



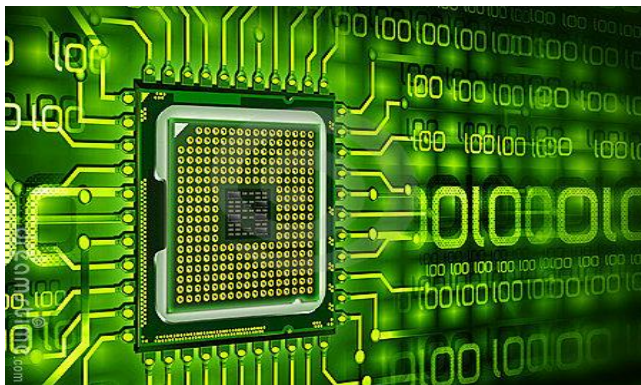
**R 1.03**

**2023 - 2024**

# **Introduction à l'architecture des ordinateurs**

## **TD n° 3**

### **Codage de l'information Codes détecteurs d'erreurs**



**ANNE Jean-François**  
*D'après le cours de M. JEANPIERRE*

Le but de ce TD est de se familiariser avec les codes de détection, voire de correction, d'erreurs et les opérateurs logiques.

## A. Distance de Hamming

1°) Quelle est la distance de Hamming de ce code binaire (4 mots de 5 bits) ?

00000 ; 01010 ; 10101 ; 11111

2°) Proposez un code binaire de 10 mots ayant une distance de Hamming de 3 bits.

## B. Code à parité

1°) Quelle est la parité de ce code (8 mots de 4 bits) ?

0 : 0001 ; 1 : 0010 ; 2 : 0100 ; 3 : 0111 ; 4 : 1000 ; 5 : 1011 ; 6 : 1101 ; 7 : 1110

2°) Quelle est la distance de Hamming du code précédent ?

3°) Codez les nombres suivants avec le code précédent

$123_8$  ;  $001010011_2$  ;  $123_{16}$  ;  $12_{16}$

4°) Codez le message ( $1234_8$ ) avec le code précédent.

5°) Quelle est la distance de Hamming ?

6°) Le mettre sous la forme d'un code de bloc, en ajoutant un contrôle de parités horizontale et verticale impaires et un contrôle de parité croisée paire.

7°) Quelle est la distance de Hamming du nouveau **message obtenu** ?

## C. Contrôle de Redondance Cyclique (CRC)

1°) Soit le polynôme générateur  $G(X) = X^5 + X + 1$

- Quelle sera la longueur du message correspondant à une donnée de 9 bits ; 10 bits ; 1024 octets ?
- Déterminer le message binaire à transmettre correspondant à la donnée octale  $456_8$ .
- Vous recevez le message «  $76543_8$  », sur 15 bits. Pouvez-vous calculer la donnée initiale ?

2°) Soit le polynôme générateur  $G(X) = X^2 + X + 1$

- a. Vous recevez le message  $T(X) = X^{10} + X^7 + X^6 + X^4 + X^3 + X$ . Est-il correct ?
- b. Calculez la donnée initiale binaire de ce message  $T(X)$
- c. Pouvez-vous trouver une altération discrète de ce message (message faux avec CRC juste) ?

**D. Code correcteur d'erreurs**

Le protocole Wifi 802.11b définit un codage du bit 1 par **10110111000**<sub>2</sub>, et le codage du bit 0 par **01001000111**<sub>2</sub>. Ce code, dit DSSS offre une très bonne résistance aux erreurs.

- a. Quelle est sa distance de Hamming ?
- b. Quelle est la donnée initiale de ce message reçu ?

11000111000010010001111011011111100110111000010010001111001011100001  
00100011110110111001

- c. Quelle serait la représentation binaire de la donnée B5<sub>16</sub> ?

**E. Opérateurs Logiques simples**

1°) **Non**

Effectuez l'opération logique NON (NO) (!) avec les exemples suivants :

00001111<sub>2</sub> ; 123<sub>10</sub> ; FA<sub>16</sub> ; 255<sub>10</sub>

2°) **ET**

Effectuez l'opération logique ET (AND) (&) avec les exemples suivants :

$\begin{array}{r} 1011 \ 0111 \\ \& \ 1101 \ 1111 \end{array}$	$\begin{array}{r} 1100 \ 1011 \\ \& \ 1001 \ 0110 \end{array}$	$\begin{array}{r} 0110 \ 1111 \\ \& \ 1110 \ 1110 \end{array}$
--	--	--

100<sub>2</sub> ET 11111101<sub>2</sub>

11101010<sub>2</sub> ET 11011100<sub>2</sub>

5<sub>16</sub> ET FB<sub>16</sub>

8<sub>16</sub> AND D<sub>16</sub>

7D<sub>16</sub> ET (! 73<sub>16</sub>)

3°) **OU**

Effectuez l'opération logique OU (OR) (|) avec les exemples suivants :

