ont effacé ces différences. Les processeurs RISC sont aujourd'hui reconnus pour leur efficacité énergétique, leur performances élevées, et leur évolutivité, ce qui les rend particulièrement adaptés aux systèmes modernes.

7.5 Les mémoires

7.5.1 Rôle des mémoires

Le traitement de données par un ordinateur nécessite plusieurs types de mémoire, chacun ayant des caractéristiques spécifiques en termes de vitesse, de capacité, de coût, de volatilité et de fonction.

On distingue principalement :

- La mémoire centrale (RAM), qui sert de zone de travail temporaire : les programmes et les données à traiter doivent y être chargés pour pouvoir être exploités par le processeur. Son contenu est perdu à l'extinction de l'appareil (mémoire volatile).
- La mémoire de stockage (ou mémoire de masse), qui conserve les données de manière permanente. Elle est assurée aujourd'hui par les disques SSD, HDD, ou encore les mémoires Flash (clés USB, cartes SD). Ces supports ne nécessitent pas d'alimentation pour conserver les données.

Le processeur communique avec la mémoire centrale par trois bus :

- Le bus d'adressage indique l'emplacement mémoire concerné.
- Le bus de données permet le transfert des informations.
- Le bus de contrôle gère les signaux de lecture/écriture.

7.5.2 La ROM

Les mémoires ROM (Read Only Memory) sont destinées à conserver des informations même en l'absence de courant électrique. Elles sont utilisées notamment pour le BIOS/UEFI, les micrologiciels (firmwares) ou certains appareils embarqués.

Voici les principales variantes :

Type	Caractéristiques	Usage
ROM	Programmée une fois pour toutes par le fabricant	Instructions de démarrage
PROM	Programmable une seule fois par l'utilisateur	Micrologiciels personnalisés
EPROM	Effaçable par lumière UV, reprogrammable	Ancienne technologie, aujourd'hui obsolète
EEPROM	Effaçable électriquement, sans retrait du support	Micrologiciels, configuration, BIOS
Flash (ou FEPROM)	Variante moderne de l'EEPROM, plus rapide	Clés USB, cartes SD, SSD

La mémoire Flash est aujourd'hui la technologie la plus répandue pour le stockage non volatil, grâce à sa vitesse d'accès et à sa durabilité.

7.5.3 Les registres

Comme cela a été vu dans le point 6.3.2 (page 61), les registres sont de petites mémoires très rapides, situées à l'intérieur du CPU. Leur accès est quasi instantané, ce qui optimise les performances.

Ils ont des rôles spécifiques selon les architectures, mais on retrouve les catégories suivantes dans la plupart des processeurs.

Type de registre	Rôle principal
Registres généraux	Stockage temporaire de données et de résultats
Accumulateur	Contient les données à traiter ou les résultats intermédiaires
Registres d'adresse	Contiennent les adresses mémoire (ex. : base, index, pointeur de pile)
Registre d'instruction (IR)	Contient l'instruction en cours d'exécution
Compteur ordinal (PC, Program Counter)	Contient l'adresse de la prochaine instruction à exécuter
Registre d'état (flags)	Stocke les indicateurs liés aux résultats (zéro, retenue, signe…) et contrôle les sauts conditionnels
Pointeur de pile (Stack Pointer)	Gère la mémoire utilisée pour les appels de fonctions, les variables locales, etc.

Notes

- Les jeux de registres et la structure exacte varient selon les familles de processeurs (Intel, AMD, ARM, RISC-V…).
- Les processeurs modernes embarquent aussi :
  - des unités de prédiction de branchement (cf. ci-après),
  - des buffers de pipeline,
  - et des unités de chargement/préchargement mémoire (load/store units).

- Prédiction de branchement

  Lorsqu'un processeur exécute un programme, il rencontre souvent des instructions de branchement (par exemple if…then…else ou une boucle).
  À ce moment, il ne sait pas encore quelle suite d'instructions devra être exécutée (la condition peut être vraie ou fausse).
  Or, avec le pipeline, il faut déjà commencer à lire et préparer les instructions suivantes avant de connaître le résultat.

  Pour éviter de perdre du temps, les processeurs modernes disposent d'unités de prédiction de branchement :
  - elles anticipent quelle voie sera suivie (branche prise ou non prise) ;
  - si la prédiction est correcte, le pipeline continue sans interruption ;
  - si la prédiction est fausse, il faut « vider » le pipeline et recharger les bonnes instructions, ce qui provoque un ralentissement (appelé misprediction penalty).

  Exemple concret :
  Dans une boucle for, le processeur peut prédire que la condition sera encore vraie et préparer le prochain tour. Comme les boucles se répètent souvent, la prédiction est très efficace dans ce cas.

7.5.4 La RAM

La mémoire vive, ou RAM (Random Access Memory), est une mémoire rapide, utilisée pour stocker les données et programmes en cours d'exécution. Contrairement à la mémoire de stockage, elle est volatile : son contenu est perdu dès que l'ordinateur est éteint.

Le terme « accès aléatoire » signifie qu'il est possible d'accéder directement à n'importe quelle case mémoire, à la différence des mémoires séquentielles comme les bandes magnétiques ou les cartes perforées utilisées dans les débuts de l'informatique, où il fallait faire défiler les données pour atteindre une information donnée.

Rôle

La RAM contient :

- les instructions du système d'exploitation et des applications en cours d'exécution ;
- les données manipulées par ces applications.

Quand la RAM est insuffisante, le système d'exploitation utilise une partie du disque dur ou SSD comme mémoire virtuelle, en y stockant temporairement les données les moins sollicitées. Ce mécanisme, bien que fonctionnel, ralentit fortement les performances.

Deux types de RAM : SRAM et DRAM

SRAM (Static RAM)

- Structure : composée de bascules électroniques (transistors, cf. ci-après).
- Avantage : très rapide, ne nécessite pas de rafraîchissement tant qu'elle est alimentée.
- Usage principal : utilisée dans les mémoires cache (L1, L2, L3) du processeur.
- Remarque : certaines anciennes cartes-mères utilisaient de la SRAM alimentée par pile pour stocker la configuration du BIOS/UEFI. Cette pratique a été en grande partie remplacée par l'usage de mémoires Flash.

- Note concernant la notion de bascule :

  Une bascule (en anglais flip-flop) est un petit circuit électronique qui peut mémoriser un seul bit d'information : soit un 0, soit un 1.

  Pour faire simple :

  Imaginons un interrupteur bistable :
  - dans une position, il reste sur 0,
  - dans l'autre, il reste sur 1.

  Analogie : Stylo à bille à clic :
  Quand on appuie une fois, la pointe sort et reste sortie.
  Quand on appuie à nouveau, la pointe rentre et reste rentrée.

  Deux positions stables, qui ne changent pas toutes seules.

  En résumé :
  - Une bascule = la plus petite « case mémoire » possible.
  - En reliant plusieurs bascules, on peut stocker plusieurs bits (donc un mot, un nombre…).
  - Dans la SRAM, chaque cellule mémoire utilise une petite bascule électronique (faite de transistors) pour garder la donnée.

DRAM (Dynamic RAM)

- Structure : chaque bit est stocké dans un condensateur qui se décharge avec le temps.
- Inconvénient : nécessite un rafraîchissement périodique pour conserver les données.

