الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Djilali BOUNAAMA Khemis Miliana
Faculté des Sciences et de la Technologie
Département des mathématiques et de l'informatique
جامعة الجيلالي بونعامة خميس مليانة
كلية العلوم والتكنولوجيا
قسم الرياضيات و الإعلام الآلي
Adresse : Rue Thniet El Had, Khemis Miliana, Ain Defla , Algérie. Tel :(213) 27556844

Intitulé du polycopié

Polycopié de Cours : Structure machine 1

Destiné aux étudiants

Niveau : Première année Licence
Spécialité : Mathématique et informatique

Auteur
MAHROUG RABIAA

Experts du polycopié	Grade	Établissement d'affiliation
Bendoumia Redha	MCA	Université de Blida 1
Hachichi Hiba	MCA	Université de D.Bounaama khemis miliana

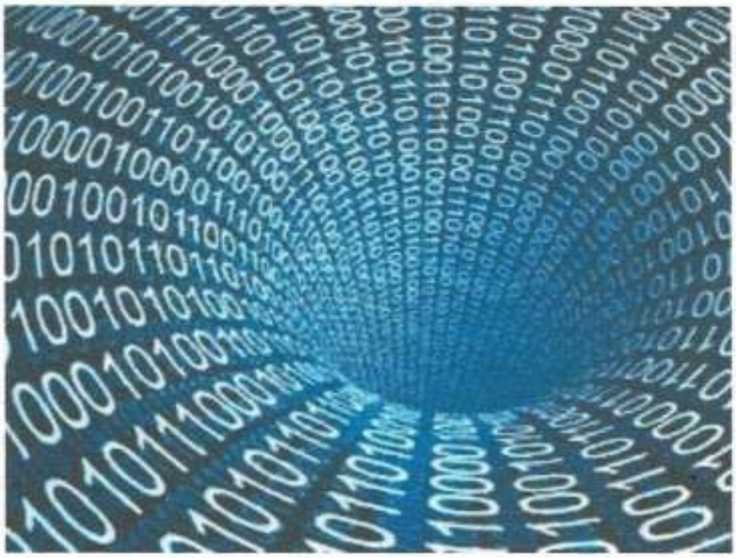
Date de validation du polycopié par l'instance scientifique habilitée
CSD et/ou CSF

CSD	26-06-2023
CSF	24-10-2023

Année universitaire : 2022/2023

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieure et de la Recherche Scientifique
Université Djilali Bounaâma de Khemis Miliana
Faculté des sciences et techniques
Département de maths et informatique
Niveau : 1ère année MI - Licence

Polycopié de Cours
Structure machine 1



Rédigé par Dr. MAHROUG RABIAA
E-mail : r.mahroug@univ-dbkm.dz
URL du fichier: https://

Année universitaire 2022-2023

Introduction

De son intitulé « Polycopié de Cours : Structure machine 1 », ce polycopié est axé sur le concept de l'informatique, le codage et la représentation de l'information, et des connaissances sur la théorie formelle basée sur l'Algèbre de Boole pour la synthèse des circuits logiques.

Ce polycopié a comme objectif d'apprendre aux étudiants les différents systèmes de numération, les différents types de codification et représentation des nombres et caractères et de prendre des connaissances sur la théorie formelle de l'Algèbre de Boole pour la synthèse des fonctions logiques. Ce travail est destiné aux étudiants LMD (1ère année licence) socle commun Mathématique et Informatique.

A l'issue de ce cours, les étudiants seront capables de :

- Maitriser le concept de systèmes de numération.
- Comprendre les règles de représentations des systèmes de numération.
- Etre capable de réaliser des conversations entre les différentes bases.
- Être capable de traiter des opérations arithmétiques et des calculs dans les différentes bases.
- Représenter les nombres réels et négatifs dans la machine.
- Être capable de faire un codage des nombres naturels et signés en complément à 1 et à 2.
- Comprendre les différents systèmes de codage de l'information.
- Savoir représenter, traiter et simuler des fonctions logiques des différents systèmes.
- Connaître les trois opérations de base et leurs différentes propriétés de l'algèbre de Boole.
- Être capable de comprendre et d'appliquer l'ensemble de lois de l'algèbre de Boole.
- Simplifier des expressions logiques graphiquement (par la table de KARNAUGH) et algébriquement (par les lois de l'algèbre de Boole).

Dans ce polycopié, nous abordons quelques notions de base qui mènent à la conception des systèmes de numération. Nous traiterons, dans la première partie, les systèmes de numération et le codage de tous types d'informations telles que les caractères, les nombres entiers signés et non signés et les réels. Puis, nous présenterons ensuite, les éléments d'algèbre booléenne qui constituent la base mathématique nécessaire au traitement de l'information, les bases de la logique booléenne sont implémentées sous forme de circuits et les différentes méthodes de simplification telle que la simplification algébrique (par les lois de l'algèbre de Boole), la simplification graphique (par la table de Karnaugh) et la simplification par la méthode McCluskey.

Prérequis : -Mathématique

-Mathématique (Algèbre Linéaire).

-Electronique de base.

Ce polycopié de cours est principalement inspiré des références en numération et codification de l'information citées dans la bibliographie, le lecteur est invité à se référer à ces ressources pour approfondir ses connaissances. Dans ce qui suit, on détaille le programme de la matière.

Table des matières

Abréviation

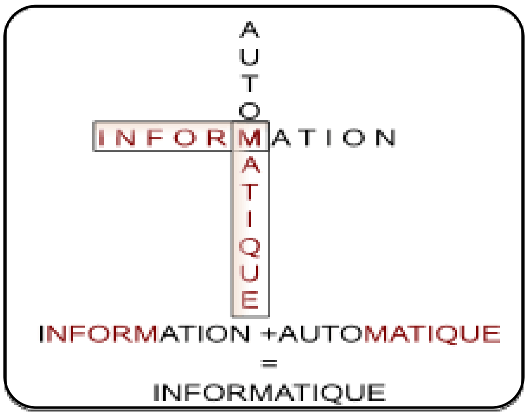
Bit	: Binary Digit
B ou b	: Base
MSB	: Most Significant Bit
LSB	: Least Significant Bit
SVA	: Signe et Valeur Absolue
Cà1	: Complément à 1
Cà2	: Complément à 2
BCD	: Binary Coded Decimal
BCD+3	: Binary Coded Decimal + 3
IEEE 754	: IEEE Standard for Floating-Point Arithmetic 754
ASCII	: Americain Standard Code Information Interchange
EBCDIC	: Extended Binary Coded Decimal Interchange Code
UTF	: Unicode Transformation Format
F ou f	: Fonction
FND	: Première forme canonique disjonctive
FNC	: Deuxième forme canonique conjonctive

Chapitre I : Introduction Générale

Chapitre I : Introduction Générale

1.1. Introduction à l'informatique

L'informatique est la science qui étudie le traitement **automatique** de **l'information** par une machine électronique (ordinateur).



1.2. Qu'est-ce qu'un ordinateur ?

Un ordinateur est une machine (un calculateur numérique) qui permet de travailler avec des informations appelées données telles que des nombres, des mots, des images et des sons…etc. Il est capable d'acquérir et de stocker des informations, d'effectuer le traitement et la récupération d'informations. C'est une machine électronique de traitement automatique de l'information.

1.3. Les composants de l'ordinateur

Les composants de l'ordinateur sont divisés en deux parties principales qui sont : Le matériel et le logiciel (Hardware et Software).

1.3.1. Matériel

Le matériel est constitué par les éléments physiques de la machine, sont constituées d'une variété de dispositifs. On peut distinguer deux catégories principales du matériel :

1.3.1.1. Éléments de base d'un ordinateur

Unité Centrale : C'est l'une des parties les plus importantes de l'ordinateur. La CPU (Central Processing Unit) reçoit les données et les instructions des modules d'entrée et les traites en informations qui peuvent être affichées ou stockées. L'unité centrale est le boitier qui contient les principaux éléments électroniques permettant à un ordinateur de fonctionner.

Écran (moniteur) : L'écran affiche visuellement des informations sous forme de textes et de graphiques. La taille d'un écran est mesurée en pouce sur la diagonale. Il existe deux principaux types de moniteur : moniteur CRT et moniteur Plat.

Clavier : est un périphérique qui permet la saisie des informations en direction de l'ordinateur. Il contient des touches pour les lettres et les chiffres et des touches spéciales (touches de fonction (Alt, Ctrl, AltGr, ...etc.) et symboles (&, %, #, @, ...etc.)). On distingue différents types de claviers comme le clavier « AZERTY » adapté pour la langue française et le clavier « QWERTY » adapté pour la langue anglaise.

Souris : Elle s'agit d'un dispositif de pointage, et se composée d'un petit boitier destiné à être tenu en main et se compose d'un ou plusieurs boutons et une molette permet de piloter l'ordinateur. Elle vous permet de sélectionner, déplacer et manipuler les éléments présents sur votre ordinateur. Elle contient principalement un bouton principal (à gauche) et un bouton secondaire (à droite).

1.3.1.2. Éléments secondaires d'un ordinateur

Ils s'agissent des éléments électroniques externes autrement dit des périphériques connectés à l'ordinateur :

Imprimante : C'est un dispositif qui permet d'imprimer sur papier des informations stockées sur l'ordinateur. Il existe plusieurs types d'imprimantes, les types les plus courants sont : les imprimantes matricielles, les imprimantes laser, les imprimantes multifonctions et les imprimantes à jet d'encre.

Scanner : C'est un appareil qui vous permet de convertir des documents (images, textes, …etc.) en données numériques pour les avoir sur votre ordinateur. Très utile pour le traitement d'image ou de texte sans avoir à taper.

Hauts Parleurs : Ce sont des périphériques connectés à l'ordinateur via la carte son, elles permettent de ressortir les sons générés par l'ordinateur.

Microphone : C'est un dispositif utilisé pour enregistrer et entrer des sons dans ordinateur.

Traceurs : ce sont des dispositifs utilisés pour reproduire des dessins, des plans, etc. sur une feuille de papier peut être de grande taille via des plumes de différentes couleurs.

Les appareils photos et caméscopes numériques : lorsqu'ils sont connectés à un ordinateur, ils agissent comme une mémoire de stockage externe et vous pouvez visualiser les photos et les vidéos qu'ils contiennent.

Les mémoires externes : Ce sont des mémoires de stockage amovibles qui servent le stockage des informations d'une façon permanente telles que : Les disques durs externes, les cartes mémoire et les clés USB.

Modem (modulateur/démodulateur) : C'est un appareil qui vous permet de se connecté à internet via une ligne téléphonique.

1.3.2. Logiciel

Un logiciel, autrement dit une application, est un ensemble de programmes qui permettent à un ordinateur ou à un système informatique d'effectuer une tâche ou une fonction spécifique. Il peut s'agir d'un :

- Système d'exploitation (SE) ;
- Logiciel d'application (logiciels bureautiques, éditeurs, navigateurs et jeux, …etc.) ;
- Progiciel à usage professionnelle.

1.4. Types de périphériques

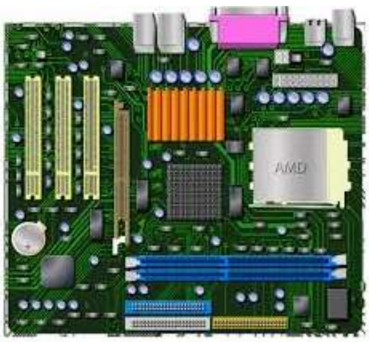
Les périphériques

- périphériques d'entrée:
  souris , clavier, scanner, microphone...etc.
- périphérique de sortie:
  écran, imprimante, hauts parleurs, traceurs..etc.
- périphérique d'entrée/sortie:
  appareils photos, caméscopes numériques, mémoires externes, modem...etc.

1.5. Composants principaux de l'unité centrale

L'unité centrale est principalement composée de :

- ✓ **Carte mère** : est un circuit imprimé servant à connecter tous les composants d'un ordinateur. La carte mère est le cœur de tout ordinateur. Elle permet aux différentes parties de l'ordinateur (processeur, clavier, mémoire, cartes d'extension, …etc.) de communiquer entre elles. La carte mère est le cœur de tout ordinateur. La carte mère est un circuit imprimé servant à connecter tous les composants d'un ordinateur. Elle permet aux différentes parties d'un ordinateur de communiquer entre elles (processeur, les mémoires, clavier, les cartes d'extensions, …etc.).



- ✓ **Processeur** : est le cerveau de l'ordinateur, il est responsable du traitement des informations et de l'exécution des commandes pour obtenir les résultats attendus. Il se caractérise par sa fréquence, exprimée en Hertz.

- ✓ **Les mémoires :** Ce sont des dispositifs qui stockent les informations de manière temporaire ou permanente. Principalement l'ordinateur utilise les types des mémoires suivantes :
  - ➢ **La mémoire principale RAM (RandomAccess Memory) :** elles s'agissent l'espace de travail du processeur. La RAM stocke temporairement les données à traiter, ce qui permet d'éviter d'accéder au disque dur qui est plus lent. Une RAM est volatile signifie que si l'ordinateur est éteint, toutes les informations de la RAM seront perdues.
  - ➢ **La mémoire Morte ROM (Read Only Memory) :** également appelée mémoire morte ou mémoire en lecture seule, c'est la mémoire interne de l'ordinateur, dont le contenu est défini lors de la fabrication. Est une mémoire non volatile, ce qui signifie qu'elle n'efface pas le contenu si l'ordinateur est mis hors tension.
  - ➢ **Les mémoires secondaires :** Elles permettent de stocker les programmes et les données de façon permanente, c'est-à-dire que les données ne sont pas effacées en cas d'une coupure de l'électricité. C'est par exemple le cas des disques durs, des bandes magnétiques, …etc.

- ✓ **Bloc alimentation** : ou l'alimentation ou alim (power supply unit en anglais PSU). L'alimentation nous permet de fournit l'énergie électrique à tous les composants de l'ordinateur. Assure la distribution de courant électrique à tous les composants de la carte mère et aux disques.

- ✓ **Carte Son** : elle est responsable de tout le traitement numérique du son, et permet la gestion de son sorti (haut-parleurs) et de son entré (microphone).

- ✓ **Carte graphique** : Elle permet d'afficher les données de l'ordinateur sur l'écran. Elle peut également s'agir d'une puce électronique intégrée à la carte mère.

✓**Carte réseau** : elle s'agit d'une carte d'extension qui permet de connecter l'ordinateur au réseau local.

✓**Lecteur/Graveur DVD** : est un lecteur qui vous permet de lire des données à partir d'un DVD ou d'un CD. Dans sa fonction de graveur, en plus de la lecture, il permet de stocker des données sur des disques (CD ou bien DVD).

1.6. Éléments de base d'une unité centrale

➲ **Panneau Avant (face Avant)**

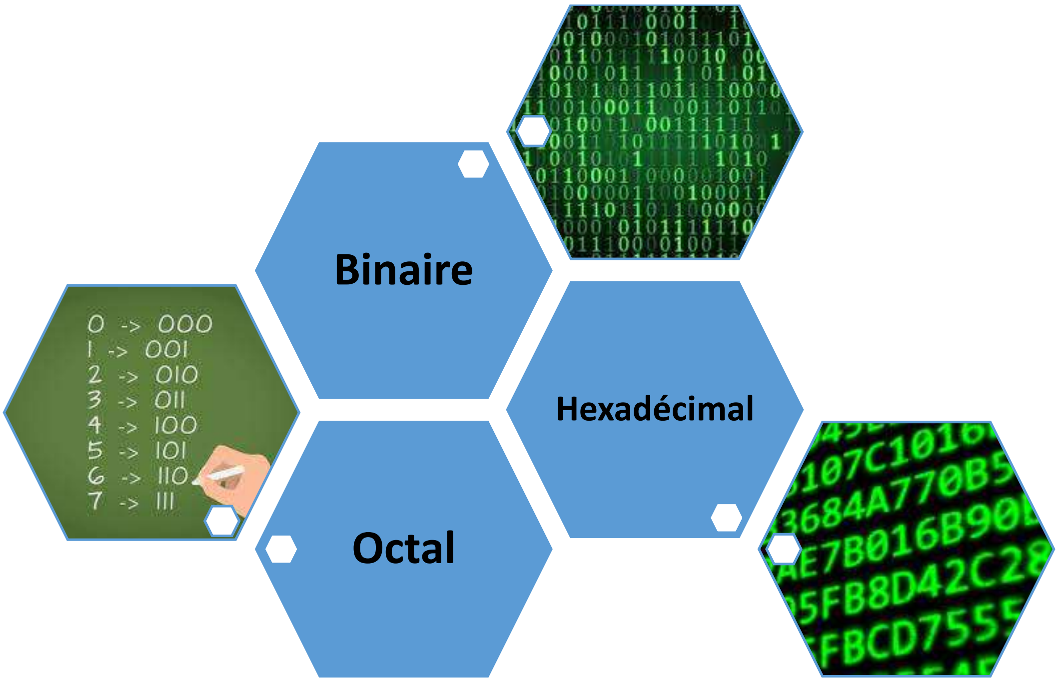
- ✓ Bouton on/off (marche/arrêt) ;
- ✓ Bouton de redémarrage (reset) ;
- ✓ Un ou bien plusieurs lecteurs optiques (Lecteur ou graveur CD/DVD, Lecteur ou Graveur BlueRay…etc.) ;
- ✓ Ports USB ;
- ✓ Lecteur de carte mémoire ;
- ✓ Prises audio (microphone, haut-parleur et casque).

➲ **Panneau Arrière (face Arrière)**

- ✓ Port série : ancien port qui est utilisé pour connecter divers périphériques à l'ordinateur.
- ✓ Port parallèle : ancien port qui est utilisé pour connecter l'imprimante à l'unité centrale.
- ✓ Prise écran : pour bancher le moniteur à l'unité centrale.

- ✓ Ports USB : permet de connecter divers périphériques tels que (Clés USB, appareil photo numérique …etc)
- ✓ Connecteur réseau (RJ45) : permet de relier l'ordinateur au réseau.
- ✓ Prises audio : Prise de sortie audio (vert) où l'on branche des haut-parleurs ou bien des casques audio.
- ✓ Prise micro (rose) : utilisée pour connecter un microphone afin d'enregistrer des sons.
- ✓ Prise ligne line in/out (bleu) : permet de relier divers outils musicaux.
- ✓ Fiche d'alimentation : utilisé pour alimenter l'ordinateur en courante électrique.

Chapitre II : Les systèmes de numération



Chapitre II : Les systèmes de numération

2.1. Définitions

Quel que soit le type d'informations traitées par un circuit électronique (texte, image, audio, vidéo), elles doivent être placées sous forme numérique (une séquence de 0 et de 1), c'est-à- dire sous la forme adaptée à celui-ci, par exemple : (11011101).

2.1.1. Bit

Un **état binaire** est appelé (en anglais **BIT** : signifie ⇒ **B**inary dig**IT**). C'est la plus petite unité d'information pouvant être traitée par un ordinateur. Cette information binaire peut être physiquement représentée par un signal électrique ou magnétique, qui dépasse un certain seuil, correspondant à la valeur 1.

Un bit ne peut prendre que deux valeurs. Selon le contexte, numérique, logique, électronique, magnétique ou optique, ils sont désignés par "0", et "1" équivaut à "faux" et "vrai", "ouvert" et "fermé", "nord" et "sud" ou "noir" et "blanc" respectivement.

2.1.2. Octet

Un octet (en anglais byte) est une unité d'information de 8 bits. Il permet de stocker un caractère tel qu'une lettre ou un chiffre. Un mot est une unité d'information composée de 16 bits (en anglais word). Un double mot est une unité d'information de 32 bits de longueur (en anglais double word, dword). Voici les unités standards :

1 Ko (kilooctet) = 10^3 octet	1 Eo (exaoctet) = 10^{18} octet
1 Mo (mégaoctet) = 10^6 octet	1 Zo (zettaoctet) = 10^{21} octet
1 Go (gigaoctet) = 10^9 octet	1 Yo (yottaoctet) = 10^{24} octet
1 To (téraoctet) = 10^{12} octet	1 Ro (ronnaoctet2) = 10^{27} octet
1 Po (pétaoctet) = 10^{15} octet	1 Qo (quettaoctet) = 10^{30} octet

2.1. 3. Systèmes de numération

La numération est la science qui s'intéresse à la dénomination et à la représentation graphique des nombres. En technologie numérique, il existe de nombreux systèmes de numération, les plus utilisés sont les systèmes : Décimal (base 10), Binaire (base 2), Tétral (base 4), Octal (base 8) et Hexadécimal (base 16). Pour ce faire, vous devez choisir un système de base B (B est un entier naturel ≥2).

2.1.4. Base, rang et poids

Base : La base d'un système numérique est le nombre de symboles distincts dont on peut réaliser n'importe quelle quantité, et ces symboles sont représentés par des chiffres ou des lettres. Par exemple en base B, les symboles disponibles sont : {0, 1, 2, 3, 4... ,(B-1)}.

• Dans une base B, pour représenter les nombres on utilise B chiffres distincts (symboles).

• La valeur de chaque chiffre doit être strictement inférieure à la base B.

• Chaque chiffre a un **POIDS** selon son **RANG**.

Exemple : En base 10, les chiffres disponibles sont : {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.

Par exemple dans le nombre réel $(5386,12)_{10}$. '5' est le chiffre des milliers, de poids 10^3, etc.

Rang	Poids	
3	Millier	$10^3 = 1000$
2	Centaine	$10^2 = 100$
1	Dizaine	$10^1 = 10$
0	Unité	$10^0 = 1$
-1	Dixième	$10^{-1} = 0.1$
-2	Centième	$10^{-2} = 0,01$

2.1.5. Nombre, Digit

Nombre : est la représentation d'une information dans un système numérique par l'association de chiffres et des lettres. Par exemple, le nombre 2023 : est l'association de chiffres 2.0.3.

Digit : un mot anglais désignant un chiffre ou une lettre.

2.2. Présentation des systèmes décimal, binaire, octal et hexadécimal

2.2.1. Représentation polynomiale

Chaque nombre N peut être décomposé en des puissances entières de la base de son système de numération. Cette décomposition est appelée la forme polynomiale du nombre N et qui est représentée par :

$N=a_nB^n + a_{n-1}B^{n-1} + a_{n-2}B^{n-2} + \ldots + a_2B^2 + a_1B^1 + a_0B^0$

⇒ B : il représente la base du système de numération, c'est à dire le nombre des différents chiffres que ce système de numération utilise.

⇒ a_i : est un chiffre (ou digit) parmi les digits du système de numération utilise.

⇒ i : rang du chiffre ai.

Exemple : la décomposition polynomiale de nombre décimal $(3740,68)_{10}$ est :

$(3740,68)_{10} = a_3 .10^3 + a_2 .10^2 + a_1 .10^1 + a_0 .10^0 + a_{-1} .10^{-1} + a_{-2} .10^{-2}$

Avec a_3=3, a_2=7, a_1=4, a_0=0, et a_{-1}=6 et a_{-2}=8.

2.2.2. Système décimal (base 10)

Le système décimale est le système de numération le plus pratique actuellement [1]. C'est un système qui est venu naturellement de l'homme à 10 doigts. Le système décimal se compose de 10 chiffres décimaux habituels qui sont {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.

Exemples : représentation de quelques nombres décimaux sous forme polynomiale.

$(2023)_{10}= 2*10^3 + 0*10^2 + 2*10^1 + 3*10^0$

$(501,468)_{10} = 5*10^2 + 0*10^1 + 1*10^0 + 4*10^{-1} + 6*10^{-2} + 8*10^{-3}$

- C'est le plus utilisé et le plus connu et que nous utilisons tous les jours.
- Il est basé sur le nombre 10 qui est la base du système décimal.
- Ces chiffres sont disposés de droite à gauche "unités, dizaines, centaine, milliers …".
- Chaque position a un poids. C’est-à-dire est un système positionnel.

2.2.3. Système binaire (base 2)

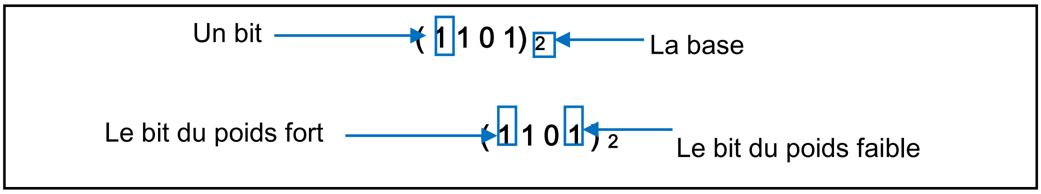
Les informations traitées par l’ordinateur sont de nature différente (Nombres, texte, Images, sons, vidéo, Programmes, …). Cependant, elles sont toujours représentées dans la machine sous forme binaire (une suite de 0 et de 1) car dans un circuit électronique on a deux niveaux de tension (Exemple 0V et 5V ou 5V et 12V) pour représenter n'importe quelle information, qu’elle soit de forme logique ou numérique. Donc, on représente les nombres en système binaire en associant par exemple la valeur binaire 0 à une tension de 0V, et la valeur binaire 1 à une tension de 5V.

Dans ce système binaire il n’y a que deux chiffres possibles qui sont appelés bits « binary digit » {0, 1}. Ce système de numération est le plus couramment utilisé dans l’informatique. Voici quelques nombres binaires sous leurs formes polynomiales.

Exemples :

$(1110011)_2 = 1*2^6 + 1*2^5 + 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

$(11011.1011)_2 = 1*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 + 1*2^{-1} + 0*2^{-2} + 1*2^{-3} + 1*2^{-4}$



2.2.4. Système tétral (base 4)

Le système tétral, également appelé système de numération de base 4 se compose de 4 chiffres qui sont :{0, 1, 2, 3}. Un nombre tétral peut être écrit sous la forme polynomiale, Ecrivons à titre d’exemple :

$(2132)_4 = 2*4^3 + 1*4^2 + 3*4^1 + 2*4^0$

$(210,23)_4 = 2*4^2 + 1*4^1 + 0*4^0 + 2*4^{-1} + 3*4^{-2}$

2.2.5. Système Octal (base 8)

Le système octal autrement dit système à base 8 se compose de 8 chiffres {0, 1, 2, 3, 4, 5, 6, 7}. Les deux chiffres 8 et 9 n’existent pas dans ce système de numération. Certains calculateurs utilisent ce système de numération, Ecrivons à titre d’exemple :

$(573)_8 = 5*8^2 + 7*8^1 + 3*8^0$

$(2374,625)_8 = 2*8^3 + 3*8^2 + 7*8^1 + 4*8^0 + 6*8^{-1} + 2*8^{-2} + 5*8^{-3}$

Anciennement, ce système octal servait au codage des nombres dans les ordinateurs de première génération. Ce système à base 8 est très peu utilisé de nos jours.

2.2.6. Système Hexadécimal (base 16)

Le système de numération à base 16, autrement dit hexadécimal, se compose de 16 chiffres et lettres qui sont {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. Les lettres {A, B, C, D, E, F} représentent les nombres {10, 11, 12, 13, 14, 15} respectivement.

Exemples :

$(A286)_{16} = 10*16^3 + 2*16^2 + 8*16^1 + 6*16^0$

$(C4F)_{16} = 12*16^2 + 4*16^1 + 15*16^0$

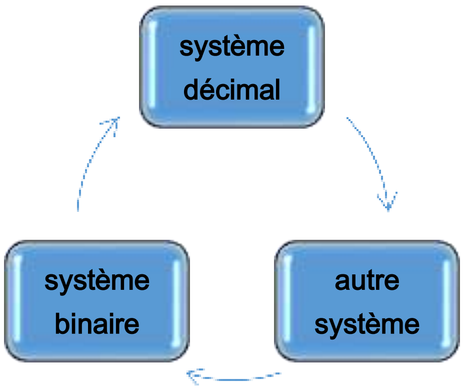
$(5B2A,EF)_{16} = 5*16^3 + 11*16^2 + 2*16^1 + 10*16^0 + 14*16^{-1} + 15*16^{-2}$

Le système hexadécimal est très utilisé dans les ordinateurs et microordinateurs notamment dans le domaine des transmissions de données.

2.3. Conversion entre ces différents systèmes

Qui consiste à convertir un nombre écrit en base B1 en son équivalent en base B2 différente de la base B1. Il existe trois types de conversions, qui sont :

- Codage : est la conversion d'un nombre représenté en système décimal vers un autre système.
- Décodage : est la conversion d'un nombre représenté en un système non décimal vers un système décimal.
- Transcodage : est la conversion entre deux systèmes non décimaux.



2.3.1 Conversion d'un nombre N de base B en un nombre décimal (décodage)

La valeur décimale d'un nombre écrit en base B (non décimale) s'obtient par sa forme polynomiale. Pour la conversion des nombres de la base B quelconque vers le décimal (exemple : Binaire–décimal, octal–décimal et hexadécimal–décimal), Il suffit de multiplier chaque chiffre par son poids correspondant puis d'additionner les résultats obtenus.

Exemples :

$(1010)_2 = (?)_{10}$

$(1010)_2 = 1*2^3 + 0*2^2 + 1*2^1 +0*2^0.$

$=8+2$

$= (10)_{10}$

$(237)_8 = (?)_{10}$

$(237)_8 = 2*8^2 +3*8^1 +7*8^0.$

$=128+24+7$

$= (159)_{10}$

$(3CA)_{16} = (?)_{10}$

$(3CA)_{16} = 3* 16^2 + 12*16^1 + 10*16^0.$

$=768+192+10.$

$= (970)_{10}$

$(1011110)_2 = (?)_{10}$

$(1011110)_2= 1*2^6 + 0*2^5 + 1*2^4 + 1*2^3 + 1*2^2 + 1*2^1+ 0*2^0$

$=(94)_{10}$

$(231103)_4 = (?)_{10}$

$(231103)_4= 2*4^5 + 3*4^4 + 1*4^3 + 1*4^2 + 0*4^1+ 3*4^0$

$=(2899)_{10}$

$(7062)_8 = (?)_{10}$

$(7062)_8= 7*8^3 + 0*8^2 + 6*8^1+ 2*8^0$

$=(3634)_{10}$

$(B7E)_{16} = (?)_{10}$

$(B7E)_{16}= 11*16^2 + 7*16^1 + 14*16^0$

$=(2942)_{10}$

$(110,011)_2 = (?)_{10}$

$(110,011)_2= 1*2^2 + 1*2^1 + 0*2^0 + 0*2^{-1} + 1*2^{-2} + 1*2^{-3}$

$=(6,375)_{10}$

$(0,122)_4 = (?)_{10}$

$(0,122)_4= 0*4^0 + 1*4^{-1} + 2*4^{-2} + 2*4^{-3}$

$=(0.40625)_{10}$

$(70,4)_8 = (?)_{10}$

$(70,4)_8= 7*8^1 + 0*8^0 + 4*8^{-1}$

$=(56,5)_{10}$

$(AE,8)_{16} = (\ ?)_{10}$

$(AE,8)_{16} = 10*16^1 + 14*16^0 + 8*16^{-1}$

$=(174,5)_{10}$

2.3.2. Conversion d'un nombre décimal vers une autre base B : (codage)

2.3.2.1. Conversion d'un nombre décimal entier

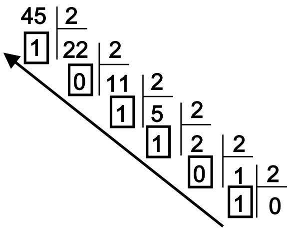
Pour convertir un entier décimal en n'importe quelle base B, des divisions entières successives doivent être effectuées par la base B, et le reste de la division doit être conservé à chaque fois. Nous nous arrêtons jusqu'à ce que le quotient devienne nul. Le nombre cherché sera obtenu en regroupant tous les restes successifs de droite à gauche [2].

La méthode des divisions successives à suivre est :

- On divise le nombre décimal par la base B.
- Puis, On divise le quotient obtenu par la base B.
- Ainsi de suite jusqu'à l'obtention d'un quotient nul.
- en regroupant la suite des restes pour obtenir le nombre cherché dans la base visée.

➦ **Décimal-Binaire**

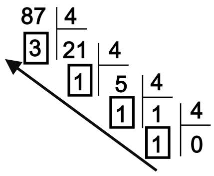
$(45)_{10}=(\ ?)_2$



$(45)_{10}=(101101)_2$

➦ **Décimal-Tétral**

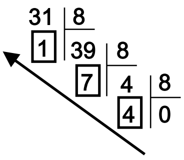
$(87)_{10}=(\ ?)_4$



$(87)_{10}=(\ 1113)_4$

➦ **Décimal-Octal**

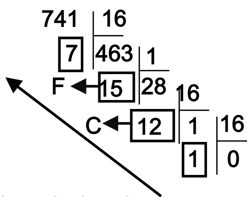
$(213)_{10}=(\ ?)_8$



$(313)_{10}=(471)_8$

➦ **Décimal-Hexadécimal**

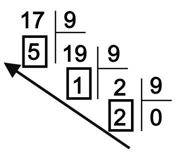
$(7415)_{10}=(\ ?)_{16}$



$(7415)_{10}=(1CF7)_{16}$

➦ **Décimal-Base 9**

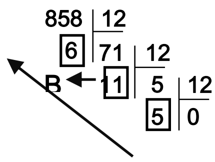
$(176)_{10}=(\ ?)_9$



$(176)_{10}=(215)_9$

➦ **Décimal-Base 12**

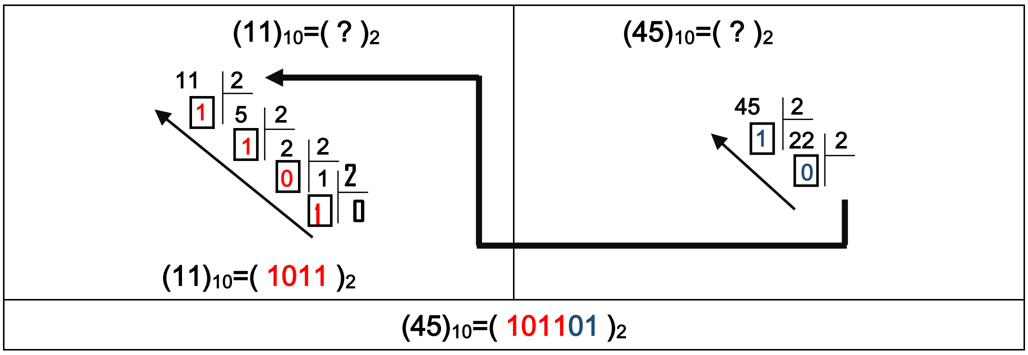
$(858)_{10}=(\ ?)_{12}$



$(858)_{10}=(\ 5B6)_{12}$

Remarque : Si on a deux nombres, et le deuxième nombre est un quotient parmi les quotients des divisions successives de conversion de premier nombre, en peut utiliser leur résultat directement sans refaire les calculs.

