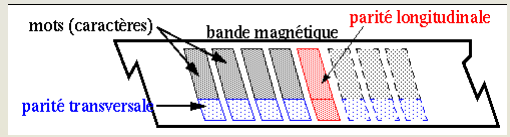
JFA - 72

Codes en blocs : Exemple simple

- On code le message sous forme de b caractères codés sur a bits. Le bloc de données est disposé sous une forme matricielle (k = a.b).
- On ajoute une parité paire sur chaque ligne : VRC (Vertical Redundancy Check : Contrôle de parité verticale)
- On ajoute une parité paire sur chaque colonne : LRC (Longitudinal Redundancy Check : Contrôle de parité horizontale)
- On ajoute une parité paire croisée de la colonne VRC et de la ligne LRC
- On obtient une matrice de taille (a+1, b+1).
- ***Historique :***



https://www.irisa.fr/armor/lesmembres/cousin/Enseignement/Reseaux-generalites/Cours/3-3.htm

JFA - 73

Codes en blocs : Exemple simple

- Si on cherche à transmettre le message : JFAnne.

Exemple :

Lettre	VRC	B6	B5	B4	B3	B2	B1	B0	Ascii
J	1	1	0	0	1	0	1	0	$4A_h$
F	1	1	0	0	0	1	1	0	46_h
A	0	1	0	0	0	0	0	1	41_h
n	1	1	1	0	1	1	1	0	$6E_h$
n	1	1	1	0	1	1	1	0	$6E_h$
e	0	1	1	0	0	1	0	1	65_h
LRC	0	0	1	0	1	0	0	0	28_h

On va donc transmettre les codes suivants :
CA C6 41 EE EE 65 28

JFA - 74

Codes en blocs : Exemple simple

- Si à la réception, on a reçu :

CA C6 41 EE EE 65 28

Lettre	VRC	B6	B5	B4	B3	B2	B1	B0	Ascii
J	1	1	0	0	1	0	1	0	CA_h
F	1	1	0	0	0	1	1	0	$C6_h$
A	0	1	0	0	0	0	0	1	41_h
n	1	1	1	0	1	1	1	0	EE_h
n	1	1	1	0	1	1	1	0	EE_h
e	0	1	1	0	0	1	0	1	65_h
LRC	0	0	1	0	1	0	0	0	28_h

On en déduit que :

- le message original était : JFAnne,
- et qu'il n'y a pas eu d'erreur.

JFA - 75

Codes en blocs : Exemple simple

- Si à la réception, on a reçu : ***CA C6 43 EE EE 65 28***

Lettre	VRC	B6	B5	B4	B3	B2	B1	B0	Ascii
J	1	1	0	0	1	0	1	0	CA_h
F	1	1	0	0	0	1	1	0	$C6_h$
~~A~~ C	~~0~~ 1	1	0	0	0	0	1	1	43_h
n	1	1	1	0	1	1	1	0	EE_h
n	1	1	1	0	1	1	1	0	EE_h
e	0	1	1	0	0	1	0	1	65_h
LRC	0	0	1	0	1	0	~~0~~ 1	0	28_h

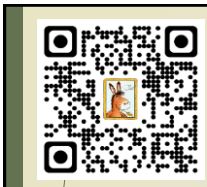
Légende : Code correct, Code recalculé à la réception.

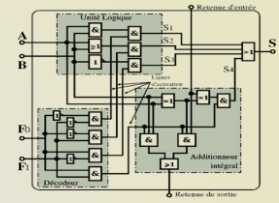
À l'aide des codes de parité erronés de VRC et LRC, on peut déterminer :

- quelle ligne et quelle colonne sont fausses,
- donc en déduire quel bit pose problème,
- et inverser le bit erroné pour corriger l'erreur,

et trouver le message original : ***CA C6 41 EE EE 65 28***

On ne peut détecter plusieurs erreurs de transmission, mais on ne peut corriger qu'une seule erreur.





Les fonctions logiques

JFA - 76

DUT Informatique – Semestre 1

Ressource R 1.03

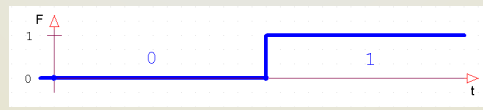
Responsable : Jean-François ANNE

07/10/2023

Fonctions logiques

JFA - 77

- L'algèbre de BOOLE s'intéresse aux opérations et aux fonctions sur les variables logiques. Elle permet de remplacer des signaux et des opérateurs par des expressions mathématiques. Chaque signal est remplacé par une variable, et son traitement par une fonction logique..
- Un circuit électrique peut avoir 2 états : marche ou arrêt (on ou off), une lampe peut être allumée ou éteinte, un interrupteur peut-être ouvert ou fermé : on peut alors leur associer deux valeurs binaires (0, 1).
- En général :
  - Un état 0 indique un arrêt, une lampe éteinte, pas de tension, ou de courant, un interrupteur ouvert.
  - Un état 1 indique une marche, une lampe allumée, une présence de tension, ou de courant, un interrupteur fermé.
- Chronogramme :
  - Le **chronogramme** est une représentation graphique de l'évolution d'un signal électrique ou d'un état en fonction du temps