- Avantage : moins chère et plus compacte que la SRAM.
- Usage principal : constitue la majorité de la mémoire vive des ordinateurs.
- Évolution actuelle : on utilise aujourd'hui des DRAM de type DDR4 ou DDR5, qui offrent des vitesses et une bande passante bien supérieures aux générations précédentes.

7.5.5 Les temps d'accès et cycle mémoire

Les commandes de lecture ou d'écriture en mémoire ne s'exécutent jamais instantanément. Deux paramètres sont essentiels pour évaluer les performances d'un module mémoire :

- Le temps d'accès : il s'agit du délai minimum entre le moment où une commande est envoyée et celui où la donnée est effectivement disponible (en lecture ou en écriture).
- Le cycle mémoire (ou temps de cycle) : il représente l'intervalle minimal entre deux accès consécutifs à la mémoire. Ce temps est souvent supérieur au simple temps d'accès, car il inclut toutes les opérations internes nécessaires à l'exécution complète de la commande précédente avant d'en lancer une nouvelle.

Différences selon le type de mémoire

- Les mémoires statiques (SRAM) ont des temps d'accès très courts (quelques nanosecondes). Elles peuvent suivre les hautes fréquences du processeur, ce qui en fait les meilleures candidates pour les mémoires « cache » (L1, L2, L3).
- Les mémoires dynamiques (DRAM), en revanche, sont plus lentes. Leur structure matricielle impose un adressage en deux temps (sélection des lignes puis des colonnes), ce qui engendre un temps de latence plus important.

Exemple concret
Le temps d'un cycle processeur est égal à l'inverse de sa fréquence d'horloge.
Par exemple, un processeur cadencé à 1 GHz (1 000 000 000 Hz) effectue un cycle toutes les :
1 / 1 000 000 000 = 1 nanoseconde (ns)

Optimisation avec le mode rafale (burst mode)
Pour accélérer les transferts, les mémoires modernes (comme la SDRAM, puis la DDR SDRAM, jusqu'à la DDR5) utilisent le mode rafale. Cette technique consiste à profiter du fait que les données sont souvent lues ou écrites de manière contiguë.

Exemple :
Prenons une SDRAM à 133 MHz (fréquence de base d'une DDR-266).
Imaginons que l'on souhaite lire 4 mots consécutifs :

- Le premier accès nécessite une latence initiale importante (par ex. 5 cycles de 7,5 ns = 37,5 ns).
- Les trois accès suivants s'enchaînent beaucoup plus rapidement, à raison d'un seul cycle chacun (7,5 ns par mot).

Résultat :

- 1er accès : 5 cycles
- Accès suivants : 1 + 1 + 1 cycle(s) → Total = 8 cycles pour 4 transferts, soit 2 cycles par mot en moyenne.

Notes

- Les mémoires DDR4 et DDR5 fonctionnent selon des principes similaires, avec des cadences beaucoup plus élevées (jusqu'à 6 400 MHz pour la DDR5).
- Les latences CAS[14] (CL) continuent de jouer un rôle central pour mesurer le délai entre la requête et la disponibilité de la donnée, mais doivent être interprétées en lien avec la fréquence réelle.
- Le mode rafale est systématiquement utilisé dans les architectures mémoire modernes, y compris dans les GPU.

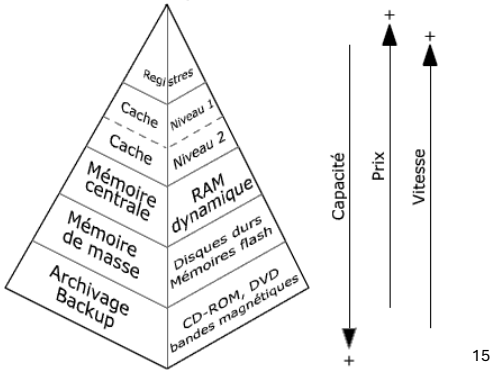
[14] CAS : Column Address Strobe

7.5.6 Hiérarchie des mémoires

Le système informatique utilise différents types de mémoires, qui se distinguent par leur vitesse, leur capacité, leur volatilité, leur coût, leur technologie (électronique, magnétique, optique), leur mode d'accès (direct ou séquentiel), et leur fonction dans le système.

Pour illustrer cette diversité, on utilise souvent une pyramide des mémoires, qui classe ces composants selon un équilibre entre rapidité, proximité du processeur et capacité de stockage :

1. Registres (au sommet de la pyramide)
   - Intégrés au cœur du processeur.
   - Accès ultrarapide (quelques nanosecondes ou moins).
   - Contiennent quelques mots seulement : opérandes, résultats intermédiaires, adresses…
   - Fonctionnent à la même fréquence que le CPU.
2. Mémoire cache (L1, L2, L3)
   - Toujours intégrée au processeur ou très proche.
   - Sert de tampon rapide entre les registres et la RAM.
   - L1 : très rapide, faible capacité (32 à 128 Ko).
   - L2 : plus grande, un peu plus lente (512 Ko à plusieurs Mo).
   - L3 : partagée entre cœurs, encore plus grande (jusqu'à plusieurs dizaines de Mo).
   - Donne un accès anticipé aux instructions et données les plus probables (principe de localité).
3. Mémoire centrale (RAM)
   - Installée sur des barrettes (DDR4, DDR5…).
   - Stocke programmes et données en cours d'exécution.
   - Volatile : son contenu disparaît à l'arrêt de l'ordinateur.
   - Accès plus lent que les caches, mais beaucoup plus grande (généralement 8 à 64 Go aujourd'hui).
   - Fonctionne en coopération avec la mémoire cache grâce au bus système.
4. Mémoire de masse
   - Non volatile, grande capacité (plusieurs To).
   - Sert au stockage permanent des données (système d'exploitation, logiciels, fichiers).
   - Technologies actuelles :
     - SSD (Solid State Drive) : rapide, silencieux, sans pièces mobiles.
     - HDD (Hard Disk Drive) : plus lent.
   - Le système d'exploitation gère également la mémoire virtuelle en utilisant une partie du disque comme extension temporaire de la RAM.
5. Supports de stockage secondaire
   Moins utilisés pour le traitement direct, mais utiles pour l'archivage ou le transfert :
   - Clés USB, cartes mémoire (technologie flash).
   - CD/DVD/Blu-ray : supports optiques, de moins en moins utilisés.
   - Bandes magnétiques : toujours en usage pour les sauvegardes de long terme, notamment en environnement professionnel.



15

[15] Source : https://projet.eu.org

7.5.7 Fonctionnement de la mémoire vive

La mémoire vive (RAM), de type DRAM (Dynamic Random Access Memory), est constituée de millions de cellules mémoire organisées en tableau (matrice) de lignes et de colonnes.

Chaque cellule mémoire contient deux composants essentiels :

- un condensateur qui stocke la charge électrique (état logique 1 ou 0),
- un transistor (de type MOS) qui permet de lire ou modifier cette charge.

Les condensateurs se déchargent rapidement. Il faut donc rafraîchir régulièrement leur contenu (toutes les 15 ns environ), ce qui justifie le nom de mémoire « dynamique ».

Temps d'accès et performances

Pour accéder à une donnée :

1) Le contrôleur mémoire sélectionne une ligne et une colonne dans la matrice.
2) L'information est lue après un temps de latence, suivi du temps de cycle mémoire.