Exemple :



2.3.2.2. Conversion d'un nombre décimal à virgule

Chaque nombre réel est composé d'une partie entière et une partie fractionnaire, pour convertir un nombre décimal réel en n'importe quelle base B, il faut :

- Convertir la partie entière en effectuant des divisions successives par B.
- Convertir la partie fractionnaire en effectuant des multiplications successives par B. et en gardant à chaque fois le nombre entier. La partie fractionnaire restante est multiplié par B et ainsi de suite jusqu'à ce qu'il la partie fractionnaire devienne nulle ou que la précision obtenue soit considéré comme suffisante.

Conversion du nombre $(60,625)_{10}$ en base 2

Parfois, si nous multiplions la partie fractionnaire par la base B, nous ne pouvons pas convertir toute la partie fractionnaire. Ceci est essentiellement dû au fait que le nombre converti n'a pas d'équivalent exact en base B et que sa partie fractionnaire est périodique.

Exemple :

$(0,6)_{10}=(\ ?\ )_2$

$0,6 * 2 = \boxed{1},2$

$0,2 * 2 = \boxed{0},4$

$0,4 * 2 = \boxed{0},8$

$0,8 * 2 = \boxed{1},6$

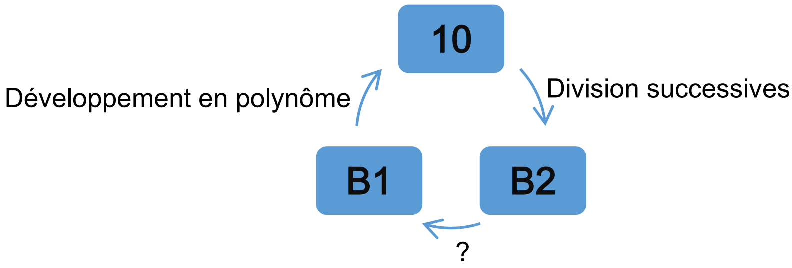
On dit que le nombre $(0,6)_{10}$ est périodique de période (1001) dans la base 2.

$\Rightarrow (0,6)_{10}=(0,10011001....)_2$

2.3.3. Conversion d'un nombre en base B1 à une base B2 (Transcodage)

2.3.3.1. Conversion d'un nombre en base B1 à une base B2 quelconque

Il n'y a aucun moyen de passer directement de la base B1 à une autre base B2. La solution est basée sur l'utilisation de la base 10 comme une base intermédiaire, c'est-à-dire de convertir le nombre de base B1 en base 10, puis de convertir le résultat de base 10 en base B2.



Exemple :

$(257)_8=(\ ?)_7$	
$(257)_8=(\ ?)_{10}$ $(257)_8=2*8^2+5*8^1+7*8^0$ $=(175)_{10}$ $(257)_8=(175)_{10}$	$(175)_{10}=(\ ?)_7$ 17 | 7 $\boxed{0}$ 25 | 7 $\boxed{4}$ 3 | 7 $\boxed{3}$ 0 $(175)_{10}=(340)_7$
$(257)_8=(340)_7$	

2.3.3.2. Conversion d'un nombre en base quelconque b_1 à une base b_2 puissance de b_1(b_1^2, b_1^3,..)

2.3.3.2.1. Conversion d'un nombre en base 2 à une base puissance de 2 (2, 4, 8, 16, …)

Mais si la base **b1** et **b2** s'écrivent respectivement sous la forme d'une puissance de 2 on peut passer par la base 2 (binaire) :

Pour convertir un nombre de n'importe quelle base b1 vers une autre base b2, il est nécessaire d'utiliser la base 10 comme une base intermédiaire entre les deux bases B1 et B2. Mais si les deux bases B1 et B2 sont respectivement écrites en puissances de 2, nous pouvons convertir via la base 2 (binaire) au lieu la base 10.

Base tétrale (base 4) : $4=2^2$ chaque chiffre tétral est converti tout seul sur 2 bits.

Base 4	Base2
0	00
1	01
2	10
3	11

Tableau 2.1 : Correspondance tétrale /Binaire.

Base octale (base 8) : $8=2^3$ chaque chiffre octal est converti tout seul sur 3 bits.

Base 8	Base 2
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Tableau 2.2 : Correspondance Octale /Binaire.

Base hexadécimale (base 16) : $16=2^4$ chaque chiffre hexadécimal est converti tout seul sur 4 bits.

Base 16	Base 2
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

Tableau 2.3 : Correspondance Hexadécimale/Binaire.

Ce remplacement se fait de :

- Droit à gauche pour la partie entière
- Gauche à droite pour la partie fractionnaire.

- Binaire vers tétrale : regrouper les bits en des sous-groupes de deux bits puis remplacer chaque groupe par le symbole correspondant dans la base Tétrale (Tableau 2.1).

- Tétrale vers binaire : Le principe de conversion Tétrale est de remplacer chaque symbole Tétral par sa valeur binaire sur 2 bits (faire des éclatements sur 2 bits).

- Binaire vers octale : regrouper les bits en des sous-groupes de trois bits puis remplacer chaque groupe par le symbole correspondant dans la base octale (Tableau 2.2).

- Octal vers binaire : Le principe de conversion octale est de remplacer chaque symbole octal par sa valeur binaire sur 3 bits (faire des éclatements sur 3 bits).

- Binaire vers Hexadécimale : regrouper les bits en sous-groupes de quatre bits puis remplacer chaque groupe par le symbole correspondant dans la base Hexadécimale (Tableau 2.3).

- Hexadécimale vers binaire : Le principe de conversion Hexadécimale est de remplacer chaque symbole Hexadécimale par sa valeur binaire sur 4 bits (faire des éclatements sur 4 bits).

Exemples :

- $(3\ 0\ 2\ 31)_4=(11\ 00\ 10\ 11\ 01)_2$ $\quad$ $(1\ 3\ 1{,}0\ 2)_4=(01\ 11\ 01{,}00\ 10)_2=(11101{,}001)_2$
- $(5\ 0\ 2\ 7\ 6)_8=(101\ 000\ 010\ \ 111\ 110)_2$ $\quad$ $(4\ 3\ 2,\ 67)_8=(100\ \ 011\ 010\ ,110\ 111)_2$
- $(9\ D\ 2\ 1\ E)_{16}=(1001\ 1101\ 0010\ 0001\ \ 1110)_2$ $\quad$ $(B3,\ FA\ 7)_{16}=(1011\ 0011\ .1111\ 1010\ 0111)_2$
- $(3\ 0\ 2)_4=(11\ 00\ 10)_2$ $\quad$ $(2\ 3\ 1{,}0\ 1)_4=(10\ \ 11\ 01\ ,\ 00\ 01)_2$
- $(1\ 7\ 0\ 5)_8=(001\ 111\ 000\ 101)_2$ $\quad$ $(6\ 5{,}1\ 0\ 7)_8=(110\ 101,\ 001\ 000\ 111)_2$
- $(A\ B\ 4)_{16}=(1010\ 1011\ 0100)_2$ $\quad$ $(E\ F{,}1\ 3)_{16}=(1110\ 1111{,}0001\ 0011)_2$

2.3.3.2.2 Conversion d'un nombre en base 3 à une base puissance de 3 (3^2, 3^3, …)

Si la base **B1** et **B2** s'écrivent respectivement sous la forme d'une puissance de 3 on peut passer par la base 3 au lieu la base 10 :

Base 9 : $9=3^2$ chaque chiffre en base 9 se convertit tout seul sur 2 bits.

Base 9	Base3
0	00
1	01
2	02
3	10
4	11
5	12
6	20
7	21
8	22

Tableau 2.4 : Correspondance Base 9/Base 3

Exemples :

$(123)_9=(010210)_3=(10210)_3$

$(87{,}4)_9=(2221{,}11)_3$

$(21102)_3=(021102)_3=(242)_9$

$(1022{,}201)_3=(1022{,}2010)_3=(38{,}63)_9$

Base 27 : $27=3^3$ chaque chiffre en base 27 se convertit tout seul sur 3 bits.

Base 27	Base 3
0	000
1	001
2	002
3	010
4	011
5	012
6	020
7	021
8	022
9	100
A	101
B	102
C	110
D	111
E	112
F	120
G	121
H	122
I	200
J	201
K	202
L	210
M	211
N	212
O	220
P	221
Q	222

Exemples :

$(A8315)_{27}=(\underline{101}\ \underline{022}\ \underline{010}\ \underline{001}\ \underline{012})_3$

$(58.01G)_{27}=(\underline{012}\ \underline{022}.\ \underline{000}\ \underline{001}\ \underline{121})_3$

$(\underline{102}\ \underline{122}\ \underline{210}\ \underline{001})_3=(BHL1)_{27}$

$(10012112221000,00100002)_3=(010\ 012\ 112\ 221\ 000,001\ 000\ 020)_3$

$=(\underline{010}\ \underline{012}\ \underline{112}\ \underline{221}\ \underline{000},\underline{001}\ \underline{000}\ \underline{020})_3$

$=(35EP0,10G)_{27}$

Tableau 2.5 : Correspondance Base 27/Base 3

Remarque : On peut appliquer la même procédure pour importe base B et ses puissances.

2.4. Opérations de base dans les différents systèmes

Les opérations arithmétiques telles que l'addition, la soustraction, la multiplication et la division en système binaire, sont similaires à celle du système décimal. La seule différence est que le système de nombres décimaux comprend les chiffres de 0 à 9, alors que le système de nombres binaires ne comprend que deux chiffres (0 et 1) [3].

2.4.1. Addition

Base binaire

$$(11011{,}01)_2 + (1001{,}11)_2 = (100101{,}00)_2$$

$$\left(\begin{array}{r} {}^{1\ \ 111\ 1} \\ 11011{,}01 \\ +\quad 1001{,}11 \\ \hline =100101{,}00 \end{array}\right)_2$$

Base tétrale

$$(323{,}01)_4 + (21{,}23)_4 = (1010{,}30)_4$$

$$\left(\begin{array}{r} {}^{11\quad 1} \\ 323{,}01 \\ +\quad 21{,}23 \\ \hline =1010{,}30 \end{array}\right)_4$$

Base octale

$$(7524)_8 + (2157)_8 = (11703)_8$$

$$\left(\begin{array}{r} {}^{11} \\ 7524 \\ +\quad 2157 \\ \hline =\quad 11703 \end{array}\right)_8$$

Base hexadécimale

$$(31A{,}E)_{16} + (95{,}BF)_{16} = (3B0{,}9F)_{16}$$

$$\left(\begin{array}{r} {}^{1\ 1} \\ 31A{,}E0 \\ +\quad 95{,}BF \\ \hline =\quad 3B0{,}9F \end{array}\right)_{16}$$

2.4.2. Soustraction

Base binaire

$$(1100001{,}11)_2 - (11100{,}111)_2 = (1010100{,}111)_2$$

$$\left(\begin{array}{r} 11{}_1 0{}_1 0{}_1 0\ 0\ 1{,}{}_1 1{}_1 1{}_1 0 \\ -\quad {}_1 1{}_1 1\ 1\ 0{}_1 0{,}{}_1 1{}_1 1\ 1 \\ \hline =1\,0\,1\,0\,1\,0\,0{,}\ 1\,1\,1 \end{array}\right)_2$$

Base tétrale

$$(32{,}31)_4 - (13{,}021)_4 = (13{,}223)_4$$

$$\left(\begin{array}{r} 3{}_1 2{,}\,3{}_1 1{}_1 0 \\ -\ {}_1 1\ 3{,}{}_1 0{}_1 2\ 1 \\ \hline =1\ 3{,}\ 2\ 2\ 3 \end{array}\right)_4$$

Base octale

$$(5702)_8 - (1764)_8 = (3716)_8$$

$$\left(\begin{array}{r} 5{}_1 7{}_1 0{}_1 2 \\ -\ {}_1 1{}_1 7{}_1 6\,4 \\ \hline =\ 3\,7\,1\,6 \end{array}\right)_8$$

Base hexadécimale

$$(E5{,}A2)_{16} - (1A{,}EE)_{16} = (CA{,}B4)_{16}$$

$$\left(\begin{array}{r} E{}_1 5{,}{}_1 A{}_1 2 \\ -\quad {}_1 1{}_1 A{,}{}_1 E\,E \\ \hline =\quad C\,A{,}\ B\,4 \end{array}\right)_{16}$$

2.4.3. Multiplication

Base binaire

$$(1101{,}11)_2 * (10{,}1)_2 = (100010{,}011)_2$$

$$\left(\begin{array}{lr}
 & 1101{,}11 \\
* & 10{,}1 \\
\hline
= & 110111 \\
+ & 000000\bullet \\
+ & 110111\bullet\bullet \\
\hline
= & 100010{,}011
\end{array}\right)_2$$

Base tétrale

$$(13{,}2)_4 * (2{,}3)_4 = (110{,}22)_4$$

$$\left(\begin{array}{lr}
 & {}^{1\ 1} \\
 & {}^{2\ 1} \\
 & 13{,}2 \\
* & 2{,}3 \\
\hline
= & 1122 \\
+ & 330\bullet \\
\hline
= & 110{,}22
\end{array}\right)_4$$

Base octale

$$(7{,}4)_8 * (3{,}5)_8 = (33{,}14)_8$$

$$\left(\begin{array}{lr}
 & {}^{1} \\
 & {}^{2} \\
 & 7{,}4 \\
* & 3{,}5 \\
\hline
= & 454 \\
+ & 264\bullet \\
\hline
= & 33{,}14
\end{array}\right)_8$$

Base hexadécimale

$$(1A{,}2)_{16} * (6{,}4)_{16} = (A3{,}48)_{16}$$

$$\left(\begin{array}{lr}
 & {}^{3} \\
 & {}^{2} \\
 & 1A{,}2 \\
* & 6{,}4 \\
\hline
= & 688 \\
+ & 9CC\bullet \\
\hline
= & A3{,}48
\end{array}\right)_{16}$$

2.4.4. Division

Base binaire

$$(100010{,}011)_2 \div (101)_2 = (110{,}111)_2$$

Base tétrale

$$(310{,}1)_4 \div (23)_4 = (10{,}3)_4$$

$$\left(\begin{array}{l|l}
310{,}1 & 23 \\
\cline{2-2}
-23 & 10{,}3 \\
\cline{1-1}
=020 & \\
-00 & \\
\cline{1-1}
=201 & \\
-201 & \\
\cline{1-1}
=000 &
\end{array}\right)_4$$

$$\left(\begin{array}{l|l}
100010{,}011 & 101 \\ \cline{2-2}
-\ 101 & 110{,}111 \\ \cline{1-1}
=0111 & \\
\quad -\ 101 & \\
\quad =\ 0100 & \\
\qquad -\ 000 & \\
\qquad =\ 1000 & \\
\qquad\ -\ 101 & \\
\qquad\ =0011 1 & \\
\qquad\qquad -101 & \\
\qquad\qquad =0101 & \\
\qquad\qquad\ -101 & \\
\qquad\qquad\ =000 &
\end{array}\right)_2$$

Base octale

$$(420{,}7)_8 \div (45)_8 = (7{,}3)_8$$

$$\left(\begin{array}{l|l}
420{,}7 & 45 \\ \cline{2-2}
-403 & 7{,}3 \\
=0157 & \\
\quad -157 & \\
\quad =\ 000 &
\end{array}\right)_8$$

Base hexadécimale

$$(164{,}9)_{16} \div (23)_{16} = (A{,}3)_{16}$$

$$\left(\begin{array}{l|l}
164{,}9 & 23 \\ \cline{2-2}
-15E & A{,}3 \\
=0069 & \\
\quad -\ 69 & \\
\quad =\ 00 &
\end{array}\right)_{16}$$

2.4.5. Autres exemples

$$(11111{,}011)_2 + (11101{,}111)_2 + (10111{,}001)_2 + (110{,}11)_2 = (1011011{,}001)_2$$

$$\left(\begin{array}{rr}
 & {\scriptstyle 10\,1110\ 10\ 10\ \ 10\,1} \\
 & 11111{,}011 \\
+ & 11101{,}111 \\
+ & 10111{,}001 \\
+ & 110{,}110 \\ \hline
= & 1011011{,}001
\end{array}\right)_2$$

$$(110001{,}11)_2 - (11100{,}111)_2 = (1010100{,}111)_2$$

$$\left(\begin{array}{rr}
 & 1\,1_1 0_1 0_1 0\ 0\ 1{,}\,_1 1_1 1_1 0 \\
- & _1 1_1 1\ 1\ 0_1 0{,}\,_1 1_1 1\ 1 \\ \hline
= & 1\ 0\ 1\ 0\ 1\ 0\ 0{,}\ 1\ 1\ 1
\end{array}\right)_2$$

$$(110{,}11)_2 * (110{,}1)_2 = (101011{,}111)_2$$

$$(1000110{,}1)_2 \div (110)_2 = (1011{,}11)_2$$

$$\left(\begin{array}{rr} & 110,11 \\ * & 110,1 \\ \hline = & 11011 \\ + & 000000\bullet \\ + & 11011\bullet\bullet \\ + & 11011\bullet\bullet\bullet \\ \hline = & 101011,111 \end{array}\right)_2$$

$$\left(\begin{array}{l|l} 10001101,1 & 110 \\ \cline{2-2} -\ 110 & 1011,11 \\ \cline{1-1} =0101 & \\ \quad -\ 000 & \\ \quad =\ 1011 & \\ \qquad -\ 110 & \\ \qquad =01010 & \\ \qquad\quad -\ 110 & \\ \qquad\quad =01001 & \\ \qquad\qquad -110 & \\ \qquad\qquad =00110 & \\ \qquad\qquad\quad -110 & \\ \qquad\qquad\quad =000 & \end{array}\right)_2$$

$$(E5,A2)_{16}+(1A,EE)_{16}=(100,9)_{16}$$

$$(E5,A2)_{16}-(1A,EE)_{16}=(CA,B4)_{16}$$

$$\left(\begin{array}{rr} & E5,A2 \\ + & 1A,EE \\ \hline = & 100,90 \end{array}\right)_{16}$$

$$\left(\begin{array}{rr} & E_{1}5,{}_{1}A_{1}2 \\ - & {}_{1}1_{1}A,{}_{1}EE \\ \hline = & CA,B4 \end{array}\right)_{16}$$

2.5. Contage dans les systèmes de numération

En base 5 :

0	0	0	0
0	0	0	1
0	0	0	2
0	0	0	3
0	0	0	4
0	0	1	0
0	0	1	1
0	0	1	2
0	0	1	3
0	0	1	4
0	0	2	0
0	0	2	1
0	0	2	2
0	0	2	3

0	0	2	4
0	0	3	0
0	0	3	1
0	0	3	2
0	0	3	3
0	0	3	4
0	0	4	0
0	0	4	1
0	0	4	2
0	0	4	3
0	0	4	4
0	1	0	0
0	1	0	1
0	1	0	2
0	1	0	3
0	1	0	4
0	1	1	0
0	1	1	1
0	1	1	2
0	1	1	3
0	1	1	4
0	1	2	0
0	1	2	1
0	1	2	2
0	1	2	3
0	1	2	4
0	1	3	0
0	1	3	1
0	1	3	2
0	1	3	3
0	1	3	4
0	1	4	0
0	1	4	1
0	1	4	2
0	1	4	3
0	1	4	4
…	…	…	…

Cas général : pour n'importe base B

Le tableau suivant est un résumé de ces systèmes

Base 10	Base 2	Base 5	Base 7	Base 8	Base 12	Base 16
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	10	2	2	2	2	2
3	11	3	3	3	3	3
4	100	4	4	4	4	4
5	101	10	5	5	5	5
6	110	11	6	6	6	6
7	111	12	10	7	7	7
8	1000	13	11	10	8	8
9	1001	14	12	11	9	9
10	1010	20	13	12	A	A
11	1011	21	14	13	B	B
12	1100	22	15	14	10	C
13	1101	23	16	15	11	D
14	1110	24	20	16	12	E
15	1111	30	21	17	13	F
16	10000	31	22	20	14	10
17	10001	32	23	21	15	11
18	10010	33	24	22	16	12
19	10011	34	25	23	17	13
20	10100	40	26	24	18	14
21	10101	41	30	25	19	15
22	10110	42	31	26	1A	16
23	10111	43	32	27	1B	17
24	11000	44	33	30	20	18
25	11001	100	34	31	21	19
26	11010	101	35	32	22	1A
27	11011	102	36	33	23	1B
28	11100	103	40	34	24	1C
29	11101	104	41	35	25	1D
30	11110	110	42	36	26	1E

Tableau 2.6 : Correspondance des nombres décimaux dans les différentes bases.

Série d'exercices N°1 (Systèmes de Numération)

Exercice N°1 :

1. Donner le tableau de correspondance des 25 premiers nombres décimaux dans les bases suivantes : 10, 2, 5, 6, 8, 11 et 16.

2. On considère les nombres décimaux suivants : $(0)_{10}$, $(11)_{10}$, $(255)_{10}$, $(34,125)_{10}$, $(13,6)_{10}$, $(54,18)_{10}$, Donner leurs représentations en binaire (base 2), en octal (base 8) et puis en hexadécimal (base 16).

3. Trouver les équivalents décimaux des nombres suivants : $(101,11)_2$, $(10000,00)_2$, $(1,1)_2$, $(1234)_8$, $(10,132)_8$, $(111,11)_8$, $(A04,12)_{16}$, $(BAC23)_{16}$.

4. Sans passer par la procédure de division, exprimer directement en binaire les nombres suivants : $X=(1320)_4$, $Y= (307,5)_8$, $Z= (BAC,BEF)_{16}$.

Exercice N°2 :

$(73)_{10}$ = (…………………….)$_7$

$(93,625)_{10}$ = (…………………….)$_2$

$(108)_{10}$ = (………………….)$_8$

$(679,93359375)_{10}$ = (…………………….)$_{16}$

$(4103)_5$ = (…………………….)$_{10}$

$(31121,232)_4$ = (…………………….)$_{10}$

$(2034)_5$=(……………………..)$_9$

$(1023,02)_4$=(……………………..)$_6$

$(104,2)_5$=(…………………….)$_6$

$(10111000,101)_2$=(…………………….)$_4$

$(10110101101,11011)_2$=(…………………….)$_8$

$(100101011100,011101)_2$=(…………………….)$_{16}$

$(135,04)_8$=(…………………….)$_2$

$(A6C,01E)_{16}$=(…………………….)$_2$

$(F92A,20F)_{16}$=(…………………….)$_8$

$(11010110101,01011)_2$=(…………….)$_4$=(………….)$_8$=(…………….)$_{16}$

Exercice N°3 :

1. Donner les nombres qui ont la même représentation dans les systèmes binaire, octal, décimal et hexadécimal.
2. Donner les nombres qui ont la même représentation dans les systèmes octal, décimal et hexadécimal.
3. Lequel des nombres suivants a une signification hexadécimale : BAC- DEUA- CAFE- NIMPORTEQUOI- BAFFE- DECADE- BEF -FA5D-F00D-C0DE-A1DE.
4. Dans une base B, combien de nombres entiers positifs peut-on représenter par n chiffres ?

Exercice N°4 :

1. Déterminer les bases (T, X, Y et Z) dans lesquelles les nombres suivants sont exprimés :

   $(24)_T = (14)_{10}$ $(13)_X = (7)_{10}$ $(70)_Y = (56)_{10}$ $(1A0)_Z = (416)_{10}$.

2. Déterminer les couples des entiers (X, Y) tel que : $(XY)_7 = (YX)_{10}$.
3. Soit le nombre décimal $X = 4a^5+2a^3+a+5$ (tel que a est un entier>5).
   a. Représenter X en base a
   b. Représenter en base a les nombres décimaux suivants : X= a, Y= a^2, Z=a^3

      (Tel que a est un entier >1)

Exercice N°5:

Effectuer les opérations suivantes :

$(1001110,11)_2 + (11011,101)_2 = (\ldots\ldots\ldots)_2$

$(11011,101)_2 + (10111,111)_2 = (\ldots\ldots\ldots)_2$

$(1110,011)_2 + (1101,11)_2 + (1110,111)_2 = (\ldots\ldots\ldots)_2$

$(101001,001)_2 - (11111,11)_2 = (\ldots\ldots\ldots)_2$

$(1011,011)_2 * (110)_2 = (\ldots\ldots\ldots)_2$

$(1001001,11)_2 / (101)_2 = (\ldots\ldots\ldots)_2$

$(73,7)_8 + (65,3)_8 = (\ldots\ldots\ldots)_8$

$(531)_8 - (167)_8 = (\ldots\ldots\ldots)_8$

$(26,5)_8 \times (4,3)_8 = (\ldots\ldots\ldots)_8$

$(31,7)_8 \times (52)_8 = (\ldots\ldots\ldots)_8$

$(C3E)_{16} + (6AD)_{16} = (\ldots\ldots\ldots)_{16}$

$(E31)_{16} - (6EC)_{16} = (\ldots\ldots\ldots)_{16}$

Chapitre III : La représentation de l'information

Chapitre III : La représentation de l’information

3.1. Introduction

Toutes les informations comprennent trois types de caractères :

-des mots composés de lettres.

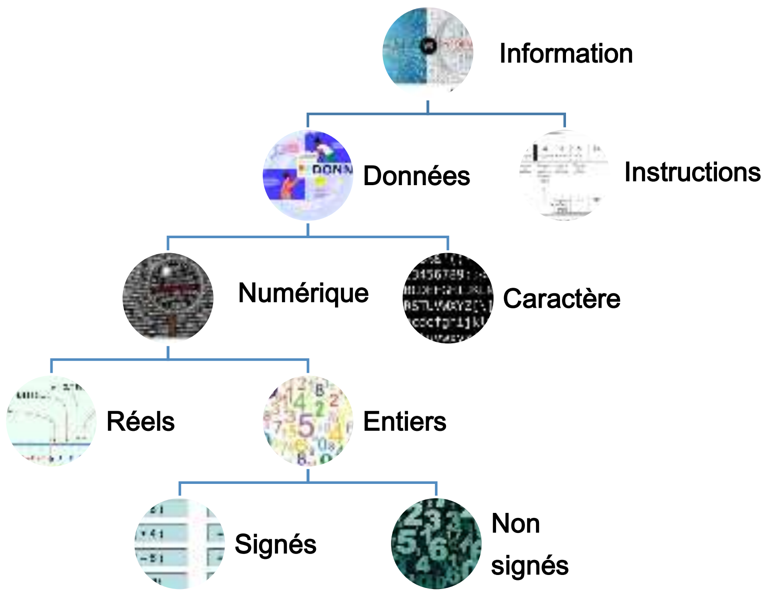
-des numéros représentés par des chiffres.

-des symboles de différentes nature (ponctuation ; opérateurs arithmétiques ou logiques ; commandes auxiliaires, etc.).

Il existe deux types de codes :

- Les codes numériques : ne permettent que l'encodage les nombres.

-Les codes alphanumériques : permet d'encoder n'importe quelle information (lettres, chiffres et symboles).



3.2. Codage de l’information :

Le codage de l’information permet d’établir une correspondance qui permet le passage d'une représentation externe de la même information à une représentation interne (sous forme binaire), selon un ensemble de règles précises [4].

En informatique, l’information est encodée principalement en trois étapes :

- Numérisation : Les informations seront représentées par une suite de chiffres
- Chaque nombre est représenté sous forme binaire (une suite de 0 et de 1)
- Chaque chiffre binaire est exprimé par un état physique.

Exemples :

- Charge électrique (RAM : Condensateur-transistor) : $\begin{cases} \text{bit} = 1 & \text{charge} \\ \text{bit} = 0 & \text{non charge} \end{cases}$
- Alvéoles (CDROM): $\begin{cases} \text{bit} = 1 & \text{Réflexion} \\ \text{bit} = 0 & \text{Pas de réflexion} \end{cases}$

• Magnétisation (Disque dur, disquette) : polarisation : $\begin{cases} \text{bit} = 1 & \text{Nord} \\ \text{bit} = 0 & \text{Sud} \end{cases}$

• Fréquences (Modem) : dans un signal sinusoïdal : $\begin{cases} \text{bit} = 1 & \text{Fréquence F1} \\ \text{bit} = 0 & \text{Fréquence F2} \end{cases}$

3.3. Codage des nombres

3.3.1. Codage des nombres entiers non signés

3.3.1.1. Code binaire pur (Code binaire naturel)

Le code binaire pur n'est utilisé que pour représenter des nombres décimaux. L'équivalent binaire d'un nombre décimal est obtenu en divisant successivement le nombre décimal par 2. Ce code est également appelé le code 8421.

N	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Tableau 3.1 : Code 8421

Exemple :

$N1=(29)_{10}=(11101)_2$

$N2=(36)_{10}=(100100)_2$

$N1+N2=(36)_{10}+(29)_{10}=(65)_{10}=(1000001)_2$

3.3.1.2. Code binaire réfléchi (ou code GRAY)

Ce code permet uniquement de représenter les nombres décimaux. Les codes Gray sont des codes adjacents (où un seul bit change lors du passage d'une valeur à la suivante). Il est également connu sous le nom de binaire réfléchi. Il est utilisé dans les tableaux de Karnaugh et aussi dans la conception numérique.

Première méthode :

Le code Gray peut être généré graphiquement par itération ou à partir du code en binaire naturel [5] :

- Nous commencions par 0 puis nous crée un axe de symétrie,
- Nous ajoutons le bit de poids fort en binaire naturel,
- Pour prolonger le code, nous passer sur N bits,
- Nous recrée un nouvel axe de symétrie sur les deux bits faibles,
- Ensuite, nous ajoutons un bit supplémentaire en binaire naturel,
- Nous recommencions si on veut ajouter un bit supplémentaire.

0	0	00	00	000	000	0000	0000	00000
1	1	01	01	001	001	0001	0001	00001
	1	11	11	011	011	0011	0011	00011
	0	10	10	010	010	0010	0010	00010
			10	110	110	0110	0110	00110
			11	111	111	0111	0111	00111
			01	101	101	0101	0101	00101
			00	100	100	0100	0100	00100
					100	1100	1100	01100
					101	1101	1101	01101
					111	1111	1111	01111
					110	1110	1110	01110
					010	1010	1010	01010
					011	1011	1011	01011
					001	1001	1001	01001
					000	1000	1000	01000
							1000	11000
							1001	11001
							1011	11011
							1010	11010
							1110	11110
							1111	11111
							1101	11101
							1100	11100
							0100	10100
							0101	10101
							0111	10111
							0110	10110
							0010	10010
							0011	10011
							0001	10001
							0000	10000

Deuxième méthode :

Nous trouvons le numéro suivant de chaque numéro gris, pour trouver le numéro suivant, nous devons suivre ces étapes :

1. On commence par 0
2. Si le nombre de 1 est pair on inverse le1 suivant après le premier 1 à droit de nombre et on garde les autres bits du nombre.
3. Si le nombre de 1 est impair on inverse le premier bit de poids faible et on garde les autres bits du nombre.

0
1
11
10
110
111
101
100
1100
1101
1111
1110
1010
1011
1001
1000

Exemple :

$(10001010001)_{Gray} \xrightarrow{\text{Suivant}} (10001010000)_{Gray}$

$(10101010111)_{Gray} \xrightarrow{\text{Suivant}} (10101010101)_{Gray}$

$(11101011010)_{Gray} \xrightarrow{\text{Suivant}} (11101011110)_{Gray}$

Le tableau (3.2) représente une comparaison entre le code binaire naturel et le code Gray :

Décimal	Binaire	Gray
0	0	0
1	1	1
2	10	11
3	11	10
4	100	110
5	101	111
6	110	101
7	111	100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Tableau 3.2 : Comparaison entre le code binaire pur et le code réfléchi (Gray).

➲ Conversion binaire pur en code Gray

Pour passer du binaire pur au code réfléchi (Gray), on cite trois méthodes :

➢ Première méthode :

Dans cette première méthode, pour passer du binaire au code Gray, il faut ajouter au nombre N la valeur de N sans le bit de poids faible. Autrement dit si **N=** $X_4X_3X_2X_1X_0$

1. On garde X_4 le bit du poids fort
2. On élimine X_0 le bit du poids faible
3. On fait l'addition suivante : **(** $X_4X_3X_2X_1X_0$ **+** $X_4X_3X_2X_1$ **)**
4. Le résultat trouvé est en **gray**

Remarque : Nous ne prenons pas la retenue lorsque nous avons 1+1=0 et retenue=1 à ignorer.

Donc 1+1=0.

Exemple :

➦ $(101101)_2=(\ ?)_{Gray}$

101101

\+ 10110

= 111011

$(101101)_2=(111011)_{Gray}$

➦ $(11110001)_2=(\ ?)_{Gray}$

11110001

\+ 1111000

= 10001001

$(11110001)_2=(10001001)_{Gray}$

- **Deuxième méthode :**

Dans cette seconde méthode, pour passer de code binaire au code binaire réfléchi (Gray), nous suivrons ces trois étapes :

1. Écrire le nombre binaire pur et laisser le MSB comme tel
2. Ajouter le MBS au nombre suivant en négligeant la retenue c'est-à-dire lorsque nous l'avons 1+1=0 et retenue=1 à ignorer. Donc 1+1=0.
3. Répéter le processus, Continuez à ajouter chaque bit avec le bit suivant jusqu'au le bit du poids le plus faible.

Exemple

$(1011)_2$

$(1110)_{Gray}$

➲ $(1011)_2=(1110)_{Gray}$

$(1101011000)_2$

$(1011110100)_{Gray}$

➲ $(1101011000)_2=(1011110100)_{Gray}$

- **Troisième méthode :**

Dans cette troisième méthode de passage du code binaire au code binaire réfléchi (Gray). Il suffit de changer de gauche vers la droite le bit qui précède directement le bit 1.

