Cours d'Automatique combinatoire & séquentielle

Automatique : science de l'étude et de la conception de machines qui commandent aux machines

Numérotation - codage

Sommaire

I° SYSTEME DE NUMERATION 5

I-1) BASE 10 (DECIMALE) 5

II° PASSAGE D'UNE BASE A L'AUTRE : 6

III° OPERATIONS ARITHMETIQUES EN BASE BINAIRE / HEXADECIMALE: 8

IV° LES CODES 10

IV-4) LIAISON SERIE ASYNCHRONE RS232: 12

V) TRAVAUX DIRIGES: 16

I° SYSTEME DE NUMERATION

Dans un système de numération en base B, un nombre A s'écrit avec des chiffres de la manière suivante :

$$A = a_{(n-1)} .B^{(n-1)} + a_{(n-2)}.B^{(n-2)} + \ldots\ldots + a_1.B^1 + a_0.B^0 + a_{-1}.B^{-1} + \ldots\ldots + a_{-m}.B^{-m}$$

Les différents a_i constituent les *chiffres* de ce *nombre* qui comporte **n** chiffres dans la partie **entière** et **m** chiffres dans la partie **fractionnaire**.

B est la base de numération exprimée en décimal. Dans une base **B**, il y a **B** chiffres différents.

Chaque chiffre a un poids qui dépend de la place qu'il occupe :

Les chiffres de gauche sont de poids FORT

Les chiffres de droite sont de poids faible

I-1) BASE 10 (DECIMALE)

Il y a dix chiffres différents : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Ex : $(165,24)_{10}$ = . . .
= . . .

Rem : Avec n chiffres, on peut représenter 10^n nombres entiers différents de 0 à 10^n -1

Par exemple avec 3 chiffres on représente 1000 nombres entiers (10^3) de 0 à 999 (1000 – 1)

I-2) BASE 2 (BINAIRE)

Il y a deux chiffres différents : 0, 1

Les systèmes automatiques présentent fréquemment deux modes de fonctionnement (interrupteur ouvert ou fermé, Vérin rentré ou sorti …) qui sont représentatifs des 2 chiffres de la base 2.

Un chiffre binaire s'appelle un bit (contraction de binary digit)

Ex : $(10110,01)_2$ = . . .
= . . .
= (. . . . . .)$_{10}$

Le bit de poids FORT est appelé MSB (Most Signifiant Bit)
Le bit de poids faible est appelé LSB (Least Signifiant Bit)

Lorsque l'on parle d'un Automate 8 bits, c'est un circuit de traitement manipulant des combinaisons de 8 chiffres binaires qu'on appelle aussi **OCTET (byte)**

Valeur mini d'un octet entier : . . .
Valeur maxi d'un octet entier : . . .

Exercice : Déterminer le nombre nécessaire de chiffre n_1 d'un nombre entier exprimé dans la base 2 pour écrire le même nombre entier utilisant n_2 chiffres dans la base 10.

I-3) BASE 16 (HEXADECIMALE)

Il y a seize chiffres différents : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Ex : $(5E6,A)_{16}$ = . . .
= . . .
= (. . . . . .)$_{10}$

Rem : $2^4 = 16$ donc chaque chiffre hexadécimal peut être représenté par un nombre de 4 bits
On utilisera donc l'hexadécimal pour représenter les grands nombres binaires
Table de correspondance :

Hexa	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Décimal	0	1	2													
Binaire	0000															

II° PASSAGE D'UNE BASE A L'AUTRE :

II-1) BASE 2 EN BASE 10

On ajoute les poids binaires validés

Table des puissances de 2 :

n	-5	-4	-3	-2	-1	0	1	2	3	4	5
2^n											

n	6	7	8	9	10	11	12	13	14	15	16
2^n											

Ex1 : 1000,001 = … = (. . . . . .)$_{10}$

Rem : En informatique 2^{10} = 1024 = 1k on fait l'approximation 1024 # 1000 !