Exemple :

- Une mémoire DDR4/DDR5 actuelle présente un temps d'accès effectif compris entre 60 et 100 ns, en tenant compte de la latence, des délais d'ouverture de ligne et du contrôleur mémoire.
- Sur un processeur cadencé à 3,5 GHz (soit un cycle processeur de ~0,29 ns), cela représente de l'ordre de 200 à 300 cycles d'attente pour accéder à une donnée en RAM, ce qui explique le rôle indispensable des mémoires cache (L1, L2, L3) pour limiter ces pénalités.

Pour améliorer cela :

- On utilise le mode rafale (burst mode) qui permet de lire plusieurs données consécutives plus rapidement après le premier accès.
- Les dernières générations de RAM (ex. DDR5) atteignent en 2025 des vitesses de 9600 MHz, avec des temps d'accès de 8 à 9 ns seulement.

7.5.8 La correction d'erreurs

Sur les systèmes où la fiabilité des données est cruciale (serveurs, applications scientifiques, bases de données critiques), il est essentiel de garantir l'intégrité des informations en mémoire. Deux technologies permettent cela :

- Bit de parité : ajoute un bit supplémentaire à chaque mot mémoire pour vérifier si le nombre de bits à 1 est pair ou impair. Ce mécanisme permet de détecter une erreur (bit inversé), mais pas de la corriger.
- ECC (Error Correction Code) : utilise plusieurs bits de contrôle pour détecter et corriger automatiquement certaines erreurs (par exemple, une erreur sur un seul bit). Légèrement plus coûteuses, ces barrettes sont réservées à des environnements où la sécurité des données prime sur les performances.

7.6 Les périphériques externes

Les ports d'entrée/sortie (ou interfaces E/S) sont les canaux matériels par lesquels l'ordinateur communique avec son environnement. Ils permettent d'échanger des données avec les périphériques connectés, qu'ils soient internes ou externes.

Aujourd'hui, les interfaces E/S les plus courantes sont :

- USB / USB-C : pour connecter claviers, souris, imprimantes, clés USB, disques durs externes, smartphones...
- HDMI / DisplayPort : pour transmettre l'image et le son vers un écran ou un projecteur.
- Ethernet / Wi-Fi : pour la connexion réseau.
- Bluetooth : pour les périphériques sans fil (casques, claviers, souris, enceintes, etc.).
- Lecteurs de cartes SD, ports audio, Thunderbolt, etc.

Les anciens ports série (COM) et parallèle (LPT) ont disparu des PC récents, remplacés par des solutions plus universelles et rapides. Les modems externes sont également obsolètes, remplacés par des connexions réseau intégrées ou via routeurs.

7.6.1 Les ports USB

L'USB (Universal Serial Bus) est une norme d'interface série universelle introduite en 1996, devenue aujourd'hui incontournable pour connecter des périphériques à un ordinateur. Elle permet à la fois l'échange de données et l'alimentation électrique des appareils connectés.

L'USB s'est imposé grâce à plusieurs avantages :

- Une connexion simple, sans redémarrage (branchement à chaud ou hot plug).
- Une alimentation électrique standardisée (jusqu'à 100 mA, puis 500 mA en USB 2.0, et plusieurs ampères en USB-C/PD[16]).
- Une topologie en étoile gérée via des hubs (concentrateurs).

Version USB	Débit théorique max	Date approximative
USB 1.1	1,5 Mbps (low) / 12 Mbps	1998
USB 2.0	480 Mbps	2000
USB 3.0 / 3.1 Gen1	5 Gbps	2008
USB 3.1 Gen2	10 Gbps	2013
USB 3.2	jusqu'à 20 Gbps	2017
USB 4.0 / 4.0 v2	jusqu'à 80/120 Gbps	2019+

L'USB-C est le connecteur réversible qui tend à remplacer les anciens formats (USB-A, USB-B, microUSB). Il est compatible avec de nombreux standards (USB 3.x, Thunderbolt, DisplayPort, Power Delivery…).

Lors de la connexion d'un périphérique USB, le système effectue une phase d'identification (appelée énumération) : il détecte l'appareil, lui attribue une adresse, puis charge le pilote adéquat.

Jusqu'à 127 périphériques peuvent être connectés à un même hôte via des hubs successifs. La longueur maximale d'un câble USB varie selon les versions, allant de 3 à 5 mètres sans amplification.

7.6.2 HDMI / DisplayPort

Ces interfaces permettent de transmettre des flux audio et vidéo numériques de haute qualité.

- HDMI (High Definition Multimedia Interface) est très répandu dans les ordinateurs, téléviseurs, consoles de jeux, projecteurs et autres équipements multimédias. Il prend en charge la vidéo jusqu'à 8K et plusieurs canaux audio.
- DisplayPort, principalement utilisé sur les ordinateurs, offre également des résolutions élevées et une meilleure compatibilité avec les écrans professionnels. Il est aussi utilisé via les connecteurs USB-C grâce à l'alternative DisplayPort Mode.

7.6.3 Ethernet / Wi-Fi

L'Ethernet désigne la connexion filaire à un réseau local (LAN). C'est une interface fiable et rapide, souvent utilisée pour les serveurs, les ordinateurs de bureau ou les situations nécessitant de grandes bandes passantes. Les débits actuels vont de 1 Gbps à 10 Gbps et au-delà.

Le Wi-Fi est une technologie sans fil qui remplace progressivement l'Ethernet dans les environnements mobiles ou domestiques. Les normes Wi-Fi ont évolué vers de très hauts débits : Wi-Fi 5 (802.11ac), Wi-Fi 6 (802.11ax) et Wi-Fi 6E étendent la portée et réduisent la latence. Le Wi-Fi 7 est en cours d'adoption.

[16] PD : Power Delivery

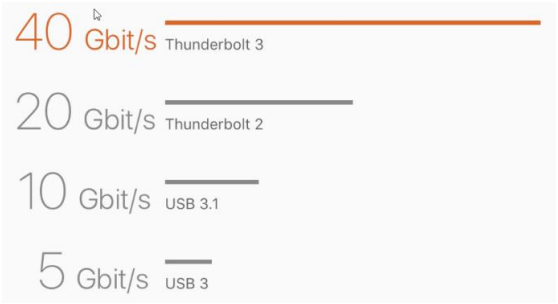
7.6.4 Bluetooth

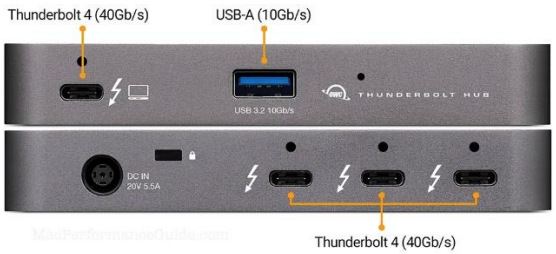
Le Bluetooth est une technologie de communication sans fil à courte portée, très utilisée pour connecter des périphériques personnels comme les claviers, souris, écouteurs, enceintes, ou montres connectées. La version 5.x de Bluetooth permet des transferts plus rapides, une portée étendue, et une consommation d'énergie plus faible.