➦ $(101101)_2=(\ ?)_{Gray}$

$(101101)_2=(111011)_{Gray}$

➦ $(11110001)_2=(\ ?)_{Gray}$

$(11110001)_2=(10001001)_{Gray}$

➲ **Conversion d'un binaire réfléchi (Gray) en binaire pur**

si **N=** $X_4X_3X_2X_1X_0$, pour passer du code Gray au binaire, il faut procéder comme suite :

1. On garde X_4 le bit du poids fort
2. On ajoute le bit du poids fort X_4 à X_3
3. Le résultat trouvé dans la 2ème étape est ajouté à X_2
4. Le résultat trouvé dans la 3ème étape est ajouté à X_1
5. Le résultat trouvé dans la 4ème étape est ajouté à X_0

$(1011)_{Gray}$

$(1101)_2$

➲ $(1011)_{Gray}=(1101)_2$

$(11010110110)_{Gray}$

$(10011011011)_2$

➲ $(11010110110)_{Gray}=(10011011011)_2$

$(111011110)_{Gray}$

$(100110111)_2$

➲$(11010110110)_{Gray}=(10011011011)_2$

3.3.1.3. Code DCB (Décimal codé binaire)

Le code BCD (**Binary Coded Decimal**) est également un code largement utilisé qui permet la représentation des nombres sous leur forme décimale. Il s'agit de coder les chiffres de 0 à 9 en binaire sur 4 bits. Dans ce code aucune valeur n'est supérieure à 9. Le principe de ce code consiste à faire des éclatements sur 4 bits et de remplacer chaque chiffre décimal par sa valeur binaire correspondante sur 4bits.

• Les combinaisons supérieures à 9 sont interdites

décimal	**BCD**
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Tableau 3.3 : Représentation des chiffres en BCD

Nous avons constaté qu'il s'agissait d'un code incomplet car il n'utilisait que 10 des 16 combinaisons possibles. Le code BCD est utile lorsque l'on veut manipuler des décimales en électronique numérique (par exemple pour le dialogue humain).

Exemples :

$(5041)_{10}=(\underline{0101}\ \underline{0000}\ \underline{0100}\ \underline{0001})_{BCD}$

$(238)_{10}=(\underline{0010}\ \underline{0011}\ \underline{1000})_{BCD}$

$(7694)_{10}=(\underline{0111}\ \underline{0110}\ \underline{1001}\ \underline{0100})_{BCD}$

BCD n'est pas un système de numération mais un code. Le code BCD est utilisé dans les systèmes d'affichage de chiffres décimaux.

La représentation en BCD a toutefois des inconvénients [6]. Parmi les inconvénients du codage BCD est qu'il ne se prête pas directement aux opérations arithmétiques : en effet, additionner deux valeurs dont la somme est comprise entre (10)10 et (15)10 produit un nombre binaire sans signification. De plus, lorsque la somme de deux décimales est supérieure à (9)10, la machine doit effectuer un ajustement décimal, qui consiste à ajouter la valeur (6)10 à la somme des deux chiffres [7].

Addition et soustraction en BCD

Pour calculer la somme de deux nombre en code BCD, il faut considérer les trois cas suivants :

- Le résultat de la somme ne dépasse pas 9 (1001) et la retenue est égale à 0, dans ce cas l'opération d'addition se fait comme en code binaire pur et le résultat obtenu est correct.

➦ $(8)_{10}+(1)_{10}$ en BDC

$$\begin{array}{lcl} 8 & \xrightarrow{\text{BCD}} & 1000 \\ + & & + \\ \underline{1} & \xrightarrow{\text{BCD}} & \underline{0001} \\ = 9 & & = \underline{1001} \\ & & \quad 9 \end{array}$$

➦ $(5)_{10}+(3)_{10}$ en BDC

$$\begin{array}{lcl} 5 & \xrightarrow{\text{BCD}} & 0101 \\ + & & + \\ \underline{3} & \xrightarrow{\text{BCD}} & \underline{0011} \\ = 8 & & = \underline{1000} \\ & & \quad 8 \end{array}$$

➦ $(33)_{10}+(52)_{10}$ en BDC

$$\begin{array}{lcl} 33 & \xrightarrow{\text{BCD}} & 0011 \;\; 0011 \\ + & & + \\ \underline{52} & \xrightarrow{\text{BCD}} & \underline{0101 \;\; 0010} \\ = 85 & & = \underline{1000} \;\; \underline{0101} \\ & & \quad 8 \qquad 5 \end{array}$$

➢ Le résultat de la somme ne dépasse pas 9 (1001) et la retenue est égale à 1, dans ce cas l'opération d'addition se fait comme en code binaire pur mais le résultat obtenu est incorrect. Pour obtenir le résultat correct, il faut ajouter 6 (0110) au résultat provisoire.

➦ **$(7)_{10}+(9)_{10}$ en BDC**

$$\begin{array}{lcr}
7 & \xrightarrow{BCD} & 0111 \\
+ & & + \quad\quad\quad \\
9 & \xrightarrow{BCD} & 1001 \\
\hline
= \ 16 & & = 0001\ 0000 \\
 & & + 0110 \\
\hline
 & & = \underline{0001}\ \ \underline{0110} \\
 & & 1 \quad\quad 6
\end{array}$$

➦ **$(27)_{10}+(19)_{10}$ en BDC**

$$\begin{array}{lcr}
27 & \xrightarrow{BCD} & 0010\ \ 0111 \\
+ & & + \quad\quad\quad\quad \\
19 & \xrightarrow{BCD} & 0001\ \ 1001 \\
\hline
= \ 46 & & = 0100\ 0000 \\
 & & + 0110 \\
\hline
 & & = \underline{0100}\ \ \underline{0110} \\
 & & 4 \quad\quad 6
\end{array}$$

- 7+9 : on voit que le résultat obtenu est incorrect car il n'est pas égal à 16, Si on divise ce nombre en deux groupes de 4 bits à partir de la gauche, on obtient 0001 0000, donc pour le corriger, il faut ajouter 6 (0110), ce qui donnera (0001 0000) + (0110) =0001 0110 qui est l'équivalent de 16 en BCD.
- 27+19 : on voit que le résultat obtenu est incorrect car il n'est pas égal à 46, Si on divise ce nombre en deux groupes de 4 bits à partir de la gauche, on obtient 0100 0000, donc pour le corriger il faut ajouter 6 (0110) sur la partie de droite qui a la retenue de 1, ce qui donnera (0100 0000)+(0110) = 0100 0110 qui est l'équivalent de 46 en BCD.

➢ Le résultat de la somme dépasse 9 et la retenue est égale à 0, dans ce cas l'opération d'addition se fait comme en code binaire pur mais le résultat obtenu est incorrect. Pour obtenir le résultat correct, il faut ajouter 6 (0110) au résultat provisoire.

➦ **$(8)_{10}+(4)_{10}$ en BDC**

$$
\begin{array}{lcr}
\ 8 & \xrightarrow{\text{BCD}} & 1000 \\
+ & & + \\
\underline{\ 4} & \xrightarrow{\text{BCD}} & \underline{0100} \\
=\ 12 & & =\ 1100 \\
 & & \underline{+\ 0110} \\
 & & =\ \underline{0001}\ \underline{0010} \\
 & & 1 \qquad 2
\end{array}
$$

➦ **$(25)_{10}+(19)_{10}$ en BDC**

$$
\begin{array}{lcr}
\ 25 & \xrightarrow{\text{BCD}} & 0010\ \ 0101 \\
+ & & + \\
\underline{\ 19} & \xrightarrow{\text{BCD}} & \underline{0001\ \ 1001} \\
=\ 44 & & =\ 0011\ \ 1110 \\
 & & \underline{+\ 0110} \\
 & & =\ \underline{0100}\ \ \underline{0100} \\
 & & 4 \qquad 4
\end{array}
$$

- 8+4 : on voit que le résultat obtenu est incorrect car il n'est pas égal à 12, donc pour le corriger, il faut ajouter 6 (0110), ce qui donnera 1100+0110 = 10010. On divise ce nombre en deux groupes de 4 bits à partir de la gauche, on obtient 0001 0010 qui est l'équivalent de 12 en BCD.
- 25+19 : on voit que le résultat obtenu est incorrect car il n'est pas égal à 44, donc pour le corriger, il faut ajouter 6 (0110) à la partie se trouvant à droite, sachant que cette dernière qui est égale à (1110) est supérieur à 9 (1001), on obtient (0011 1110) + (0110)=(0100 0100) qui est l'équivalent de 44 en BCD.

Exemple :

➦ **$(357)_{10}+(579)_{10}$ en BDC**

$$
\begin{array}{lcr}
\ 357 & \xrightarrow{\text{BCD}} & 0011\ 0101\ \ 0111 \\
+ & & + \\
\underline{\ 579} & \xrightarrow{\text{BCD}} & \underline{0101\ \ 0111\ 1001} \\
=\ 936 & & =\ 1000\ \ 1101\ 0000 \\
 & & \underline{+\ 0110} \\
 & & =\ 1000\ 1101\ \ 0110 \\
 & & \underline{+\ 0110} \\
 & & =\ \underline{1001}\ \ \underline{0011}\ \ \underline{0110} \\
 & & 9 \qquad 3 \qquad 6
\end{array}
$$

-Le premier bloc (4 premiers bits) du résultat est incorrect, on ajoute +6 (0110) au groupe de droite, on garde le nibble et on ajoute la retenue au bloc suivant,

-Le bloc suivant, après l'addition, donne une retenue, on garde le nibble et on ajoute la retenue au bloc suivant.

Principe de la soustraction en code BCD

Lors d’une opération de soustraction en code BCD ; L'ordre relatif de leurs valeurs absolues doit être pris en compte ; le signe du résultat est le signe du plus grand nombre ; lorsqu'une retenue terminale est détectée, une correction doit être apportée.

Le processus de correction consiste à soustraire 6 (0110) ; si une retenue terminale apparaît pour le dernier quartet, le résultat est incorrect car le classement des valeurs absolues n’a pas été respecté.

- **$(92)_{10}$-$(48)_{10}$ en BDC**

$$
\begin{array}{lcr}
92 & \xrightarrow{\text{BCD}} & 1001\ 0010 \\
- & & - \\
\underline{48} & \xrightarrow{\text{BCD}} & \underline{0100\ 1000} \\
= 44 & & =\ 0100\ \ 1010 \\
 & & \underline{-\ 0110} \\
 & & =\ \underline{0100}\ \ \underline{0100} \\
 & & 4 \qquad 4
\end{array}
$$

- **$(244)_{10}$-$(89)_{10}$ en BDC**

$$
\begin{array}{lcr}
244 & \xrightarrow{\text{BCD}} & 0010\ 0100\ 0100 \\
- & & - \\
\underline{89} & \xrightarrow{\text{BCD}} & \underline{0000\ \ 1000\ 1001} \\
=\ 155 & & = 0001\ 1011\ 1011 \\
 & & \underline{-0110-0110} \\
 & & = \underline{0001}\ \ \underline{0101}\ \underline{0101} \\
 & & 1 \qquad 5 \qquad 5
\end{array}
$$

3.3.1.4. Code excède de trois (Le codage EXCESS3 ou BCD+3)

Le code décimal binaire Excess 3 (XS3), également connu sous le nom code de Stibitz, est un système numérique non pondéré auto réfléchi, principalement utilisé par les anciens processeurs pour représenter les nombres décimaux. Dans ce code XS3, chaque chiffre du nombre décimal est représenté pour par sa représentation binaire sur quatre bits additionné de 3 (0011). Ainsi le code XS3 d'un nombre décimal est similaire à son code BCD, sauf que chaque groupe de quatre bits est incrémenté de trois. Pour passer de 0 à 9, il suffit de remplacer les uns par des zéros et des zéros dans les uns comme pour faire le complément de 1 en binaire pur. Pour avoir le code BDC+3 du chiffre 8, il suffit de prendre le code BCD+3 du chiffre 1 et chercher son cà1.

décimal	BCD	BCD+3
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001
7	0111	1010
8	1000	1011
9	1001	1100

Tableau 3.4 : Représentation des chiffres en BCD+3

Exemple :

$(238)_{10}=(\underline{0101}\ \underline{0110}\ \underline{1011})_{BCD+3}$

$(7694)_{10}=(\underline{1010}\ \underline{1001}\ \underline{1100}\ \underline{0111})_{BCD+3}$

Addition en BCD+3

Pour calculer la somme de deux nombres en code XS3, vous devez commencer par la conversion des deux nombres en BCD+3, puis additionner. Pour additionner deux nombre BCD+3, Le résultat obtenu est incorrect, pour le corriger il faut prendre en compte les trois cas suivants :

1. Il faut enlever 3 (0011) si la retenue=0.
2. Il faut ajouter 3 (0011) si la retenue=1.
3. Il faut ajouter 3 (0011) si on a un nouveau groupe de 4 bits.

➦ **$(6)_{10}+(2)_{10}$ en BDC+3**

$$\begin{array}{lcr} 6 & \xrightarrow{BCD+3} & 1001 \\ + & & + \\ 2 & \xrightarrow{BCD+3} & 0101 \\ \hline = 8 & & = 1110 \\ & & -0011 \\ \hline & & =\underline{1011} \\ & & 8 \end{array}$$

Si la retenue=0 → Le résultat de l’addition est incorrect, pour le corriger, il faut enlever trois (0011).

➦ $(72)_{10}+(19)_{10}$ en BDC+3

72	$\xrightarrow{\text{BCD+3}}$	1010 0101
+		+
19	$\xrightarrow{\text{BCD+3}}$	0100 1100
= 91		= 1111 0001
		-0011 +0011
		= 1100 0100
		9 1

Si la retenue=1 → Le résultat de l'addition est incorrect, pour le corriger, il faut ajouter trois (0011).

3.3.2. Codage des nombres entiers signés

Pour effectuer les opérations de base (addition, soustraction, multiplication et division), Il existe trois représentations possibles qui sont :

- Représentation en signe et valeur absolue SVA (module et signe MS)
- Représentation en complément à « 1 » → (Cà1)
- Représentation en complément à « 2 » → (Cà2)

3.3.2.1. Représentation par signe et valeur absolue

• Si on travaille sur n bits, alors le bit du poids fort est utilisé pour indiquer le signe du nombre :

$$\begin{cases} S = 0 & \text{si le nombre} > 0 \\ S = 1 & \text{si le nombre} < 0 \end{cases}$$

• Les (n -1) bits suivants désignent la valeur absolue du nombre.

Exemple : Si on a 6 bits

(+18) $\xrightarrow{\text{SVA sur 6 Bits}}$ 0 $\boxed{10010}$

Signe Valeur absolue

010010 est la représentation de +18

(+18) $\xrightarrow{\text{SVA sur 6 Bits}}$ 1 $\boxed{10010}$

Signe Valeur absolue

010010 est la représentation de -18

Exemple : Représenter le nombre (-15) en signe et valeur absolue sur 8 bits

$(-15)_{10}= (-1111)_2 \xrightarrow{\text{SVA sur 8 Bits}}$

1	0	0	0	1	1	1	1

Il est impossible de représenter le chiffre -15 sur 4 bits car sa valeur absolue $|-(15)_{10}|$ qui est égale à $(1111)_2$ prends déjà 4 bits et donc on aura besoin au minimum de 5 bits pour pouvoir représenter son bit de signe.

S	VA				Nombre en décimal
0	0	0	0	0	+0
0	0	0	0	1	+1
0	0	0	1	0	+2
0	0	0	1	1	+3
0	0	1	0	0	+4
0	0	1	0	1	+5
0	0	1	1	0	+6
0	0	1	1	1	+7
0	1	0	0	0	+8
0	1	0	0	1	+9
0	1	0	1	0	+10
0	1	0	1	1	+11
0	1	1	0	0	+12
0	1	1	0	1	+13
0	1	1	1	0	+14
0	1	1	1	1	+15

S	VA				Nombre en décimal
1	0	0	0	0	-0
1	0	0	0	1	-1
1	0	0	1	0	-2
1	0	0	1	1	-3
1	0	1	0	0	-4
1	0	1	0	1	-5
1	0	1	1	0	-6
1	0	1	1	1	-7
1	1	0	0	0	-8
1	1	0	0	1	-9
1	1	0	1	0	-10
1	1	0	1	1	-11
1	1	1	0	0	-12
1	1	1	0	1	-13
1	1	1	1	0	-14
1	1	1	1	1	-15

Tableau 3.5 : Représentation des nombres par la méthode Signe et Valeur Absolue

D'après le tableau (3.5), on peut représenter sur 5 bits, l'intervalle des nombres entiers : de [-(2^4-1) , (2^4-1)] soit de [-15 , +15].

Plus généralement, si on travaille sur n bits, l'intervalle nombres entiers qu'on peut représenter en SVA est : [-(2^{n-1} - 1), +(2^{n-1} - 1)].

Cette méthode présente deux inconvénients :

- Le zéro possède deux représentations distinctes (00000) et (10000) soit +0 et -0 [8];
- Les opérations de l'addition et de la multiplication sont compliquées, car le bit de signe doit être traité à part.

3.3.2.2. Représentation par Complément restreint (ou Complément à 1)

La représentation en complément à 1 est concerné les nombres négatifs

$$A = A_n A_{n-1} A_{n-2} \cdots\cdots\cdots A_2 A_1 A_0 \quad \Rightarrow \quad (A)_{Cà1} = \overline{A}_n \overline{A}_{n-1} \overline{A}_{n-2} \cdots\cdots\cdots \overline{A}_2 \overline{A}_1 \overline{A}_0$$

• En Complément restreint, le bit du poids fort nous indique le signe du nombre :

$$\text{bit de signe} = \begin{cases} 1 & \text{si le nombre positif} \\ 0 & \text{si le nombre négatif} \end{cases}$$

• Cà1 du Cà1 d'un nombre est égale au nombre lui-même, c'est-à-dire Cà1(Cà1(N))= N.

• La somme d'un nombre binaire et de son Cà1 est un nombre binaire composé uniquement de 1.

•Le bit de poids fort est utilisé pour représenter le bit de signe du nombre :

Exemple : Représentation en Cà1 des nombres suivants sur 8 bits :

$$(-20) \xrightarrow{\text{SVA sur 8 Bits}} 1|0010100 \xrightarrow{\text{Cà1}} 1|\ 1101011$$

$$(+20) \xrightarrow{\text{SVA sur 8 Bits}} 0|0010100 \xrightarrow{\text{Cà1}} 0|0010100$$

Nombre en Cà1					Nombre en SVA	Nombre en décimal
0	0	0	0	0	00000	+0
0	0	0	0	1	00001	+1
0	0	0	1	0	00010	+2
0	0	0	1	1	00011	+3
0	0	1	0	0	00100	+4
0	0	1	0	1	00101	+5
0	0	1	1	0	00110	+6
0	0	1	1	1	00111	+7
0	1	0	0	0	01000	+8
0	1	0	0	1	01001	+9
0	1	0	1	0	01010	+10
0	1	0	1	1	01011	+11
0	1	1	0	0	01100	+12
0	1	1	0	1	01101	+13
0	1	1	1	0	01110	+14
0	1	1	1	1	01111	+15

Nombre en Cà1					Nombre en SVA	Nombre en décimal
1	0	0	0	0	11111	-15
1	0	0	0	1	11110	-14
1	0	0	1	0	11101	-13
1	0	0	1	1	11100	-12
1	0	1	0	0	11011	-11
1	0	1	0	1	11010	-10
1	0	1	1	0	11001	-9
1	0	1	1	1	11000	-8
1	1	0	0	0	10111	-7
1	1	0	0	1	10110	-6
1	1	0	1	0	10101	-5
1	1	0	1	1	10100	-4
1	1	1	0	0	10011	-3
1	1	1	0	1	10010	-2
1	1	1	1	0	10001	-1
1	1	1	1	1	10000	-0

Tableau 3.6 : Représentation des nombres par la méthode Cà1.

D'après le tableau (3.6), on peut déduire que sur 5 bits :

- Le plus grand nombre positif est (01111), ce qui donne (2^4–1), soit (+15)
- Le plus petit nombre négatif est (11111), ce qui donne –(2^4-1), soit (-15).

On constate donc que sur 5 bits nous pouvons représenter des nombres en cà1 qui sont dans l'intervalle [-15 ,+15], soit [-(2^4-1) , +(2^4-1)].

En général, si on travaille sur n bits, l'intervalle des valeurs que l'on peut représenter en Cà1 est : [-(2^{n-1} -1), +(2^{n-1} -1)].

Cette méthode présente un inconvénient. Le zéro possède deux représentations distinctes. Par exemple sur 8 bits $+(0)_{10}=(00000000)_2$, $-(0)_{10}=$ Cà1(10000000)= $(11111111)_2$.

Addition en complément à 1 :

- $(+20)_{10} + (+11)_{10}$ en complément à 1 sur 8 bits

$$\begin{array}{rcccl}
+20 & \xrightarrow{\text{SVA sur 8 Bits}} & 0|0010100 & \longrightarrow & 0|0010100 \\
+ & & & & + \\
\underline{+11} & \xrightarrow{\text{SVA sur 8 Bits}} & 0|0001011 & \longrightarrow & \underline{0|0001001} \\
= +31 & & & & =0|0011101 \\
 & & & & \downarrow \quad \downarrow \\
 & & & & (+ \quad 31)_{10}
\end{array}$$

- $(+20)_{10} + (-11)_{10}$ en complément à 1 sur 8 bits

$$\begin{array}{rcccl}
+20 & \xrightarrow{\text{SVA sur 8 Bits}} & 0|0010100 & \longrightarrow & 0|0010100 \\
+ & & & & + \\
\underline{-11} & \xrightarrow{\text{SVA sur 8 Bits}} & 1|0001011 & \xrightarrow{\text{C à 1}} & \underline{1|1110100} \\
= +9 & & & & =\boxed{1}\,0|0001000 \\
 & & & & \underline{+ \qquad\qquad 1} \\
 & & & & = 0|0001001 \\
 & & & & \downarrow \quad \downarrow \\
 & & & & (+ \quad 9)_{10}
\end{array}$$

- $(-20)_{10} + (+11)_{10}$ en complément à 1 sur 8 bits

$$\begin{array}{rcccl}
-20 & \xrightarrow{\text{SVA sur 8 Bits}} & 1|0010100 & \xrightarrow{\text{C à 1}} & 1|1101011 \\
+ & & & & + \\
\underline{+11} & \xrightarrow{\text{SVA sur 8 Bits}} & 0|0001011 & \longrightarrow & \underline{0|0001011} \\
= -9 & & & & =1|1110110 \\
 & & & & \downarrow \quad \downarrow \text{C à 1} \\
 & & & & \quad 0001001 \\
 & & & & \downarrow \quad \downarrow \\
 & & & & (- \quad 9)_{10}
\end{array}$$

➦ **$(-20)_{10} + (-11)_{10}$ en complément à 1 sur 8 bits**

$$-20 \xrightarrow{\text{SVA sur 8 Bits}} 1|0010100 \xrightarrow{\text{C à 1}} 1|1101011$$

$$+ \qquad\qquad\qquad\qquad\qquad\qquad +$$

$$-11 \xrightarrow{\text{SVA sur 8 Bits}} 1|0001011 \xrightarrow{\text{C à 1}} \underline{1|1110100}$$

$$= -31 \qquad\qquad\qquad\qquad = 11|1011111$$

$$\underline{+ \longrightarrow 1}$$

$$= 1|1100000$$

$$\downarrow \text{C à 1}$$

$$0011111$$

$$\downarrow$$

$$(-\ 31)_{10}$$

3.3.2.3. Représentation par Complément Vrai (ou Complément à 2)

Le complément vrai ou Cà2 d'un nombre négatif est Cà1 de ce nombre auquel on ajoute un 1 au bit de poids faible. $(A)_{Cà2}= (A)_{Cà1} +1$. On représente un nombre positif de manière standard par son écriture binaire [9].

Exemple : Représentation en Cà2 les nombres suivants sur 8 bits

$$-20 \xrightarrow{\text{SVA sur 8 Bits}} 1|0010100 \xrightarrow{\text{Cà1}} 1|\ 1101011 \xrightarrow{\text{Cà2}} 1|\ 1101100$$

$$+20 \xrightarrow{\text{SVA sur 8 Bits}} 0|0010100 \xrightarrow{\text{Cà1}} 0|0010100 \xrightarrow{\text{Cà2}} 0|0010100$$

Il existe quatre méthodes pour calculer le complément à 2 d'un nombre binaire.

Première méthode

La première méthode consiste à le soustraire à la puissance de 2 immédiatement supérieure.

Exemple : Cà2 du nombre $N=(10001110)_2$

```
  100000000
-  10001110
-----------
=  01110010
```

Donc Cà2(10001110) = 01110010

Deuxième méthode

Elle consiste à trouver d'abord le complément à 1 et à ajouter 1 au résultat. Par exemple : $N=(10001110)_2$, Cà1(10001110) = 01110001

```
  01110001
+        1
----------
= 01110010
```

Donc Cà2(10001110) = 01110010

Troisième méthode

Cette méthode consiste à conserver tous les bits nuls (égaux à 0) à partir de bit de poids plus faible jusqu'au premier 1 compris et en inversant les autres bits restants. Pour l'exemple précédent, on obtient :

Cà2(100011$\boxed{10}$) = 011100$\boxed{10}$

bits conservés

Quatrième méthode

Cette méthode consiste à conserver tous les bits nuls à partir du bit du poids faible jusqu'au premier 1 compris, et à trouver le Cà2 pour ce bit et le Cà1 pour les autres restants. Pour l'exemple précédent, on obtient :

Cà2($\boxed{100011}$ $\boxed{1}$ $\boxed{0}$) = $\boxed{011100}$ $\boxed{1}$ $\boxed{0}$

Cà1 Cà2 bits conservés

Nombre en Cà2					Nombre en SVA	Nombre en décimal
1	0	0	0	0	10000	-16
1	0	0	0	1	11111	-15
1	0	0	1	0	11110	-14
1	0	0	1	1	11101	-13
1	0	1	0	0	11100	-12
1	0	1	0	1	11011	-11
1	0	1	1	0	11010	-10
1	0	1	1	1	11001	-9
1	1	0	0	0	11000	-8
1	1	0	0	1	10111	-7
1	1	0	1	0	10110	-6
1	1	0	1	1	10101	-5
1	1	1	0	0	10100	-4
1	1	1	0	1	10011	-3
1	1	1	1	0	10010	-2
1	1	1	1	1	10001	-1

Nombre en Cà2					Nombre en SVA	Nombre en décimal
0	0	0	0	0	00000	+0
0	0	0	0	1	00001	+1
0	0	0	1	0	00010	+2
0	0	0	1	1	00011	+3
0	0	1	0	0	00100	+4
0	0	1	0	1	00101	+5
0	0	1	1	0	00110	+6
0	0	1	1	1	00111	+7
0	1	0	0	0	01000	+8
0	1	0	0	1	01001	+9
0	1	0	1	0	01010	+10
0	1	0	1	1	01011	+11
0	1	1	0	0	01100	+12
0	1	1	0	1	01101	+13
0	1	1	1	0	01110	+14
0	1	1	1	1	01111	+15

Tableau 3.7 : Représentation des nombres par la méthode Cà2.

Sachant que le bit du poids fort est utilisé pour représenter le signe du nombre, on peut déduire que sur 5 bits :

- Le plus grand nombre positif représentable est donc 01111 ce qui représente 2^4-1 soit +15
- Le plus petit négatif est codé par 10000, ce qui donne la valeur binaire -1000, soit -16.

Donc, d'après le tableau (3.7), on constate que sur 5 bits, on peut représenter les nombres qui sont dans l'intervalle [-16, +15], soit $[-2^4, +(2^4-1)]$.

- En général, si on travaille sur n bits, l'intervalle des nombres que l'on peut représenter en Cà2 est : $[-2^{n-1}, +(2^{n-1}-1)]$.

Addition en complément à 2

L'avantage de la représentation en Cà2 est qu'on ne traite pas les signes lors des calculs.

➦ **$(+20)_{10}$ + $(+11)_{10}$ en complément à 2 sur 8 bits**

$$+20 \xrightarrow{\text{SVA sur 8 Bits}} 0|0010100 \longrightarrow 0|0010100$$

$$+$$

$$+11 \xrightarrow{\text{SVA sur 8 Bits}} 0|0001011 \longrightarrow 0|0001001$$

$$= +31 \qquad\qquad = 0|0011101 \rightarrow (+\ 31)_{10}$$

➦ **$(+20)_{10}$ + $(-11)_{10}$ en complément à 2 sur 8 bits**

$$+20 \xrightarrow{\text{SVA sur 8 Bits}} 0|0010100 \longrightarrow 0|0010100$$

$$+$$

$$-11 \xrightarrow{\text{SVA sur 8 Bits}} 1|0001011 \xrightarrow{\text{C à 2}} 1|1110101$$

$$= +9 \qquad\qquad = \boxed{\cancel{1}}\,0|0001001 \rightarrow (+\ 9)_{10}$$

➦ **$(-20)_{10}$ + $(+11)_{10}$ en complément à 2 sur 8 bits**

$$-20 \xrightarrow{\text{SVA sur 8 Bits}} 1|0010100 \xrightarrow{\text{C à 2}} 1|1101100$$

$$+$$

$$+11 \xrightarrow{\text{SVA sur 8 Bits}} 0|0001011 \longrightarrow 0|0001011$$

$$= -9 \qquad\qquad = 1|1110111 \xrightarrow{\text{C à 2}} 0001001 \rightarrow (-\ 9)_{10}$$

➦ **$(-20)_{10} + (-11)_{10}$ en complément à 2 sur 8 bits**

$$
\begin{array}{llll}
 & -20 \xrightarrow{\text{SVA sur 8 Bits}} 1|0010100 \xrightarrow{\text{C à 2}} & 1|1101100 \\
+ & & + \\
 & -11 \xrightarrow{\text{SVA sur 8 Bits}} 1|0001011 \xrightarrow{\text{C à 2}} & \underline{1|1110101} \\
= & -31 & \boxed{\cancel{1}}\ 1|\ 1100001 \\
 & & \downarrow \text{C à 2} \\
 & & 0011111 \\
 & & (-\ \ 31)_{10}
\end{array}
$$

3.3.3. Codage des nombres fractionnaires

Les nombres fractionnaires sont ceux qui contiennent des chiffres après la virgule [10]. Dans le système décimal, on écrit par exemple :

$78{,}25 = 7*10^1+8*10^0+2*10^{-1}+5*10^{-2}$

En général, en base b, on écrit :

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

3.3.3.1. Virgule fixe

Un nombre fractionnaire binaire se compose de deux parties, une pour les valeurs entières et l'autre pour les valeurs fractionnaires, la position de la virgule fixe est fictive. Si on a n bits, alors

- le bit du poids fort est utilisé pour indiquer le signe :

$$\begin{cases} S = 0 & \text{si le nombre} > 0 \\ S = 1 & \text{si le nombre} < 0 \end{cases}$$

- Les autres bits (n -1) désignent les valeurs entière et fractionnaire du nombre.

Soit un nombre N=-(1101010,10101)₂. Le codage du nombre en virgule fixe consiste à définir la position de la virgule selon un format donné, c'est-à-dire la taille de la partie entière ainsi que la taille de partie fractionnaire.

1	0	0	0	0	1	1	0	1	0	1	0	1	0	1	0	1	0	0	0
Signe	Partie entière sur 11bits											Partie fractionnaire sur 8bits							

Tableau 3.8 : Représentation d'un nombre réel par la méthode de la virgule fixe

La représentation des nombres à virgule fixe présente des avantages en terme de simplicité d'implémentation (elle est souvent privilégiée pour cela dans les processeurs embarqués de faible puissance) [11].

Addition et soustraction en virgule fixe

Le processus d'addition et de soustraction des nombres binaires en virgule fixe ainsi que les problèmes débordement sont exactement les mêmes que pour les nombres signés en Cà2.

+6,25	0110,01
+2,75	0011,11
=9,00	1001,00

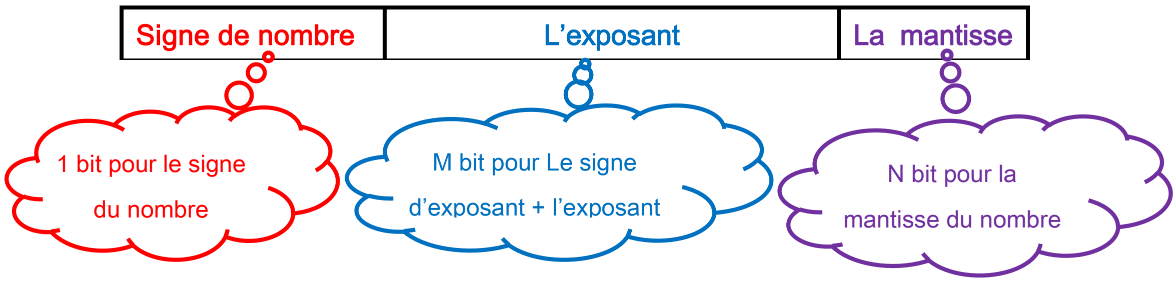
3.3.3.2. Virgule flottante

La représentation en machine des nombres réels (on parle souvent en informatique de nombres flottants) diffère de la représentation en machine des entiers.

• Tout nombre réel peut s'écrire comme suit : $N= (-1)^S * M * B^e$

- M : mantisse
- B : la base
- e : l'exposant
- S : signe

Le format de représentation des nombres flottants est comme suite :



Exemple :

$(15,6)_{10} =(-1)^0 * 0,156 * 10^{+2}$

$-(110,101)_2 = (-1)^1 * 0,110101 * 2^{+3}$

$(0,00101)_2=(-1)^0 * 0,101 * 2^{-2}$

La représentation de nombre (1101,101)2 selon le format virgule flottante donnée suivante : 1 bit de signe, 6 bits pour l'exposant plus son signe et 9bits pour la mantisse. On a :

$(1101,101)_2=(-1)^0 * 0,1101101 * 2^4 \Rightarrow$

0	0	0	0	1	0	0	1	1	0	1	1	0	1	0	0
Signe de nombre	Le signe d'exposant + l'exposant						La mantisse								

• Dans cette représentation en virgule flottante sur **n** bits :

– La mantisse est sous la forme signe/valeur absolue

• 1 bit pour le signe

• et **K** bits pour la valeur.