JFA - 78

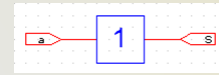
Opérateur Oui

- L'opérateur OUI effectue l'égalité entre deux variables. Il sert à transmettre et à amplifier l'information.

- ***Table de vérité :***

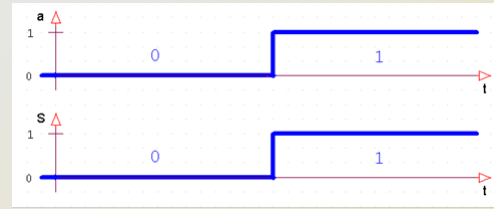
a	S
0	0
1	1

- ***Symbole :***



- ***Équation :*** $S = a$

- ***Chronogramme :***



JFA - 79

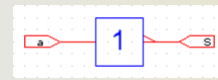
Opérateur Non (NO)

- L'opérateur NON effectue le complément entre deux variables. Il sert à transmettre et à amplifier l'information en l'inversant.

- ***Table de vérité :***

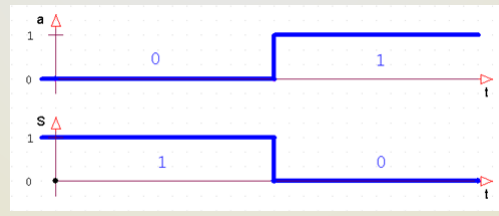
a	S
0	1
1	0

- ***Symbole :***



- ***Équation :*** $S = \bar{a}$

- ***Chronogramme :***



JFA - 80

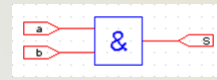
Opérateur ET (AND)

- L'opérateur ET aussi appelé Intersection, appliqué à 2 variables effectue le produit de ces variables. On la note par le signe «•». Le résultat de la sortie est égal à 1 si toutes les variables d'entrée sont à 1.

❑ ***Table de vérité :***

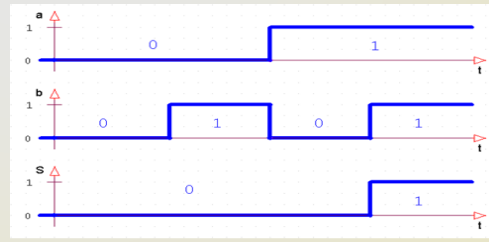
b	a	S
0	0	0
0	1	0
1	0	0
1	1	1

❑ ***Symbole :***



❑ ***Équation :*** $S = a.b$

❑ ***Chronogramme :***



❑ ***Propriétés :***

$1 . a = a$
$0 . a = 0$
$a . a = a$
$a . \bar{a} = 0$

JFA - 81

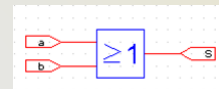
Opérateur OU « Inclusif » (OR)

- L'opérateur OU (inclusif) aussi appelé Union, appliqué à 2 variables effectue la « somme » de ces variables. On la note par le signe «+». Le résultat de la sortie est égal à 1 si au moins une des variables d'entrée est à 1.

❑ ***Table de vérité :***

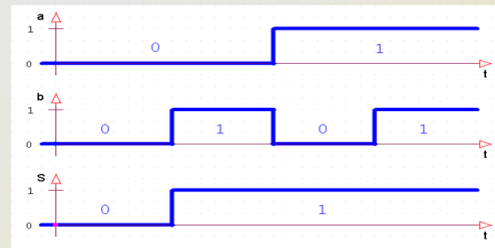
b	a	S
0	0	0
0	1	1
1	0	1
1	1	1

❑ ***Symbole :***



❑ ***Équation :*** $S = a + b$

❑ ***Chronogramme :***



❑ ***Propriétés :***

$1 + a = 1$
$0 + a = a$
$a + a = a$
$a + \bar{a} = 1$

JFA - 82

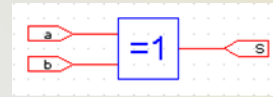
Opérateur OU Exclusif (XOR)

- L'opérateur OU (exclusif) aussi appelé Anti coïncidence, appliqué à 2 variables effectue la « non-égalité » de ces variables. On la note par le signe «⊕». Le résultat de la sortie est égal à 1 si le nombre d'entrées à 1 est impair.