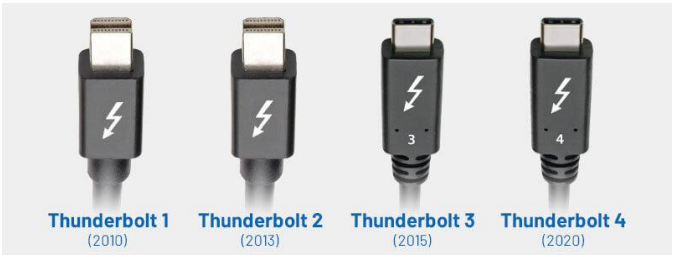
Bluetooth est également utilisé dans le cadre de l'Internet des Objets (IoT), notamment dans les objets domotiques et les dispositifs de santé connectée.

7.6.5 Thunderbolt

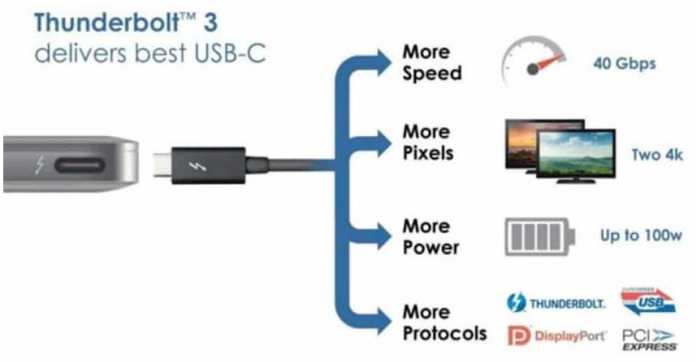
Thunderbolt est une technologie développée par Intel, initialement en partenariat avec Apple. Elle combine plusieurs protocoles (PCI Express, DisplayPort, alimentation, données) sur un même câble via un connecteur USB-C. La version Thunderbolt 4 offre un débit jusqu'à 40 Gbps, permet de chaîner plusieurs appareils et de connecter simultanément des écrans, des SSD externes ultra-rapides et d'autres périphériques.







Cette interface est prisée dans les environnements professionnels nécessitant un haut débit, comme la création vidéo, la conception 3D ou le traitement scientifique.



[17]

7.6.6 Autres interfaces

- Lecteurs de cartes SD : parfois intégrés aux ordinateurs portables ou accessibles via adaptateur, ils permettent de lire les cartes mémoire des appareils photo ou caméras.
- Ports audio (jack 3,5 mm) : ils assurent la connexion des casques, microphones ou haut-parleurs analogiques.
- Ports d'alimentation propriétaires : certains fabricants conservent encore des connecteurs d'alimentation spécifiques pour leurs ordinateurs portables, bien que l'USB-C Power Delivery tende à s'imposer.

[17] Source : https://www.youtips.com/astuce-mac-du-jour-quest-ce-que-le-port-thunderbolt/

7.7 Les dispositifs d'entrée et de sortie

7.7.1 Dispositifs d'entrée

Les dispositifs (ou périphériques) d'entrée permettent à l'utilisateur ou à l'environnement de fournir des données ou des commandes à l'ordinateur.

Exemples courants :

Dispositif	Fonction principale
Clavier	Saisir du texte et des commandes
Souris	Pointer, cliquer, sélectionner
Écran tactile	Entrée par contact direct avec l'écran
Scanner	Numériser des documents ou images
Microphone	Saisir des sons ou des instructions vocales
Webcam	Capturer des images ou vidéos en temps réel
Joystick / Manette	Contrôler des jeux ou des simulations
Lecteur de codes-barres	Lire des informations codées (code produit, etc.)
Capteur biométrique	Reconnaissance d'empreinte ou de visage
Capteur de mouvement	Contrôle par gestes (ex. Kinect, gyroscope)

7.7.2 Dispositifs de sortie

Les dispositifs (ou périphériques) de sortie permettent à l'ordinateur de restituer les résultats d'un traitement à l'utilisateur ou à l'environnement.

Exemples courants :

Dispositif	Fonction principale
Écran / moniteur	Afficher du texte, des images, des vidéos
Imprimante	Produire une copie papier d'un document
Enceintes	Reproduire des sons ou de la musique
Casque audio	Restituer du son de manière individuelle
Vidéoprojecteur	Afficher sur un grand écran
Traceur (plotter)	Imprimer des dessins techniques
Imprimante 3D	Réaliser des objets physiques à partir de fichiers numériques
Retour haptique	Produire des vibrations pour simuler un contact ou un retour physique (sur smartphone, manette…)

7.7.3 Dispositifs d'entrée/sortie (mixtes)

Certains périphériques combinent des fonctions d'entrée et de sortie. On parle alors de périphériques mixtes. Ils permettent à la fois de recevoir des données de l'utilisateur ou de l'environnement, et de transmettre des informations vers ce dernier.

Dispositif	Fonctions
Écran tactile	Entrée par contact tactile et sortie visuelle
Clé USB / disque externe	Lecture (entrée) et écriture (sortie) de données
Modem / carte réseau	Envoi (sortie) et réception (entrée) de données via le réseau
Lecteur de cartes à puce	Lecture et écriture sécurisée de données
Casque audio avec micro	Émission de son (sortie) et capture de voix (entrée)
Imprimante multifonction	Impression (sortie), numérisation et envoi (entrée et sortie)

7.8 Le système d'affichage

Le système d'affichage est l'interface visuelle principale entre l'ordinateur et l'utilisateur. Il repose sur deux éléments fondamentaux : le moniteur (ou écran) et la carte graphique, parfois appelée processeur graphique (GPU).

7.8.1 Le moniteur

Aujourd'hui, la grande majorité des écrans utilisent la technologie LCD à rétroéclairage LED ou OLED, remplaçant les anciens tubes cathodiques (CRT).

Types d'écrans modernes :

- LCD/LED : Écrans à cristaux liquides avec rétroéclairage LED. Bon compromis entre qualité, consommation et coût.
- OLED : Diodes électroluminescentes organiques, offrant un contraste élevé, des noirs profonds et une meilleure finesse d'image.
- Écrans tactiles : Intègrent des capteurs de contact, souvent utilisés sur les ordinateurs portables, tablettes et bornes interactives.

Caractéristiques techniques :

Caractéristique	Définition
Taille	Diagonale de l'écran, exprimée en pouces (1" = 2,54 cm)
Définition	Nombre total de pixels (ex. : 1920×1080 pour le Full HD)
Résolution	Nombre de pixels par pouce (DPI – Dots Per Inch)
Fréquence de rafraîchissement	Nombre d'images affichées par seconde (60 Hz, 120 Hz, 144 Hz...)
Temps de réponse	Temps nécessaire à un pixel pour changer d'état (important pour les jeux)
Connectique	HDMI, DisplayPort, USB-C, parfois VGA ou DVI (anciens écrans)

7.8.2 La carte graphique

La carte graphique (GPU – Graphics Processing Unit) est un composant spécialisé dans le traitement de l'affichage. Elle peut être :

- intégrée dans le processeur (solution économique, pour bureautique ou usage modéré),
- ou dédiée, sous forme de carte d'extension (NVIDIA, AMD...), pour des besoins plus intensifs (jeux, graphisme, calcul scientifique).

Fonctions principales du GPU :

- gestion de l'affichage 2D et 3D,
- accélération du rendu graphique (textures, ombres, animations...),
- prise en charge de plusieurs écrans,
- encodage/décodage vidéo (accélération matérielle).