2^{10} bits = 1 kilo bit	2^{20} bits = 1 Méga bit	$2^{\cdots}$ bits = 1 Giga bit
2^{12} bits = . . .	2^{23} bits = . . .	$2^{\cdots}$ bits = 16 Giga bits

Ex : 1 kilo octet = ... octets = ... k bit = ... bits

II-2) BASE 16 EN BASE 10

On ajoute les poids hexadécimaux validés

Table des puissances de 16 :

n	-2	-1	0	1	2	3	4	5
16^n	0,003906	0,0625						1048576

Ex : 1C,0A = . . . = (. . . . .)$_{10}$

II-3) BASE 10 EN BASE 16

Règle 1: on divise la partie entière du nombre à convertir par la base jusqu'à ce que le quotient soit nul, le résultat est constitué des différents restes lus de bas en haut :

Ex : $(1387,5)_{10}$ = ? $_{16}$

1387	16		
107	86	16	
11	6	5	16
		5	0

11 = B, poids faible, 5 est le poids fort donc 1387 = $(56B)_{16}$

Règle 2: on multiplie la partie fractionnaire du nombre à convertir par la base jusqu'à la précision attendue, le résultat est constitué des différentes parties entière des produits lus de haut en bas:

0,5
x 16
= 8,0 → 0
x 16
= 0,0

0, 5 = (0, 80) $_{16}$

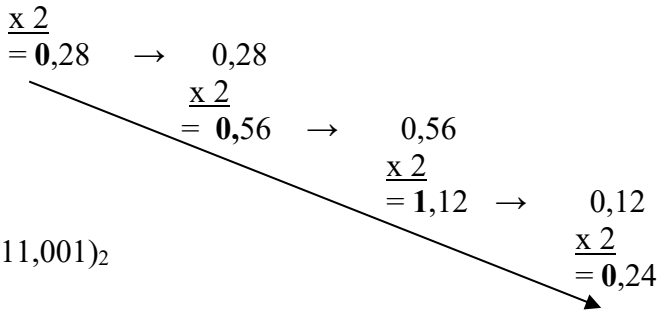
Donc 1387, 5 = $(56B, 8)_{16}$

EX : $(1024)_{10}$ = ? $_{16}$

II-4) BASE 10 EN BASE 2

On divise la partie entière par 2 et on multiplie la partie fractionnaire par 2

Ex : 3,14 partie entière 3 = $(11)_2$
Partie fractionnaire 0,14 : 0,14



Une valeur approchée de 0,14 est :
$(0,0010)_2 = 0,125$

Donc $3,14 = (11,001)_2$

Une valeur plus précise :
$(11,001\ 001)_2 = 3 + 0.125 + 0,015625$
$= 3, 1406$

II-5) BASE 16 EN BASE 2

On procède en 2 temps : Base 16 en base 10 puis Base 10 en base 2

Autre méthode : on transcrit chaque chiffre hexa en binaire sur 4 bits

Ex : $(5E6,3)_{16} = (\ldots)_2$

II-6) BASE 2 EN BASE 16 :

On découpe le nombre en quartet à partir de la droite et on transcrit chaque quartet binaire en hexadécimal :

Ex1 : $(100\ 1110\ 1010, 11)_2 = (\ldots)_{16}$ Ex2 : $(1111\ 1111)_2 = (\ldots)_{16}$

II-7) SYMBOLISATION :

Binaire : préfixe **%** ou **0b** : $(101\ 1110\ 0110)_2$ = **%**101 1110 0110 = **0b**101 1110 0110

Hexadécimal : préfixe **$** ou **H** ou **0x:** $(5E6)_{16}$ = **$** 5E6 = **H** 5E6 = **0x**5E6

Décimal : pas de préfixe

III° OPERATIONS ARITHMETIQUES EN BASE BINAIRE / HEXADECIMALE:

III-1 : L'ADDITION :

Table d'addition :

a	b	S = a + b	Retenue
0	0		
0	1		
1	0		
1	1		

Exercice 1:

0b 0 0 1 0 1 0 1 0	⇔	$(\ldots\ldots)_{10}$	*Rem :* le résultat en base 10 est exprimé sur 3 chiffres, il pourrait s'exprimer sur 9 bits en base 2
+ 0b 0 0 1 1 1 1 0 0		+ $(\ldots\ldots)_{10}$	
= 0b		= $(\ldots\ldots)_{10}$	

Exercice 2 :

0x 2 A	⇔	$(\ldots\ldots)_{10}$	0xA = 10 0xC = 12 0xA + 0xC = 22 = 0x16 0x66 = 6 * 16 + 6 = 102
+ 0x 3 C		+ $(\ldots\ldots)_{10}$	
= 0x		= $(\ldots\ldots)_{10}$	

III-2 : LA SOUSTRACTION :

Table de soustraction :

a	b	S = a - b	Report
0	0		
0	1		
1	0		
1	1		

Exercice 1 :

0b 1 0 1 0 1 0, 1 1	⇔	$(\ldots\ldots)_{10}$
- 0b 0 1 1 1 0 0, 0 1		- $(\ldots\ldots)_{10}$
= 0b ,		= $(\ldots\ldots)_{10}$

Exercice 2 : Une mémoire de 2 k octets est implantée à l'adresse de base 0x2000

- Quelle est, en octet et en bits, la capacité de stockage exacte de cette mémoire ?
- Quelle est l'adresse haute (en hexadécimal) de la dernière case mémoire ?