Table de vérité :

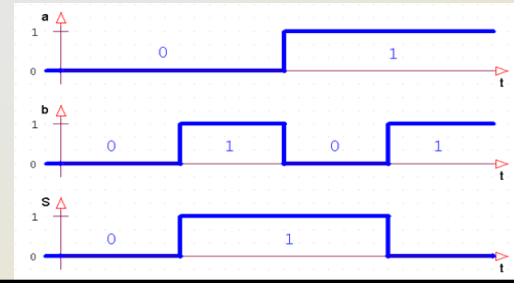
b	a	S
0	0	0
0	1	1
1	0	1
1	1	0

Symbole :



Équation : $S = a \oplus b = \bar{a}.b + a.\bar{b}$

Chronogramme :



Propriétés :

$1 \quad a = \bar{a}$

$0 \quad a = a$

$a \oplus a = 0$

$a \oplus \bar{a} = 1$

JFA - 83

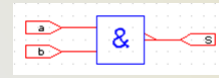
Opérateur NON ET (NAND)

- L'opérateur NON ET, appliquée à 2 variables effectue le complément du produit de ces variables. On la note par le signe «•» avec une barre sur les variables. Le résultat de la sortie est égal à 1 si une des variables d'entrée est à 0.

Table de vérité :

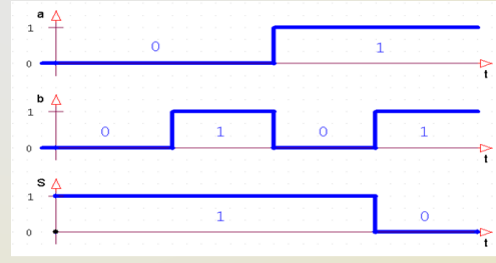
b	a	S
0	0	1
0	1	1
1	0	1
1	1	0

Symbole :



Équation : $S = \overline{a.b} = \bar{a} + \bar{b}$

Chronogramme :



Propriétés :

$\overline{1.a} = \bar{a}$

$\overline{0.a} = 1$

$\overline{a.a} = \bar{a}$

$\overline{a.\bar{a}} = 1$

JFA - 84

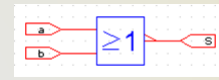
Opérateur NON OU « Inclusif » (NOR)

- L'opérateur NON OU (inclusif), appliqué à 2 variables effectue le complément de la « somme » de ces variables. On la note par le signe «+» avec une barre sur les variables. Le résultat de la sortie est égal à 1 si toutes les variables d'entrée sont à 0.

❑ ***Table de vérité :***

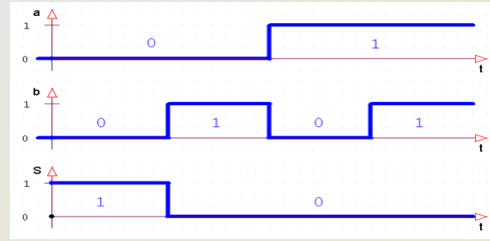
b	a	S
0	0	1
0	1	0
1	0	0
1	1	0

❑ ***Symbole :***



❑ ***Équation :*** $S = \overline{a + b} = \bar{a} \, . \, \bar{b}$

❑ ***Chronogramme :***



❑ ***Propriétés :***

$$\overline{1 + a} = 0$$
$$\overline{0 + a} = \bar{a}$$
$$\overline{a + a} = \bar{a}$$
$$\overline{a + \bar{a}} = 0$$

JFA - 85

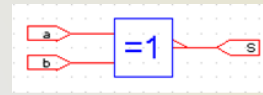
Opérateur NON OU Exclusif (EXNOR)

- L'opérateur NON OU Exclusif aussi appelé égalité, appliquée à 2 variables teste « l'égalité » de ces variables. On la note par le signe «⊕» avec une barre sur les variables. Le résultat de la sortie est égal à 1 si les valeurs des entrées sont identiques.

❑ ***Table de vérité :***

b	a	S
0	0	1
0	1	0
1	0	0
1	1	1

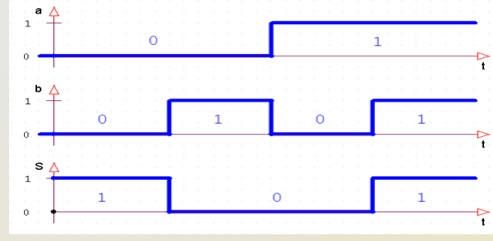
❑ ***Symbole :***



❑ ***Équation :***

$$S = \overline{a \oplus b} = a.b + \bar{a}.\bar{b}$$

❑ ***Chronogramme :***



❑ ***Propriétés :***

$$\overline{1 \oplus a} = a$$
$$\overline{0 \oplus a} = \bar{a}$$
$$\overline{a \oplus a} = 1$$
$$\overline{a \oplus \bar{a}} = 0$$

