Section : ***S***	Option : ***Sciences de l'ingénieur***	Discipline : ***Génie Électrique***	
Les différents codes utilisés en électronique			
Domaine d'application : **Transmission de l'information**	Type de document : **Cours**	Classe : **Terminale**	Date :

I – Introduction

Le rôle de l'électronique numérique est de traiter des *informations* numériques représentant des *grandeurs physiques* (lumière, température, tension, vitesse, son, pression, nombre d'objets comptés, etc.). Mais avant de traiter une grandeur il faut la convertir en une image numérique, représentative de la grandeur et pouvant être manipulée par l'électronique. On utilise pour cela un ***code***, qui traduit fidèlement et sans ambiguïté un nombre décimal (image de la grandeur) en une succession de bits appelée ***mot-code*** (image numérique de la grandeur). Un mot-code sera matérialisé par une succession de lignes électriques ne pouvant prendre que 2 valeurs en fonction de l'état de chacun des bits (exemple : bit à 0 → 0 volt sur la ligne, bit à 1 → 5 volts sur la ligne). Ainsi codé, l'image de la grandeur pourra être analysée par des circuits numériques (circuits logiques, microcontrôleurs, calculateurs, etc.).

Mais il existe plusieurs codes binaires (code *binaire* signifie que chaque élément (appelé *élément binaire* ou *bit*) ne peut prendre *que 2 valeurs différents*), ayant chacun leurs avantages et leurs inconvénients, et possédant des propriétés utilisées dans des applications spécifiques. Certains sont adaptés pour le calcul numérique, d'autres seront utilisés pour réaliser des capteurs de position délivrant un code sans aléas, d'autres encore permettent le détection et la correction d'erreurs et seront alors utilisés pour la transmission d'informations numériques.

Lors du traitement numérique d'une information, il sera donc souvent nécessaire de passer d'un code à un autre. Les circuits logiques permettant de réaliser cette opération sont appelés *codeurs*, *décodeurs*, *encodeurs*, ou encore *transcodeurs* selon les cas.

II – Les codes binaires codant tous les nombres entiers

II – 1 – Le binaire naturel

Le binaire naturel est un système de numération à base 2, et deux symboles (habituellement 0 et 1) suffisent pour représenter tous les nombres entiers naturels. Un bit peut donc prendre les valeurs 0 ou 1. Deux bits sont nécessaires pour représenter les nombres décimaux de 0 à 3 ; 3 bits pour les nombres décimaux de 0 à 7, etc. D'une manière générale, n bits permettent de représenter en binaire naturel les nombres décimaux de 0 à 2^n-1. Un nombre exprimé en binaire naturel se présente sous la forme d'une succession de bits.

Le binaire naturel est un code pondéré, et les poids de chaque bit correspondent aux puissances successives de deux : 1 2 4 8 16 32 etc. Pour coder un nombre entier naturel en binaire naturel, il suffit de l'écrire sous la forme d'une somme finie de puissances de 2.

Exemples de conversion **décimal → binaire naturel** :

41 = 1 + 8 + 32, donc 41 s'écrit 101001 en binaire naturel.

2000 = 1024 + 512 + 256 + 128 + 64 + 16
donc $2000_{(10)} \equiv 11111010000_{(2)}$

Exemple de conversion **binaire naturel → décimal** :

$1101011001_{(2)} \equiv 2^9 + 2^8 + 2^6 + 2^4 + 2^3 + 2^0{}_{(10)}$
$\equiv 857_{(10)}$

Tableau de conversion **Décimal → Binaire**	
Décimal	Binaire naturel
0	0
1	1
2	1 0
3	1 1
4	1 0 0
5	1 0 1
6	1 1 0
7	1 1 1
8	1 0 0 0
9	1 0 0 1
10	1 0 1 0
11	1 0 1 1
Etc.	Etc.

II – 2 – Le binaire réfléchi (ou code GRAY)

La propriété principale du code gray est que le passage d'un mot-code au suivant entraînera toujours le changement d'un bit et d'un seul. Ainsi, les transitions s'effectuent sans ambiguïté, éliminant les risques d'aléas. De plus le code est cyclique pour un nombre de mot-code égal à une puissance de 2. En revanche le code gray n'est pas pondéré, il n'est donc pas adapté pour le calcul numérique. Tout comme le binaire naturel, le binaire réfléchi peut coder n'importe quel nombre entier naturel.

Remarque : le code Gray est le code utilisé dans les tableaux de Karnaugh, afin de coder la valeur des différentes entrées de telle sorte qu'une seule d'entre elles ne change de valeur d'une ligne à l'autre, ou d'une colonne à l'autre.