– **P** bits pour l'exposant (positif ou négatif).

La partie de mantisse m est représentée en base b. La précision est le nombre de chiffres de la mantisse, l'exposant est un entier signé.

En informatique nous avons la base binaire. Un nombre réel a un nombre infini de représentations mantisse–exposant [12]. La représentation en virgule flottante a été normalisée (norme IEEE 754) afin que les programmes se comportent de manière identique d'une machine à l'autre [13].

➢ **Virgule flottante selon la norme IEEE 754**

Dans le domaine informatique, la norme IEEE 754 (IEEE Standard for Floating-Point Arithmetic) est la norme la plus largement utilisée pour représenter les nombres rationnels. La représentation en virgule flottante selon la norme IEEE 754 est un cas particulier de la représentation en virgule flottante des nombres réels [14], pour que tout le monde obtienne les mêmes résultats. Pour normaliser une représentation selon la norme IEEE 754 on décale la virgule jusqu'à avoir 1, donc si X est un nombre flottant:

$$X = (-1)^{S} * 1,ZZZZZ... * 2^{e}$$

Où :

S : est le signe de nombre X,

e :est l'exposant signé

ZZZZZ...: est la partie fractionnaire de la mantisse.

Dans la représentation en VF il y a quatre formats : simple précision, double précision, simple précision étendue et double précision étendue [15]. Voici des exemples de formats courants :

Nombre de bits	format	valeur max	précision max
32	1 + 8 + 23	2128 ≈ 10^{+38}	$2^{-23} \approx 10^{-7}$
64	1 + 11 + 52	21024 ≈ 10^{+308}	$2^{-52} \approx 10^{-15}$
80	1 + 15 + 64	216384 ≈ 10^{+4932}	$2^{-64} \approx 10^{-19}$

Tableau 3.9 : Différents formats de la représentation en virgule flottante selon la norme IEEE 754.

➦ **Représentations en virgule flottante sous format IEEE754 simple précision**

Représentation simple précision est sur **32 bits** organisés comme suite :

- **1** bit de signe
- **8** bits pour la partie d'exposant décalé
- **23** bits pour la partie de mantisse

signe mantisse	Exposant décalé	partie fractionnaire de la mantisse (non signé)
1bit	8 bits	23 bits

Pour calculer l'exposant décalé on a : **Exposant Décalé = Exposant Réel + Décalage**

Décalage= 2^{n-1}- 1

n = nombre de bit de l'exposant décalé (n= 8 pour la représentation sous format IEEE754 simple précision), Décalage = 2^{8-1} - 1 = 2^{7} - 1 = 128 – 1 = 127

Exemples :

- $A=(-42,75)_{10}=(-101010,11)_2$

  $=(-1)^1.1,0101011.2^{+5}$

S=1

$M=(01010110........0)_2$

$E-127=e \Rightarrow E=(127+5)_{10}$

$=(132)_{10}$

$=(10000100)_2$

$A=(-42,75)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{1|\ 10000100|\ 010101100............0}_{32\text{ bits}}$

- $B=(-13,125)_{10}=(-1101,001)_2$

  $=(-1)^1.1,101001.2^{+3}$

S=1

$M=(1010010........0)_2$

$E-127=3 \Rightarrow E=(127+3)_{10}$

$=(130)_{10}$

$=(10000010)_2$

$B=(-13,125)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{1|10000010|\ 101001000.............0}_{32\text{ bits}}$

- $C=(+30,0625)_{10}=(+11110,0001)_2$

  $=(-1)^0\ x1.1100001x2^{+4}$

S=0

$M=(111000010.........0)_2$

$E-127=+4 \Rightarrow E=127+4=(131)_{10}$

$=(10000011)_2$

$C=(27,25)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{0|\ 10000011\ |\ 11100001000000000000000}_{32\text{ bits}}$

- $D=(+0,21875)_{10}=(+0,00111)_2$

  $=(-1)^0\ x1.11x2^{-3}$

S=0

$M=(11000.........0)_2$

$E-127=-3 \Rightarrow E=127-3=(124)_{10}$

$=(1111100)_2$

$D=(+0,21875)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{0|01111100|\ 11000000000000000000000}_{32\text{ bits}}$

- $E=(+0,375)_{10}=(-0,011)_2$
  $=(-1)^0 \times 1.1 \times 2^{-2}$

S=0

$M=(100\ldots\ldots\ldots0)_2$

$E-127=-2 \Rightarrow E=127-2=(125)_{10}$
$=(01111101)_2$

$$E=(+0.375)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{0|01111101|\ 10000000000000000000000}_{32\text{ bits}}$$

➦ Représentations en virgule flottante sous format IEEE754 double précision

Représentation simple précision est sur **64 bits** organisés comme suite :

- **1** bit de signe
- **11** bits pour la partie d'exposant décalé
- **52** bits pour la partie de mantisse

signe mantisse	Exposant décalé	partie fractionnaire de la mantisse (non signé)
1bit	11 bits	52 bits

Pour calculer l'exposant décalé on a : **Exposant Décalé = Exposant Réel + Décalage**

Décalage= $2^{n-1}-1$

n=nombre de bit de l'exposant décalé (n=11 pour la représentation sous format IEEE754 simple précision), Décalage = $2^{11-1}-1 = 2^{10}-1 = 1024 - 1 = 1023$

Exemples :

- $X=(-42,75)_{10}=(-101010,11)_2$
  $=(-1)^1.1,0101011.2^{+5}$

S=1

$M=(01010110\ldots\ldots..0)_2$

$E-1023=e \Rightarrow E=(1023+5)_{10}$
$=(1028)_{10}$
$=(10000000100)_2$

$$X=(-42,75)_{10} \xrightarrow[\text{Précision}]{\text{VF Double}} \underbrace{1|\ 10000000100|\ 010101100\ldots\ldots\ldots\ldots0}_{64\text{ bits}}$$

- $Y=(+0,21875)_{10}=(+0,00111)_2$
  $=(-1)^0 \times 1.11 \times 2^{-3}$

S=0

$M=(11000\ldots\ldots\ldots0)_2$

$E-1023=-3 \Rightarrow E=1023-3=(1020)_{10}$
$=(1111111100)_2$

$$Y=(+0,21875)_{10} \xrightarrow[\text{Précision}]{\text{VF Double}} \underbrace{0|01111111100|\ 1100000000000000000000}_{64\text{ bits}}$$

➦ La représentation décimale

A/ Conversion de la représentation en virgule flottante sous format IEEE754 simple précision vers la représentation décimale

• $Z=(43C00000)_{16}=(0|10000111|10000000000000000000000)_2$

$S=0$

$M=(1000……..0)_2$

$E=(10000111)_2=(135)_{10}\Rightarrow e=E-127$

$=135-127$

$=+8$

$Z=(-1)^0 x1,1x2^{+8}$

$=+1,1x2^{+8}$

$=(+11000000)_2$

$=+(2^8+2^7)_{10}$

$=(+384)_{10}$

• $W=(C29A0000)_{16}=1|10000101|00110100000000000000000$

$S=1$

$M=(001101000………0)_2$

$E=(10000101)_2=(133)_{10}\Rightarrow e=E-127$

$=133-127$

$=+6$

$W=(-1)^1 x1,001101x2^{+6}$

$=(-1,001101\ x2^6)_2$

$=(-1001101)_2$

$=-(2^6+2^3+2^1+2^0)_{10}$

$=(-75)_{10}$

B/ Conversion de la représentation en virgule flottante sous format IEEE754 double précision vers la représentation décimale

• $Z=(C038\ 0000\ 0000\ 0000)_{16}=(1100\ 0000\ 0011\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000…………………)_2$

$S=1$

$M=(100……..0)_2$

$E=(10000000011)_2=(1027)_{10}\Rightarrow e=E-1023$

$=1027-1023$

$=+4$

$Z=(-1)^1.1,1.2^{+4}$

$=-1,1.2^{+4}=(-11000)_2=(-24)_{10}$

•$W=(BF8C\ 0000\ 0000\ 0000)_{16}=(1011\ 1111\ 1000\ 1100\ 0000\ 0000\ 0000\ 0000\ 0000\ldots\ldots)_2$

$S=1$

$M=(1100\ldots\ldots.0)_2$

$E=(01111111000)_2=(1016)_{10}\Rightarrow e=E-1023$

$=1016-1023$

$=-7$

$W=(-1)^1 x1,11x2^{-7}$

$=-1,11.2^{-7}=(-0.000000111)_2$

$=-(2^{-7}+2^{-8}+2^{-9})_{10}$

$=(-0.013671875)_{10}$

3.4. Codage des caractères

Les caractères comprennent :

- Les lettres alphabétiques (A, a, B , b,..) ,
- Les chiffres (1, 2, 3, …) ,
- Les symboles (> , ; / : ….) .

3.4.1. Code ASCII (American Standard Code for Information Interchange)

Non seulement les nombres doivent être codés en binaire, mais aussi les caractères comme les lettres alphabétiques et les symboles (signes de ponctuation, …etc). Le code le plus connu et le plus utilisé est l'ASCII (American Standard Code for Information Interchange).

Le codage ASCII attribue un code à sept bits aux principaux caractères mais pas aux caractères accentués [16]. L'ASCII ne spécifie que 128 caractères numérotés de 0 à 127 et codés en binaire de 0000000 à 1111111. Cependant, comme presque tous les ordinateurs fonctionnent sur des multiples de huit bits (multiples d'octets) depuis les années 1970, chaque caractère de texte ASCII est généralement stocké dans un octet où le huitième octet est égal à 0.

Les caractères de numéro 0 à 31 et le 127 correspondent à des commandes qui contrôlent le terminal de l'ordinateur.

- Le caractère numéro 127 est la commande de suppression.
- Le caractère numéro 32 est un espace.
- Les autres caractères sont des chiffres arabes, des lettres latines minuscules et majuscules sans accent, les signes de ponctuation, les opérateurs et quelques autres symboles.

Parfois, le huitième bit est utilisé pour donner le code ASCII étendu avec la possibilité de faire $2^8=256$ caractères.

		000	001	010	011	100	101	110	111
		0	1	2	3	4	5	6	7
0000	0	NUL	DLE	SP	0	@	P	`	p
0001	1	SOH	DC1	|	1	A	Q	a	q
0010	2	STX	DC2	″	2	B	R	b	r
0011	3	ETX	DC3	#	3	C	S	c	s
0100	4	EOT	DC4	$	4	D	T	d	t
0101	5	ENQ	NAK	%	5	E	U	e	u
0110	6	ACK	SYN	&	6	F	V	f	v
0111	7	BEL	ETB	‘	7	G	W	g	w
1000	8	BS	CAN	(	8	H	X	h	x
1001	9	HT	EM	)	9	I	Y	i	y
1010	10	LF	SUB	*	:	J	Z	j	z
1011	11	VT	ESC	+	;	K	[	k	{
1100	12	FF	FS	,	<	L	\	l	!
1101	13	CR	GS	-	=	M	]	m	}
1110	14	SO	RS	.	>	N	‘	n	~
1111	15	SI	US	/	?	O	-	o	DEL

Tableau 3.10 : Codage ASCII.

NUL	Absence de caractère,blanc,espace
SOH	Start of Heading : début en-tête
STX	Start of Text
ETX	End of Text
EOT	End of Transmission
ENQ	Enquiry Demande
ACK	Acknowledge, accusé réception
BEL	Bell, sonnette
BS	Backspace marche arrière 1 caractère
HT	Horizontale Tabulation
LF	Line Fed retour à la nvelle ligne
VT	Vertical Tabulation
FF	Form Fed, passage page suivante
CR	Carriage Return, retour chariot
SO	Shift Out caractère suivant non std
SI	Shift In retour au caractères std
DLE	DataLink Escape chgmt de signific.
NAK	Negative Acknoledgment
SYN	Synchronous, caractère de synchro.
ETB	End Of Transmission Block
CAN	Cancel annulation de la donnée précédente
SUB	Substitute remplacement
ESC	Escape caractère de ctrl d'extension
FS	File Separator
GS	Groupe Separator
RS	Record Separator
US	United Separator
SP	Space Espace
DEL	Delete, suppression

DC1 à DC4 : caractères de commandes

Exemple :

•Le code de caractère A =(41)ASCII-hexadécimal

=(01000001)ASCII-binaire

=(65)ASCII-décimal

•Le code de caractère a =(61)ASCII-hexadécimal

=(01000001)ASCII-binaire

=(97)ASCII-décimal

•Le code de caractère X =(58)ASCII-hexadécimal

=(01011000)ASCII-binaire

=(88)ASCII-décimal

	.0	.1	.2	.3	.4	.5	.6	.7	.8	.9	.A	.B	.C	.D	.E	.F
0.	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	NP	CR	SO	SI
1.	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2.		!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3.	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4.	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5.	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6.	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7.	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	Del

Tableau 3.11 : Codage ASCII.

Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	`
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(	72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29	)	73	01001001	111	49	I	105	01101001	151	69	i
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01010000	120	50	P	112	01110000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01010001	121	51	Q	113	01110001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01010010	122	52	R	114	01110010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01010011	123	53	S	115	01110011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	01010100	124	54	T	116	01110100	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	01010101	125	55	U	117	01110101	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	01010110	126	56	V	118	01110110	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	01010111	127	57	W	119	01110111	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	01011000	130	58	X	120	01111000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	01011001	131	59	Y	121	01111001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	01011010	132	5A	Z	122	01111010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	;	91	01011011	133	5B	[	123	01111011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	01011100	134	5C	\	124	01111100	174	7C	|
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	01011101	135	5D	]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	01011110	136	5E	^	126	01111110	176	7E	~
31	00011111	037	1F	US	63	00111111	077	3F	?	95	01011111	137	5F	_	127	01111111	177	7F	DEL

Tableau 3.12 : Codage ASCII.

(BONJOUR)$_{\text{caractère}}$= (66 79 78 74 79 85 82)$_{\text{ASCII-décimal}}$

=(42 4F 4E 4A 4F 55 52)$_{\text{ASCII-hexadécimal}}$

=(01000010 01001111 01001110 01001010 01001111 01010101 01010010)$_{\text{ASCII-binaire}}$

(STRUCTURE Machine 1)$_{\text{caractère}}$

=(83 84 82 85 67 84 85 82 69 32 109 97 99 104 105 110 101 32 73)$_{\text{ASCII-décimal}}$

=(53 54 52 55 43 54 55 52 45 20 6D 61 63 68 69 6E 65 20 49)$_{\text{ASCII-hexadécimal}}$

=(01010010 01010100 01010010 01010101 01000011 01010100 01010101 01010010 01000101 00100000 01101101 01100001 01100011 01101000 01101001 01101110 01100101 00100000 01001001)$_{\text{ASCII-binaire}}$

3.4.2. Code EBCDIC

Le code EBCDIC (Extended Binary Coded Decimal Interchange Code) est un code des caractères sur 8 bits créé par IBM à l'époque des cartes perforées. Il y a au moins six versions différentes bien documentées, qui ne sont pas compatibles entre elles. Ce code a été critiqué pour cette raison, mais aussi pour l'absence de certains signes de ponctuation dans certaines versions. Le code EBCDIC a été créé pour améliorer le code BCD et est utilisé dans les systèmes AS/400 d'IBM et les mainframes sous MVS, VM ou DOS/VSE. Ce code, précurseur du code ASCII, a été dévoilé entre 1963 et 1964 lors du lancement de la série 360 d'IBM.

b8-b5 / b4-b1		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	NUL	DLE	DS		SP	&	-						{	}	⁄	0
0001	1	SOH	DC1	SOS				/		a	j	~		A	J		1
0010	2	STX	DC2	FS	SYN					b	k	s		B	K	S	2
0011	3	ETX	DC3	WUS	IR					c	l	t		C	L	T	3
0100	4	SEL	RES	BYP	PP					d	m	u		D	M	U	4
0101	5	HT	NL	LF	TRN					e	n	v		E	N	V	5
0110	6	RNL	BS	ETB	NBS					f	o	w		F	O	W	6
0111	7	DEL	POC	ESC	EOT					g	p	x		G	P	X	7
1000	8	GE	CAN	SA	SBS					h	q	y		H	Q	Y	8
1001	9	SPS	EM	SFE	IT				`	i	r	z		I	R	Z	9
1010	A	RPT	UBS	SM	REF	¢	!		:								
1011	B	VT	CU1	CSP	CU3	.	$	,	#								
1100	C	FF	FS	MFA	DC4	<	*	%	@								
1101	D	CR	GS	ENQ	NAK	(	)	_	'								
1110	E	SO	RS	ACK		+	;	>	=								
1111	F	SI	US	BEL	SUB	|	¬	?	"								E0

Tableau 3.13 : Code EBCDIC Standard.

(BONJOUR)$_{\text{caractère}}$= (C2 D6 D5 D1 D6 E4 D9)$_{\text{EBCDIC-hexadécimal}}$

=(11000010 11010110 11010101 11010001 11010110 11100110 11011001)$_{\text{EBCDIC-binaire}}$

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	PF	HT	LC	DEL	GE	RLF	SMM	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	TM	RES	NL	BS	IL	CAN	EM	CC	CU1	IFS	IGS	IRS	IUS
2	DS	SOS	FS		BYP	LF	ETB	ESC			SM	CU2		ENQ	ACK	BEL
3			SYN		PN	RS	UC	EOT				CU3	DC4	NAK		SUB
4											¢	.	<	(	+	|
5	&										!	$	*	)	;	¬
6	-	/									¦	,	%	_	>	?
7										`	:	#	@	'	=	"
8		a	b	c	d	e	f	g	h	i						
9		j	k	l	m	n	o	p	q	r						
A		~	s	t	u	v	w	x	y	z						
B																
C	{	A	B	C	D	E	F	G	H	I			⑀		⑂	
D	}	J	K	L	M	N	O	P	Q	R						
E	\		S	T	U	V	W	X	Y	Z			⑁			
F	0	1	2	3	4	5	6	7	8	9	|					EO

Tableau 3.14 : Code EBCDIC Standard.

3.4.3. Code UTF

Des représentations plus avancées telles qu'Unicode avec 16 bits peuvent représenter jusqu'à 2016 de 65536 symboles. Avec ce code, plusieurs alphabets peuvent être représentés. L'absence de caractères de langues étrangères comme l'anglais rend cette norme inadaptée aux textes étrangers comme le français, ce qui oblige à utiliser d'autres codages.

Unicode (1991)

Unicode créé en 1991 a pour vocation de rassembler tous ces codes afin d'avoir un code commun et unique pour toutes les langues au niveau mondial [17]. Il contient plus de 1 million de caractères jusqu'à 4 octets soit plus de $2^{4x8} = 2^{32}$ = 4,3 milliard de possibilités soit 4 fois plus de place qu'en ASCII. Chaque caractère abstrait est identifié par un nom unique et associé à un point de code. Dans ce code tous les caractères sont codés de façon unique et universelle.

UTF-8 (1996)

UTF-8 (Unicode Transformation Format 8 bits) est dérivé de l'Unicode, rétro-compatible avec la norme ASCII et identique au code ASCII, alors que les autres caractères sont codés selon Unicode, sur 2 à 4 octets. UFT-8 donne le moyen de différencier les codes ASCII des codes Unicode.

Dans le code ASCII, le bit de poids fort est nul et les sept autres bits donnent le code ASCII (surlignés en vert, comme le montre sur la figure ci-dessous). Mais, dans le code Unicode, le bit de poids fort est non nul et suit la règle : 110, 1110, 11110... Le reste des bits donnent le code Unicode (surlignés en vert, comme le montre sur la figure ci-dessous). Alors, UTF-8 combine les avantages de l'ASCII (fichier de petite taille) et de l'Unicode (caractères universels).

Représentation binaire UTF-8	Signification
0xxxxxxx	1 octet codant 1 à 7 bits
110xxxxx 10xxxxxx	2 octets codant 8 à 11 bits
11100000 101xxxxx 10xxxxxx	3 octets codant 12 à 16 bits
11100001 10xxxxxx 10xxxxxx	
1110001x 10xxxxxx 10xxxxxx	
111001xx 10xxxxxx 10xxxxxx	
111010xx 10xxxxxx 10xxxxxx	
11101100 10xxxxxx 10xxxxxx	
11101101 100xxxxx 10xxxxxx	
1110111x 10xxxxxx 10xxxxxx	
11110000 1001xxxx 10xxxxxx 10xxxxxx	4 octets codant 17 à 21 bits
11110000 101xxxxx 10xxxxxx 10xxxxxx	
11110001 10xxxxxx 10xxxxxx 10xxxxxx	
1111001x 10xxxxxx 10xxxxxx 10xxxxxx	
11110100 1000xxxx 10xxxxxx 10xxxxxx	

reprise ASCII

reprise Unicode

Tableau 3.15 : Code UTF-8.

UTF-16 et UTF-32

UTF-16 et UTF-32 suivent le même principe qu'UTF-8 mais le codage se fait sur 16 bits (UTF-16) ou 32 bits (UTF-32). Par conséquent, pour la plupart des langues latines qui utilisent largement les caractères ASCII, UTF-8 nécessite moins d'octets qu'UTF-16 ou UTF-32. En revanche, pour les langues qui utilisent beaucoup de caractères en dehors de l'ASCII, UTF-8 prend beaucoup plus d'espace. UTF-32 ne sera plus efficace que pour les textes qui utilisent majoritairement des textes anciens ou rares encodés en dehors du niveau multilingue de base, c'est-à-dire de U+10000, mais il peut aussi être utile localement dans certains traitements pour simplifier les algorithmes.

Chaque symbole d'écriture est représenté par un nom et une valeur hexadécimale préfixée par «**U+** ». Exemple :

- Le code de A « lettre majuscule latine A » est **U+0041.**
- Le code de é « lettre minuscule latine e accent aigü » est U**+00E9.**
- Le code de € « symbole euro » est **U+20AC.**

Série d'exercices N°2 (La représentation de l'information)

Exercice N°1

A - Compléter le tableau ci-dessous.

décimal	Binaire pur sur 7 bits	gray	BCD
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19			
20			

B - Expliquer la différence entre le code Gray (binaire réfléchi) et le code binaire pur.

Exercice N°2

Donner le nombre suivant pour chaque nombre donné en code Gray :

- $(1101010010)_{Gray}$
- $(1011011011)_{Gray}$
- $(111110001)_{Gray}$
- $(110110000)_{Gray}$
- $(110011100)_{Gray}$

Exercice N°3

Convertissez les nombres binaires suivants en code Gray :

- $(11011)_2$
- $(1001010)_2$
- $(11101101110)_2$
- $(11000110)_2$
- $(101101)_2$

Exercice N°4

Convertissez chaque code Gray en binaire :

- $(1010)_{Gray}$
- $(10010)_{Gray}$
- $(11000010001)_{Gray}$
- $(10101111)_{Gray}$
- $(1000111)_{Gray}$

Exercice N°5

1. Donner la représentation des nombres suivants en code BCD, Excédent+3, binaire et en Gray
   - $(4389)_{10}$ = (…………..)$_{BCD}$ = (……………..)$_{Excedent+3}$ = (…………..)$_2$ = (………………)$_{gray}$
   - $(2023)_{10}$ = (…………..)$_{BCD}$ = (……………..)$_{Excedent+3}$ = (…………..)$_2$ = (………………)$_{gray}$
   - $(512)_{10}$ = (…………..)$_{BCD}$ = (……………..)$_{Excedent+3}$ = (…………..)$_2$ = (………………)$_{gray}$
2. Donner la représentation des nombres suivants en code BCD+3
   - $(1001101)_2$
   - $(11101101)_{Gray}$
   - $(50421)_8$

Exercice N°6 :

Trouver les résultants des opérations suivantes en code BCD.

- 158+641
- 77+33
- 199+644
- 781-167
- 1000-1001
- 903-878

Exercice N°7 :

Trouver les résultants des opérations suivantes en code BCD+3.

- 1000+1001
- 66+57
- 1527+4543
- 371+983

Exercice N°8 :

1. Donner la représentation binaire pur de chacun des nombres BCD suivants :

   a. $(100001100111)_{BCD}$

   b. $(001010011000)_{BCD}$

2. Donner le code binaire pur, puis le code BCD de $(2024)_{10}$ et comparez le coût de représentation.

3. Combien de bits sont nécessaires pour représenter un nombre décimal à 8 chiffres en code BCD ?

4. Exprimer la valeur 67 puis 14 en code excess 3. Quel est le résultat de leur addition ?

Exercice N°9:

Soit les nombres entiers suivants : X= (11011011) en Cà1, Y=(10111100) en Cà2, Z= (01010011) en Cà1 et T=(01000111) en Cà2. Donner le signe et la valeur décimale de chaque nombre.

Exercice N°10:

Soit les nombres entiers suivants : A= 25 et B= 38.

1- représenter sur 6 bits en Cà1 et Cà2 les nombres suivants : A, -A, B et -B.

Cette codification est-elle possible sur 5 bits ? Justifier.

2- effectuer, si possible, les opérations suivantes en Cà1 et Cà2 : A-B, B-A, -A-B.

Exercice N°11:

Trouver les compléments des nombres suivants :

-sur 8 bits binaires le complément à 1 de $(00101011)_2$

-sur 6 bits binaires le complément à 2 de $(11010)_2$

-sur 10 bits octaux le complément à 8 de $(1776010)_8$

-sur 9 bits décimaux le complément à 10 de $(2990700)_{10}$

-sur 7 bits hexadécimaux le complément à 16 de $(A02CBF0)_{16}$

Exercice N°12:

1. Donner la représentation en virgule flottante sous format IEEE754 simple précision des nombres suivants : $(-128,25)_{10}$, $(+18,125)_{10}$, $(-0,375)_{10}$
2. Donner la représentation en virgule flottante sous format IEEE754 double précision des nombres suivants : $(1000)_2$, $(-12,625)_{10}$, $(-64)_{10}$

Exercice N°13:

a- Donner la représentation décimale des nombres codés en virgule flottante (IEEE754 simple précision) :

1 • $(1011\ 1101\ 1110\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

2• $(1100\ 0011\ 1101\ 1010\ 0000\ 0000\ 0000\ 0000)_2$

3• $(0100\ 0000\ 1101\ 1000\ 0000\ 0000\ 0000\ 0000)_2$

b- Donner la représentation décimale des nombres codés en virgule flottante (IEEE754 double précision) :

1• $(403D\ 8000\ 0000\ 0000)_{16}$

2• $(C040\ 0000\ 0000\ 0000)_{16}$

3• $(3FB8\ 0000\ 0000\ 0000)_{16}$

Exercice N°14:

Soit le format M à 15 bits en virgule fixe :

$A_8|A_7A_6A_5A_4A_3A_2A_1A_0|A_{-1}A_{-2}A_{-3}A_{-4}A_{-5}A_{-6}$. Tel que A_8 constitue le bit de signe, et $A_7A_6A_5A_4A_3A_2A_1A_0$ la partie entière et $A_{-1}A_{-2}A_{-3}A_{-4}A_{-5}A_{-6}$ est la partie fractionnaire.

a) Représenter les équivalents binaires des nombres N_1=-26 et N_2=+34,0625 selon le format M de virgule fixe.

b) Effectuez l'opération d'addition des opérandes N_1 et N_2 et représentez le résultat selon le format M de virgule fixe.

Exercice N°15:

1. Représenter le nombre réel $(-16{,}375)_{10}$ en format virgule fixe (1 bit de signe, 8 bit pour la partie entière et 7 bits pour la partie fractionnaire).
2. Quelle est le plus petit nombre positif représentable dans ce format.
3. Quelle est le plus grand nombre positif représentable dans le même format.

Exercice N°16 :

1. convertissez les nombres décimaux suivants en code ASCII : 1, 102, 29, 58, 706.
2. convertissez les lettres suivantes en code ASCII : A, a, d, W, SM.
3. Déterminez chaque caractère ASCII : 00111111-00100000-01000000-01011000-00111110

Exercice N°17 :

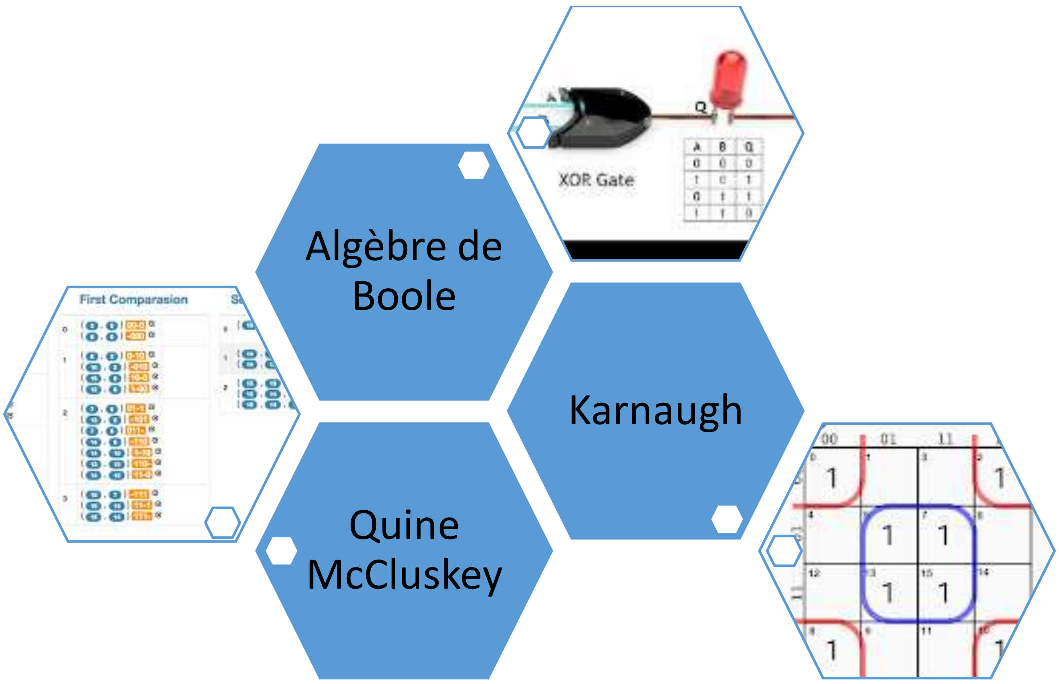
• Trouver le texte représenté en code ASCII binaire par la suite de bits :

**01001101011000010111010001101000011100110010000001100101011101000010000001001001
01101110011001100110111101110010011011010110000101110100011010010111000101110101
01100101**

• Codez votre nom en hexadécimal avec les codages ASCII et EBCDIC.

• Le codage UTF-8 est un codage de longueur variable, où les mots sont composés de 1, 2, 3 ou 4 octets. Combien de caractères le codage UTF-8 code-t-il ?

Chapitre IV : L'algèbre de Boole binaire



Chapitre IV : L'algèbre de Boole binaire

4.1. Définition et axiomes de l'algèbre de Boole

George Boole, né le 2 novembre 1815 à Lincoln (Royaume-Uni) était un logicien, mathématicien, philosophe britannique et autodidacte. Il est le créateur de la logique moderne, basée sur une structure algébrique et sémantique, qui s'appelle l'algèbre booléenne. Il a également travaillé dans d'autres domaines des mathématiques, les équations différentielles, les probabilités et l'analyse. Il a aussi travaillé dans d'autres domaines mathématiques, des équations différentielles aux probabilités en passant par l'analyse. Mort le 8 décembre 1864 à Ballintemple (Irlande) [18].

Les variables logiques sont généralement définies en électronique numérique par les valeurs 0 ou 1. Ces variables suivent des règles algébriques spécifiques qu'il faut maîtriser avant d'effectuer l'analyse ou la synthèse de circuits numériques. Dans ce chapitre nous allons énoncer les principes et les règles de l'algèbre de Boole, puis nous les appliquerons à la représentation et à la manipulation de fonctions booléennes.

4.1.1. Variables et fonctions logiques

4.1.1.1. Variables logiques

La variable logique est une variable qui prendre deux valeurs définies conventionnellement, le 0 et le 1, en d'autres termes variables binaires. Ces deux valeurs sont liées à une grandeur physique, telle que la tension de collecteur d'un transistor. Ce qui permet nous de relier une étude théorique utilisant l'algèbre de Boole à un circuit électronique [19].

4.1.1.2. Fonctions logiques

F est une fonction booléenne (logique) de n variables logiques (x_1, x_2,…, x_n), notée par exemple F = F(x_1, x_2,..., x_n), Chaque variable booléenne xi pouvant prendre la valeur 0 ou 1, il existe au total 2^n combinaisons possibles des variables booléennes (x_1, x_2,..., x_n). Et on définit une fonction complètement booléenne en donnant sa valeur à chacune de ces combinaisons [20].

Parmi les représentations possibles d'une fonction logique, nous avons la table de vérité, qui donne la valeur de cette fonction pour chaque combinaison des variables logiques. Une table de vérité est une énumération complète de toutes les combinaisons des variables d'entrées du circuit logique avec les valeurs des sorties associées [21]. L'écriture de tables de vérité fait partie de l'analyse d'une fonction logique donnée.

Exemple :

Soit F une fonction de deux variables **X** et **Y**. donc, il existe $2^2 = 4$ combinaisons possibles de ces deux variables. Une table de vérité donne la valeur de F pour chacune des quatre combinaisons possibles de **X** et **Y**. il existe généralement deux façons de représenter la table de vérité, comme indiqué ci-dessous.

X	Y	F
0	0	0
0	1	1
1	0	0
1	1	1

Y \ X	0	1
0	0	1
1	0	1

4.2. Les opérateurs de base

L'outil mathématique utilisé pour décrire les systèmes combinatoires est l'algèbre de Boole [22]. En algèbre de Boole, il existe trois lois, ou fonctions logiques de base, NON, OU, ET, En combinant ces trois fonctions de base NON, OU, ET, nous pourrons décrire chaque sortie en termes d'entrées.