…

…

La mémoire a une capacité de …… octets = 0x……., comptée de ……. à ……...
L'adresse du premier octet est 0x2000, l'adresse du dernier sera :

…

IV° LES CODES

Pour communiquer entre les différents sous système d'un dispositif automatique ou informatique (par ex : clavier → unité centrale → écran d'un ordinateur), les informations sont « encodées » pour pouvoir être transportées sans erreurs.

IV-1) CODE BCD (BINARY CODED DECIMAL = DECIMAL CODE EN BINAIRE)

Il est principalement utilisé pour la communication Homme Machine (I.H.M. Interface Homme - Machine)

Chaque Chiffre décimal est codé en binaire sur 4 bits :

Préfixe : BCD Ex : 32 = BCD 0011 0010 ≠ 0b 10 0000

Rem : Les combinaisons > 1001 n'existent pas en BCD

IV-2) CODE GRAY (OU CODE BINAIRE REFLECHI)

Le code Gray est dérivé du code binaire naturel (la base 2). Il est basé sur le principe suivant :
Le passage d'un nombre au nombre suivant nécessite le changement d'un seul bit (0→1 ou 1→0).
Le code Gray est donné dans le tableau suivant :

Code décimal	Code Gray	Code binaire naturel	
N	$G_3G_2G_1G_0$	$B_3B_2B_1B_0$	
0	0000	0000	↑ Code BCD
1	0001	0001	
2	0011	0010	
3	0010	0011	
4	0110	0100	
5	0111	0101	
6	0101	0110	
7	0100	0111	
8	1100	1000	
9	1101	1001	↓
10	1111	1010	
11	1110	1011	
12	1010	1100	
13	1011	1101	
14	1001	1110	
15	1000	1111	

Le code Gray est souvent employé dans les tables de Karnaugh, pour éviter les aléas dans les automatismes.

Les codeurs optiques de position utilisent le code Gray pour éliminer les aléas de fonctionnement dus à des défauts d'alignement.

Pour exploiter, pour des calculs, les nombres binaires en code Gray, il est nécessaire de les transcoder en binaire naturel.

Equations de transcodage : cf fiche de TD.

IV-3) CODE ASCII (AMERICAN STANDARD CODE FOR INFORMATION INTERCHANGE)

C'est le code international utilisé pour les échanges d'information entre objets techniques (souris → ordinateur → imprimante…)

Les Lettres, chiffres et caractères sont codés sur 8 bits suivant une table internationale :

Préfixe : ‘ Ex : ‘3 = 0x… = 0b… ‘M = 0x… = 0b…

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

IV-4) LIAISON SERIE ASYNCHRONE RS232:

Les échanges d'information entre objets techniques (souris → ordinateur → imprimante…) sont réalisés sous forme série (un bit à la fois) pour réduire le nombre de conducteur. Une des norme utilisée est le protocole RS232.

IV-4-1) Principe de la sérialisation des informations :

Le mot à émettre, en code ASCII, est émis bit par bit au rythme d'une horloge de transmission.
Des informations supplémentaires peuvent être ajoutées : bit de contrôle (……), de début (……..), de fin (……)…

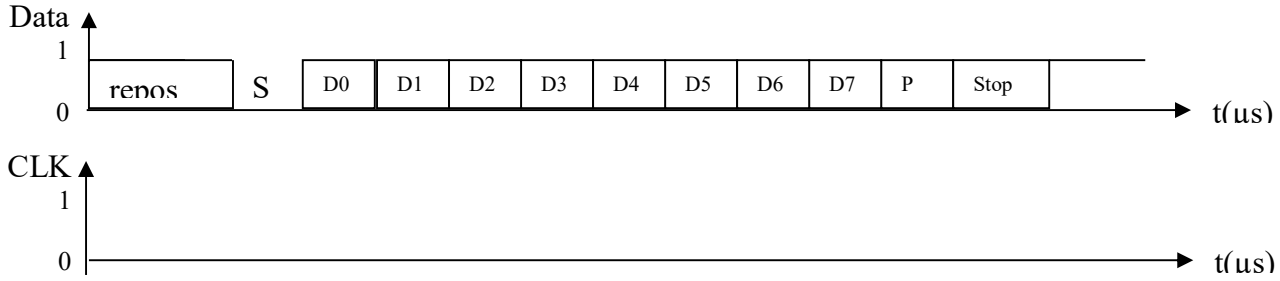
Le débit d'information est exprimé en **……** (bits d'informations par seconde) ex : 9600 Bds

Côté récepteur, à l'apparition d'un bit de Start, les bits sont regroupés par 8 au rythme de l'horloge de réception de même fréquence que celle d'émission pour reconstituer l'octet. Un contrôle de parité peut être effectué.