Tableau de conversion **Décimal → Code GRAY**	
Décimal	Code GRAY
0	0 0 0 0
1	0 0 0 1
2	0 0 1 1
3	0 0 1 0
4	0 1 1 0
5	0 1 1 1
6	0 1 0 1
7	0 1 0 0
8	1 1 0 0
9	1 1 0 1
10	1 1 1 1
11	1 1 1 0
Etc.	Etc.

Le lien entre un mot-code n codé en binaire réfléchi et un mot-code N codé en binaire naturel et codant le même nombre est le suivant :

$$n = \left| \frac{N \oplus 2N}{2} \right|$$

III – Les codes décimaux

Les codes décimaux sont utilisés pour la représentation des chiffres du système décimal (symboles 0 à 9). Ils contiennent par conséquent dix mots-code. Les 4 codes suivants sont des codes décimaux, et ne concernent donc que le codage des chiffres de 0 à 9. Pour représenter un nombre (c'est à dire une succession de chiffres) avec un code décimal, on utilise une chaîne d'éléments binaires regroupés par quatre. Chaque groupe de 4 bits (appelé une *tétrade*) est représentatif d'un chiffre.

III – 1 – Le binaire codé décimal (ou code BCD)

Travailler sur des nombres en binaire naturel est intéressant dans les calculateurs, car ces nombres sont pondérés, mais lorsqu'on veut une image rapide de l'équivalent décimal, on est amené à effectuer un transcodage long et fastidieux. Il est plus commode dans certaines applications, comme par exemple l'affichage en décimal du contenu de compteurs, d'utiliser la représentation BCD.

Le BCD (*Binary Coded Decimal*, ou Décimal Codé en Binaire en français) est le code décimal le plus utilisé en électronique. Il contient des mots-code qui sont la traduction en binaire naturel (sur 4 bits) de chacun des dix chiffres du système décimal. Chaque élément binaire d'un mot-code a un poids comme en binaire naturel : 8 4 2 1. Le BCD est donc un code **pondéré**. Pour retrouver un chiffre décimal à partir de son mot-code en BCD il suffit d'effectuer une conversion binaire → décimal pour chacune des tétrades composant le code BCD.

Tableau de conversion **Décimal → BCD**	
Décimal	BCD
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

Exemple de codage en BCD

	1	9	8	2	en décimal
	↓	↓	↓	↓	
devient	0001	1001	1000	0010	en BCD
	0101	0000	0111	0011	en BCD
	↓	↓	↓	↓	
devient	5	0	7	3	en décimal

III – 2 – Le code à excès de trois (ou code de STIBITZ)

Le code à excès de trois (*Excess 3* noté XS 3 en abrégé) s'obtient en ajoutant 3 à chaque mot-code du code BCD. Tout comme le BCD, le code à excès de 3 est un code décimal, son tableau de conversion ne concerne donc que les chiffres de 0 à 9. Comme en BCD, pour coder un nombre en code à excès de trois, il faudra concaténer une succession de tétrades, traduisant chacune un chiffre du nombre à coder.

Tableau de conversion **Décimal → Code à excès de 3**	
Décimal	Code à excès de 3
0	0 0 1 1
1	0 1 0 0
2	0 1 0 1
3	0 1 1 0
4	0 1 1 1
5	1 0 0 0
6	1 0 0 1
7	1 0 1 0
8	1 0 1 1
9	1 1 0 0

Exemple de codage en XS 3

	1	9	8	2	en décimal
	↓	↓	↓	↓	
devient	0100	1100	1011	0101	en XS 3
	1000	0011	1010	0110	en XS 3
	↓	↓	↓	↓	
devient	5	0	7	3	en décimal

Propriété du code XS 3 :

Le code à excès de trois a été créé pour permettre la réalisation simple des opérations de soustraction. Le complément à 1 d'un mot-code représente le complément à 9 dans l'ensemble source : les codes possédant cette propriété sont appelés des codes *auto-complémentaires*.

Exemple avec le chiffre 7 :

En XS 3, le chiffre 7 se code 1010. En XS 3 le complément à 1 de 7 est donc 0101 (complément de chacun des bits). En décimal le complément à 9 de 7 est 2 (9-7=2). On remarque que 0101 est bien le mot-code de 2 en XS 3. Et cette propriété d'auto-complémentarité peut être vérifiée pour les 10 chiffres : **Le complément à 1 d'un mot-code en XS 3 correspond au complément à 9 du chiffre en décimal**.

Remarque : le code XS 3 est un code **auto-complémentaire**, mais il n'est pas pondéré.

III – 3 – Le code Aïken

Le code Aïken regroupe les deux propriétés des codes BCD et XS 3 précédents : c'est un code décimal **pondéré** et **auto-complémentaire**. Les poids des éléments binaires sont 2 4 2 1. La différence entre le code Aïken et le code BCD est le poids du premier bit à gauche : il valait 8 en BCD alors qu'il vaut 2 en code Aïken.