JFA - 86

Application d'un masque sur une donnée

- Toute donnée informatique est stockée en mémoire sous la forme d'une combinaison de bits.
- On peut vouloir ne récupérer qu'une partie de la donnée en utilisant des masques et des opérations logiques. La réalisation de l'opération de masquage se fait en plusieurs temps :
  - **le choix de l'opérateur,**
  - **la construction du masque**
  - **et son application.**

- Les opérateurs de bits permettent de modifier, de conserver ou de tester un ou plusieurs bits d'une donnée, suivant ce que l'on désire conserver. Ces opérateurs sont :
  - **NON (NOT)**
  - **ET (AND)**
  - **OU (OR)**
  - **OU exclusif (XOR)**
  - **Décalage à droite (SHR)**
  - **Décalage à gauche (SHL)**

Opérateur	Signification	Exemple
&	ET bit à bit	a & b
|	OU bit à bit	a | b
^	OU EXCLUSIF bit à bit	a ^ b
>>	décalage à droite	a >> 2 (décalage à droite de 2 bits)
<<	décalage à gauche	a << 5 (décalage à gauche de 5 bits)
~	complément à 1 (NON bit à bit)	~a

Les opérateurs de manipulation de bit

JFA - 87

Opération NON

- L'opération NON effectue le complément à 1 de la valeur de la variable. C'est-à-dire un non bit à bit.

- ***Exemple :***
  on désire inverser (trouver le complément à 1) du nombre a :

En base 2 :

```
 a = 0110 1010
~a = 1001 0101
```

En base 16 :

```
 a = 6A
~a = 95
```

JFA - 88

Opération Décalage à gauche (SHL)

- L'opération SHL effectue un décalage à gauche des bits du nombre en binaire. Les chiffres ajoutés à droite sont des 0.
- ***Exemple :***
  on désire multiplier par 16 donc, décaler à gauche de 4 bits, les bits du nombre contenu dans a :
  a = a << 4

En base 2 :

a = 0110 1010
a<<4 = 1010 0000

En base 16 :

a = 6A
a<<4 = A0

JFA - 89

Opération Décalage à droite (SHR)

- L'opération SHR effectue un décalage à droite des bits du nombre en binaire. Les chiffres ajoutés à gauche sont des 0.
- ***Exemple :***
  on désire diviser par 8 donc, décaler à droite de 3 bits, les bits du nombre contenu dans a :
  a = a >> 3

En base 2 :

a = 0110 1010
a>>3 = 0000 1101

En base 16 :

a = 6A
a>>3 = 0D

JFA - 90

Application d'un ET avec masque de donnée

- Ainsi, un opérateur **ET** permet de conserver la valeur de la donnée, en utilisant pour 2ème variable la valeur 1, ou de la forcer à 0 en utilisant la valeur 0.
- ***Exemple :***
  on désire conserver les deux bits de poids faible et forcer la valeur des autres à zéro dans un mot de 8 bits. On choisit donc l'opérateur ET. Puis on choisit un masque comportant des 1 à la même position que les bits à conserver et des zéro ailleurs :

En base 2 :

```
  0110 1010
& 0000 0011
------------
  0000 0010
```

En base 16 :

```
  6A
& 0F
-----
  0A
```

JFA - 91

Application d'un OU avec masque de donnée

- Un opérateur **OU** permet de mettre une valeur à 1 en utilisant pour 2ème variable la valeur 1, et un 0 pour ne pas la modifier.

- ***Exemple :***
  on désire mettre à 1 les deux bits de poids faible et laisser les autres à leur valeur d'origine dans un mot de 8 bits. On choisit donc l'opérateur OU. Puis on choisit un masque comportant des 1 à la même position que les bits à modifier et des zéros ailleurs :

En base 2 :

```
  0110 1010
| 0000 0011
------------
  0110 1011
```

En base 16 :

```
  6A
| 0F
-----
  6F
```

JFA - 92

Application d'un OUEX avec un masque

- Un opérateur **OU Exclusif** permet d'inverser la valeur du bit si c'est un 1, et un 0 pour ne pas la modifier.

- ***Exemple :***
  on désire inverser les deux bits de poids faible et laisser les autres à leur valeur d'origine dans un mot de 8 bits. On choisit donc l'opérateur OU Exclusif. Puis on choisit un masque comportant des 1 à la même position que les bits à modifier et des zéros ailleurs :

En base 2 :

```
  0110 1010
^ 0000 0011
------------
  0110 1001
```

En base 16 :

```
  6A
^ 0F
-----
  65
```