$\begin{array}{r} 1010 \ 1101 \\   \ 1001 \ 1111 \end{array}$	$\begin{array}{r} 1010 \ 0100 \\   \ 0100 \ 1001 \end{array}$	$\begin{array}{r} 1101 \ 1011 \\   \ 0111 \ 1111 \end{array}$
---	---	---

101101<sub>2</sub> OU 11001100<sub>2</sub>

$11100110_2$  OU  $10010100_2$

$9_{16}$  OR  $0_{16}$

$A8_{16}$  OU  $7E_{16}$

$7D_{16}$  OU ( $! 73_{16}$ )

#### 4°) OU Exclusif

Effectuez l'opération logique OU Exclusif (XOR) ( ^ ) avec les exemples suivants :

$$\begin{array}{r} 1010 \ 1101 \\ \wedge \ 1001 \ 1111 \\ \hline \end{array}$$

$$\begin{array}{r} 1010 \ 0100 \\ \wedge \ 0100 \ 1001 \\ \hline \end{array}$$

$$\begin{array}{r} 1101 \ 1011 \\ \wedge \ 0111 \ 1111 \\ \hline \end{array}$$

$1001010_2$  XOR  $11011011_2$

$D_{16}$  XOR  $3_{16}$

$5_{16}$  XOR  $E_{16}$

$4B_{16}$  XOR  $12_{16}$

$7D_{16}$  XOR ( $! 73_{16}$ )

#### 5°) NON ET

$100_2$  NON ET  $11111101_2$

$11101010_2$  !ET  $11011100_2$

$5_{16}$  !ET  $FB_{16}$

$8_{16}$  !AND  $D_{16}$

$7D_{16}$  NAND ( $! 73_{16}$ )

#### 6°) NON OU

$101101_2$  NON OU  $11001100_2$

$11100110_2$  !OU  $10010100_2$

$9_{16}$  !OR  $0_{16}$

$A8_{16}$  !OU  $7E_{16}$

$7D_{16}$  NOR ( $! 73_{16}$ )

#### 7°) NON OU Exclusif

$1001010_2$  XNOR  $11011011_2$

$D_{16}$  !XOR  $3_{16}$

$5_{16}$  !XOR  $E_{16}$

$4B_{16}$  !XOR  $12_{16}$

$7D_{16}$  !XOR ( $! 73_{16}$ )

## **F. Décalages**

### **1°) A gauche**

- ❖ Que devient le nombre hexadécimal D après un décalage à gauche de 2 bits ?
- ❖ Que devient le nombre hexadécimal 5 après un décalage à gauche de 2 bits ? :
- ❖ Que devient le nombre hexadécimal D5 après un décalage à gauche de 1 bit ?
- ❖ Que devient le nombre hexadécimal AAAA après un décalage à gauche de 4 bits ?

### **2°) A droite**

- ❖ Que devient le nombre hexadécimal 6 après un décalage à droite de 2 bits ?
- ❖ Que devient en hexadécimal, le nombre hexadécimal A7 après un décalage à droite de 2 bits ?
- ❖ Que devient le nombre hexadécimal 101 après un décalage à droite de 4 bits ?
- ❖ Que devient le nombre hexadécimal 15A après un décalage à droite de 1 bit ?

## **G. Masques**

- ❖ Comment ne garder que les 4 bits de gauche du nombre  $AF_{16}$  ?
- ❖ Comment ne garder qu'un bit sur 2 du nombre  $A5_{16}$  à partir de la droite, et mettre les autres à 0 ?
- ❖ Comment ne garder que les 4 bits de droite du nombre  $FA_{16}$  ?
- ❖ Comment mettre à 0 les 4 bits de gauche du nombre  $AF_{16}$  ?
- ❖ Comment mettre à 1 les 4 bits de gauche du nombre  $AF_{16}$  ?
- ❖ Comment mettre à 1 les 4 bits de gauche du nombre  $AF_{16}$  avec le masque  $0F_{16}$  ?
- ❖ Comment mettre à 0 les 4 bits de droite du nombre  $A5_{16}$  avec le masque  $0F_{16}$  ?

## **H. Programmes**

### **1°) Exercice 1**

Donnez le résultat du programme suivant :

```
#include<stdio.h>
int main()
{
    int a,b;
    int c = 6, d = 1;
    a = c&d;
    b = c|d;
    printf("a=%d ; b=%d ;", a, b);
    return 0;
}
```

### **2°) Exercice 2**

Donnez le résultat du programme suivant :

```
#include<stdio.h>
int main()
```

```

{
    int a,b;
    a = 0b11001101;
    b = 0b00001111 & a;
    printf("a=%d ; b=%d ;", a, b);
    return 0;
}

```

### 3°) Exercice 3

Donnez le résultat du programme suivant :

```

#include<stdio.h>
int main()
{
    int a,b;
    int c = 6, d = 1;
    a = c && d;
    b = c || d;
    printf("a=%d ; b=%d ;", a, b);
    return 0;
}

```