Tableau de conversion **Décimal → Code Aïken**	
Décimal	Code Aïken
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	1 0 1 1
6	1 1 0 0
7	1 1 0 1
8	1 1 1 0
9	1 1 1 1

Exemple de codage en code Aïken

	1	9	8	2	en décimal
	↓	↓	↓	↓	
devient	0001	1111	1110	0010	en Aïken
	1011	0100	0111	1100	en Aïken
	↓	↓	↓	↓	
devient	5	4	7	6	en décimal

Propriété d'auto-complémentarité du code Aïken :

Exemple avec le chiffre 8 :

- En décimal le complément à 9 de 8 est 1
- En Aïken le complément à 1 de 1110 (mot-code du 8) est 0001 (mot-code du 1)

Et cette propriété d'auto-complémentarité peut être vérifiée pour les 10 chiffres : **Le complément à 1 d'un mot-code en code Aïken correspond au complément à 9 du chiffre en décimal**.

III – 4 – Les codes « 2 parmi 5 »

Avec les codes « 2 parmi 5 », les mots-code comprennent 5 bits dont 2 sont à 1 (et les 3 autres à 0). Il existe plusieurs codes « 2 parmi 5 », et les plus utilisés sont le code « 8 4 2 1 0 » et le code « 7 4 2 1 0 ». Ces deux codes sont pondérés, la liste des poids figurant dans la dénomination du code.

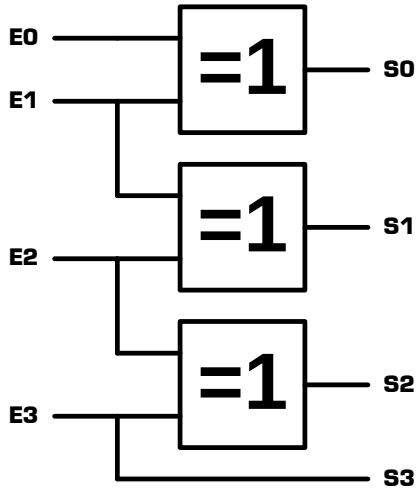
Tableau de conversion **Décimal → Code « 2 parmi 5 »**		
Décimal	Code « 2 parmi 5 » 8 4 2 1 0	Code « 2 parmi 5 » 7 4 2 1 0
0	1 0 1 0 0	1 1 0 0 0
1	0 0 0 1 1	0 0 0 1 1
2	0 0 1 0 1	0 0 1 0 1
3	0 0 1 1 0	0 0 1 1 0
4	0 1 0 0 1	0 1 0 0 1
5	0 1 0 1 0	0 1 0 1 0
6	0 1 1 0 0	0 1 1 0 0
7	1 1 0 0 0	1 0 0 0 1
8	1 0 0 0 1	1 0 0 1 0
9	1 0 0 1 0	1 0 1 0 0

Les codes « 2 parmi 5 » font partie des codes spécialement conçus pour la transmission de l'information et pour la détection des erreurs. En effet, si on reçoit un nombre codé en « 2 parmi 5 », pour détecter une éventuelle erreur dans ce nombre il suffit de compter le nombre de 1 logiques présents dans chacun des groupes de 5 bits. Si un groupe ne présente pas deux 1 logiques, on peut en déduire avec certitude qu'il est erroné.

Remarque : contrairement à d'autres codes plus perfectionnés (code de Hamming par exemple), les codes « 2 parmi 5 » permettent de détecter une erreur, mais ne permettent pas de la corriger. De plus, si lors de la transmission, 2 bits de valeurs différentes changent simultanément d'état, aucune erreur ne pourra être détectée à l'arrivée.

IV – Les transcodeurs

Les transcodeurs sont des circuits en logique combinatoire, permettant de convertir un mot-code d'un code vers un autre. Par exemple, le montage ci-dessous utilisant 3 portes logiques OU-Exclusif est un transcodeur 4 bits. Le mot-code d'entrée, exprimé sur 4 bits (E0 à E3, E0 étant le LSB) est exprimé en binaire naturel :



Mot-code d'entrée (sur 4 bits) codée en binaire naturel				Mot-code de sortie (sur 4 bits) codée en			
E3	**E2**	**E1**	**E0**	**S3**	**S2**	**S1**	**S0**
0	0	0	0				
0	0	0	1				
0	0	1	0				
0	0	1	1				
0	1	0	0				
0	1	0	1				
0	1	1	0				
0	1	1	1				
1	0	0	0				
1	0	0	1				
1	0	1	0				
1	0	1	1				
1	1	0	0				
1	1	0	1				
1	1	1	0				
1	1	1	1				

Complétez la table de vérité de ce transcodeur, et déterminez quel est le code de sortie :

Ce montage est un transcodeur **binaire naturel** → .....