4.2.1. Fonction inversion NON (NOT)

La fonction NON est un opérateur qui affecte à une variable de sortie l'état complémentaire d'une variable d'entrée. Cette fonction est aussi appelée complément

Notation : $S = \overline{X}$

Table de vérité

X	$S = \overline{X}$
0	1
1	0

$$S = \overline{X}$$

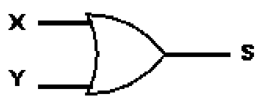
4.2.2. Fonction OU (OR)

La fonction OU est une fonction de deux variables, elle est aussi appelée somme logique.

Notation : $S = X + Y$

Table de vérité

X	Y	$S = X + Y$
0	0	0
0	1	1
1	0	1
1	1	1



$$S = X + Y$$

La fonction logique OU vaut 1 si au moins une entrée parmi les variables d'entrée de cette fonction vaut 1.

4.2.3. Fonction ET (AND)

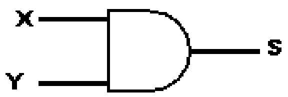
La fonction ET est une fonction de deux variables, elle est aussi appelée produit logique.

Notation : $S = X \cdot Y$

Table de vérité

X	Y	$S = X \cdot Y$
0	0	0
0	1	0
1	0	0
1	1	1

$S = X \cdot Y$



La fonction ET ne vaut 1 que si toutes les variables d'entrée de cette fonction valent 1.

4.2.4. Autres opérateurs logiques

4.2.4.1. Circuits NAND et NOR

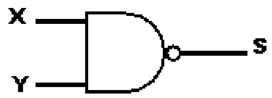
La fonction NAND est une fonction de deux variables, elle est également appelée complément d'un produit logique.

Notation : $S = \overline{X \cdot Y}$

Table de vérité

X	Y	$S = \overline{X \cdot Y}$
0	0	1
0	1	1
1	0	1
1	1	0

$S = \overline{X \cdot Y}$



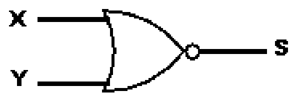
La fonction NOR est une fonction de deux variables, elle est aussi appelée complément d'une somme logique.

Notation : $S = \overline{X + Y}$

Table de vérité

X	Y	$S = \overline{X + Y}$
0	0	1
0	1	0
1	0	0
1	1	0

$S = \overline{X + Y}$



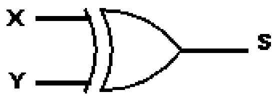
4.2.4.2. Ou exclusif et NON-OU exclusif

- **Ou exclusif**

Table de vérité

X	Y	$S = X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

$$S = X \oplus Y = \overline{X}Y + X\overline{Y}$$

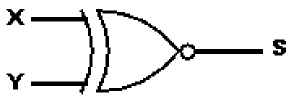


- **NON-OU exclusif – XNOR**

Table de vérité

X	Y	$S = X \otimes Y$
0	0	1
0	1	0
1	0	0
1	1	1

$$S = X \otimes Y = \overline{X \oplus Y} = \overline{X}\,\overline{Y} + XY$$



4.3. Propriétés et théorèmes de l'algèbre de Boole

Propriétés des opérateurs logiques

• Associativité : L'ordre des opérations n'a pas d'importance si elles ont la même priorité.

$$(X + Y) + Z = X + (Y + Z) = X + Y + Z$$
$$(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) = X \cdot Y \cdot Z$$

• Commutativité : L'ordre des variables n'a pas d'importance.

$$X + Y = Y + X$$
$$X \cdot Y = Y \cdot X$$

• Distributivité : Pour passer de la distributivité de l'addition par rapport à la multiplication vers la distributivité de la multiplication par rapport à l'addition, il suffit de permuter les deux opérateurs « · » et « + ».

$$X(Y + Z) = XY + XZ$$
$$X + (YZ) = (X + Y)(X + Z)$$

• **Idempotence :** Une variable ajoutée ou multipliée par elle conserve toujours sa valeur.

$$X + X + X + \dots\dots\dots + X = X$$
$$X \cdot X \cdot X \dots\dots\dots\dots \cdot X = X$$

• **L'identité :** une variable ajoutée par 0 ou multiplié par 1 conserve toujours sa valeur.

$$X + 0 = X$$
$$X \cdot 1 = X$$

• **Absorption :** A absorbe B, c'est à dire A conserve toujours sa valeur.

$$X + XY = X$$
$$X(X+Y) = X$$

• **Éléments neutres :** Une variable ajoutée par 1 donne 1 et multipliée par 0 donne 0.

$$X \cdot 0 = 0$$
$$X + 1 = 1$$

• **La double négation :** La négation de la négation d'une variable va donner la variable elle-même : $\overline{\overline{X}} = X$

• **Complémentarité :** une variable ajoutée à son complément donne 1, et multiplié à son complément donne 0

$$X \cdot \overline{X} = 0$$
$$X + \overline{X} = 1$$

Théorème de De Morgan

Auguste (ou Augustus) De Morgan : né le 27 juin 1806 à Madurai (Tamil Nadu) en Inde, était un mathématicien et logicien britannique. Il est le fondateur de la logique moderne avec Boole ; En particulier, il a formulé les lois de De Morgan. Il a introduit le terme induction mathématique et il donne le concept strict. Ensuite, il formalise la première algèbre relationnelle. Décédé le 18 mars 1871 [23].

C'est l'une des propriétés les plus importantes des en algèbre de Boole. Elle est basée sur les conversion suivantes : Les relations caractéristiques des lois ET et OU sont invariantes dans leur ensemble au cours de la transformation $+ \rightarrow \cdot$, $\cdot \rightarrow +$, $X \rightarrow \overline{X}$, $\overline{X} \rightarrow X$.

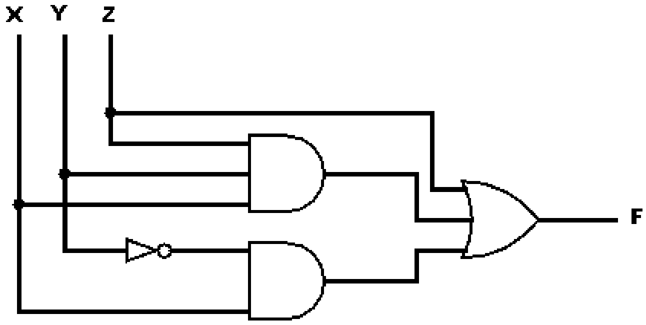
Le théorème de De Morgan est donnée par [24] :

$$\begin{cases} \overline{X + Y} = \overline{X} \cdot \overline{Y} \\ \overline{X \cdot Y} = \overline{X} + \overline{Y} \end{cases}$$

4.4. Représentation des fonctions logiques

Les différentes fonctions booliennes sont décrites sous plusieurs formes :

• Une représentation logique (symbole logique= logigramme)

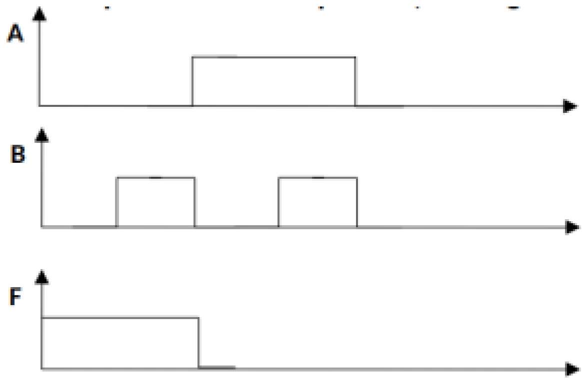


• Une représentation arithmétique (table de vérité)

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

• Une représentation temporelle (chronogramme)

Il s'agit de représenter la fonction logique en fonction du temps pour différentes valeurs des variables d'entrée [25].



• Une représentation algébrique (équation algébrique)

$$F(X,Y,Z) = XYZ + X\overline{Y} + Z$$

4.5. Table de vérité d'une fonction logique

Une table de vérité définit la relation entre les entrées et les sorties en répertoriant toutes les possibilités dans la table à la fois.

$$F(X,Y,Z) = XYZ + X\overline{Y} + Z$$

1ère méthode :

X	Y	Z	XYZ	$\overline{Y}$	$X\overline{Y}$	F
0	0	0	0	1	0	0
0	0	1	0	1	0	1
0	1	0	0	0	0	0
0	1	1	0	0	0	1
1	0	0	0	1	1	1
1	0	1	0	1	1	1
1	1	0	0	0	0	0
1	1	1	1	0	0	1

2ème méthode :

$$F(X,Y,Z) = XYZ + X\overline{Y} + Z$$
$$= 111 + 10\_ + \_\_1$$

X	Y	Z	F
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

4.6. Les formes canoniques d'une fonction logique

Une forme est dite canonique lorsque toutes les variables composent l'entrée apparaissent dans les termes qui représentant la sortie de la fonction. pour une fonction logique donnée , Il existe deux formes canoniques, appelées 1ère forme canonique et 2ème forme canonique [26]:

- ⇒ La première forme canonique disjonctive : FND : mintermes = somme des produits.
- ⇒ La deuxième forme canonique conjonctive : FNC : maxtermes = produits des sommes.

Exemple :

$$F(X,Y,Z) = XYZ + X\overline{Y} + Z$$

X	Y	Z	[redacted]	F
0	0	0		0
0	0	1		1
0	1	0		0
0	1	1		1
1	0	0		1
1	0	1		1
1	1	0		0
1	1	1		1

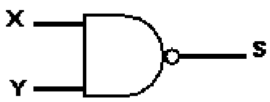
$$F_{FND}(X,Y,Z) = \overline{X}\,\overline{Y}Z + \overline{X}YZ + X\overline{Y}\,\overline{Z} + X\overline{Y}Z + XYZ$$

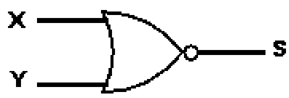
$$F_{FNC}(X,Y,Z) = (X+Y+Z)(X+\overline{Y}+Z)(\overline{X}+\overline{Y}+Z)$$

2ème méthode pour la forme FND

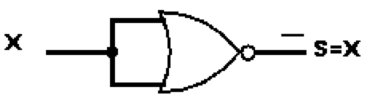
$$F(X,Y,Z) = XYZ + X\overline{Y} + Z$$
$$= XYZ + X\overline{Y}(Z+\overline{Z}) + (X+\overline{X})(Y+\overline{Y})Z$$
$$= XYZ + X\overline{Y}Z + X\overline{Y}\,\overline{Z} + XYZ + X\overline{Y}Z + \overline{X}YZ + \overline{X}\,\overline{Y}Z$$
$$= XYZ + X\overline{Y}Z + X\overline{Y}\,\overline{Z} + \overline{X}YZ + \overline{X}\,\overline{Y}Z$$

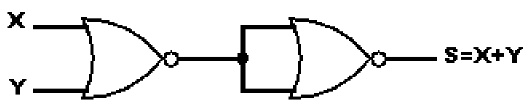
4.7. Expression d'une fonction logique avec des circuits NANDs ou NOR exclusivement

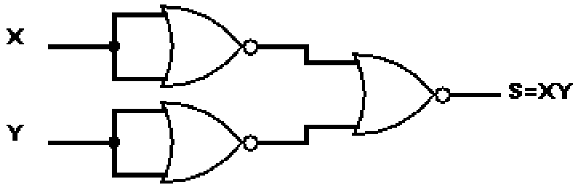
NAND (NON ET) 

NOR (NON OU) 

Réalisation des fonctions NON, OU et ET en utilisant uniquement des portes NOR ou NAND







4.8. Simplification d'une fonction logique

Simplifier une fonction consiste à rendre son expression la plus compacte possible pour réduire le nombre d'opérateurs booléens nécessaires à sa réalisation. La simplification des fonctions logiques a pour but de :

– réduire le nombre de termes dans l'expression d'une fonction logique.

– réduire le nombre de variables dans les termes de l’expression d’une fonction logique.

Ceci afin de réduire le nombre de portes logiques utilisées et le coût du circuit électronique. Il existe plusieurs méthodes de simplification, dans ce chapitre nous distinguons trois méthodes de simplification qui sont :

– La Méthode algébrique (utilisant les lois de l’algèbre de Boole).

– Les Méthodes graphiques : (ex : tableaux de Karnaugh).

– La Méthode de Quine Mc-Cluskey.

4.8.1. Simplification algébrique

La simplification algébrique s'appuie sur les théorèmes de l'algèbre de Boole étudiés précédemment. Exemple :

$$
\begin{aligned}
F(X,Y,Z) &= XY\overline{Z} + X\overline{Y}Z + XYZ \\
&= XY\left(\overline{Z} + Z\right) + X\overline{Y}Z \\
&= XY + X\overline{Y}Z \\
&= X\left(Y + \overline{Y}Z\right) \\
&= X\left(\left(Y + \overline{Y}\right)\left(Y + Z\right)\right) \\
&= XY + XZ
\end{aligned}
$$

$$
\begin{aligned}
G(X,Y) &= \overline{X}\overline{Y} + X\overline{Y} + \overline{X}Y \\
&= \overline{X}\overline{Y} + \overline{X}\overline{Y} + X\overline{Y} + \overline{X}Y \\
&= \overline{X}\left(\overline{Y} + Y\right) + \overline{Y}\left(X + \overline{X}\right) \\
&= \overline{X} + \overline{Y}
\end{aligned}
$$

4.8.2. Simplification par table de Karnaugh (Méthode graphique) :

Maurice Karnaugh, né le 4 octobre 1924 à New York et mort le 8 novembre 2022, est un ingénieure en télécommunications américain américain. Il a développé la table de Karnaugh aux laboratoires Bell en 1953. De **nationalité** américaine, c'est un spécialiste de physique et mathématique logique (**Algèbre de Boole**).

Activités: Physicien, mathématicien, professeur d’université, ingénieur, informaticien. **A travaillé pour** : IBM, École d'ingénierie Tandon de l'université de New York. **Membre de** :Institute of Electrical and Electronics Engineers [27].

La simplification par tableau de Karnaugh repose sur l’utilisation des tableaux de Karnaugh.

a. Table de Karnaugh

La table de KARNAUGH est un outil de simplification des expressions logiques [28]. Est une table de vérité à deux dimensions. L'intersection d'une ligne et d'une colonne forme une case. Les variables d’entrées sont divisées en variables de ligne et en variables de colonne. Pour extraire l’expression simplifier d’une fonction logique à partir de la table de Karnaugh, on utilise obligatoirement le code Gray.

Voici quelques exemples sur les tables de Karnaugh représentants 2, 3, 4 et 5 variables d’entrée.

b. Description de la table de Karnaugh

X / Y	0	1
0		
1		

Table à 2 variables

Z \ XY	00	01	11	10
0				
1				

X / YZ	0	1
00		
01		
11		
01		

Table à 3 variables

XY / ZW	00	01	11	10
00				
01				
11				
10				

Table à 4 variables

XYZ / WT	000	001	011	010	110	111	101	100
00								
01								
11								
10								

Table à 5 variables

b. Règles de regroupement

1. Seuls les cases adjacentes vraies (contenant des 1) de la fonction sont regroupées.
2. Vous pouvez faire des groupements contenant 2^k cases adjacentes.
3. la même case vraie de la fonction peut être utilisée plusieurs fois dans des groupements différents.
4. Toutes les cases vraies de la fonction doivent être contenues dans au moins un groupement.
5. Il faut chercher le plus grand groupement possible pour minimiser le nombre des variables.
6. Si une fonction est représentée par N variables, alors le regroupement de 2^k cases donne un terme produit simplifié de (N–k) variables. Les k variables qui sont éliminés sont celle qui ont varié dans le regroupement.
7. Lorsqu'il ne reste plus d'une case vraie isolée, les regroupements sont terminés.
8. La fonction simplifiée est la réunion des différents groupements.
9. Il est aussi possible et parfois facile de regrouper les cases contenant 0 de la fonction F et de considérer que l'on étudie F.

c. Principe de simplification

- Chaque regroupement obtenu représente un impliquant premier.
- Un impliquant premier contenant au moins 1 ne pouvant être inclus dans aucun autre impliquant premier est dit impliquant premier essentiel.
- Nous sélectionnons d'abord les impliquants premiers essentiels pour obtenir la forme minimale.
- Puis, nous choisissons parmi les impliquants premiers restants celles nécessaires pour couvrir l'ensemble de la fonction d'origine.
- La forme minimale est unique, si elle contient des impliquants premiers essentiels.

Exemples :

AB \ C	00	01	11	10
0	1	1	1	1
1	1	0	0	1

$$F(A,B,C) = \overline{B} + \overline{C}$$

A \ BC	0	1
00	1	1
01	0	1
11	0	0
10	1	1

$$F(A,B,C) = A\overline{B} + \overline{C}$$

AB \ CD	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	1	1	1	1
10	0	1	1	0

$$F(A,B,C,D) = B + D$$

AB \ CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	1	1	1
10	1	0	0	1

$$F(A,B,C,D) = \overline{B} + D$$

AB \ CDE	00	01	11	10
000	0	0	0	1
001	1	1	0	1
011	1	1	0	0
010	1	1	0	0
110	0	0	0	0
111	0	0	0	0
101	1	1	0	0
100	1	1	0	0

$$F(A,B,C,D,E) = \overline{A}C\overline{D} + \overline{A}\overline{C}E + \overline{A}\overline{C}D + A\overline{B}\overline{C}\overline{D}$$

ABC \ DE	000	001	011	010	110	111	101	100
00	1	1	1	1	0	0	0	1
01	1	1	1	1	0	0	0	1
11	0	1	1	0	0	0	0	1
10	0	1	1	0	0	0	0	1

$$F(A,B,C,D,E) = A\overline{B}\overline{C} + \overline{A}C + \overline{A}\overline{D}$$

4.8.3. Simplification par la méthode de Quine Mc Cluskey

Edward Joseph McCluskey, Edward Joseph McCluskey, né le 16 octobre 1929 à New York (États-Unis), est un professeur émérite à l'université Stanford. Il est considéré comme l'un des pionniers dans le domaine de l'électrotechnique. Il a travaillé sur les systèmes de commutation électronique chez Bell Telephone Laboratories de 1955 à 1959 [29].

Il a servi de mentor plus de 70 doctorants et a une famille grandissante des universitaires « petits-enfants ». Il est décédé le 13 Février 2016 à New York (États-Unis) [29].

La simplification par la méthode de Quine Mc Cluskey est plus complexe dans son application que la simplification par Karnaugh, cette méthode n'est généralement utilisée que lorsque le nombre de variables de la fonction est important (plus de cinq variables). Elle a l'avantage d'être systématique et donc programmable.

Méthode :

La technique de simplification des fonctions logique par Quine-McCluskey s'applique de la même façon aux expressions disjonctives qu'aux expressions conjonctives. Dans la suite, nous nous intéresserons au cas des expressions disjonctives.

Le principe de la méthode se présente comme suit [30] :

1. Représenter la fonction sous la première forme canonique disjonctive ;
2. Représenter les minterms de la forme canonique disjonctive sous forme binaire ;
3. Regrouper les termes selon leur poids ;
4. Unir les termes deux à deux ;
5. Répéter la quatrième étape autant de fois que nécessaire ;
6. Identifier les impliquants premiers ;
7. Identifier les impliquants premiers essentiels ;
8. Si la fonction est représentée uniquement par ses impliquants premiers essentiels, arrêter;
9. Sinon, choisir les impliquants premiers non essentiels permettant une couverture complète.

Exemple :

Considérons la fonction F représentée par sa table de vérité :

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

1. Représentation de la fonction sous forme canonique disjonctive

$$F(A,B,C,D) = \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}BC\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D + A\overline{B}C\overline{D} + A\overline{B}CD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABC\overline{D}$$

2. Représentation des minterms sous forme binaire

$$F(A,B,C,D) = 0010+0011+0110+1000+1001+1010+1011+1100+1101+1110$$

Pour chaque minterm, les variables sont remplacées par leur équivalent binaire. Si nous avons le complément de la variable on met 0, sinon on met 1. Exemple :

$\overline{A}\,\overline{B}C\overline{D} \Rightarrow 0010$

$\overline{A}\,\overline{B}CD \Rightarrow 0011$

$\overline{A}BC\overline{D} \Rightarrow 0110$

$A\overline{B}\,\overline{C}\,\overline{D} \Rightarrow 1000$

$A\overline{B}\,\overline{C}D \Rightarrow 1001$

3. Regroupement des termes selon leurs poids

Le mot poids signifie le nombre de 1 contenus dans la forme binaire des minterms de la fonction.

1.	0010	poids 1
2.	1000	
3.	0011	
4.	0110	
5.	1001	
6.	1010	poids 2
7.	1100	
8.	1011	
9.	1101	poids 3
10.	1110	

4. Union des termes deux à deux

Il s'agit de créer une nouvelle colonne de termes en combinant deux à deux les termes de la colonne précédente.

1.	0010	✓	001x	généré en combinant 1 et 3
2.	1000	✓	0x10	généré en combinant 1 et 4
3.	0011	✓	x010	généré en combinant 1 et 6
4.	0110	✓	100x	généré en combinant 2 et 5
5.	1001	✓	10x0	généré en combinant 2 et 6
6.	1010	✓	1x00	généré en combinant 2 et 7
7.	1100	✓	x011	généré en combinant 3 et 8
8.	1011	✓	x110	généré en combinant 4 et 10
9.	1101	✓	10x1	généré en combinant 5 et 8
10.	1110	✓	1x01	généré en combinant 5 et 9
			101x	généré en combinant 6 et 8
			1x10	généré en combinant 6 et 10
			110x	généré en combinant 7 et 9
			11x0	généré en combinant 7 et 10

5. Répétition de la quatrième étape autant de fois que nécessaire

On réunit les termes de la nouvelle colonne, la même procédure que nous l'avons fait à l'étape quatre. Afin d'unifier les deux termes, en plus des conditions précédentes, la valeur de x doit être au même endroit.

1.	001x	✓	x01x	généré en combinant 1 et 11
2.	0x10	✓	xx10	généré en combinant 2 et 12
3.	x010	✓	x01x	généré en combinant 3 et 7
4.	100x	✓	xx10	généré en combinant 3 et 8
5.	10x0	✓	10xx	généré en combinant 4 et 11
6.	1x00	✓	1x0x	généré en combinant 4 et 13
7.	x011	✓	10xx	généré en combinant 5 et 9
8.	x110	✓	1xx0	généré en combinant 5 et 14
9.	10x1	✓	1x0x	généré en combinant 6 et 10
10.	1x01	✓	1xx0	généré en combinant 6 et 12
11.	101x	✓		
12.	1x10	✓		
13.	110x	✓		
14.	11x0	✓		

Si le même terme est généré plusieurs fois, une seule copie sera conservée. Ensuite, nous répétons la quatrième étape :

x01x		
xx10		
~~x01x~~		x01x
~~xx10~~		xx10
10xx	⇒	10xx
1x0x		1x0x
~~10xx~~		1xx0
1xx0		
~~1x0x~~		
~~1xx0~~		

- Il est impossible de combiner aucun des termes parce que n'existe pas des termes possédant les x au même endroit.
- Impossible de réunir aucune paire de termes, la cinquième étape est terminée.

6. Identification des impliquants premiers

Le tableau suivant résume toutes les étapes qui ont été effectuées :

0010 ✓	001x ✓	x01x
1000 ✓	0x10 ✓	xx10
0011 ✓	x010 ✓	10xx
0110 ✓	100x ✓	1x0x
1001 ✓	10x0 ✓	1xx0
1010 ✓	1x00 ✓	
1100 ✓	x011 ✓	
1011 ✓	x110 ✓	
1101 ✓	10x1 ✓	
1110 ✓	1x01 ✓	
	101x ✓	
	1x10 ✓	
	110x ✓	
	11x0 ✓	

Tous les termes qui ne sont pas marqués (✓) sont des impliquants premiers : x01x, xx10, 10xx, 1x0x, 1xx0. Cette écriture binaire se lit $\overline{B}C, C\overline{D}, A\overline{B}, A\overline{C}$ et $A\overline{D}$ respectivement.

7. Identifier les impliquants premiers essentiels

Pour identifier les impliquants premiers essentiels, nous utilisons un tableau de sorte que dans les lignes nous ayons tous les impliquants premiers identifiés et nous ayons les minterms de la fonction sur les colonnes. Ensuite, nous passons à l'identification :

	0010	0011	0110	1000	1001	1010	1011	1100	1101	1110
x01x	✓	✓				✓	✓			
xx10	✓		✓			✓				✓
10xx				✓	✓	✓	✓			
1x0x				✓	✓			✓	✓	
1xx0				✓		✓		✓		✓

Un impliquant premier est considéré comme essentiel s'il est le seul à être associé à au moins un minterm. Ainsi, un minterm appartient à un impliquant premier essentiel si sa colonne ne contient qu'une seule astérisque (*).

	0010	0011	0110	1000	1001	1010	1011	1100	1101	1110
x01x	✓	(✓)				✓	✓			
xx10	✓		(✓)			✓				✓
10xx				✓	✓	✓	✓			
1x0x				✓	✓			✓	(✓)	
1xx0				✓		✓		✓		✓

Un impliquant premier est essentiel s'il comporte au moins une étoile entre parenthèses. Pour notre exemple, les impliquants premiers essentiels sont : x01x, xx10 et 1x0x.

8. Vérifier si la fonction est entièrement exprimée par ses impliquants essentiels

Pour cela il faut refaire le tableau tel que en ne gardant que les impliquants essentiels :

	0010	0011	0110	1000	1001	1010	1011	1100	1101	1110
x01x	✓	(✓)				✓	✓			
xx10	✓		(✓)			✓				✓
1x0x				✓	✓			✓	(✓)	

Pour que la fonction soit entièrement décrite par ses impliquant essentiels, chaque colonne doit contenir au moins une étoile. Pour notre exemple, la technique de Quine-McCluskey se termine ici :

$$\begin{aligned} \mathrm{F}(\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}) &= \mathrm{x01x}+\mathrm{xx10}+\mathrm{1x0x} \\ &= \overline{\mathrm{B}}\mathrm{C}+\mathrm{C}\overline{\mathrm{D}}+\mathrm{A}\overline{\mathrm{C}} \end{aligned}$$

- **Deuxième exemple sur la méthode de Quine Mc-Cluskey**

$$F(A,B,C,D)=\overline{A}\overline{B}CD+A\overline{C}D+ABD+AB\overline{D}+\overline{A}C$$

- **Représentation de la fonction sous la forme canonique disjonctive (FND)**

$$\begin{aligned} F(A,B,C,D) &= \overline{A}\overline{B}CD+A\overline{C}D+ABD+AB\overline{D}+\overline{A}C \\ &= \overline{A}\overline{B}CD+A(B+\overline{B})\overline{C}D+AB(C+\overline{C})D+AB(C+\overline{C})\overline{D}+\overline{A}(B+\overline{B})C(D+\overline{D}) \\ &= \overline{A}\overline{B}CD+AB\overline{C}D+A\overline{B}\overline{C}D+ABCD+AB\overline{C}D+ABC\overline{D}+AB\overline{C}\,\overline{D}+\overline{A}BCD+\overline{A}BC\overline{D}+\overline{A}\overline{B}CD+\overline{A}\overline{B}C\overline{D} \\ &= \overline{A}\overline{B}C\overline{D}+\overline{A}\overline{B}CD+\overline{A}BC\overline{D}+\overline{A}BCD+A\overline{B}\overline{C}D+AB\overline{C}\,\overline{D}+AB\overline{C}D+ABC\overline{D}+ABCD \end{aligned}$$

- **Transformation en nombres binaires**

$$\begin{aligned} F(A,B,C,D) &= \overline{A}\overline{B}C\overline{D}+\overline{A}\overline{B}CD+\overline{A}BC\overline{D}+\overline{A}BCD+A\overline{B}\overline{C}D+AB\overline{C}\,\overline{D}+AB\overline{C}D+ABC\overline{D}+ABCD \\ &= 0010+0011+0110+0111+1001+1100+1101+1110+1111 \end{aligned}$$

- **Classification**

1.	0010	poids 1
2.	0011	
3.	0110	poids 2
3.	1001	
4.	1100	
5.	0111	
6.	1101	poids 3
7.	1110	
8.	1111	poids 4

➦ **Comparaisons et Identification des impliquants**

0010	✓	001x	✓	0x1x
0011	✓	0x10	✓	~~0x1x~~
0110	✓	0x11	✓	x11x
1001	✓	011x	✓	~~x11x~~
1100	✓	x110	✓	11xx
0111	✓	1x01		~~11xx~~
1101	✓	110x	✓	
1110	✓	11x0	✓	
1111	✓	x111	✓	
		11x1	✓	
		111x	✓	

➦ **Identification des impliquants essentiels**

	0010	0011	0110	0111	1001	1100	1101	1110	1111
1x01					(✓)		✓		
0x1x	(✓)	(✓)	✓	✓					
x11x			✓	✓				✓	✓
11xx						(✓)	✓	✓	✓

Les impliquants essentiels sont : 1x01, 0x1x et 11xx. Puisqu'ils sont suffisants pour représenter toutes les solutions, la fonction est donc simplifiée comme suit :

$$F(A,B,C,D) = A\overline{C}D + \overline{A}C + AB$$

4.9. Fonctions incomplètement définies

Certaines combinaisons de variables n'ont aucune signification physique et n'apparaissent jamais en réalité. Par conséquent, il n'est pas nécessaire de spécifier une valeur de fonction pour telles combinaisons d'entrées. Dans ce cas, le concepteur peut commodément attribuer une valeur de 0 ou 1 à ces cases pour obtenir le nombre maximum de regroupements.

• il faut mettre un X pour les cas impossibles ou interdites dans la table de vérité.

• dans la table de Karnaugh, les cas impossibles ou interdites sont également représentés par des X.

•Il est possible d'utiliser les X dans des regroupements, soit les considérer comme 1 ou les considérer comme 0.

•Ne formez pas de regroupements contenant uniquement X.

Table de vérité

X	Y	Z	W	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	0
1	1	1	1	1

Simplification de F par l'utilisation de la table de Karnaugh

XY / ZW	00	01	11	10
00	0	0	X	1
01	1	1	X	1
11	0	0	1	X
10	1	1	0	0

$$F(X,Y,Z,W) = \overline{Z}W + X\overline{Z} + XW + \overline{X}Z\overline{W}$$

Série d'exercices N°3 (Algèbre de Boole)

Exercice N°1:

Démontrer algébriquement les relations suivantes :

$a)\quad XY+\bar{X}Z=(\bar{X}+Y)(X+Z)$

$b)\quad XY+\bar{X}Z+YZ=XY+\bar{X}Z$

$c)\quad (X+Y)(\bar{X}+Z)(Y+Z)=(X+Y)(\bar{X}+Z)$

$d)\quad XY+X\bar{Y}Z=XY+XZ$

Exercice N°2:

Mettre les fonctions suivantes sous la première et la deuxième forme canonique :

$1)\quad X=\bar{a}b\bar{c}+abd+bcd$

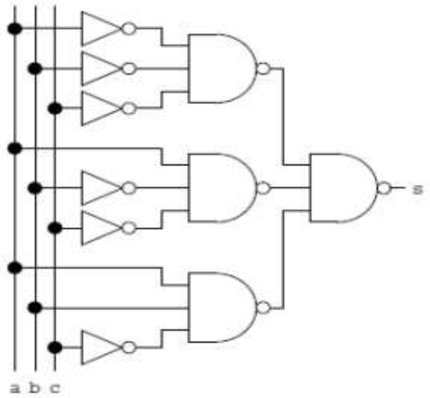
$2)\quad Y=a(b+c)(\bar{c}+\bar{d})$

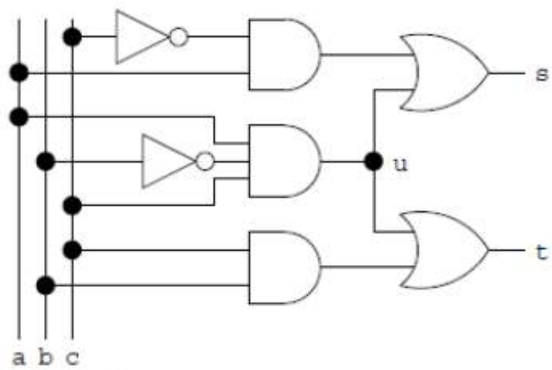
$3)\quad Z=(a+d)(\bar{a}+c+d)+\bar{a}\bar{b}$

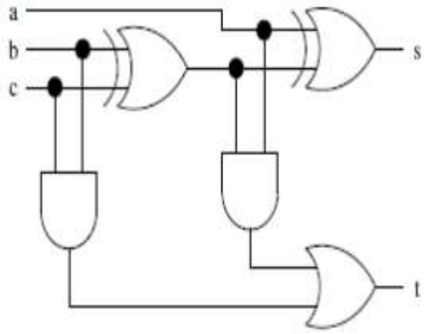
Exercice N°3:

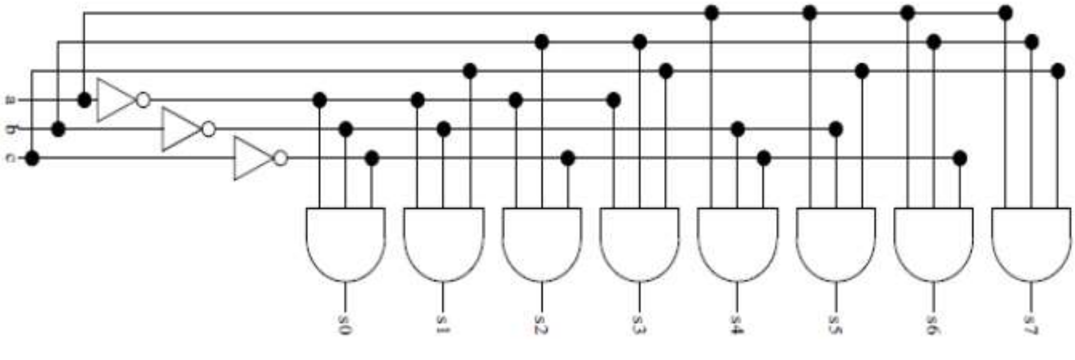
a/ donner le logigramme de la fonction suivante : $Y=(A\oplus B)\left(\overline{A\bar{B}C+A\overline{BC}+AB\bar{C}}\right)$

b/ à partir de ces schémas donner les équations de sortie :









Exercice N°4:

Extraire des équations simplifiées à l'aide des tables de KARNAUGH suivantes.