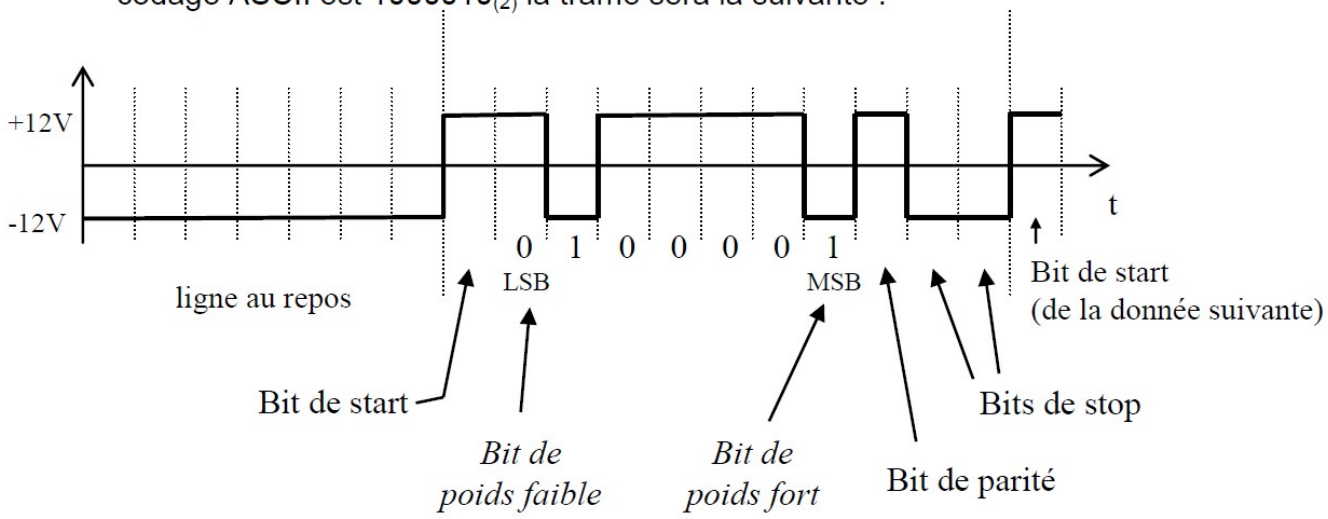
Parité : Un nombre binaire est considéré PAIR si le nombre de bits à "1" qu'il contient est ....., sinon IMPAIR.

Des périphériques intégrés USART (Universal Sérial Asynchronous Receiver & Transmiter) réalisent automatiquement cette fonction.

IV-4-2) Structure du message :



exemple : Soit à transmettre en parité paire, avec 2 bits de stop, le caractère B dont le codage ASCII est $1000010_{(2)}$ la trame sera la suivante :



Côté récepteur, la détection d'une transition 1-0 (**……**) sur la ligne correspond au passage de l'état repos à celui de travail et recale l'horloge de réception sur le bit de start. Les informations sont alors lues au milieu de chaque données.

Le bit de parité « P » indique **……**. Le récepteur recalcule la parité et compare avec le bit reçu. Il accepte le message si les parités correspondent et le refuse dans le cas contraire.

Le(s) bit(s) de « Stop » permet **……**..

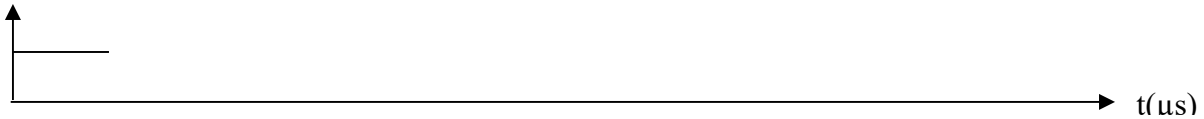
IV-4-3) Ex : On souhaite transmettre le caractère "G" sur une liaison série RS 232:

a) donner les codes Décimal, hexa et binaire du caractère "G":

b) Déterminer sa parité, indiquer la valeur du bit "P" pour une transmission Paire de l'information. Même question pour une transmission Impaire.

c) déterminer le code "binaire inversé" du caractère "G" :

d) Représenter le message émis en transmission 7 bits, avec parité paire à 9600 bauds (indiquer l'échelle)



IV-5) CODE COMPLEMENT A 2:

Le code complément à 2 est utilisé pour représenter les nombres signés suivant le principe que le bit de poids le plus fort (MSB) représente le signe et qu'un nombre + son opposé vaut 0.