Caractéristiques clés :

Élément	Description
Mémoire vidéo (VRAM)	Espace dédié pour les textures, tampons d'image, etc. (ex. 4 Go, 8 Go, 12 Go)
Interface	PCIe (actuellement version 4.0 ou 5.0)
Sorties vidéo	HDMI, DisplayPort, USB-C...
Technologies associées	Ray tracing, DLSS (chez NVIDIA), FSR (chez AMD), G-Sync, FreeSync...

Note :
Les cartes graphiques modernes déchargent fortement le processeur central, notamment pour les tâches de rendu 3D, de traitement vidéo ou d'intelligence artificielle.

7.9 Les périphériques de stockage

7.9.1 Définition

Les périphériques de stockage sont des dispositifs permettant l'enregistrement, la conservation et la récupération de données numériques. Ils servent à stocker les programmes, les documents, les fichiers multimédias et les systèmes d'exploitation.

On distingue :

- les stockages internes, intégrés à l'ordinateur,
- les stockages externes, connectés temporairement ou en complément (USB, réseau…),
- les solutions dématérialisées, comme le cloud computing.

7.9.2 Classification des périphériques de stockage

7.9.2.1 Périphériques de stockage internes

Ces supports sont installés à l'intérieur de l'unité centrale et assurent le fonctionnement quotidien du système.

Dispositif	Description
Disque dur (HDD)	Stockage magnétique, grande capacité (jusqu'à 20 To), prix réduit, mais plus lent et sensible aux chocs. Encore répandu pour l'archivage.
Disque SSD	Basé sur la mémoire flash, rapide, silencieux et résistant. Temps d'accès très court. Devient la norme dans les PC récents.
Mémoire eMMC	Variante simplifiée de la mémoire flash, moins rapide que les SSD, souvent présente dans les appareils d'entrée de gamme (tablettes, netbooks).
NVMe (via PCIe)	Variante de SSD très rapide, directement connectée au bus PCI Express, avec des vitesses de transfert bien supérieures aux SSD SATA.

7.9.2.2 Périphériques de stockage externes

Utilisés pour le transport, la sauvegarde ou l'échange de données.

Dispositif	Utilisation principale
Clé USB	Mémoire flash portable, de 8 Go à 2 To. Pratique pour les transferts rapides.
Disque dur externe (HDD/SSD)	Grande capacité. Idéal pour les sauvegardes, les archives ou les bibliothèques multimédias.
Carte mémoire (SD, microSD)	Présente dans les smartphones, caméras, drones, consoles… Différentes classes de vitesse selon l'usage (photo, vidéo 4K…).
Boîtiers externes SSD/NVMe[18]	Permettent de transformer un SSD interne en solution externe rapide via USB-C ou Thunderbolt.

7.9.2.3 Supports optiques (aujourd'hui moins utilisés)

Ces supports sont basés sur la lecture/écriture de données via un faisceau laser.

Support	Capacité	Usage typique
CD-ROM / CD-RW	700 Mo	Sauvegarde ponctuelle, musique (en déclin)
DVD / DVD-RW	4,7 à 9 Go	Données, films (remplacé par le numérique)
Blu-ray (BD)	25 à 100 Go (BD-XL)	Films HD, archivage (moins utilisé aujourd'hui)

Les lecteurs optiques sont rarement intégrés dans les ordinateurs récents.

[18] L'acronyme NVMe signifie Non-Volatile Memory Express. Il s'agit d'un protocole de communication de stockage et une interface pour les disques SSD basés sur la mémoire flash, conçue pour exploiter la vitesse de l'interface PCI Express (PCIe) et offrir des performances élevées avec des temps de réponse réduits et un débit élevé par rapport aux anciens protocoles comme SATA.

7.9.2.4 Stockage en réseau ou à distance

Ces solutions permettent de partager et d'accéder aux données depuis plusieurs appareils ou à distance.

Dispositif / Service	Fonction principale
NAS (Network Attached Storage)	Serveur de fichiers personnel ou d'entreprise accessible via un réseau local ou Internet. Idéal pour les sauvegardes automatisées.
Cloud (Google Drive, Dropbox, OneDrive…)	Stockage en ligne, accessible depuis n'importe quel appareil connecté. Synchronisation automatique, collaboration facilitée.
Stockage hybride	Certains services combinent stockage local et sauvegarde cloud pour plus de sécurité.

7.9.3 Principes physiques utilisés

Les périphériques de stockage se distinguent par les principes physiques sur lesquels ils reposent. Ces principes influencent directement leurs performances (temps d'accès, vitesse de transfert), leur fiabilité, leur encombrement et leur coût.

Type de technologie	Principe physique	Exemples de supports
Magnétique	Les données sont codées sous forme de zones magnétiques sur une surface rotative ou linéaire.	Disque dur (HDD), bande magnétique
Électronique (Flash)	Les données sont enregistrées dans des cellules de mémoire à base de transistors (MOSFET[19]), conservant la charge électrique.	SSD, clé USB, carte mémoire, NVMe…
Optique	Un laser lit ou écrit les données en modifiant la surface d'un disque via des marques réfléchissantes	CD-ROM, DVD, Blu-ray
Électromécanique	Certains dispositifs combinent des parties mobiles mécaniques et des composants électroniques.	Disques durs (têtes de lecture, plateaux)
Réseau / Cloud	Données stockées sur des serveurs distants, accessibles par transmission de données via Internet (couche logicielle sur des technologies physiques diverses).	NAS, services cloud (OneDrive, Dropbox…)

7.9.4 Comparaison des périphériques de stockage

Support	Type de technologie	Principe physique	Avantages	Inconvénients
SSD (Solid State Drive)	Électronique (Flash)	Stockage dans des cellules à transistors (MOSFET), retenant une charge électrique	Très rapide, silencieux, résistant aux chocs, faible consommation	Prix au Go plus élevé qu'un HDD, usure à long terme (nombre d'écritures limité)
Disque dur (HDD)	Magnétique + électromécanique	Codage des données sous forme	Grande capacité, prix bas par Go,	Sensible aux chocs, plus lent,

[19] MOSFET est l'acronyme anglais de Metal Oxide Semiconductor Field Effect Transistor, ce qui se traduit en français par « transistor à effet de champ à structure métal-oxyde-semiconducteur ». Il s'agit d'un type de transistor utilisé pour amplifier ou commuter des signaux électroniques, et il est un composant essentiel dans de nombreux appareils, des microprocesseurs aux alimentations électriques.