AB / CD	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	0	1	1	0
10	0	1	1	0

AB / CD	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1

AB / CD	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

AB / CD	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	1	1	1	1
10	1	1	1	1

AB / CD	00	01	11	10
00	0		1	0
01	1	1	1	1
11	1	1	1	1
10	0	1	1	0

AB / CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	0
10	1	1	1	1

AB / CD	00	01	11	10
00	1	0	1	1
01	1	0	1	1
11	0	0	1	1
10	1	1	1	1

AB / CD	00	01	11	10
00	1	1	1	1
01	0	1	1	0
11	0	1	1	0
10	1	1	1	1

AB / CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	1	0	0
10	0	0	0	0

AB / CD	00	01	11	10
00	0	0	1	0
01	1	0	1	1
11	1	1	1	1
10	0	0	1	0

Exercice N°5:

Soit la fonction suivante :

$$F(A,B,C,D) = A\overline{C} + ABD + AB\overline{D} + \overline{A}C + \overline{AB}$$

a) Simplifier par la méthode de karnaugh (4 variables)

b) Retrouvez le même résultat algébriquement.

Exercice N°6:

Soit la fonction algébrique suivante :

$$F(A,B,C,D) = ABCD + AB\overline{C}D + A\overline{B}C + \overline{A}BCD + AB + C\overline{D}$$

a\. Donner la table de vérité de F.

b\. Ecrire la 1er et la 2ème formes canoniques de F (FND et FNC).

c\. Simplifier F en utilisant les lois de l'algèbre de Boole.

d\. Représenter la fonction F sous forme de table de Karnaugh, puis simplifier F.

e\. Faire les logigrammes simplifiés.

Exercice N°7:

Soit la fonction algébrique suivante :

$$F(A,B,C) = AB\overline{C} + A\overline{B} + \overline{A}\overline{B}\overline{C} + \overline{B}C$$

a\. Donner la table de vérité de F.

b\. Ecrire la 1er et la 2ème formes canoniques de F (FND et FNC).

c\. Simplifier F en utilisant les lois de l'algèbre de Boole.

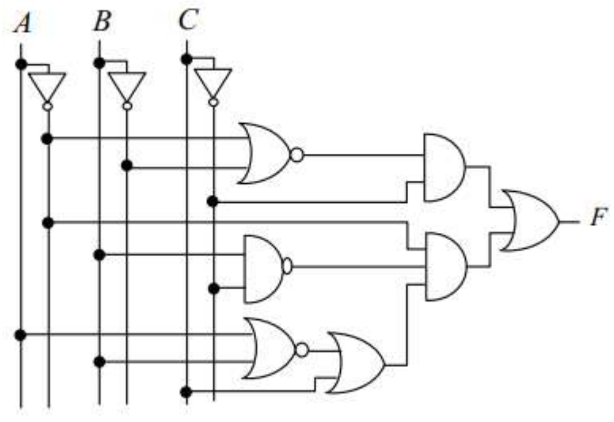
d\. Représenter la fonction F sous forme de table de Karnaugh, puis simplifier F.

e\. Simplifier F en utilisant la méthode de Quine Mc-Cluskey.

f\. Faire les logigrammes simplifiés.

Exercice N°8:

Soit la fonction logique f définie par le logigramme suivant :

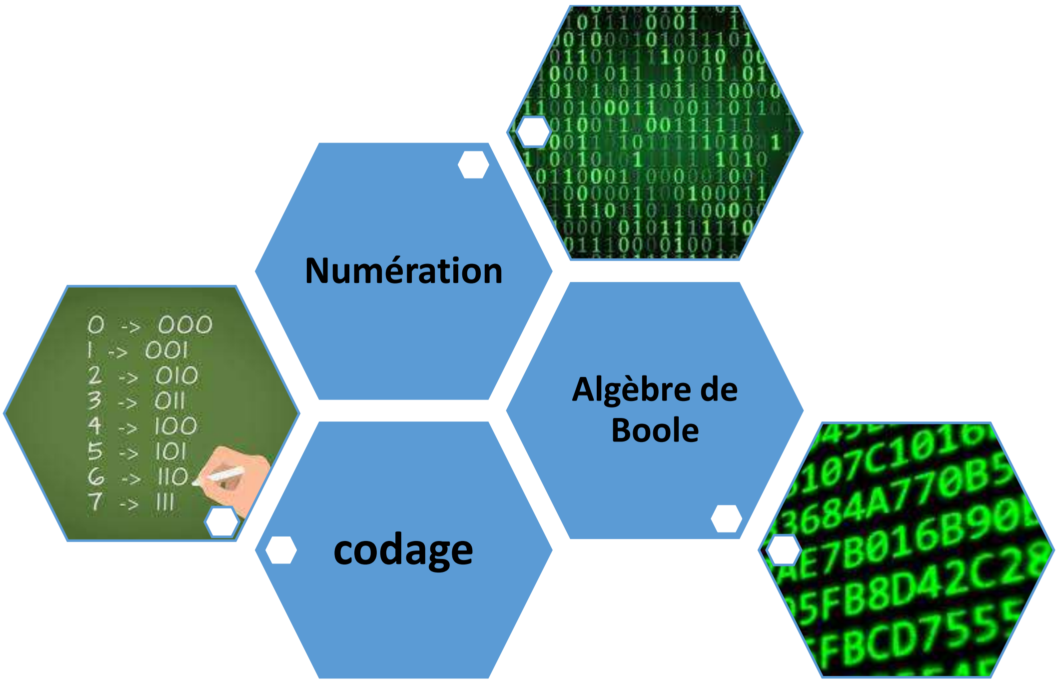


1. Donner l'expression logique de la fonction de sortie F.
2. Donner la table de vérité de F.
3. Simplifier par la méthode de karnaugh.
4. Simplifier par la méthode de Quine-McCluskey.
5. Simplifier F en utilisant les lois de l'algèbre de Boole.

Exercice N°9:

Réaliser un circuit complément à 1 à 4 bits.

Solution des séries d'exercices



Solution des séries d'exercices

1. Solution de la série d'exercices N°1 (Systèmes de Numération)

Exercice N°1 :

1. Tableau de correspondance des 25 premiers entiers dans les bases suivantes : 2, 5, 6, 8, 11 et 16.

Base 10	Base 2	Base 5	Base 7	Base 8	Base 12	Base 16
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	10	2	2	2	2	2
3	11	3	3	3	3	3
4	100	4	4	4	4	4
5	101	10	5	5	5	5
6	110	11	6	6	6	6
7	111	12	10	7	7	7
8	1000	13	11	10	8	8
9	1001	14	12	11	9	9
10	1010	20	13	12	A	A
11	1011	21	14	13	10	B
12	1100	22	15	14	11	C
13	1101	23	16	15	12	D
14	1110	24	20	16	13	E
15	1111	30	21	17	14	F
16	10000	31	22	20	15	10
17	10001	32	23	21	16	11
18	10010	33	24	22	17	12
19	10011	34	25	23	18	13
20	10100	40	26	24	19	14
21	10101	41	30	25	1A	15
22	10110	42	31	26	20	16
23	10111	43	32	27	21	17
24	11000	44	33	30	22	18

2. Les représentations en binaire (base 2), en octal (base 8) et puis en hexadécimal (base 16) des nombres décimaux suivants : $(0)_{10}$, $(11)_{10}$, $(255)_{10}$, $(34,125)_{10}$, $(13,6)_{10}$, $(54,18)_{10}$

➦ $(0)_{10}$

$(0)_{10} = (0)_2$

$= (0)_8$

$= (0)_{16}$

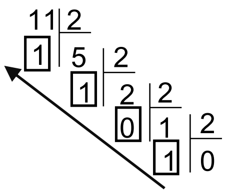
$$\begin{array}{c|c} 0 & 2 \\ \cline{2-2} \boxed{0} & 0 \end{array}$$

➦ $(11)_{10}$

$(11)_{10} = (1011)_2$

$=(001\ 011)_2 = (13)_8$

$=(1011)_2 = (B)_{16}$

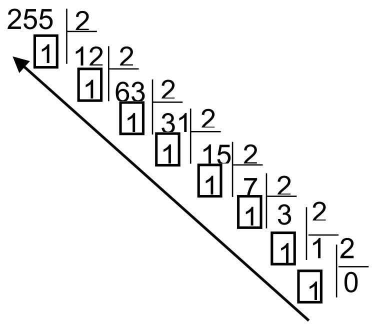


➦ $(255)_{10}$

$(255)_{10} = (11111111)_2$

$=(011\ 111\ 111)_2 = (377)_8$

$=(1111\ 1111)_2 = (FF)_{16}$



➦ $(34{,}125)_{10}$

$(34{,}125)_{10} = (100010{,}001)_2$

$=(100\ 010{,}001)_2 = (42{,}1)_8$

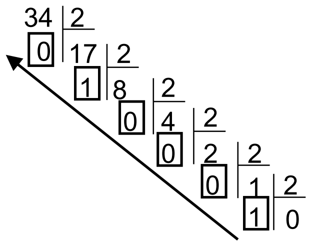
$=(0010\ 0010{,}0010)_2$

$=(22{,}2)_{16}$

$0{,}125*2=\boxed{0},25$

$0{,}25*2=\boxed{0},5$

$0{,}5*2=\boxed{1},0$



➦ $(13{,}6)_{10}$

$(13{,}6)_{10} = (1101{,}100110011001\ldots)_2$

$=(001\ 101{,}100\ 110\ 011\ 001\ldots)_2$

$=(15{,}4631..)_8$

$=(1101{,}1001\ 1001\ 1001\ldots)_2$

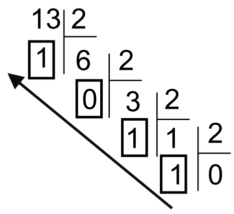
$=(D{,}999\ldots)_{16}$

$0{,}6*2=\boxed{1},2$

$0{,}2*2=\boxed{0},4$

$0{,}4*2=\boxed{0},8$

$0{,}8*2=\boxed{1},6$



➦ $(54{,}18)_{10}$

$(54{,}18)_{10} = (110110{,}001011100001\ldots\ldots)_2$

$=(110\ 110{,}001\ 011\ 100\ 001\ldots\ldots)_2$

$=(66{,}1341\ldots)_8$

$=(0011\ 0110{,}0010\ 1110\ 0001\ldots\ldots)_2$

$=(36{,}2E1\ldots..)_{16}$

$0{,}18*2=\boxed{0}.36$

$0{,}36*2=\boxed{0},72$

$0{,}72*2=\boxed{1},44$

$0{,}44*2=\boxed{0},88$

$0{,}88*2=\boxed{1}.76$

$0{,}76*2=\boxed{1},52$

$0{,}52*2=\boxed{1},04$

$0{,}04*2=\boxed{0},08$

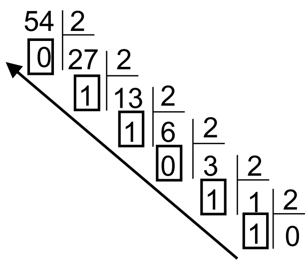
$0{,}08*2=\boxed{0}.16$

$0{,}16*2=\boxed{0},32$

$0{,}32*2=\boxed{0},64$

$0{,}64*2=\boxed{1},28$

. . . .



3. Les équivalents décimaux des nombres suivants :

➦ **$(101,11)_2 = (……………………)_{10}$**

$(101,11)_2 = (?)_{10}$

$(101,11)_2 = 1*2^2 +0*2^1 +1*2^0 +1*2^{-1}+1*2^{-2}$.

$=4+1+0,5+0,25$

$= (5,75)_{10}$

➦ **$(10000,00)_2 = (……………………)_{10}$**

$(10000,00)_2 = (?)_{10}$

$(10000,00)_2 = 1*2^4 +0*2^3 +0*2^2 +0*2^1+0*2^0+0*2^{-1}+0*2^{-2}$.

$=16$

$= (16)_{10}$

➦ **$(1,1)_2 = (……………………)_{10}$**

$(1,1)_2 = (?)_{10}$

$(1,1)_2 = 1*2^0 +1*2^{-1}$.

$=1+0,5$

$= (1,5)_{10}$

➦ **$(1234)_8 = (……………………)_{10}$**

$(1234)_8 = (?)_{10}$

$(1234)_8 = 1*8^3 +2*8^2 +3*8^1 +4*8^0$.

$=512+128+24+4$

$= (668)_{10}$

➦ **$(10,132)_8 = (……………………)_{10}$**

$(10,132)_8 = (?)_{10}$

$(10,132)_8 = 1*8^1 +0*8^0 +1*8^{-1} +3*8^{-2}+2*8^{-3}$.

$=8+0,125+0,046875+0,00390625$

$= (8,17578125)_{10}$

➦ **$(111,11)_8 = (……………………)_{10}$**

$(111,11)_8 = (?)_{10}$

$(111,11)_8 = 1*8^2 +1*8^1 +1*8^0 +1*8^{-1}+1*8^{-2}$

$=64+8+1+0.125+0,015625$

$= (73,140625)_{10}$

➦ **$(A04,12)_{16}$ = (……………………)$_{10}$**

$(A04,12)_{16} = (?)_{10}$

$(A04,12)_{16} = A*16^2 +0*16^1 +4*16^0 +1*16^{-1}+2*16^{-2}$

$= 10*16^2 +0*16^1 +4*16^0 +1*16^{-1}+2*16^{-2}$

$= 2560+4+0{,}0625+0{,}0078125$

$= (2564{,}0703125)_{10}$

➦ **$(BAC23)_{16}$ = (……………………)$_{10}$**

$(BAC23)_{16} = (?)_{10}$

$(BAC23)_{16} = B*16^4 +A*16^3 +C*16^2 +2*16^1+3*16^0.$

$= 11*16^4 +10*16^3 +12*16^2 +2*16^1+3*16^0$

$= 720896+40960+3072+32+3$

$= (764963)_{10}$

4. Représentation directe des nombres suivants en base 2 (sans passer par la méthode de division successive) :

➦ **X=$(1320)_4$ = (……………………)$_2$**

X=$(1320)_4$

$= (01\ 11\ 10\ 00)_2$

$= (1111000)_2$

➦ **Y=$(307,5)_8$ = (……………………)$_2$**

Y=$(307,5)_8$

$= (011\ 000\ 111{,}101)_2$

$= (11000111{,}101)_2$

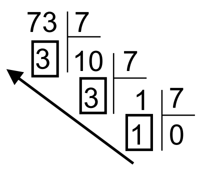
➦ **Z=$(BAC,BEF)_{16}$ = (……………………)$_2$**

Z=$(BAC,BEF)_{16}$

$= (1011\ 1010\ 1100{,}1011\ 1110\ 1111)_2$

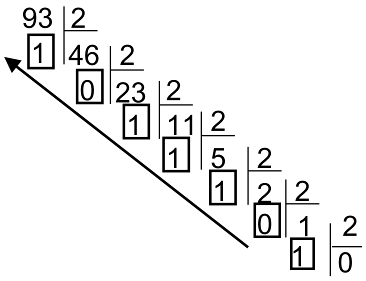
Exercice N°2 :

➦ **$(73)_{10}$ = (…………………)$_7$**



$(73)_{10} = (\ 133)_7$

➦ **$(93{,}625)_{10}$ = (…………………….. $)_2$**



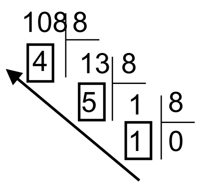
$0{,}625*2 = \boxed{1},25$

$0{,}25*2 = \boxed{0},5$

$0{,}5*2 = \boxed{1},0$

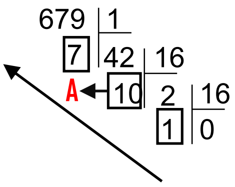
$(93{,}625)_{10} = (1011101{,}101)_2$

➦ **$(108)_{10}$ = (………………….. $)_8$**



$(108)_{10} = (154)_8$

➦ **$(679{,}93359375)_{10}$ = (……………………. $)_{16}$**



$0{,}93359375*16 = \boxed{14},9375 \longrightarrow E$

$0{,}9375*16 = \boxed{15},0 \longrightarrow F$

$(679{,}93359375)_{10} = (1A7{,}EF)_{16}$

➦ **$(4103)_5$ = (……………………. $)_{10}$**

$(4103)_5 = (?)_{10}$

$(4103)_5 = 4*5^3 + 1*5^2 + 0*5^1 + 3*5^0.$

$= 500+25+3$

$= (528)_{10}$

➦ **$(31121{,}232)_4$ = (……………………. $)_{10}$**

$(31121{,}232)_4 = (?)_{10}$

$(31121{,}232)_4 = 3*4^4 + 1*4^3 + 1*4^2 + 2*4^1 + 1*4^0 + 2*4^{-1} + 3*4^{-2} + 2*4^{-3}.$

$= 768+64+16+8+1+0{,}5+0{,}1875+0{,}03125$

$= (857{,}71875)_{10}$

➦ **$(2034)_5=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_9$**

$(2034)_5=(\ ?)_9$

$(2034)_5=(\ ?)_{10}$

$(2034)_5=2*5^3+0*5^2+3*5^1+4*5^0$

$=(269)_{10}$

$(269)_{10}=(\ ?)_9$

$$\begin{array}{llll}
269 & \vert\, 9 & & \\
\boxed{8} & \vert\, 29 & \vert\, 9 & \\
 & \boxed{2} & \vert\, 3 & \vert\, 9 \\
 & & \boxed{3} & \vert\, 0
\end{array}$$

$(2034)_5=(269)_{10}$

$(269)_{10}=(328)_9$

$(2034)_5=(328)_9$

➦ **$(1023,02)_4=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_6$**

$(1023,02)_4=(\ ?)_6$

$(1023,02)_4=(\ ?)_{10}$

$(1023,02)_4=1*4^3+0*4^2+2*4^1+3*4^0+0*4^{-1}+2*4^{-2}$

$=(75,125)_{10}$

$(75,125)_{10}=(\ ?)_6$

$$\begin{array}{llll}
75 & \vert\, 6 & & \\
\boxed{3} & \vert\, 12 & \vert\, 6 & \\
 & \boxed{0} & \vert\, 2 & \vert\, 6 \\
 & & \boxed{2} & \vert\, 0
\end{array}$$

$(1023,02)_4=(75,125)_{10}$

$0,125*6=\boxed{0},75$

$0,75*6\ \ =\boxed{4},5$

$0,5*6\ \ \ \ =\boxed{3},0$

$(75,125)_{10}=(203,043)_6$

$(1023,02)_4=(203,043)_6$

➦ **$(104,2)_5=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_6$**

$(104,2)_5=(\ ?)_6$

$(104,2)_5=(\ ?)_{10}$

$(29,4)_{10}=(\ ?)_6$

$(104,2)_5=1*5^2+0*5^1+4*5^0+2*5^{-1}$

$=(29,4)_{10}$

$$\begin{array}{lll}
29 & \vert\, 6 & \\
\boxed{5} & \vert\, 4 & \vert\, 6 \\
 & \boxed{4} & \vert\, 0
\end{array}$$

$0,4*6=\boxed{2},4$

$(104,2)_5=(29,4)_{10}$

$(29,4)_{10}=(54,222\ldots..)_6$

$(104,2)_5=(54,222\ldots\ldots)_6$

➦ **$(10111000,101)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_4$**

$(10111000,101)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_4$

$(10111000,101)_2=(\underline{10}\ \underline{11}\ \underline{10}\ \underline{00},\underline{10}\ \underline{10})_2$

$=(2320,22)_4$

➦ **$(10110101101,11011)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_8$**

$(10110101101,11011)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_8$

$(10110101101,11011)_2=(\underline{010}\ \underline{110}\ \underline{101}\ \underline{101},\underline{110}\ \underline{110})_2$

$=(2655,66)_8$

- **$(100101011100{,}011101)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_{16}$**

$(100101011100{,}011101)_2=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_{16}$

$(100101011100{,}011101)_2=(\underline{1001}\ \underline{0101}\ \underline{1100}{,}\underline{0111}\ \underline{0100})_2$

$=(95C{,}74)_{16}$

- **$(135{,}04)_8=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$**

$(135{,}04)_8=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$

$(135{,}04)_8=(\underline{001}\ \underline{011}\ \underline{101}{,}\underline{000}\ \underline{100})_2$

$=(1011101{,}0001)_2$

- **$(A6C{,}01E)_{16}=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$**

$(A6C{,}01E)_{16}=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$

$(A6C{,}01E)_{16}=(\underline{1010}\ \underline{0110}\ \underline{1100},\ \underline{0000}\ \underline{0001}\ \underline{1110})_2$

$=(101001101100{,}00000001111)_2$

- **$(F92A{,}20F)_{16}=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_8$**

$(F92A{,}20F)_{16}=(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_8$

$(F92A{,}20F)_{16}=(1111\ 1001\ 0010\ 1010{,}0010\ 0000\ 1111)_2$

$=(\underline{001}\ \underline{111}\ \underline{100}\ \underline{100}\ \underline{101}\ \underline{010}{,}\underline{001}\ \underline{000}\ \underline{001}\ \underline{111})_2$

$=(174452{,}1017)_8$

- **$(11010110101{,}01011)_2=(\ldots\ldots\ldots\ldots\ldots)_4=(\ldots\ldots\ldots\ldots)_8=(\ldots\ldots\ldots\ldots\ldots)_{16}$**

$(11010110101{,}01011)_2=(\ldots\ldots\ldots\ldots\ldots)_4=(\ldots\ldots\ldots\ldots)_8=(\ldots\ldots\ldots\ldots\ldots)_{16}$

$(11010110101{,}01011)_2=(01\ 10\ 10\ 11\ 01\ 01{,}01\ 01\ 10)_2=(122311{,}112)_4$

$=(011\ 010\ 110\ 101{,}010\ 110)_2=(3265{,}26)_8$

$=(0110\ 1011\ 0101{,}0101\ 1000)_2=(6B5{,}58)_{16}$

Exercice N°3 :

1. Les nombres qui possède la même représentation dans les systèmes binaire, octal, décimal et hexadécimal sont : $\{0,1\}$

2. Les nombres qui possède la même représentation dans les systèmes octal, décimal et hexadécimal sont : $\{0,1,2,3,4,5,6,7\}$

3. Les nombres qui ont un sens en hexadécimal sont : BAC- CAFE- BAFFE- DECADE- BEF - FA5D-F00D-C0DE- A1DE.

4. Le nombre des entiers positifs qui peuvent être représentés avec n chiffres en base b est : B^n

Exercice N°4 :

1. Détermination des bases (X,Y, Z et W) dans lesquelles les nombres sont représentés:

➦ $(24)_X = (14)_{10}$

$(24)_X = (14)_{10} \Rightarrow 2*X^1+4*X^0=14$

$\Rightarrow 2X+4=14$

$\Rightarrow X=(14-4)/2$

$\Rightarrow X=5$

➦ $(13)_Y = (7)_{10}$

$(13)_Y = (7)_{10} \Rightarrow 1*Y^1+3*Y^0=7$

$\Rightarrow Y+3=7$

$\Rightarrow Y=7-3$

$\Rightarrow Y=4$

➦ $(70)_Z = (56)_{10}$

$(70)_Z = (56)_{10} \Rightarrow 7*Z^1+0*Z^0=56$

$\Rightarrow 7Z=56$

$\Rightarrow Z=56/7$

$\Rightarrow Z=8$

➦ $(1A0)_W = (416)_{10}$

$(1A0)_W = (416)_{10} \Rightarrow 1*W^2+A*W^1+0*W^0=416$

$\Rightarrow W^2+10W=416$

$\Rightarrow W^2+10W-416=0$

$$\begin{aligned} \Delta &= B^2 - 4AC \\ &= 10^2 - 4*1*(-416) \\ &= 1764 \quad \Rightarrow \sqrt{\Delta} = 42 \end{aligned}$$

$$\begin{cases} W_1 = \dfrac{-B+\sqrt{\Delta}}{2A} = \dfrac{-10+42}{2} = 16 \in N \\ W_2 = \dfrac{-B-\sqrt{\Delta}}{2A} = \dfrac{-10-42}{2} = -26 \notin N \end{cases} \qquad \Rightarrow W = 16$$

2. Détermination des entiers (X, Y) tel que : $(XY)_7 = (YX)_{10}$:

$(XY)_7 = (YX)_{10} \Rightarrow X*7^1+Y*7^0=Y*10^1+X*10^0$

$\Rightarrow 7X+Y=10Y+X$

$\Rightarrow 10Y-Y=7X-X$

$\Rightarrow 9Y=6X$

$\Rightarrow Y=\frac{6}{9}X$

$\Rightarrow Y=\frac{2}{3}X$

En plus (XY) c'est un nombre dans la base 7, c'est-à-dire donc 0<X<7 et 0<Y<7, alors :

$$\begin{cases} \text{si} \quad X = 0 \in N \Rightarrow Y = 0 \in N \\ \text{si} \quad X = 1 \in N \Rightarrow Y = \frac{2}{3} \notin N \\ \text{si} \quad X = 2 \in N \Rightarrow Y = \frac{4}{3} \notin N \\ \text{si} \quad X = 3 \in N \Rightarrow Y = 2 \in N \\ \text{si} \quad X = 4 \in N \Rightarrow Y = \frac{8}{3} \notin N \\ \text{si} \quad X = 5 \in N \Rightarrow Y = \frac{10}{3} \notin N \\ \text{si} \quad X = 6 \in N \Rightarrow Y = 4 \in N \end{cases} \Rightarrow \text{donc il existe trois solutions sont :} \begin{cases} X = 0 \text{ et } Y = 0 \\ X = 3 \text{ et } Y = 2 \\ X = 6 \text{ et } Y = 2 \end{cases}$$

3. Représentation des nombres décimaux en base a :

X= $(4a^5+2a^3+a+5)_{10}$
=(402015)a

Y= $(a)_{10}$
=(10)a

Z= $(a^2)_{10}$
=(100)a

W= $(a^3)_{10}$
=(1000)a

Exercice N°5:

➦ $(1001110{,}11)_2 + (11011{,}101)_2 = (\ldots\ldots)_2$

➦ $(11011{,}101)_2 + (10111{,}111)_2 = (\ldots\ldots)_2$

$$\left(\begin{array}{rr} & {\scriptstyle 111111} \\ & 1001110{,}110 \\ + & 11011{,}101 \\ \hline = & 1101010{,}011 \end{array}\right)_2 \qquad \left(\begin{array}{rr} & {\scriptstyle 11\,111\;\;11} \\ & 11011{,}101 \\ + & 10111{,}111 \\ \hline = & 110011{,}100 \end{array}\right)_2$$

$$(1001110{,}11)_2 + (11011{,}101)_2 = (1101010{,}011)_2 \qquad (11011{,}101)_2 + (10111{,}111)_2 = (110011{,}100)_2$$

➦ $(1110{,}011)_2 + (1101{,}11)_2 + (1110{,}111)_2 = (\ldots\ldots)_2$

$$\left(\begin{array}{rr} & {\scriptstyle 10\ 1\ 1\ 10\ \ 101} \\ & 1110{,}011 \\ + & 1101{,}110 \\ + & 1110{,}111 \\ \hline = & 101011{,}000 \end{array}\right)_2$$

$$(1110{,}011)_2 + (1101{,}11)_2 + (1110{,}111)_2 = (101011)_2$$

➦ $(101001{,}001)_2 - (11111{,}11)_2 = (\ldots\ldots\ldots)_2$

$$\left(\begin{array}{rr} & 1_1 0_1 1_1 0_1 0_1 1,\ _1 0_1 01 \\ - & {}_1\,{}_1 1_1 1_1 1_1 1_1 1,\ _1 1\,10 \\ \hline = & 0\ 0\ 1\ 0\ 0\ 1,\ 0\ \ 1\ 1 \end{array}\right)_2$$

$$(101001{,}001)_2 - (11111{,}11)_2 = (1001{,}011)_2$$

➦ $(1011{,}011)_2 * (110)_2 = (\ldots\ldots\ldots\ldots\ldots\ldots)_2$

$$\left(\begin{array}{rr} & 1011{,}011 \\ * & 110 \\ \hline = & 0000000 \\ + & 1011011\bullet \\ + & 1011011\bullet\bullet \\ \hline = & 1000100{,}010 \end{array}\right)_2$$

$$(1011{,}011)_2 * (110)_2 = (1000100{,}01)_2$$

➦ $(1001001{,}11)_2 / (101)_2 = (\ldots\ldots\ldots\ldots\ldots..)_2$

$$\left(\begin{array}{l|l} 1001001{,}11 & 101 \\ \cline{2-2} -\ 101 & 1110{,}11 \\ =1001 & \\ \quad -\ 101 & \\ \quad =\ 0110 & \\ \qquad -\ 101 & \\ \qquad =\ 0011 & \\ \qquad\ -\ 000 & \\ \qquad\ =\ 0111 & \\ \qquad\quad -101 & \\ \qquad\quad =\ 0101 & \\ \qquad\qquad -101 & \\ \qquad\qquad =000 & \end{array}\right)_2$$

$$(1001001{,}11)_2 \div (101)_2 = (1110{,}11)_2$$

➦ $(73{,}7)_8 + (65{,}3)_8 = (\ldots\ldots\ldots\ldots\ldots..)_8$

$$\left(\begin{array}{rr} & {\scriptstyle 1\ 1} \\ & 73{,}7 \\ + & 65{,}3 \\ \hline = & 161{,}2 \end{array}\right)_8$$

$$(73{,}7)_8 + (65{,}3)_8 = (161{,}2)_8$$

➦ $(531)_8 - (167)_8 = (\ldots\ldots\ldots\ldots\ldots..)_8$

$$\left(\begin{array}{rr} & 5_1 3_1 1 \\ - & {}_1 1_1 67 \\ \hline = & 3\,4\,2 \end{array}\right)_8$$

$$(531)_8 - (167)_8 = (342)_8$$

➦ $(26,5)_8$ x $(4,3)_8$ =(……………..)$_8$

$$\left(\begin{array}{lr} & {}^{3\,2} \\ & {}^{2\,1} \\ & 26,5 \\ * & 4,3 \\ \hline = & 737 \\ + & 1322\bullet \\ \hline = & 141,57 \end{array}\right)_8$$

$$(26,5)_8 * (4,3)_8 = (141,57)_8$$

➦ $(31,7)_8$ x$(52)_8$ =(……………..)$_8$

$$\left(\begin{array}{lr} & {}^{1\,4} \\ & {}^{1} \\ & 31,7 \\ * & 52 \\ \hline = & 636 \\ + & 2013\bullet \\ \hline = & 2076,6 \end{array}\right)_8$$

$$(31,7)_8 * (52)_8 = (2076,6)_8$$

➦ $(C3E)_{16}$ +$(6AD)_{16}$=(……………..)$_{16}$

$$\left(\begin{array}{lr} & {}^{1} \\ & C3E \\ + & 6AD \\ \hline = & 12EB \end{array}\right)_{16}$$

$$(C3E)_{16} + (6AD)_{16} = (12EB)_{16}$$

➦ $(E31)_{16}$-$(6EC)_{16}$=(……………..)$_{16}$

$$\left(\begin{array}{lr} & E_1 3_1 1 \\ - & {}_1 6_1 EC \\ \hline = & 7\;4\;5 \end{array}\right)_{16}$$

$$(E31)_{16} - (6EC)_{16} = (745)_{16}$$

2. Solution de la série d'exercices N°2 (La représentation de l'information)

Exercice N°1

A - Le tableau/

décimal	Binaire pur sur 7 bits	gray	BCD
0	0000000	0	0000
1	0000001	1	0001
2	0000010	11	0010
3	0000011	10	0011
4	0000100	110	0100
5	0000101	111	0101
6	0000110	101	0110
7	0000111	100	0111
8	0001000	1100	1000
9	0001001	1101	10001
10	0001010	1111	0001 0000
11	0001011	1110	0001 0001
12	0001100	1010	0001 0010
13	0001101	1011	0001 0011
14	0001110	1001	0001 0100
15	0001111	1000	0001 0101
16	0010000	11000	0001 0110
17	0010001	11001	0001 0111
18	0010010	11011	0001 1000
19	0010011	11010	0001 1001
20	0010100	11110	0010 0000

B - La différence entre le code binaire pur et le code Gray (binaire réfléchi) est :

Le binaire pur : autrement dit le code binaire naturel est un code pondéré dont les poids sont exprimés par les puissances successives de 2.

Le binaire réfléchi (code GRAY) : il s'agit d'un type de codage binaire qui permet de modifier un seul bit à la fois lorsque le nombre est augmenté d'une unité. C'est à dire est un code construit d'une manière qu'à partir du chiffre 0 chaque nombre consécutif diffère du précédent immédiat d'un seul chiffre.