Par convention : le signe + est noté 0 ;
Le signe – est noté 1 ;
0 est considéré positif et s'écrit 0b0000 0000

Ex : sur 8 bits : 0b 0 0 0 1 0 0 0 1 = . . . 0b 1 0 0 1 0 0 0 1 < 0 $\neq$ - 17

Représentation de – 17 : on applique la règle + 17 + (– 17) = 0 donc (-17) = 0 - 17

0b 0 0 0 0 0 0 0 0 = 0
\- 0b 0 0 0 1 0 0 0 1 = 17
= 0b 1 1 1 0 1 1 1 1 = - 17

Vérification :
0b 0 0 0 1 0 0 0 1 = 17
\+ 0b 1 1 1 0 1 1 1 1 = - 17
0b 1 0 0 0 0 0 0 0 0 = 0

Rem : la représentation est limitée à 8 bits le 9ème n'est pas pris en considération !

Règle : pour déterminer l'opposé d'un nombre signé on le complémente à 2 pour cela on complémente le nombre bit à bit (y compris le bit de signe) et on ajoute 1

Ex : + 17 = 0b 0 0 0 1 0 0 0 1 → complément 1 = 0b . . . . . → + 1 = 0b . . . . . < 0

Ex : - 17 = 0b 1 1 1 0 1 1 1 **1** → complément 1 = 0b . . . . . → + 1 = 0b . . . . . > 0

Avantage : les soustractions se transforment en addition et complémentation (plus simple pour un calculateur).

Rem : Sur 8 bits le plus grand nombre positif s'écrit : 0b 0111 1111 et vaut 127

Il y a 128 nombres positifs de 0 à +127

Sur 8 bits le plus grand nombre négatif s'écrit : 0b 1000 0000 et vaut -128

Il y a 128 nombres négatifs de -1 à -128

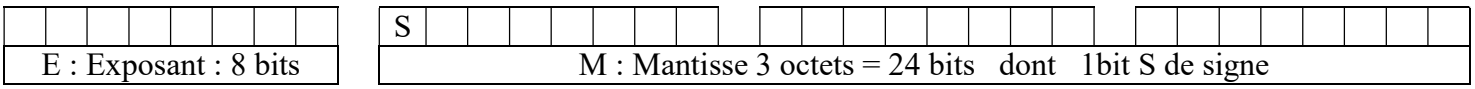
+127	0b 0111 1111
…	…..
+1	0b 0000 0001
0	0b 0000 0000
-1	0b 1111 1111
….	……
-127	0b 1000 0001
-128	0b 1000 0000

IV-6) CODAGE DES NOMBRES EN VIRGULE FLOTTANTE:

Dans un ordinateur, la représentation des nombres réels utilise le codage en « virgule flottante » sur 4 octets avec Exposant (E), Mantisse (M) et Signe (S).

Un nombre N est alors le résultat de l'expression : $N = (-1)^S \times 2^{E-129} \times [(M / 2^{23}) + 1]$

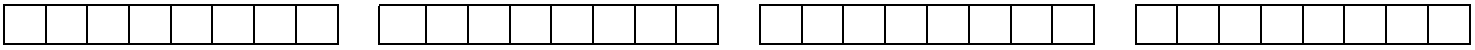
Représentation binaire :



Exemple : déterminer la valeur décimale du nombre représenté par les 4 octets 0x82 0x49 0x0F 0xDB

Représentation binaire :

S = … donc



E = … = … M = …x16^5 + …x16^4 + …x16^3 + …x16^2 + ...x16^1 + …

M = …

$N = (-1)^{\cdots} \times 2^{\cdots\cdots-129} \times [(. . . . . /2^{23})+1]$

N = … x … x (…….. + 1)

N = … x … = ….. = …

V) Travaux dirigés:

V-1) Conversions de base : Convertir les nombres suivants dans les bases demandées:

Base décimale (10)	Base Hexadécimale (16)	Base binaire (2)	Décimal Codé Binaire
514			
	0x CBA		
		0b 1000 1111 1010	
			BCD 0001 0110 1001
25,2			

V-2) Opérations: Soient L, K, J, trois nombres binaires codés sur 8 bits:
L = 0b 1010 1010 K = 0b 0101 0101 J = 0b 0101 0110

V-2-1) Réaliser les opérations: L + K ; J + L Donner les résultats sur 8 bits sans tenir compte de la dernière retenue éventuelle. Exprimer en base 10.