Support	Type de technologie	Principe physique	Avantages	Inconvénients
		magnétique sur des plateaux tournants, lus par une tête mobile	bonne longévité si bien utilisé	bruyant, consomme davantage d'énergie
Clé USB	Électronique (Flash)	Données stockées dans des cellules Flash (similaire à un SSD)	Petite, portable, plug-and-play, pas besoin d'alimentation externe	Risque de perte, durée de vie limitée, vitesse variable selon modèles
Carte mémoire (SD/microSD)	Électronique (Flash)	Idem clé USB : cellules Flash	Compact, utile pour smartphones, appareils photo	Fragile, vitesse dépend du type (SDHC, SDXC…), durée de vie limitée
CD-ROM / DVD / Blu-ray	Optique	Lecture/écriture au laser via des zones réfléchissantes	Bon marché, utile pour l'archivage, faible consommation	Vitesse faible, sensible aux rayures, obsolescence progressive
NAS (Serveur de stockage réseau)	Réseau + disque magnétique ou SSD	Données accessibles via réseau local ou Internet, support physique interne variable	Partage de fichiers, sauvegarde centralisée, accessible à distance	Coût, configuration plus complexe, dépendance au réseau
Cloud (Google Drive, OneDrive…)	Réseau / Internet	Données stockées sur des serveurs distants, accessibles via Internet	Accessibilité partout, synchronisation automatique, pas de support physique	Dépend d'Internet, coût à long terme, confidentialité des données
Bande magnétique (Backup)	Magnétique (linéaire)	Données codées sur une bande défilante via magnétisation	Très grande capacité, coût faible, durable pour archivage	Très lent, accès séquentiel, équipement spécialisé
eMMC	Électronique (Flash)	Variante compacte de mémoire flash intégrée, souvent soudée à la carte mère	Peu encombrant, consommation réduite, bon rapport performance/prix pour appareils mobiles	Moins performant qu'un SSD, non remplaçable sur certains appareils

7.10 Comment vérifier la compatibilité d'une carte mère avec d'autres composants ?

Source : https://art-of-pc.fr/comment-verifier-la-compatibilite-de-votre-carte-mere-avec-dautres-composants/

Lorsqu'il s'agit de construire ou d'améliorer un PC de bureau, l'un des aspects les plus critiques est de s'assurer que la carte mère est compatible avec les autres composants. Une incompatibilité peut causer divers problèmes techniques et même empêcher complètement le bon fonctionnement de l'ordinateur. Penchons-nous sur les moyens pratiques permettant de vérifier la compatibilité de la carte mère avec des composants clés comme le processeur (CPU), la mémoire vive (RAM) et la carte graphique (GPU).

7.10.1 Identification des besoins spécifiques de notre configuration

Avant de plonger dans les détails de la compatibilité, il est essentiel de déterminer vos besoins spécifiques. Quel type de tâches prévoyez-vous d'effectuer avec votre PC ? Cette question influencera grandement les choix en matière de matériel, notamment en termes de performance et de budget.

Une fois vos besoins définis, passez à l'étape suivante : examiner les spécifications de votre carte mère actuelle ou future. Cela inclut des éléments tels que le chipset, le socket et le facteur de forme. Chaque composant doit correspondre aux exigences techniques de votre carte mère pour garantir une harmonie parfaite entre tous les éléments matériels.

7.10.2 Compatibilité du processeur (CPU)

7.10.2.1 Vérifier le socket

L'une des premières choses à vérifier pour la compatibilité du processeur est le socket. Le socket est l'emplacement sur la carte mère où le processeur sera installé. Les fabricants utilisent différents types de sockets pour différents modèles de CPU, et ceux-ci ne sont pas interchangeables. Identifiez le socket supporté par votre carte mère (par exemple, AM4 pour AMD ou LGA 1200 pour Intel) et choisissez un processeur qui utilise le même socket.

Exemple : Si vous avez une carte mère avec un socket LGA 1851, assurez-vous que votre processeur est également conçu pour ce socket.

7.10.2.2 Examinez le chipset

Le chipset de la carte mère joue également un rôle crucial dans la compatibilité avec le processeur. Il détermine les fonctionnalités supportées par la carte mère et influence directement les performances du CPU. Assurez-vous de consulter les spécifications de la carte mère et du site du fabricant pour vérifier que le chipset supporte bien le modèle de processeur que vous envisagez d'acheter.

7.10.3 Compatibilité de la mémoire vive (RAM)

7.10.3.1 Type et fréquence de la RAM

La mémoire vive est un autre composant déterminant pour le bon fonctionnement de votre PC. Tout d'abord, vérifiez le type de RAM supporté par votre carte mère (DDR3, DDR4, DDR5 etc.). Ensuite, examinez la fréquence maximale de la RAM que la carte mère peut gérer. Une différence trop importante dans ces spécifications risque de réduire les performances ou même d'empêcher le démarrage du système.

Les informations sur la compatibilité RAM se trouvent souvent dans le manuel de la carte mère ou sur le site du fabricant. Consultez ces ressources pour confirmer que la mémoire que vous souhaitez utiliser est acceptée.

7.10.3.2 Capacité maximale et nombre de slots

Il est également important de noter la capacité maximale de RAM que votre carte mère peut supporter ainsi que le nombre de slots disponibles. Par exemple, si votre carte mère supporte jusqu'à 64 Go de RAM mais qu'elle n'a que deux slots de RAM, vous devrez utiliser des barrettes de 32 Go chacune pour maximiser votre capacité.

Astuces : Répartir la RAM sur plusieurs modules peut améliorer les performances grâce au Dual Channel, alors considérez cette option si elle est disponible.

7.10.4 Compatibilité de la carte graphique (GPU)

7.10.4.1 Interface PCIe

Pour la carte graphique, la majorité des cartes mères modernes utilisent des emplacements PCI Express (PCIe). Vérifiez que votre carte mère dispose d'au moins un slot PCIe x16 pour accueillir le GPU. La version de PCIe (par exemple, PCIe 3.0 ou PCIe 4.0) importe également, même si elles sont généralement rétrocompatibles. Cependant, utiliser le même standard maximise les performances.

Consultez le manuel ou le site du fabricant pour connaître le type exact de slots PCIe disponibles sur votre carte mère. Ces sources contiennent des diagrammes détaillés qui peuvent aider à bien situer chaque slot.

7.10.4.2 Alimentation électrique requise

Assurez-vous aussi que votre alimentation peut fournir suffisamment de puissance pour faire fonctionner la carte graphique. Les puissantes cartes graphiques modernes nécessitent parfois plusieurs connecteurs d'alimentation supplémentaires. Vérifiez donc si votre bloc d'alimentation est capable de répondre à cette demande.

7.10.5 Autres considérations importantes

7.10.5.1 Facteur de forme de la carte mère

Le facteur de forme de la carte mère détermine sa taille et ses points de montage. Les cartes mères sont disponibles en différents formats tels que ATX, Micro-ATX et Mini-ITX. Le choix dépend de la taille de votre boîtier PC et de vos besoins en termes d'expansions futures. Une carte mère Micro-ATX pourrait être plus adaptée pour un usage basique tandis qu'une carte mère ATX offre plus d'options d'expansion.

7.10.5.2 Connectiques et ports supplémentaires

Ne négligez pas les ports et connectiques supplémentaires dont vous pourriez avoir besoin. Cela inclut les ports USB, les slots M.2 pour les SSD NVMe, les headers pour RGB, et les connecteurs pour ventilateurs supplémentaires. Prenez le temps d'examiner les connectivités offertes par votre carte mère et comparez-les avec vos besoins.

7.10.6 Utilisation des outils de vérification en ligne

De nos jours, nombreux sont les outils en ligne disponibles pour simplifier la vérification de la compatibilité des composants. Des sites comme PCPartPicker (https://fr.pcpartpicker.com/), configurateurPC (https://configurateurpc.com/, PC Builder (https://fr.pc-builder.io/), pangoly (https://pangoly.com) permettent de sélectionner des composants et de vérifier automatiquement leur compatibilité. Utiliser ces outils peut vous éviter des erreurs coûteuses et gagner du temps.