Exercice N°2

Le nombre suivant pour chaque nombre donné en code Gray :

$$(1101010010)_{Gray} \xrightarrow{\text{Suivant}} (1101010110)_{Gray}$$
$$(1011011011)_{Gray} \xrightarrow{\text{Suivant}} (1011011001)_{Gray}$$
$$(111110001)_{Gray} \xrightarrow{\text{Suivant}} (111110000)_{Gray}$$
$$(110110000)_{Gray} \xrightarrow{\text{Suivant}} (110110001)_{Gray}$$
$$(110011100)_{Gray} \xrightarrow{\text{Suivant}} (110010100)_{Gray}$$

Exercice N°3

Conversion des nombres binaires suivants vers le code Gray :

$(11011)_2$
$\downarrow$
$(10110)_{Gray}$

➲ $(11011)_2=(10110)_{Gray}$

$(1001010)_2$
$\downarrow$
$(1101111)_{Gray}$

➲ $(1001010)_2=(1101111)_{Gray}$

$(11101101110)_2$
$\downarrow$
$(10011011001)_{Gray}$

➲ $(11101101110)_2=(10011011001)_{Gray}$

$(11000110)_2$
$\downarrow$
$(10100101)_{Gray}$

➲ $(11000110)_2=(10100101)_{Gray}$

$(101101)_2$
$\downarrow$
$(111011)_{Gray}$

➲ $(101101)_2=(111011)_{Gray}$

Exercice N°4

La Conversion de chaque code Gray en binaire :

$(1010)_{Gray}$
$(1100)_2$

➲ $(1010)_{Gray}=(1100)_2$

$(10010)_{Gray}$
$(11100)_2$

➲ $(10010)_{Gray}=(11100)_2$

$(11000010001)_{Gray}$

$(10000011110)_2$

➲$(11000010001)_{Gray}=(10000011110)_2$

$(10101111)_{Gray}$

$(11001010)_2$

➲$(10101111)_{Gray}=(11001010)_2$

$(1000111)_{Gray}$

$(1111010)_2$

➲$(1000111)_{Gray}=(1111010)_2$

Exercice N°5

1. Les représentations des nombres suivants en code BCD, Excédent+3, binaire et en Gray

➲$(4389)_{10}$ = $(0100\ 0011\ 1000\ 1001)_{BCD}$

= $(0111\ 0110\ 1011\ 1100)_{Excedent+3}$

= $(1000100100101)_2$

= $(1100110110111)_{gray}$

$(1000100100101)_2$

$(1100110110111)_{Gray}$

➲ $(1000100100101)_2=(1100110110111)_{Gray}$

➲$(2023)_{10}$ = $(0010\ 0000\ 0010\ 0011)_{BCD}$

= $(0101\ 0011\ 0101\ 0110)_{Excedent+3}$

= $(11111100111)_2$

= $(10000010100)_{gray}$

$(11111100111)_2$

$(10000010100)_{Gray}$

➲ $(11111100111)_2=(10000010100)_{Gray}$

➲$(512)_{10}$ = $(0101\ 0001\ 0010)_{BCD}$

= $(1000\ 0100\ 0101)_{Excedent+3}$

= $(1000000000)_2$

= $(1100000000)_{gray}$

$(1000000000)_2$

$(1100000000)_{Gray}$

➲ $(1000000000)_2=(1100000000)_{Gray}$

2. Les représentations des nombres suivants en code BCD+3

➦ **$(1001101)_2$=(………………….)$_{XS3}$**

$(1001101)_2 = (77)_{10}$

$= (1010\ 1010)_{XS3}$

➦ **$(11101101)_{Gray}$=(………………….)$_{XS3}$**

$(11101101)_{Gray}$=(…………………….)$_2$

$(11101101)_{Gray}$

$(10110110)_2$

➲ $(11101101)_{Gray} = (10110110)_2$

$(11101101)_{Gray} = (10110110)_2$

$= (182)_{10}$

$= (0100\ 1011\ 0101)_{XS3}$

➦ **$(50421)_8$=(………………….)$_{XS3}$**

$(50421)_8 = (20753)_{10}$

$= (0101\ 0011\ 1010\ 1000\ 0110)_{XS3}$

Exercice N°6 :

Les résultants des opérations suivantes en code BCD :

➦ **$(158)_{10}+(641)_{10}$ en BDC**

$$
\begin{array}{llr}
158 & \xrightarrow{BCD} & 0001\ 0101\ 1000 \\
+ & & + \\
\underline{641} & \xrightarrow{BCD} & \underline{0110\ 0100\ 0001} \\
= 799 & & = \underline{0111}\ \underline{1001}\ \underline{1001} \\
 & & 7 \quad\ \ 9 \quad\ \ 9
\end{array}
$$

➦ **$(77)_{10}+(33)_{10}$ en BDC**

$$
\begin{array}{llr}
77 & \xrightarrow{BCD} & 0111\ \ 0111 \\
+ & & + \\
\underline{33} & \xrightarrow{BCD} & \underline{0011\ 0011} \\
= 110 & & = 1010\ 1010 \\
 & & \underline{+\ 0110} \\
 & & = \ 1011\ 0000 \\
 & & \underline{+0110\quad\quad} \\
 & & = \underline{0001}\ \underline{0001}\ \underline{0000} \\
 & & 1 \quad\ \ 1 \quad\ \ 0
\end{array}
$$

- $(199)_{10}+(644)_{10}$ **en BDC**

$199 \xrightarrow{\text{BCD}} 0001\ 1001\ 1001$

$+ \qquad\qquad +$

$644 \xrightarrow{\text{BCD}} 0110\ 0100\ 0100$

$=843 \qquad = 0111\ 1101\ 1101$

$+\ 0110$

$= 0111\ 1110\ 0011$

$+0110$

$= \underline{1000}\ \underline{0100}\ \underline{0011}$

8 4 3

- $(781)_{10}-(167)_{10}$ **en BDC**

$781 \xrightarrow{\text{BCD}} 0111\ 1000\ 0001$

$- \qquad\qquad -$

$167 \xrightarrow{\text{BCD}} 0001\ 0110\ 0111$

$= 614 \qquad = 0110\ 0001\ 1010$

$-\ 0110$

$= \underline{0110}\ \underline{0001}\ \underline{0100}$

6 1 4

- $(1000)_{10}-(1001)_{10}$ **en BDC**

$1000 \xrightarrow{\text{BCD}} 0001\ 0000\ 0000\ 0000$

$- \qquad\qquad -$

$1001 \xrightarrow{\text{BCD}} 0001\ 0000\ 0000\ 0001$

$=9999 \qquad =1111\ 1111\ 1111\ 1111$

$-\ 0110-0110-0110-0110$

$=\underline{1001}\ \underline{1001}\ \underline{1001}\ \underline{1001}$

9 9 9 9

- $(903)_{10}-(878)_{10}$ **en BDC**

$903 \xrightarrow{\text{BCD}} 1001\ 0000\ 0011$

$- \qquad\qquad -$

$878 \xrightarrow{\text{BCD}} 1000\ 0111\ 1000$

$= 025 \qquad = 0000\ 1000\ 1011$

$-\ 0110-\ 0110$

$=\underline{0000}\ \underline{0010}\ \underline{0101}$

0 2 5

Exercice N°7 :

Les résultants des opérations suivantes en code BCD+3 :

- **$(1000)_{10}+(1001)_{10}$ en BCD+3**

$$
\begin{array}{lcl}
1000 & \xrightarrow{BCD+3} & 0001\ 0000\ 0000\ 0000 \\
+ & & + \\
\underline{1001} & \xrightarrow{BCD+3} & \underline{0001\ 0000\ 0000\ 0001} \\
= 2001 & & = 1000\ 0110\ 0110\ 0111 \\
 & & \underline{-\ 0011 - 0011 - 0011 - 0011} \\
 & & = \underline{0101}\ \underline{0011}\ \underline{0011}\ \underline{0100} \\
 & & \quad\ \ 2 \qquad\ 0 \qquad\ 0 \qquad\ 1
\end{array}
$$

- **$(66)_{10}+(57)_{10}$ en BCD+3**

$$
\begin{array}{lcl}
66 & \xrightarrow{BCD+3} & 1001\ 1001 \\
+ & & + \\
\underline{57} & \xrightarrow{BCD+3} & \underline{1000\ 1010} \\
=123 & & =0001\ 0010\ 0011 \\
 & & \underline{+0011+0011+0011} \\
 & & = \underline{0100}\ \underline{0101}\ \underline{0110} \\
 & & \quad\ \ 1 \qquad\ 2 \qquad\ 3
\end{array}
$$

- **$(1527)_{10}+(4543)_{10}$ en BCD+3**

$$
\begin{array}{lcl}
1527 & \xrightarrow{BCD+3} & 0100\ 1000\ 0101\ 1010 \\
+ & & + \\
\underline{4543} & \xrightarrow{BCD+3} & \underline{0111\ 1000\ 0111\ 0110} \\
=6070 & & =1100\ 0000\ 1101\ 0000 \\
 & & \underline{-0011+0011-0011+0011} \\
 & & =\underline{1001}\ \underline{0011}\ \underline{1010}\ \underline{0011} \\
 & & \quad\ \ 6 \qquad\ 0 \qquad\ 7 \qquad\ 0
\end{array}
$$

- **$(371)_{10}+(983)_{10}$ en BCD+3**

$$
\begin{array}{lcl}
371 & \xrightarrow{BCD+3} & 0110\ 1010\ 0100 \\
+ & & + \\
\underline{983} & \xrightarrow{BCD+3} & \underline{1100\ 1011\ 0110} \\
=1354 & & = 0001\ 0011\ 0101\ 1010 \\
 & & \underline{+0011+0011+0011-0011} \\
 & & =\underline{0100}\ \underline{0110}\ \underline{1000}\ \underline{0111} \\
 & & \quad\ \ 1 \qquad\ 3 \qquad\ 5 \qquad\ 4
\end{array}
$$

Exercice N°8 :

1. La représentation en binaire pur de chacun des nombres BCD suivants :

a. $(100001100111)_{BCD}=(867)_{10}$

$=(1101100011)_2$

b. $(001010011000)_{BCD}=(298)_{10}$

$=(100101010)_2$

2. Le code binaire pur, puis le code BCD de $(2024)_{10}$

$(2024)_{10}=(11111101000)_2$

$=(0010000000100100)_{BCD}$

le coût de représentation de nombre $(2024)_{10}$ dans le code BCD (16 bits) est plus grand que le coût de représentation de même nombre dans le code binaire pur (11 bits)⇒16 bits (BCD)> 11 bits (binaire pur).

3. le nombre de bits qui faut-il utilisé pour représenter un nombre décimal de 8 chiffres en BCD est : 4*8 bits=32 bits.

4. La valeur 67 puis 14 en code excess 3. Et le résultat de leur addition :

➦ **$(67)_{10}+(14)_{10}$ en BDC+3**

$$\begin{array}{lcr} 67 & \xrightarrow{BCD+3} & 1001\ 1010 \\ + & & + \\ \underline{14} & \xrightarrow{BCD+3} & \underline{0100\ 0111} \\ =81 & & =1110\ 0001 \\ & & \underline{-0011+0011} \\ & & = \underset{8}{\underline{1011}}\ \underset{1}{\underline{0100}} \end{array}$$

Exercice N°9:

➦ détermination du signe et de la valeur décimale des nombres suivants :

$X=(11011011)_{Cà1} \xrightarrow{C\ à\ 1} (10100100)_{SVA} \xrightarrow{Binaire} -(100100)_2=(-36)_{10}$

$Y=(10111100)_{Cà2} \xrightarrow{C\ à\ 2} (11000100)_{SVA} \xrightarrow{Binaire} -(1000100)_2=(-68)_{10}$

$Z=(01010011)_{Cà1} \xrightarrow{C\ à\ 1} (01010011)_{SVA} \xrightarrow{Binaire} +(1010011)_2=(+83)_{10}$

$T=(01000111)_{Cà2} \xrightarrow{C\ à\ 2} (01000111)_{SVA} \xrightarrow{Binaire} +(1000111)_2=(+71)_{10}$

Exercice N°10:

On a les deux nombres A et B tel que : A= 25 et B= 38.

1- **Représentation sur six bits en Cà1 et Cà2 des nombres suivants : A, -A, B et –B**

$$A=(+25)_{10}=+(11001)_2 \xrightarrow{\text{SVA sur 8 Bits}} 0|011001$$
$$\xrightarrow{\text{C à 1}} 0|011001$$
$$\xrightarrow{\text{C à 2}} 0|011001$$

$$-A=(-25)_{10}=-(11001)_2 \xrightarrow{\text{SVA sur 8 Bits}} 1|011001$$
$$\xrightarrow{\text{C à 1}} 1|100110$$
$$\xrightarrow{\text{C à 2}} 1|100111$$

$$B=(+38)_{10}=+(100110)_2 \xrightarrow{\text{SVA sur 8 Bits}} 0|100110$$
$$\xrightarrow{\text{C à 1}} 0|100110$$
$$\xrightarrow{\text{C à 2}} 0|100110$$

$$-B=(-38)_{10}=-(100110)_2 \xrightarrow{\text{SVA sur 8 Bits}} 1|100110$$
$$\xrightarrow{\text{C à 1}} 1|011001$$
$$\xrightarrow{\text{C à 2}} 1|011010$$

Cette codification est impossible sur Cinque bits car il manque le bit de signe.

2- **Réalisation des opérations suivantes en Cà1 et Cà2 : A-B, B-A, -A-B**

➦ **$(25)_{10}$ + $(-38)_{10}$ en complément à 1 sur 8 bits**

$$\begin{array}{rlll}
 & +25 \xrightarrow{\text{SVA sur 8 Bits}} 0|0011001 & \longrightarrow & 0|0011001 \\
+ & & & + \\
 & \underline{-38} \xrightarrow{\text{SVA sur 8 Bits}} 1|0100110 & \xrightarrow{\text{C à 1}} & \underline{1|1011001} \\
= & -13 & & = 1|1110010 \\
 & & & \quad \downarrow \text{C à 1} \\
 & & & \quad 0001101 \\
 & & & (-\ \ 13\ )_{10}
\end{array}$$

➦ **$(25)_{10}$ + $(-38)_{10}$ en complément à 2 sur 8 bits**

$$\begin{array}{rlll}
 & +25 \xrightarrow{\text{SVA sur 8 Bits}} 0|0011001 & \longrightarrow & 0|0011001 \\
+ & & & + \\
 & \underline{-38} \xrightarrow{\text{SVA sur 8 Bits}} 1|0100110 & \xrightarrow{\text{C à 2}} & \underline{1|1011010} \\
= & -13 & & =1|1110011 \\
 & & & \quad \downarrow \text{C à 2} \\
 & & & \quad 0001101 \\
 & & & (-\ 13\ )_{10}
\end{array}$$

➦ **$(38)_{10}$ + $(-25)_{10}$ en complément à 1 sur 8 bits**

$$
\begin{array}{lll}
+38 \xrightarrow{\text{SVA sur 8 Bits}} 0|0100110 & \longrightarrow & 0|0100110 \\
+ & & + \\
-25 \xrightarrow{\text{SVA sur 8 Bits}} 1|0011001 & \xrightarrow{\text{C à 1}} & \underline{1|1100110} \\
= +13 & & = \boxed{1}\ 0|0001100 \\
 & & \underline{+ \longrightarrow 1} \\
 & & = 0|0001101 \\
 & & (+\ \ 13\ )_{10}
\end{array}
$$

➦ **$(38)_{10}$ + $(-25)_{10}$ en complément à 2 sur 8 bits**

$$
\begin{array}{lll}
+38 \xrightarrow{\text{SVA sur 8 Bits}} 0|0100110 & \longrightarrow & 0|0100110 \\
+ & & + \\
-25 \xrightarrow{\text{SVA sur 8 Bits}} 1|0011001 & \xrightarrow{\text{C à 2}} & \underline{1|1100111} \\
= \ \ -63 & & = \boxed{\cancel{1}}\ 0|0001101 \\
 & & (+\ \ 13\ )_{10}
\end{array}
$$

➦ **$(-25)_{10}$ + $(-38)_{10}$ en complément à 1 sur 8 bits**

$$
\begin{array}{lll}
-25 \xrightarrow{\text{SVA sur 8 Bits}} 1|0011001 & \xrightarrow{\text{C à 1}} & 1|1100110 \\
+ & & + \\
-38 \xrightarrow{\text{SVA sur 8 Bits}} 1|0100110 & \xrightarrow{\text{C à 1}} & \underline{1|1011001} \\
= \ \ -63 & & = \boxed{1}\ 1|0111111 \\
 & & \underline{+ \longrightarrow 1} \\
 & & = 1|1000000 \\
 & & \quad \downarrow \text{C à 1} \\
 & & \quad 0111111 \\
 & & (-\ \ 63\ )_{10}
\end{array}
$$

➦ **$(-25)_{10}$ + $(-38)_{10}$ en complément à 2 sur 8 bits**

$$
\begin{array}{lll}
-25 \xrightarrow{\text{SVA sur 8 Bits}} 1|0011001 & \xrightarrow{\text{C à 2}} & 1|1100111 \\
+ & & + \\
-38 \xrightarrow{\text{SVA sur 8 Bits}} 1|0100110 & \xrightarrow{\text{C à 2}} & \underline{1|1011010} \\
= -63 & & = \boxed{\cancel{1}}\ 1|1000001 \\
 & & \quad \downarrow \text{C à 2} \\
 & & \quad 0111111 \\
 & & (-\ \ 63\ )_{10}
\end{array}
$$

Exercice N°11 :

➦ **Les compléments des nombres suivants :**

1ère méthode :

➦ $(00101011)_2 \xrightarrow[\text{Sur 8 bits}]{\text{C à 1}} (11010100)_2$

➦ $(011010)_2 \xrightarrow[\text{Sur 6 bits}]{\text{C à 1}} (100101)_2$

$$\begin{array}{r} (100101)_2 \\ + \quad\quad 1 \\ \hline \end{array}$$

$\xrightarrow[\text{Sur 6 bits}]{\text{C à 2}} = (100111)_2$

➦ $(0001776010)_8 \xrightarrow[\text{Sur 10 bits}]{\text{C à 7}} (7776001767)_8$

$$\begin{array}{r} (7776001767)_8 \\ + \quad\quad\quad 1 \\ \hline \end{array}$$

$\xrightarrow[\text{Sur 10 bits}]{\text{C à 8}} =(7776001770)_8$

➦ $(002990700)_{10} \xrightarrow[\text{Sur 9 bits}]{\text{C à 9}} (997009299)_{10}$

$$\begin{array}{r} (997009299)_{10} \\ + \quad\quad\quad 1 \\ \hline \end{array}$$

$\xrightarrow[\text{Sur 9 bits}]{\text{C à 10}} =(997009300)_{10}$

➦ $(A02CBF0)_{16} \xrightarrow[\text{Sur 7 bits}]{\text{C à 15}} (5FD340F)_{16}$

$$\begin{array}{r} (5FD340F)_{16} \\ + \quad\quad\quad 1 \\ \hline \end{array}$$

$\xrightarrow[\text{Sur 7 bits}]{\text{C à 16}} =(5FD3410)_{16}$

2ème méthode :

➦ $(\overbrace{00101011}^{\text{C à 1}})_2 \xrightarrow[\text{Sur 8 bits}]{\text{C à 1}} (11010100)_2$

➦ $(\overbrace{0110}^{\text{C à 1}}\overbrace{1}^{\text{C à 2}}0)_2 \xrightarrow[\text{Sur 6 bits}]{\text{C à 2}} (100110)_2$

➦ $(\overbrace{00017760}^{\text{C à 17}}\overbrace{1}^{\text{C à 8}}\ 0)_8 \xrightarrow[\text{Sur 10 bits}]{\text{C à 8}} (7776001770)_8$

➦ $(\overbrace{002990}^{\text{C à 9}}\overbrace{7}^{\text{C à 10}}00)_{10} \xrightarrow[\text{Sur 9 bits}]{\text{C à 10}} (997009300)_{10}$

➦ $(\overbrace{A02CB}^{\text{C à 15}}\overbrace{F}^{\text{C à 16}}0)_{16} \xrightarrow[\text{Sur 7 bits}]{\text{C à 16}} (5FD3410)_{16}$

Exercice N°12:

a- La représentation en virgule flottante sous format IEEE754 simple précision des nombres suivants :

- $A=(-128,25)_{10} =(-10000000,01)_2$

  $=(-1)^1*1,000000001*2^{+7}$

S=1

$M=(0000000010………0)_2$

$E-127=+7 \Rightarrow E=127+7=(134)_{10}$

$=(10000110)_2$

$A=(-128,25)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{1|10000110|00000000100000000000000}_{32 \text{ bits}}$

- $B=(+18,125)_{10} =(+10010,001)_2$

  $=(-1)^1*1,0010001*2^{+4}$

S=0

$M=(00100010………0)_2$

$E-127=+4 \Rightarrow E=127+4=(131)_{10}$

$=(10000011)_2$

$B=(+18,125)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{0|10000011|00100010000000000000000}_{32 \text{ bits}}$

- $C=(-0.375)_{10} =(-0,011)_2$

  $=(-1)^1 x1,1x2^{-2}$

S=1

$M=(100………0)_2$

$E-127=-2 \Rightarrow E=127-2=(125)_{10}$

$=(01111101)_2$

$C=(-0.375)_{10} \xrightarrow[\text{Précision}]{\text{VF Simple}} \underbrace{1|01111101|10000000000000000000000}_{32 \text{ bits}}$

b- La représentation en virgule flottante sous format IEEE754 double précision des nombres suivants :

- $D=(1000)_2$

  $=(-1)^0*1,0*2^{+3}$

  S=0

  $M=(00000……..0)_2$

  $E-1023=3 \Rightarrow E=1023+3=(1026)_{10}$

  $=(10000000010)_2$

$D=(1000)_2 \xrightarrow[\text{Précision}]{\text{VF Double}} \underbrace{0|10000000010|000000000000000……..0}_{64 \text{ bits}}$

• $F=(-12.625)_{10}=(-1100{,}101)_2$

$=(-1)^1*1{,}100101*2^{+3}$

S=1

$M=(10010100……..0)_2$

$E-1023=3 \Rightarrow E=1023+3=(1026)_{10}$

$=(10000000010)_2$

$$F=(-12{,}625)_{10} \xrightarrow[\text{Précision}]{\text{VF Double}} \underbrace{1|10000000010|1001010000000000……..0}_{\text{64 bits}}$$

• $G=(-64)_{10}=(-1000000)_2$

$=(-1)^1*1{,}0*2^{+6}$

S=1

$M=(00000……..0)_2$

$E-1023=6 \Rightarrow E=1023+6=(1029)_{10}$

$=(10000000101)_2$

$$G=(-64)_{10} \xrightarrow[\text{Précision}]{\text{VF Double}} \underbrace{1|10000000101|000000000000000……..0}_{\text{64 bits}}$$

Exercice N°13:

a- La représentation décimale des nombres codés en virgule flottante (IEEE754 simple précision) :

• $(1011\ 1101\ 1110\ 0000\ 0000\ 0000\ 0000\ 0000)_2$

$X=(1\ |\ 01111011\ |\ 11000000000000000000000)_2$

S=1

$M=(1100……..0)_2$

$E=(01111011)_2=(123)_{10} \Rightarrow e=E-127$

$=123-127$

$=-4$

$X=(-1)^1*1{,}11*2^{-4}$

$=-1{,}11*2^{-4}$

$=(-0{,}000111)_2$

$=(-0{,}109375)_{10}$

• $(1100\ 0011\ 1101\ 1010\ 0000\ 0000\ 0000\ 0000)_2$

$Y=(1\ |\ 10000111\ |\ 10110100000000000000000)_2$

S=1

$M=(1011010……..0)_2$

$E=(10000111)_2=(135)_{10} \Rightarrow e=E-127$

$=135-127$

$=+8$

$Y=(-1)^{1}*1,101101*2^{+8}$

$=-1,101101*2^{+8}$

$=(-110110100)_2$

$=(-436)_{10}$

• $(0100\ 0000\ 1101\ 1000\ 0000\ 0000\ 0000\ 0000)_2$

$Z=(0 \mid 10000001 \mid 10110000000000000000000)_2$

S=0

$M=(101100……..0)_2$

$E=(10000001)_2 =(129)_{10} \Rightarrow e=E-127$

$=129-127$

$=+2$

$Z=(-1)^{0}*1,1011*2^{+2}$

$=+1,1011*2^{+2}$

$=(+110,11)_2$

$=(+6,75)_{10}$

b- Donner la représentation décimale des nombres codés en virgule flottante (IEEE754 double précision) :

• $(403D\ 8000\ 0000\ 0000)_{16}$

$A=(403D\ 8000\ 0000\ 0000)_{16}$

$=(0100\ 0000\ 0011\ 1011\ 1000\ 0000\ 0000\ 0000\ 0000…………………)_2$

$=(0 \mid 10000000011 \mid 1011100000000000000000000…………………)_2$

S=0

$M=(101110……..0)_2$

$E=(10000000011)_2 =(1027)_{10} \Rightarrow e=E-1023$

$=1027-1023$

$=+4$

$A=(-1)^{0}*1,10111*2^{+4}$

$=+1,10111*2^{+4}$

$=(+11011,1)_2$

$=(+27,5)_{10}$

• $(C040\ 0000\ 0000\ 0000)_{16}$

$B=(C040\ 0000\ 0000\ 0000)_{16}$

$=(1100\ 0000\ 0100\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000…………………)_2$

$=(1 \mid 10000000100 \mid 0000000000000000000000000…………………)_2$

S=1

$M=(000……..0)_2$

$E=(10000000100)_2=(1028)_{10} \Rightarrow e=E-1023$

$=1028-1023$

$=+5$

$B=(-1)^1*1,0*2^{+5}$

$=-1,0*2^{+5}$

$=(-100000)_2$

$=(-32)_{10}$

- $(3FB8\ 0000\ 0000\ 0000)_{16}$

$C=(3FB8\ 0000\ 0000\ 0000)_{16}$

$=(0011\ 1111\ 1011\ 1000\ 0000\ 0000\ 0000\ 0000\ 0000\ldots\ldots)_2$

$=(0\ |\ 01111111011\ |\ 10000000000000000000000\ldots\ldots)_2$

S=0

$M=(100000\ldots\ldots0)_2$

$E=(01111111011)_2=(1019)_{10} \Rightarrow e=E-1023$

$=1019-1023$

$=-4$

$A=(-1)^0*1,1*2^{-4}$

$=+1,1*2^{-4}$

$=(+0,00011)_2$

$=(+0,09375)_{10}$

Exercice N°14:

Soit le format M à 15 bits en virgule fixe : $A_8|A_7A_6A_5A_4A_3A_2A_1A_0|A_{-1}A_{-2}A_{-3}A_{-4}A_{-5}A_{-6}$. Tel que A_8 constitue le bit de signe, et $A_7A_6A_5A_4A_3A_2A_1A_0$ la partie entière et $A_{-1}A_{-2}A_{-3}A_{-4}A_{-5}A_{-6}$ est la partie fractionnaire.

a. Représentation des équivalents binaires des nombres $N_1=-26$ et $N_2=+34,0625$ selon le format M de virgule fixe

$(26)_{10}=(11010)_2$

$(34,0625)_{10}=(100010,0001)_2$

$-(26)_{10} \xrightarrow[\text{Format M}]{\text{Virgule Fixe}} 1|\ 00011010\ |\ 000000$

$+(34.0625)_{10} \xrightarrow[\text{Format M}]{\text{Virgule Fixe}} 0|\ 00100010\ |\ 000100$

b. L'addition des opérandes N_1 et N_2 et la représentation du résultat selon le format M de virgule fixe.

$(-26)_{10}+(34,0625)_{10}=(+8,0625)_{10}$

$=(1000,0001)_2$

$(-26)_{10}+(34,0625)_{10}=(+8,0625)_{10} \xrightarrow[\text{Format M}]{\text{Virgule Fixe}} 0|\ 00001000\ |\ 000100$

Exercice N°15:

1. Tout d'abord, nous convertissons le nombre en binaire afin de pouvoir le représenter dans la machine.

On a : $-(16.375)_{10}= -(10000,011)_2$.

Ensuite, on représente le nombre selon le format indiqué.

$-(16.375)_{10} \xrightarrow{\text{Virgule Fixe}}$ 1|00010000|0110000

2. le plus petit nombre positif est représenté comme suit : 0|00000000|0000001 ce qui donne la valeur $N_{min}=(2^{-7})_{10}$

3. le plus grand nombre positif est représenté comme suit : 0|11111111|1111111, Pour donner l'équivalent en décimale, calculons la partie entière max (PE_{max}) et la partie fractionnaire max (PF_{max}).

$$PE_{max}=2^0+2^1+\ldots+2^7$$
$$= 2^8-1$$
$$PF_{max} =2^{-1}+2^{-2}+\ldots+2^{-7}$$
$$=1- 2^{-7}$$
$$N_{max}=PE_{max}+PF_{max}$$
$$=2^8-1 +1- 2^{-7}$$
$$= 2^8- 2^{-7}$$
$$= (255,9921875)_{10}$$

Exercice N°16:

1. Conversion des nombres décimaux suivants en code ASCII :

$(1)_{10}=(00110001)_{\text{ASCII-binaire}}=(31)_{\text{ASCII-hexadécimal}}$

$(102)_{10}=(001100010011000000110010)_{\text{ASCII-binaire}}=(31\ 30\ 32)_{\text{ASCII-hexadécimal}}$

$(29)_{10}=(0011001000111001)_{\text{ASCII-binaire}}=(32\ 39)_{\text{ASCII-hexadécimal}}$

$(58)_{10}=(001101010011000)_{\text{ASCII-binaire}}=(35\ 38)_{\text{ASCII-hexadécimal}}$

$(706)_{10}=(001101110011000000110110)_{\text{ASCII-binaire}}=(37\ 30\ 36)_{\text{ASCII-hexadécimal}}$

2. Conversion des nombres décimaux suivants en code ASCII :

$(A)_{\text{caractère}}=(01000001)_{\text{ASCII-binaire}} =(41)_{\text{ASCII-hexadécimal}}$

$(a)_{\text{caractère}}=(01100001)_{\text{ASCII-binaire}} =(61)_{\text{ASCII-hexadécimal}}$

$(d)_{\text{caractère}}=(01100100)_{\text{ASCII-binaire}} =(64)_{\text{ASCII-hexadécimal}}$

$(W)_{\text{caractère}}=(01010111)_{\text{ASCII-binaire}} =(57)_{\text{ASCII-hexadécimal}}$

$(SM)_{\text{caractère}}=(0101001101001101)_{\text{ASCII-binaire}} =(53\ 4D)_{\text{ASCII-hexadécimal}}$

3. Détermination chaque caractère ASCII :

$(00111111)_{ASCII\text{- }binaire}=(?)_{caractère}$

$(00100000)_{ASCII\text{- }binaire}$=(la commande espace=SP)$_{caractère}$

$(01000000)_{ASCII\text{- }binaire}=(@)_{caractère}$

$(01011000)_{ASCII\text{- }binaire}=(X)_{caractère}$

$(00111110)_{ASCII\text{- }binaire}=(>)_{caractère}$

Exercice N°17:

• recherche du texte représenté en ASCII-binaire par la suite de bits suivante :

0100110101100001011101000110100001110011001000000110010101110100001000000100100101101110011001100110111101110010011011010110000101110100011010010111000101110101 01100101

01001101	01100001	01110100	01101000	01110011	00100000	01100101	01110100	00100000
M	a	t	h	s	espace	e	t	espace
01001001	01101110	01100110	01101111	01110010	01101101	01100001	01110100	01101001
I	n	f	o	r	m	a	t	i
01110001	01110101	01100101						
q	u	e						

Le texte est : Maths et Informatique

• Codage de votre nom en hexadécimal avec les codages ASCII et EBCDIC.