V-2-2) Donner la valeur de J en base 10. En considérant le résultat précédant, et en supposant que J est un nombre positif, quel est le signe de L ? Quelle est sa valeur ?

V-2-3) On pose M = 0b 1010 1011 = - 85. Donner la valeur binaire sur 8 bits du nombre X valant 85 en décimal.
Contrôler que M + X = 0.

V-2-4) Réaliser les opérations: 2 x K. Contrôler en décimal. Réaliser l'opération L/2 . Contrôler en décimal. En déduire une règle simple de multiplication et de division par la base.

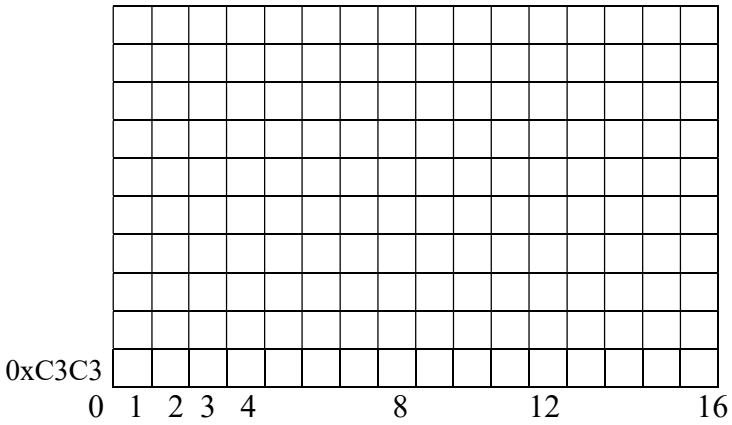
V-3) Codage Bitmap des images :

Le codage Bitmap d'une image consiste à coder la couleur de chaque pixel (point) de l'image par un nombre binaire de longueur fixée. Par ex pour représenter une image en noir et blanc, il suffit d'un seul bit (0= blanc ; 1 = noir) Les formats les plus fréquents codent chaque pixel sur 1bit (2 couleurs), 4 bits (16 couleurs), 8 bits (256 couleurs) et 24 bits (16 millions de couleurs)

Une image noir et blanc de 10 lignes de 16 pixels est codée en bitmap par la suite hexadécimale ci-dessous :

0xC3C3 85A1 0990 1188 0180 07E0 0180 0240 8241 C183

Dessiner l'image du bas vers le haut :



V-4) Sécurité des informations :

Pour échanger des informations numériques entre 2 ordinateurs, on transmet en série les codes ASCII sur 7 bits des caractères à émettre. Afin de sécuriser les échanges on ajoute, en fin d'octet, à l'émission, un 8em bit dit "de parité", de façon à rendre les octets transmis tous Pairs ou tous Impairs. A la réception, on contrôle la parité pour détecter une éventuelle erreur de transmission.

Nombre Pair : le nombre de « 1 » dans l'octet est pair.

Nombre Impair : le nombre de « 1 » dans l'octet est impair.

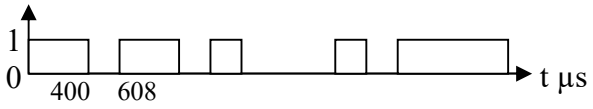
On relève la séquence suivante lors d'une transmission en parité Paire :

0x C4 DE DD D4 DF EB E4

V-4-1) détecter les valeurs défectueuses

V-4 -2) reconstituer le message transmis (cf table des codes ASCII)

V-4-3) On observe la trame asynchrone logique ci-dessous : indiquer dans quel format elle a été émise :



a) 8 bits de donnée, sans parité
b) 7 bits de donnée, parité paire
c) 8 bits de donnée, parité impaire
d) 8 bits de donnée, parité paire
e) 9,6 kbauds
f) 19,2 kbauds
g) 4800 bauds
h) 2400 bauds

V-4-4) Quelle est la durée de transmission minimale, sur une ligne RS232, d'un fichier de caractères ASCII de 10 kOctet à 19,6 kbits/s ? format 8 bits, 1stop, parité paire

V-5) Capacités mémoire :

Une image de 4 Méga octets est stockée sur une clé USB d'1 Go à partir de l'adresse 0x 10C0 0000.
Quelle est l'adresse de fin de stockage de cette image ?

En considérant que tout l'espace mémoire de la clé est utilisé entre 0x0000 0000 (adresse de base de la clé) et l'adresse de fin de stockage de l'image calculée ci-dessus, calculer l'espace libre restant sur la clé.