N'oubliez pas de visiter régulièrement le site du fabricant de votre carte mère pour obtenir les dernières mises à jour de bios ou de firmware. Cela garantit non seulement la compatibilité mais améliore aussi la stabilité et les performances de votre système.

8 Informatisation d'un problème posé

8.1 Étapes de l'informatisation d'un problème posé

L'informatisation d'un problème consiste à traduire une situation, une question ou un besoin réel en solutions numériques structurées. Ce processus suit une démarche méthodique qui facilite la réussite de l'informatisation, que ce soit dans une entreprise, une organisation ou un contexte pédagogique.

8.1.1 Analyse et compréhension du problème

- Examiner précisément la situation et les besoins.
- Identifier les enjeux, les objectifs à atteindre et les contraintes du contexte.
- Reformuler le problème sous une forme claire, éventuellement accompagnée des fonctions attendues par la future solution informatique.

8.1.2 Définition des objectifs et des besoins

- Énoncer des objectifs atteignables et mesurables.
- Déterminer les processus ou activités à informatiser.
- Lister les besoins fonctionnels (opérations à réaliser, résultats attendus) et techniques (interopérabilité, sécurité, etc.).

8.1.3 Évaluation des ressources et élaboration d'un cahier des charges

- Identifier les ressources disponibles (humaines, financières, matérielles).
- Rédiger un cahier des charges fonctionnel qui précise les exigences, les délais et les contraintes.
- Fixer un échéancier pour les différentes étapes du projet.

8.1.4 Planification et organisation du projet

- Définir le plan d'action et la chronologie de mise en œuvre.
- Organiser l'accès aux données ou aux documents à informatiser.
- Impliquer les parties prenantes dès le départ pour garantir l'adhésion au projet.

8.1.5 Choix des outils, matériels et logiciels

- Effectuer une veille sur les solutions existantes.
- Sélectionner des outils adaptés (logiciels, matériels, applications, plateformes).
- Vérifier la compatibilité avec les équipements déjà présents.

8.1.6 Développement spécifique (si nécessaire)

Dans certains projets, les logiciels existants ne répondent pas totalement aux besoins exprimés. Il faut alors procéder à un développement spécifique :

- Création d'un logiciel : traduction des besoins en algorithmes, puis en programmes écrits dans un langage de programmation.
- Mise en place d'une base de données : conception du modèle de données (tables, relations, règles d'intégrité), puis création de la base via un SGBD (par ex. MySQL, SQL Server...).
- Tests unitaires : vérifier que chaque composant fonctionne correctement avant l'intégration finale.

8.1.7 Installation, paramétrage et tests

- Installer l'infrastructure matérielle et logicielle choisie.
- Paramétrer les systèmes selon les besoins identifiés.
- Réaliser des tests pour valider la configuration et l'adéquation de la solution aux attentes.

8.1.8 Saisie et migration des données

- Saisir, migrer ou convertir les données nécessaires au fonctionnement du système informatisé.
- Procéder par catégories logiques, tout en assurant la qualité des données.

8.1.9 Formation des utilisateurs et accompagnement au changement

- Former les utilisateurs à l'utilisation des nouveaux outils et processus numériques.
- Accompagner et soutenir le changement pour une adoption efficace et durable de la solution.

8.1.10 Suivi, maintenance et amélioration continue

- Surveiller les performances du système informatisé.
- Assurer sa maintenance (mises à jour, sauvegardes, corrections).
- Recueillir les retours d'expérience pour améliorer le dispositif et adapter les outils si nécessaire.

Chaque étape contribue à structurer la transition vers une réponse numérique efficace et pertinente au problème posé, tout en limitant les risques d'échec ou d'insatisfaction. Cette démarche est essentielle dans tout projet d'informatisation, quel que soit le domaine d'application.

8.2 L'algorithmique

8.2.1 Qu'est-ce qu'un algorithme ?

Un algorithme est une suite d'instructions, ordonnées et non ambiguës, qui permettent de résoudre un problème ou d'atteindre un objectif.

Exemple de la vie courante :

1) Avancer jusqu'au bord du trottoir.
2) Regarder à gauche.
3) Regarder à droite.
4) Si une voiture arrive, attendre.
5) Sinon, traverser.
6) Arriver sur l'autre trottoir.

De la même manière, un ordinateur exécute des algorithmes pour effectuer des calculs, trier des données, afficher une page web, etc.

8.2.2 Caractéristiques d'un algorithme

Un algorithme doit être :

- Fini : il doit se terminer après un nombre limité d'étapes.
- Précis : chaque instruction doit être claire et sans ambiguïté.
- Exécutable : il doit être possible de le réaliser en pratique (par un humain ou une machine).

8.2.3 Représentations d'un algorithme

Les algorithmes peuvent être représentés de plusieurs manières :

- En langage naturel (phrases) – comme l'exemple ci-avant.
- En pseudo-code – un langage intermédiaire proche de l'informatique, mais lisible par l'humain.
- En ordinogramme (organigramme) – un schéma visuel utilisant des symboles normalisés pour représenter les étapes.

8.2.4 Exemple simple : calculer la somme de deux nombres

En pseudo-code :

```
DEBUT
  Lire A
  Lire B
  S ← A + B
  Afficher S
FIN
```

8.2.5 Importance de l'algorithmique

- C'est la base de la programmation : chaque programme n'est que la traduction d'un algorithme dans un langage compréhensible par la machine.
- Elle favorise la résolution de problèmes et le raisonnement logique.
- Elle est utilisée partout : recherche Google, GPS, tri de mails, sécurité informatique, etc.

8.3 Les langages de programmation

8.3.1 Pourquoi des langages de programmation ?

Un ordinateur ne comprend que le langage machine (suite de 0 et de 1).
Écrire directement en binaire serait fastidieux et source d'erreurs.
Les langages de programmation ont donc été inventés pour permettre aux humains de rédiger des instructions de manière plus claire, ensuite traduites en langage machine par des programmes : compilateurs, interpréteurs ou assembleurs.

8.3.2 Types de langages

On distingue plusieurs niveaux :

- Langage machine : instructions en binaire, directement comprises par le processeur.
- Langage assembleur : proche du matériel, utilise des mnémoniques (ADD, MOV, etc.).
- Langages de haut niveau : proches du langage humain, permettent d'écrire des programmes plus facilement (C, Python, Java, etc.).

8.3.3 Familles de langages

- Procéduraux : le programme décrit une suite d'ordres à exécuter (C, Pascal).
- Orientés objets : organisent le code autour d'objets et de classes (Java, C++, Python).
- Fonctionnels : centrés sur les fonctions mathématiques (Lisp, Haskell).
- Langages spécialisés : destinés à un usage particulier (SQL pour les bases de données, HTML/CSS pour la mise en page web).

8.3.4 Exemple simple en Python

Traduction de l'algorithme « calculer la somme de deux nombres » :

```
# Déclaration / initialisation des variables
a = 0
b = 0
s = 0

# Saisie des valeurs
a = int(input("Entrez un premier nombre : "))
b = int(input("Entrez un deuxième nombre : "))

# Traitement
s = a + b

# Affichage
print("La somme est :", s)
```

Ce petit exemple montre comment un algorithme devient un programme exécutable par un ordinateur.