(CHERIEF Yasser)$_{caractère}$=(**43 48 45 52 49 45 46 20 59 61 73 73 65 72**)$_{ASCII\text{-}hexadécimal}$

(CHERIEF Yasser)$_{caractère}$=(**C3 C8 C5 D9 C9 C5 C6 40 E8 81 A2 A2 85 99**)$_{EBCDIC\text{-}hexadécimal}$

• Le nombre de caractères que le codage UTF-8 permet-il de coder

- ✓ Pour le codage UTF-8 sur 1 octet (8bits) on a : 2^8=256 possibilité
- ✓ Pour le codage UTF-16 sur 2 octet (16bits) on a : 2^{16}=65536 possibilité
- ✓ Pour le codage UTF-32 sur 3 octet (32bits) on a : 2^{32}=4294967296 possibilité
- ✓ Pour le codage UTF-64 sur 4 octet (64bits) on a : 2^{64}=18446744073709551616 possibilité

3. Solution de la série d'exercices N°3 (Algèbre de Boole)

Exercice N°1:

Démonstration algébrique des relations suivantes :

1/ $XY + \overline{X}Z = (\overline{X} + Y)(X + Z)$

$$
\begin{aligned}
(\overline{X} + Y)(X + Z) &= \overline{X}X + \overline{X}Z + XY + YZ \\
&= XY + \overline{X}Z + (X + \overline{X})YZ \\
&= XY + \overline{X}Z + XYZ + \overline{X}YZ \\
&= XY(1 + Z) + \overline{X}Z(1 + Y) \\
&= XY + \overline{X}Z
\end{aligned}
$$

$\Rightarrow$La relation est vraie

2/ $XY + \overline{X}Z + YZ = XY + \overline{X}Z$

$$
\begin{aligned}
XY + \overline{X}Z + YZ &= XY + \overline{X}Z + (X + \overline{X})YZ \\
&= XY + \overline{X}Z + XYZ + \overline{X}YZ \\
&= XY(1 + Z) + \overline{X}Z(1 + Y) \\
&= XY + \overline{X}Z
\end{aligned}
$$

$\Rightarrow$La relation est vraie

3/ $(X + Y)(\overline{X} + Z)(Y + Z) = (X + Y)(\overline{X} + Z)$

$$
\begin{aligned}
(X + Y)(\overline{X} + Z)(Y + Z) &= (X\overline{X} + XZ + \overline{X}Y + YZ)(Y + Z) \\
&= XYZ + \overline{X}YY + YYZ + XZZ + \overline{X}YZ + YZZ \\
&= (X + \overline{X})YZ + \overline{X}Y + YZ + XZ \\
&= \overline{X}Y + YZ + XZ + X\overline{X} \\
&= \overline{X}(X + Y) + Z(X + Y) \\
&= (X + Y)(\overline{X} + Z)
\end{aligned}
$$

$\Rightarrow$La relation est vraie

4/ $XY + X\overline{Y}Z = XY + XZ$

$$
\begin{aligned}
XY + X\overline{Y}Z &= X(Y + \overline{Y}Z) \\
&= X((Y + \overline{Y})(Y + Z)) \\
&= X(Y + Z) \\
&= XY + XZ
\end{aligned}
$$

$\Rightarrow$La relation est vraie

Exercice N°2:

Représentation des fonctions sous FND et FNC (la première et la deuxième forme canonique) :

$$
\begin{aligned}
1/\quad X &= \bar{a}b\bar{c} + abd + bcd \\
&= \bar{a}b\bar{c}\left(d+\bar{d}\right) + a\left(c+\bar{c}\right)bd + \left(a+\bar{a}\right)bcd \\
&= \bar{a}b\bar{c}d + \bar{a}b\bar{c}\bar{d} + abcd + ab\bar{c}d + abcd + \bar{a}bcd \\
&= \bar{a}b\bar{c}d + \bar{a}b\bar{c}\bar{d} + ab\bar{c}d + abcd + \bar{a}bcd \\
&= \bar{a}b\bar{c}\bar{d} + \bar{a}b\bar{c}d + \bar{a}bcd + ab\bar{c}d + abcd \\
&= 0100 + 0101 + 0111 + 1101 + 1111 \\
&= \sum\left(4,5,7,D,F\right) \\
&= \prod\left(0,1,2,3,6,8,9,A,B,C,E\right)
\end{aligned}
$$

➲ la première forme canonique

$$
\begin{aligned}
X_{FND}\left(a,b,c,d\right) &= \sum\left(4,5,7,D,F\right) \\
&= \bar{a}b\bar{c}\bar{d} + \bar{a}b\bar{c}d + \bar{a}bcd + ab\bar{c}d + abcd
\end{aligned}
$$

➲ la deuxième forme canonique

$$
\begin{aligned}
X_{FNC}\left(a,b,c,d\right) &= \prod\left(0,1,2,3,6,8,9,A,B,C,E\right) \\
&= \left(a+b+c+d\right)\left(a+b+c+\bar{d}\right)\left(a+b+\bar{c}+d\right)\left(a+b+\bar{c}+\bar{d}\right)\left(a+\bar{b}+\bar{c}+d\right)\left(\bar{a}+b+c+d\right) \\
&\quad \left(\bar{a}+b+c+\bar{d}\right)\left(\bar{a}+b+\bar{c}+d\right)\left(\bar{a}+b+\bar{c}+\bar{d}\right)\left(\bar{a}+\bar{b}+c+d\right)\left(\bar{a}+\bar{b}+\bar{c}+d\right)
\end{aligned}
$$

$$
\begin{aligned}
2/\quad Y &= a\left(b+c\right)\left(\bar{c}+\bar{d}\right) \\
&= \left(ab+ac\right)\left(\bar{c}+\bar{d}\right) \\
&= ab\bar{c} + ab\bar{d} + a\cancel{c}\bar{c} + ac\bar{d} \\
&= ab\bar{c}\left(d+\bar{d}\right) + ab\left(c+\bar{c}\right)\bar{d} + a\left(b+\bar{b}\right)c\bar{d} \\
&= ab\bar{c}d + ab\bar{c}\bar{d} + abc\bar{d} + ab\cancel{\bar{c}}\bar{d} + ab\cancel{c}\bar{d} + a\bar{b}c\bar{d} \\
&= a\bar{b}c\bar{d} + ab\bar{c}\bar{d} + ab\bar{c}d + abc\bar{d} \\
&= 1010 + 1100 + 1101 + 1110 \\
&= \sum\left(A,C,D,E\right) \\
&= \prod\left(0,1,2,3,4,5,6,7,8,9,B,F\right)
\end{aligned}
$$

➲ la première forme canonique

$$
\begin{aligned}
Y_{FND}\left(a,b,c,d\right) &= \sum\left(A,C,D,E\right) \\
&= a\bar{b}c\bar{d} + ab\bar{c}\bar{d} + ab\bar{c}d + abc\bar{d}
\end{aligned}
$$

➲ la deuxième forme canonique

$$
\begin{aligned}
Y_{FNC}\left(a,b,c,d\right) &== \prod\left(0,1,2,3,4,5,6,7,8,9,B,F\right) \\
&= \left(a+b+c+d\right)\left(a+b+c+\bar{d}\right)\left(a+b+\bar{c}+d\right)\left(a+b+\bar{c}+\bar{d}\right)\left(a+\bar{b}+c+d\right)\left(a+\bar{b}+c+\bar{d}\right)\left(a+\bar{b}+\bar{c}+d\right) \\
&\quad \left(a+\bar{b}+\bar{c}+\bar{d}\right)\left(\bar{a}+b+c+d\right)\left(\bar{a}+b+c+\bar{d}\right)\left(\bar{a}+b+\bar{c}+\bar{d}\right)\left(\bar{a}+\bar{b}+\bar{c}+\bar{d}\right)
\end{aligned}
$$

3/ $\quad Z=(a+d)(\bar{a}+c+d)+\bar{a}\bar{b}$

⇒ **Première méthode (à partir de l'expression logique)**

$$\begin{aligned}
Z&=(a+d)(\bar{a}+c+d)+\bar{a}\bar{b}\\
&=\cancel{a}\bar{a}+ac+ad+\bar{a}d+cd+dd+\bar{a}\bar{b}\\
&=ac+ad+\bar{a}d+cd+d+\bar{a}\bar{b}\\
&=a(b+\bar{b})c(d+\bar{d})+a(b+\bar{b})(c+\bar{c})d+\bar{a}(b+\bar{b})(c+\bar{c})d+(a+\bar{a})(b+\bar{b})cd+(a+\bar{a})(b+\bar{b})(c+\bar{c})d+\bar{a}\bar{b}(c+\bar{c})(d+\bar{d})\\
&=abcd+abc\bar{d}+a\bar{b}cd+a\bar{b}c\bar{d}+ab\cancel{c}d+a\cancel{b}cd+ab\bar{c}d+a\bar{b}\bar{c}d+\bar{a}bcd+\bar{a}b\bar{c}d+\bar{a}\bar{b}cd+\bar{a}\bar{b}\bar{c}d+\cancel{a}bcd+\bar{a}bc\bar{d}+a\cancel{b}cd\\
&\quad+\bar{a}\cancel{\bar{b}}cd+ab\cancel{c}d+\bar{a}\cancel{b}cd+a\cancel{\bar{b}}cd+\bar{a}\cancel{\bar{b}}cd+a\cancel{b}\bar{c}d+\bar{a}\cancel{b}\bar{c}d+a\cancel{\bar{b}}\bar{c}d+\cancel{a}\bar{b}\bar{c}d+\bar{a}\cancel{\bar{b}}cd+\bar{a}\bar{b}c\bar{d}+\cancel{a}\bar{b}\bar{c}d+\bar{a}\bar{b}\bar{c}\bar{d}\\
&=abcd+abc\bar{d}+a\bar{b}cd+a\bar{b}c\bar{d}+ab\bar{c}d+a\bar{b}\bar{c}d+\bar{a}bcd+\bar{a}b\bar{c}d+\bar{a}\bar{b}cd+\bar{a}\bar{b}\bar{c}d+\bar{a}bc\bar{d}+\bar{a}\bar{b}c\bar{d}+\bar{a}\bar{b}\bar{c}\bar{d}\\
&=\bar{a}\bar{b}\bar{c}\bar{d}+\bar{a}\bar{b}\bar{c}d+\bar{a}\bar{b}c\bar{d}+\bar{a}\bar{b}cd+\bar{a}b\bar{c}d+\bar{a}bc\bar{d}+\bar{a}bcd+a\bar{b}\bar{c}d+a\bar{b}c\bar{d}+a\bar{b}cd+ab\bar{c}d+abc\bar{d}+abcd\\
&=0000+0001+0010+0011+0101+0110+0111+1001+1010+1011+1101+1110+1111\\
&=\sum(0,1,2,3,5,6,7,9,A,B,E,F)\\
&=\prod(4,8,C,D)
\end{aligned}$$

➲ La première forme canonique

$$\begin{aligned}
Z_{FND}(a,b,c,d)&=\sum(0,1,2,3,5,6,7,9,A,B,D,E,F)\\
&=\bar{a}\bar{b}\bar{c}\bar{d}+\bar{a}\bar{b}\bar{c}d+\bar{a}\bar{b}c\bar{d}+\bar{a}\bar{b}cd+\bar{a}b\bar{c}d+\bar{a}bc\bar{d}+\bar{a}bcd+a\bar{b}\bar{c}d+a\bar{b}c\bar{d}+a\bar{b}cd+ab\bar{c}d+abc\bar{d}+abcd
\end{aligned}$$

➲ La deuxième forme canonique

$$\begin{aligned}
Z_{FNC}(a,b,c,d)&=\prod(4,8,C)\\
&=(a+\bar{b}+c+d)(\bar{a}+b+c+d)(\bar{a}+\bar{b}+c+d)
\end{aligned}$$

⇒ **Deuxième méthode (à partir de la table de vérité)**

a	b	c	d		$\bar{a}$	$\bar{b}$	$\bar{a}\bar{b}$	$(a+d)$	$(\bar{a}+c+d)$	$Z=(a+d)(\bar{a}+c+d)+\bar{a}\bar{b}$
0	0	0	0		1	1	1	0	1	1 FND
0	0	0	1		1	1	1	1	1	1 FND
0	0	1	0		1	1	1	0	1	1 FND
0	0	1	1		1	1	1	1	1	1 FND
0	1	0	0		1	0	0	0	1	0 FNC
0	1	0	1		1	0	0	1	1	1 FND
0	1	1	0		1	0	0	1	1	1 FND
0	1	1	1		1	0	0	1	1	1 FND
1	0	0	0		0	1	0	1	0	0 FNC
1	0	0	1		0	1	0	1	1	1 FND
1	0	1	0		0	1	0	1	1	1 FND
1	0	1	1		0	1	0	1	1	1 FND
1	1	0	0		0	0	0	1	0	0 FNC
1	1	0	1		0	0	0	1	1	1 FND
1	1	1	0		0	0	0	1	1	1 FND
1	1	1	1		0	0	0	1	1	1 FND

➲ la première forme canonique

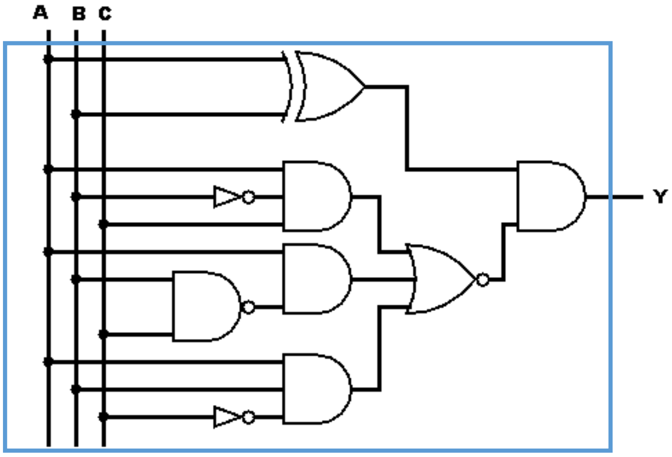
$$Z_{FND}(a,b,c,d)=\bar{a}b\bar{c}\bar{d}+\bar{a}\bar{b}\bar{c}d+\bar{a}\bar{b}c\bar{d}+\bar{a}\bar{b}cd+\bar{a}b\bar{c}d+\bar{a}bc\bar{d}+\bar{a}bcd+a\bar{b}\bar{c}d+a\bar{b}c\bar{d}+a\bar{b}cd+ab\bar{c}d+abc\bar{d}+abcd$$

➲ la deuxième forme canonique

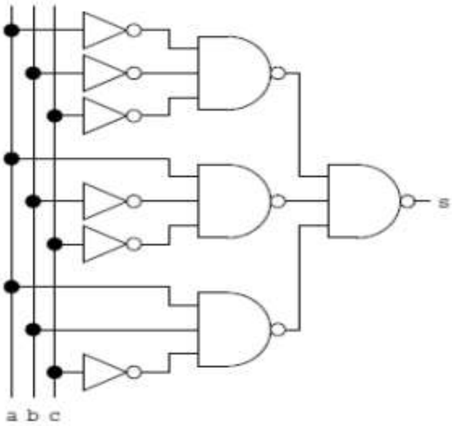
$$Z_{FNC}(a,b,c,d)=\left(a+\bar{b}+c+d\right)\left(\bar{a}+b+c+d\right)\left(\bar{a}+\bar{b}+c+d\right)$$

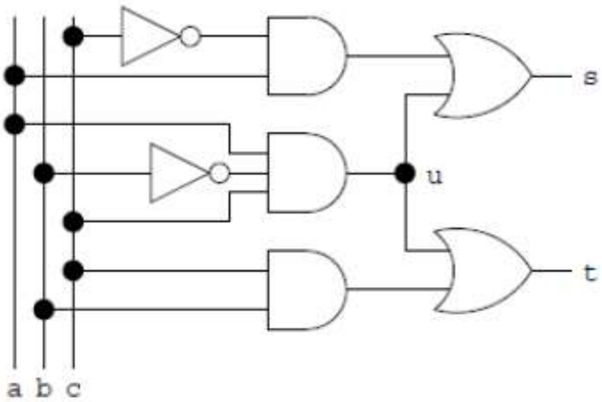
Exercice N°3:

a/ Logigramme de la fonction $Y=(A\oplus B)\left(\overline{A\bar{B}C+A\overline{BC}+AB\bar{C}}\right)$



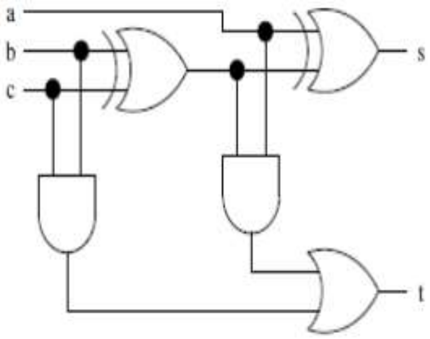
b/ Les équations de sorties :



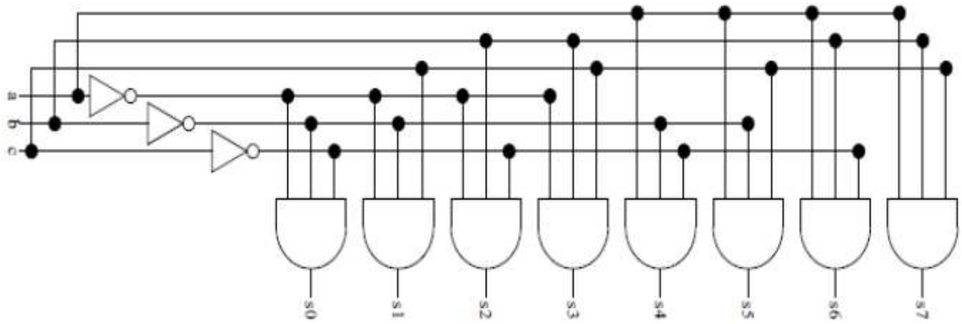


$$S(a,b,c)=\overline{\overline{\bar{a}\bar{b}\bar{c}}\;\overline{a\bar{b}\bar{c}}\;\overline{ab\bar{c}}}$$

$$\begin{cases} S(a,b,c)=a\bar{c}+a\bar{b}c \\ u(a,b,c)=a\bar{b}c \\ t(a,b,c)=a\bar{b}c+bc \end{cases}$$



$$\begin{cases} S(a,b,c)=a\oplus(b\oplus c) \\ t(a,b,c)=a(b\oplus c)+bc \end{cases}$$



$$\begin{cases} S_0(a,b,c)=\bar{a}\bar{b}\bar{c} \\ S_1(a,b,c)=\bar{a}\bar{b}c \\ S_2(a,b,c)=\bar{a}b\bar{c} \\ S_3(a,b,c)=\bar{a}bc \\ S_4(a,b,c)=a\bar{b}\bar{c} \\ S_5(a,b,c)=a\bar{b}c \\ S_6(a,b,c)=ab\bar{c} \\ S_7(a,b,c)=abc \end{cases}$$

Exercice N°4:

Les expressions simplifiées des tableaux de KARNAUGH suivants :

AB \ CD	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	0	1	1	0
10	0	1	1	0

$F(A,B,C,D)=B+\bar{C}D$

AB \ CD	00	01	11	10
00	1	0	0	1
01	0	1	1	0
11	0	1	1	0
10	1	0	0	1

$F(A,B,C,D)=BD+\bar{B}\bar{D}$

AB \ CD	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$F(A,B,C,D)=\bar{C}D+C\bar{D}$

AB \ CD	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	1	1	1	1
10	1	1	1	1

$F(A,B,C,D)=A\bar{B}+\bar{A}B+C$

AB / CD	00	01	11	10
00	0	1	1	0
01	1	1	1	1
11	1	1	1	1
10	0	1	1	0

$$F(A,B,C,D) = B + D$$

AB / CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	0	1	0
10	1	1	1	1

$$F(A,B,C,D) = AB + C\overline{D}$$

AB / CD	00	01	11	10
00	1	0	1	1
01	1	0	1	1
11	0	0	1	1
10	1	1	1	1

$$F(A,B,C,D) = A + \overline{B}\overline{C} + C\overline{D}$$

AB / CD	00	01	11	10
00	1	1	1	1
01	0	1	1	0
11	0	1	1	0
10	1	1	1	1

$$F(A,B,C,D) = B + \overline{D}$$

AB / CD	00	01	11	10
00	1	0	0	1
01	1	1	1	1
11	1	1	0	0
10	0	0	0	0

$$F(A,B,C,D) = \overline{A}D + \overline{C}D + \overline{B}\overline{C}$$

AB / CD	00	01	11	10
00	0	0	1	0
01	1	0	1	1
11	1	1	1	1
10	0	0	1	0

$$F(A,B,C,D) = AB + CD + \overline{A}D$$

Exercice N°5:

$$F(A,B,C,D) = A\overline{C} + ABD + AB\overline{D} + \overline{A}C + \overline{AB}$$

➦ Table de vérité

A	B	C	D	F
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

➦ Simplification par la méthode de karnaugh

AB / CD	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

$$F(A,B,C,D) = 1$$

➦ Simplification algébrique

$$\begin{aligned}
F(A,B,C,D) &= A\overline{C} + ABD + AB\overline{D} + \overline{A}C + \overline{AB} \\
&= A\overline{C} + AB(D+\overline{D}) + \overline{A}C + \overline{A} + \overline{B} \\
&= A\overline{C} + (AB + \overline{B}) + \overline{A}C + \overline{A} \\
&= A\overline{C} + ((A+\overline{B})(B+\overline{B})) + \overline{A}C + \overline{A} \\
&= A\overline{C} + \overline{A} + A + \overline{B} + \overline{A}C \\
&= A\overline{C} + 1 + \overline{B} + \overline{A}C \\
&= 1
\end{aligned}$$

Exercice N°6:

$$F(A,B,C,D) = ABCD + ABC\overline{C}D + A\overline{B}C + \overline{A}BCD + AB + C\overline{D}$$

➦ Table de vérité

A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

➦ La 1er et la 2ème formes canoniques de F (FND et FNC)

$$F_{FND}(A,B,C,D) = \overline{A}\,\overline{B}C\overline{D} + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}C\overline{D} + A\overline{B}CD + AB\overline{C}\;\overline{D} + AB\overline{C}D + ABC\overline{D} + ABCD$$

$$F_{FNC}(A,B,C,D) = (A+B+C+\overline{D})(A+B+\overline{C}+\overline{D})(A+\overline{B}+C+D)(A+\overline{B}+C+\overline{D})(\overline{A}+B+C+D)(\overline{A}+B+C+\overline{D})$$

➦ Simplification de F en utilisant les lois de l'algèbre de Boole

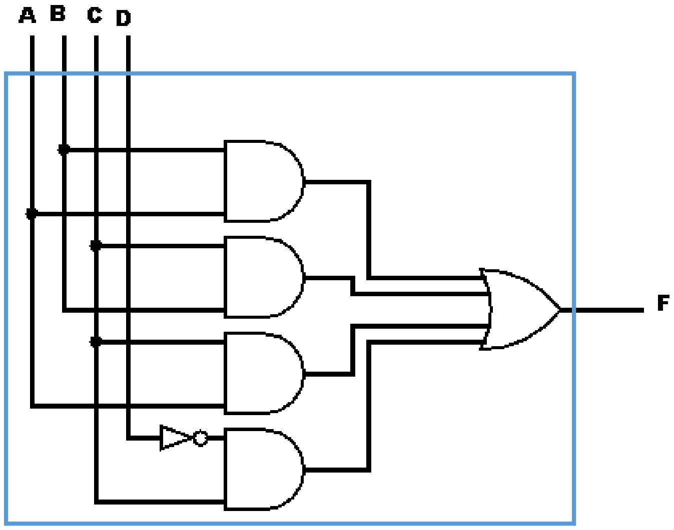
$$\begin{aligned}
F(A,B,C,D) &= ABCD + AB\overline{C}D + A\overline{B}C + \overline{A}BCD + AB + C\overline{D} \\
&= ABDC + ABCD + AB\overline{C}D + A\overline{B}C + \overline{A}BCD + AB + C\overline{D} \\
&= ABD(C+\overline{C}) + A\overline{B}C + BCD(A+\overline{A}) + AB + C\overline{D} \\
&= ABD + A\overline{B}C + BCD + AB + C\overline{D} \\
&= AB(1+D) + A\overline{B}C + BCD + C\overline{D} \\
&= A(B+\overline{B}C) + C(BD+\overline{D}) \\
&= A((B+\overline{B})(B+C)) + C((B+\overline{D})(D+\overline{D})) \\
&= A(B+C) + C(B+\overline{D}) \\
&= AB + AC + BC + C\overline{D}
\end{aligned}$$

- Simplification de F en utilisant la table de Karnaugh

AB / CD	00	01	11	10
00	0	0	1	0
01	0	0	1	0
11	0	1	1	1
10	1	1	1	1

$$F(A,B,C,D) = AB + BC + AC + C\overline{D}$$

- les logigrammes simplifiés



Exercice N°7:

$$F(A,B,C) = AB\overline{C} + A\overline{B} + \overline{A}\overline{B}\overline{C} + \overline{B}C$$

- Table de vérité

A	B	C	F
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

- Les formes canoniques

$$F_{FND}(A,B,C) = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + A\overline{B}\overline{C} + A\overline{B}C + AB\overline{C}$$

$$F_{FNC}(A,B,C) = \left(A + \overline{B} + C\right)\left(A + \overline{B} + \overline{C}\right)\left(\overline{A} + \overline{B} + \overline{C}\right)$$

➦ Simplification par la table de karnaugh

AB \ C	00	01	11	10
0	1	0	1	1
1	1	0	0	1

$$F(A,B,C)=\overline{B}+A\overline{C}$$

➦ Simplification par la méthode de quine MC-cluskey

$$F_{FND}(A,B,C)=\overline{A}\,\overline{B}\,\overline{C}+\overline{A}\,\overline{B}C+A\overline{B}\,\overline{C}+A\overline{B}C+AB\overline{C}$$
$$=000+001+100+101+110$$

000✓	00X✓	X0X
001✓	X00✓	~~X0X~~
100✓	X01✓	
101✓	10X✓	
110✓	1X0	

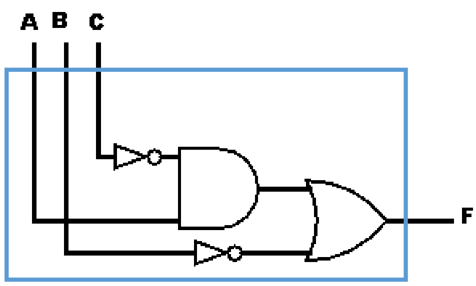
	000	001	100	101	110
1X0	X		X		⊗
X0X	X	⊗	X	⊗	

$$F_{FND}(A,B,C)=1X0+X0X$$
$$=A\overline{C}+\overline{B}$$

➦ Simplification algébrique :

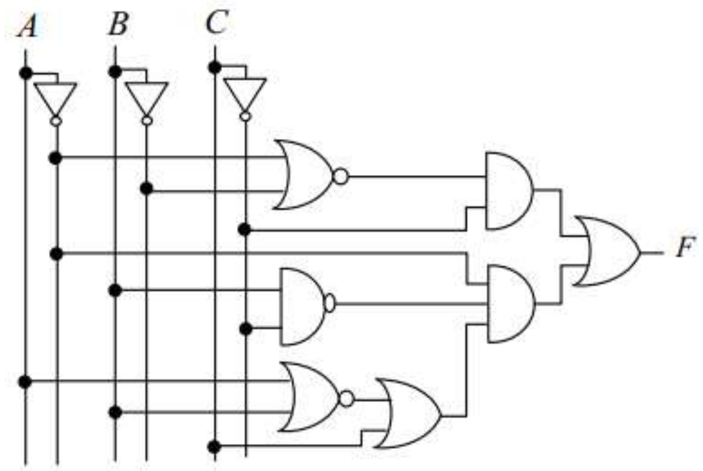
$$F_{FND}(A,B,C)=AB\overline{C}+A\overline{B}+\overline{A}\,\overline{B}\,\overline{C}+\overline{B}C$$
$$=A\left(B\overline{C}+\overline{B}\right)+\overline{B}\left(\overline{A}\,\overline{C}+C\right)$$
$$=A\left(\left(\overline{C}+\overline{B}\right)\left(B+\overline{B}\right)\right)+\overline{B}\left(\left(\overline{A}+C\right)\left(\overline{C}+C\right)\right)$$
$$=A\left(\overline{C}+\overline{B}\right)+\overline{B}\left(\overline{A}+C\right)$$
$$=A\overline{C}+A\overline{B}+\overline{A}\,\overline{B}+\overline{B}C$$
$$=A\overline{C}+\overline{B}\left(A+\overline{A}\right)+\overline{B}C$$
$$=A\overline{C}+\overline{B}+\overline{B}C$$
$$=A\overline{C}+\overline{B}(1+C)$$
$$=A\overline{C}+\overline{B}$$

➦ Les logigrammes simplifiés



Exercice N°8:

➢ L'expression logique :



$$F(A,B,C)=\overline{\left(\overline{A}+\overline{B}\right)}\overline{C}+\overline{A}\,\overline{B\overline{C}}\left(\left(\overline{A+B}\right)+C\right)$$
$$=AB\overline{C}+\overline{A}\left(\overline{B}+C\right)\left(\overline{A}\,\overline{B}+C\right)$$
$$=AB\overline{C}+\overline{A}\,\overline{B}+\overline{A}\,\overline{B}C+\overline{A}C$$

➢ Table de vérité :

A	B	C		F
0	0	0		1
0	0	1		1
0	1	0		0
0	1	1		1
1	0	0		0
1	0	1		0
1	1	0		1
1	1	1		0

➢ Les formes canoniques

$$F_{FND}(A,B,C)=\overline{A}\,\overline{B}\,\overline{C}+\overline{A}\,\overline{B}C+\overline{A}BC+AB\overline{C}$$
$$F_{FNC}(A,B,C)=\left(A+\overline{B}+C\right)\left(\overline{A}+B+C\right)\left(\overline{A}+B+\overline{C}\right)\left(\overline{A}+\overline{B}+\overline{C}\right)$$

➢ Simplification par la table de karnaugh

AB / C	00	01	11	10
0	1	0	1	0
1	1	1	0	0

$$F(A,B,C)=AB\overline{C}+\overline{A}\,\overline{B}+\overline{A}C$$

- Simplification par la méthode de quine MC-cluskey

$$F_{FND}(A,B,C) = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}BC + AB\overline{C}$$
$$= 000 + 001 + 011 + 110$$

000 ✓	00X
001 ✓	0X1
011 ✓	
110	

	000	001	011	110
110				✓
00X	✓	✓		
0X1		✓	✓	

$$F_{FND}(A,B,C) = 110 + 00X + 0X1$$
$$= AB\overline{C} + \overline{A}\,\overline{B} + \overline{A}C$$

Simplification algébrique :

$$F(A,B,C) = \left(\overline{\overline{A}+\overline{B}}\right)\overline{C} + \overline{A}\,\overline{B\overline{C}}\left(\left(\overline{A+B}\right)+C\right)$$
$$= AB\overline{C} + \overline{A}\left(\overline{B}+C\right)\left(\overline{A}\,\overline{B}+C\right)$$
$$= AB\overline{C} + \overline{A}\,\overline{B} + \overline{A}\,\overline{B}C + \overline{A}C$$
$$= AB\overline{C} + \overline{A}\,\overline{B}(1+C) + \overline{A}C$$
$$= AB\overline{C} + \overline{A}\,\overline{B} + \overline{A}C$$

Exercice N°9:

Réalisation d'un circuit complément à 1 à 4 bits.

- La table de vérité

X_3	X_2	X_1	X_0	C_3	C_2	C_1	C_0
0	0	0	0	1	1	1	1
0	0	0	1	1	1	1	0
0	0	1	0	1	1	0	1
0	0	1	1	1	1	0	0
0	1	0	0	1	0	1	1
0	1	0	1	1	0	1	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	0	0
1	0	0	0	0	1	1	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	0	1
1	0	1	1	0	1	0	0
1	1	0	0	0	0	1	1
1	1	0	1	0	0	1	0
1	1	1	0	0	0	0	1
1	1	1	1	0	0	0	0

➦ Les équations de sortie

X_3X_2 \ X_1X_0	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	1	1	0	0
10	1	1	0	0

$$C_3(X_3, X_2, X_1, X_0) = \overline{X}_3$$

X_3X_2 \ X_1X_0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	1	0	0	1
10	1	0	0	1

$$C_2(X_3, X_2, X_1, X_0) = \overline{X}_2$$

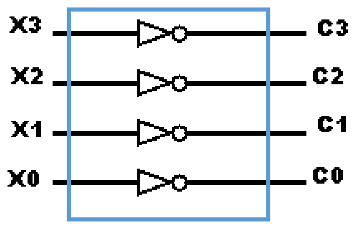
X_3X_2 \ X_1X_0	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	0	0	0	0

$$C_1(X_3, X_2, X_1, X_0) = \overline{X}_1$$

X_3X_2 \ X_1X_0	00	01	11	10
00	1	1	1	1
01	0	0	0	0
11	0	0	0	0
10	1	1	1	1

$$C_0(X_3, X_2, X_1, X_0) = \overline{X}_0$$

➦ Le logigramme simplifié



Complément à 1 à 4 bits.

Bibliographies

[1] : J. Christophe Dubacq, « Introduction à l'informatique », Cours complet IUT de Villetaneuse, S1 2016.

[2] : J. Jacques, « Architectures des l'ordinateurs », Ed. EYROLLES, 2005.

[3] : C. Frayssinet et M.Hibou, « numérique et science informatique », Spécialité NSI - 2020/2021 Version 2 - 2021.

[4] : D. Yedjour, « Codification et Représentation de l'Information », Polycopié de cours, Université des Sciences et de la Technologie d'Oran, 2018.

[5] : C. Alexandre, « Circuits numériques : 2ère partie », Polycopié de cours : Electronique A4, 2004. ».

[6] : J.M. Muller, « Arithmétique des ordinateurs, opérateurs et fonctions élémentaires, Etudes et recherches en informatique », Masson, 1989.

[7] : A. Cazes et J. Delacroix, « Architecture des machines et des systèmes informatiques, Dunod, 2005. • T », Cormen C Leiserson R Rivest Introd. À Algorithmique Dunod, vol. 1, 2002.

[8] : P. M. F. Bouami, « Architecture des Ordinateurs ». Support de Cours Filière SMI, Semestre 4, faculté pluridisciplinaire de Nador. 2020.

[9] : E. Lazard, « Architecture de l'ordinateur », Pearson Education France, 2006.

[10] : E. Viennet, « Architecture des ordinateurs », GTR, 1999.

[11] : F. GoualaRd et C. JeRmann, « Le calcul sur ordinateur », LS2N, 2023.

[12] : V. Risch, « Eléments d'Architecture des Ordinateurs », département d'informatique Inst Technol. Univ. Méditerranée, 2015.

[13] : O. Temam, « Architecture des ordinateurs », Polycopié, EP, 2008.

[14] : J. Privat, « Chapitre 10 Arithmétique réelle », Université du Québec à Montréal INF2170 Organisation des ordinateurs et assembleur Automne 2013.

[15] : C. ecile Germain et D. Etiemble, « Architecture des Ordinateurs Premi ere partie », 2009.

[16] : O. Carton, « Circuits et architecture des ordinateurs », Université Paris Diderot, 2015.

[17] : S. Fratani et P. Niebert, « Cours d’Architecture des ordinateurs », Aix Marseille Université - Laboratoire d’Informatique Fondamentale de Marseille, 2014.

[18]: N. Gessler, « George Boole et l'algèbre de la logique», Travaux de logique 9, 123-169, 1994.

[19] : Luc Museur, « Electronique numérique Logique combinatoire et séquentielle », Université Paris 13, Institut Galilée, 2015.

[20] : D. Etiemble, « Algèbre De Boole Et Fonctions Booléennes », S4-CLM, Université Paris Sud, 2004.

[21] : R. Strandh et I. Durand, « Architecture de l’ordinateur : Portes logiques, circuits combinatoires, arithmétique binaire, circuits séquentiels et mémoires », Exe. Dunod, 2005.

[22] : D. Gozim, K. Guesmi, « Logique Combinatoire Et Séquentielle », support de cours, Université Ziane Achor de Djelfa, 2019.

[23] : H. Anellis Irving « The Rise of Modern Logic : From Leibniz to Frege », review-Essay of Handbook of the History of Logic, Volume 3." (2007): 13-90.

[24] : J.M. Poitevin, « Aide-mémoire Electronique analogique et numérique ». Dunod, 2008.

[25] : E. G. Almouzni, « Architecture Des Ordinateurs », support de Cours, TD, TP. EISTI, 2013.

[26] : A. Kachouri, « Logique combinatoire », support de cours, Université Virtuelle de Tunis, 2006.

[27] : V.D. Ambeth Kumar et S.G. Amuthan, « Static structure simplification of Boolean function for ‘N’variables–A novel approach », J. Microelectron, 2016, vol. 1, no 4, p. 160-167.

[28] : D. Mange, « Analyse et synthèse des systèmes logiques », vol. 5. PPUR presses polytechniques, 1995.

[29] : D.P. Siewiorek et E.J. Mccluskey, « Switch complexity in systems with hybrid redundancy», IEEE Transactions on Computers, 1973, vol. 100, no 3, p. 276-282.

[30] : E.J. McCluskey. « Logic design principles : with emphasis on testable semicustom circuits» .Editions Prentice Hall International, ISBN 0-13-539768-5, Englewood Cliffs, New Jersey, 1986.