I Conversions de base : Convertir les nombres suivants dans les bases demandées:

Base décimale (10)	Base Hexadécimale (16)	Base binaire (2)	Décimal Codé Binaire
513			
	0x ABC		
		0b 1100 1011 1110	
			BCD 0101 0100 1000
37,2			

II) Opérations: Soient L, K, J, trois nombres binaires codés sur 8 bits:
L = 0b 1010 1010 K = 0b 0101 0101 J = 0b 0101 0110

II-1) Réaliser les opérations: L + K ; J + L Donner les résultats sur 8 bits sans tenir compte de la dernière retenue éventuelle. Exprimer en base 10.

II-2) Donner la valeur de J en base 10. En considérant le résultat précédant, et en supposant que J est un nombre positif, quel est le signe de L ? Quelle est sa valeur ?

II-3) On pose M = 0b 1010 1011 = - 85. Donner la valeur binaire sur 8 bits du nombre X valant 85 en décimal. Contrôler que M + X = 0.

II-4) Réaliser les opérations: 2 x K. Contrôler en décimal. Réaliser l'opération L/2 . Contrôler en décimal. En déduire une règle simple de multiplication et de division par la base.

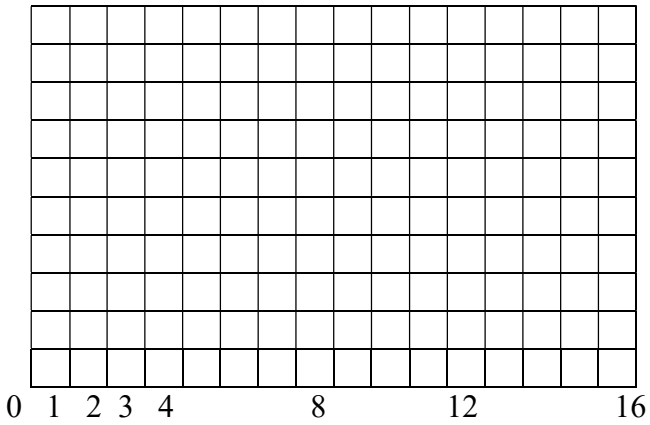
III) Codage Bitmap des images :

Le codage Bitmap d'une image consiste à coder la couleur de chaque pixel (point) de l'image par un nombre binaire de longueur fixée. Par ex pour représenter une image en noir et blanc, il suffit d'un seul bit (0= blanc ; 1 = noir) Les formats les plus fréquents codent chaque pixel sur 1bit (2 couleurs), 4 bits (16 couleurs), 8 bits (256 couleurs) et 24 bits (16 millions de couleurs)

Une image noir et blanc de 10 lignes de 16 pixels est codée en bitmap par la suite hexadécimale ci-dessous :

0x0000 7FFE 4182 5998 5998 4182 7FFE 3188 1B08 0E00

Dessiner l'image du bas vers le haut :



IV) Sécurité des informations :

Pour échanger des informations numériques entre 2 ordinateurs, on transmet en série les codes ASCII sur 7 bits des caractères à émettre. Afin de sécuriser les échanges on ajoute, en fin d'octet, à l'émission, un 8em bit dit "de parité", de façon à rendre les octets transmis tous Pairs ou tous Impairs. A la réception, on contrôle la parité pour détecter une éventuelle erreur de transmission.

Nombre Pair : le nombre de « 1 » dans l'octet est pair.

Nombre Impair : le nombre de « 1 » dans l'octet est impair.

On relève la séquence suivante lors d'une transmission en parité Paire :

0x C4 DE DD D4 DF EB E4

IV-1) détecter les valeurs défectueuses

IV -2) reconstituer le message transmis (cf table des codes ASCII)

V) Capacités mémoire :

Une image de 4 Méga octets est stockée sur une clé USB d'1 Go à partir de l'adresse 0x 10C0 0000.
Quelle est l'adresse de fin de stockage de cette image ?

En considérant que tout l'espace mémoire de la clé est utilisé entre 0x0000 0000 (adresse de base de la clé) et l'adresse de fin de stockage de l'image calculée ci-dessus, calculer l'espace libre restant sur la clé.