8.3.5 Importance des langages de programmation

- Ils permettent de traduire une idée (algorithme) en instructions compréhensibles par la machine.
- Ils sont adaptés aux besoins : site web, intelligence artificielle, gestion de données, électronique embarquée, etc.
- Ils évoluent sans cesse, pour répondre à de nouveaux usages.

En résumé :

- **L'algorithmique = réflexion et conception.**
- **Le langage de programmation = mise en œuvre pratique pour l'ordinateur.**

9 Domaines d'intervention dans un système informatique et métiers s'y rapportant

Le champ de l'informatique est vaste et couvre plusieurs domaines d'intervention complémentaires. Chaque domaine correspond à des expertises et débouche sur des métiers spécifiques.

9.1 Matériel et infrastructure

Domaine : Conception, installation, maintenance et gestion des équipements physiques (serveurs, postes de travail, périphériques, composants réseau...).

Métiers :

- Technicien(ne) informatique
- Administrateur(trice) systèmes et réseaux
- Ingénieur(e) infrastructure
- Gestionnaire de parc informatique

9.2 Réseaux et télécommunications

Domaine : Mise en place, sécurisation et administration des réseaux informatiques (locaux, étendus, sans fil), gestion du câblage, du routage, des connexions Internet et des protocoles de communication.

Métiers :

- Technicien(ne) réseaux
- Administrateur(trice) réseau
- Ingénieur(e) réseau et télécoms
- Architecte réseau

9.3 Logiciels et développement

Domaine : Création, développement, test, maintenance et optimisation des applications logicielles, sites web, applications mobiles, outils métier et scripts.

Métiers :

- Développeur(se) d'applications
- Développeur(se) web/mobile
- Ingénieur(e) en développement logiciel
- Analyste-programmeur(se)
- Chef(fe) de projets informatiques

9.4 Systèmes d'information et gestion de données

Domaine : Déploiement et administration des systèmes d'information (ERP[20], CRM[21]...), gestion et exploitation des bases de données, analyse de données.

Métiers :

- Administrateur(trice) de bases de données (DBA)
- Data analyst / Data scientist
- Consultant(e) ERP/CRM
- Ingénieur(e) décisionnel

[20] ERP signifie « Enterprise Resource Planning » et se traduit en français par Progiciel de Gestion Intégré (PGI). Un ERP est un système logiciel qui unifie les différentes fonctions et processus d'une entreprise (comptabilité, gestion des ressources humaines, production, logistique, etc.) dans une seule base de données pour une gestion plus efficace et une meilleure prise de décision.

[21] CRM signifie « Customer Relationship Management », soit « Gestion de la Relation Client » en français. Il désigne à la fois une stratégie de gestion des interactions avec les clients et prospects, et l'outil informatique (logiciel) qui permet de mettre en œuvre cette stratégie en centralisant les données et en automatisant les processus.

9.5 Cybersécurité

Domaine : Sécurisation des systèmes, des réseaux, des applications et des données ; gestion des accès, mises à jour et audits de sécurité.

Métiers :

- Analyste sécurité
- Consultant(e) cybersécurité
- Ingénieur(e) sécurité informatique
- RSSI (Responsable Sécurité des Systèmes d'Information)

9.6 Support, assistance et formation

Domaine : Assistance technique aux utilisateurs, maintenance logicielle et matérielle, gestion des incidents, formation et sensibilisation aux outils numériques.

Métiers :

- Technicien(ne) support/helpdesk
- Assistant(e) informatique
- Formateur(trice) informatique
- Conseiller(ère) numérique

9.7 Gouvernance, gestion de projets et conseil

Domaine : Pilotage de projets informatiques, alignement stratégique, audit, conseil, accompagnement au changement numérique dans les organisations.

Métiers :

- Chef(fe) de projets informatiques
- Consultant(e) en systèmes d'information
- Architecte IT
- DSI (Directeur(trice) des Systèmes d'Information)

Résumé :

Domaine	Métiers liés
Matériel & Infrastructure	Technicien(ne) informatique, Ingénieur(e) infrastructure, Gestionnaire de parc
Réseaux & Télécoms	Technicien(ne) ou ingénieur(e) réseaux, architecte réseau
Logiciel & Développement	Développeur(se), analyste-programmeur(se), chef(fe) de projet
Systèmes d'Information & Données	DBA, data analyst/scientist, consultant(e) ERP/CRM
Cybersécurité	Analyste ou ingénieur(e) sécurité, consultant(e)
Support & Assistance	Technicien(ne) support, formateur(trice), assistant(e)
Gouvernance, Gestion de projet & Conseil	Chef(fe) de projet, consultant(e) SI, architecte IT

L'informatique moderne repose sur cette pluralité de domaines, avec des spécialisations qui répondent aux besoins variés des organisations, du support technique jusqu'à la stratégie numérique. Les métiers cités sont évolutifs, certains pouvant combiner plusieurs domaines selon la taille de l'entreprise ou la structure du service informatique.

10 Bibliographie

1NSI - Modèle d'architecture d'un ordinateur. (s. d.). https://www.info-mounier.fr/premiere_nsi/archi_os/modele-archi

Architecture. (s. d.). Qkzk. https://qkzk.xyz/docs/nsi/cours_premiere/architecture/

Arrow Electronics, Inc. (2023, 17 avril). *Principes de base des convertisseurs analogique-numérique.* Arrow. https://www.arrow.com/fr-fr/research-and-events/articles/engineering-resource-basics-of-analog-to-digital-converters

Art-Of-Pc. (2025, 27 février). *Comment vérifier la compatibilité de votre carte mère avec d'autres composants - Art of Pc - Spécialiste de l'assemblage PC sur mesure.* Art Of Pc - Spécialiste de L'assemblage PC Sur Mesure. Consulté le 10 juillet 2025, à l'adresse https://art-of-pc.fr/comment-verifier-la-compatibilite-de-votre-carte-mere-avec-dautres-composants/

De Mey, L. (s. d.-c). *Le CPU.* Consulté le 8 juillet 2025, à l'adresse https://courstechinfo.be/Techno/CPU.html

De Mey, L. (s. d.-b). *Architecture matérielle d'un système informatique.* https://courstechinfo.be/Techno/ArchMat.html

De Mey, L. (s. d.-c). *Information / informatique.* https://courstechinfo.be/Techno/NumeriqueVsAnalogique.html

De Mey, L. (s. d.-c). *La mémoire.* Consulté le 8 juillet 2025, à l'adresse https://courstechinfo.be/Techno/Memoire.html

De Mey, L. (s. d.-d). *Information / informatique*. Consulté le 20 juillet 2025, à l'adresse https://courstechinfo.be/Techno/NumeriqueVsAnalogique.html

De Mey, L. (s. d.-e). *Numération.* https://courstechinfo.be/Techno/Numeration.html

Documentation of Operating Systems and Producers - Independent Website. (s. d.). Consulté le 8 juillet 2025, à l'adresse https://www.operating-system.org/index.html

Zanella, P., Ligier, Y., & Lazard, E. (s. d.). *Architecture et technologie des ordinateurs - 6e éd. : Cours et exercices corrigés*. Dunod

Source de l'image en couverture : https://pixabay.com/fr/photos/mat%C3%A9riel-la-technologie-carte-m%C3%A8re-4955429/