Logique combinatoire

Algèbre de Boole

Logique combinatoire

Table des matières

I Variables et fonctions binaires **23**
I.1 Définition et état d'une variable binaire **23**
I.2 types de logique **23**
Logique combinatoire **23**
Logique séquentielle **23**
I.3 représentation des fonctions binaires **23**
I.4 Les quatre fonctions à une variable 26
I.5 Les 16 fonctions à 2 variables 27
II propriétés des opérations logiques **30**
II.1 élément neutre **30**
II.2 Commutativité **30**
II.3 Associativité **30**
II.4 Distributivité **30**
II.5 Idem potence 30
II.6 Complémentarité **30**
II.7 Involution **30**
II.8 Produit logique avec le terme 0 ..... **31**
II.9 Somme logique avec le terme 1 ..... **31**
II.10 Absorption **31**
II.11 Inclusion **31**
II.12 Exclusion **31**
III dualité ou théorème de DEMORGAN **31**
IV formes canoniques **32**
IV.1 Table de vérité **32**
IV.2 Définitions **32**
IV.3 Première forme canonique **32**
IV.4 Deuxième forme canonique **32**
V Simplifications algébriques **32**
V.1 Mise en facteur **32**
VI simplification graphique **33**
VI.1 Matrice de Karnaugh **33**
VII Travaux dirigés **33**

Méthodes de résolution, simplification des fontions logiques **41**

Aléas **55**

Convertion de fontions logiques en NOR ou en NAND **60**

Exercices **66**

I. variables et fonctions binaires

I.1) définition et état d'une variable binaire

Le mathématicien, logicien Georges Boole (1815-1864) est le père fondateur de la logique moderne. Il a créé une algèbre binaire, dont les valeurs ne peuvent être que 0 ou 1. De nombreux dispositifs physique fonctionnent en tout ou rien, c'est à dire en deux états qui peuvent être selon les technologies utilisées :

Marche / arrêt ; Ouvert / fermé ; Vrai / Faux ; Conduction / Blocage ; High / Low…

C'est le système binaire. Une variable binaire peut donc prendre deux valeurs : **0 ou 1.**

I.2) types de logique

I.2.a) Logique combinatoire :

En logique combinatoire, l'état de sortie ne dépend que de l'état des entrées.

I.2.b) Logique séquentielle :

En logique séquentielle, l'état de sortie dépend de l'état des entrées mais également de l'état antérieur de la sortie.

I.3) représentation des fonctions binaires

Dans le chapitre suivant, nous représenterons la fonction OU dont la valeur de sortie est à 1 si l'entrée a OU l'entrée b est à la valeur logique 1.

$$\{a, b \in (0, 1) ; \quad S \in (0, 1)\}$$

I.3.a) Table de Vérité :

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

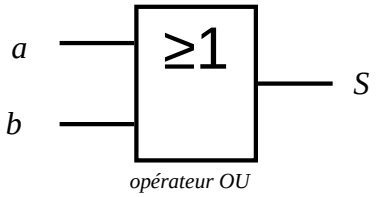
Généralisation :

Pour n variables, on a : **n + 1 colonnes, 2^n lignes.** A chaque ligne correspond une des 2^n combinaisons possibles des n variables suivie de la valeur de la fonction.

I.3.b) Logigramme :

Il s'agit d'une représentation schématique.

On utilise des schémas blocs pour les fonctions utilisées.



opérateur OU

$S = a + b$

I.3.c) équation algébrique :

C'est l'écriture sous la forme d'une somme de produits. On ajoute le produit des variables d'entrée pour chacune des lignes de la table de vérité où la sortie est à 1.

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

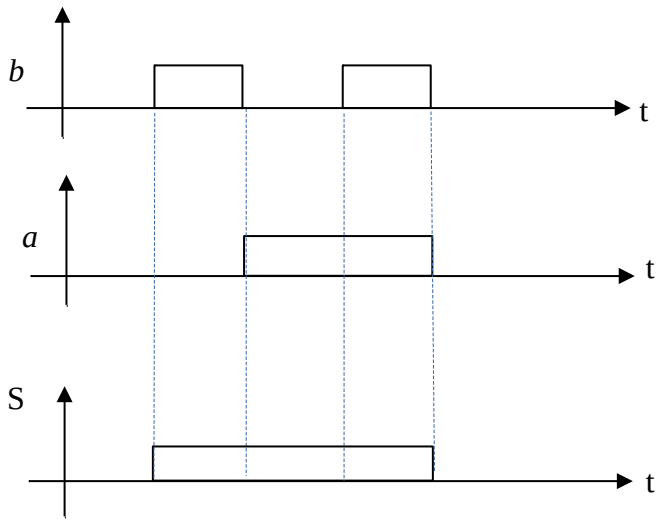
I.3.d) table de Karnaugh :

On représente la fonction en indiquant dans chaque case **la valeur de la sortie** correspondant à chaque **combinaison des variables d'entrée** :

S	$\bar{a}$	a
$\bar{b}$	0	1
b	1	1

I.3.e) diagramme des états :

C'est une représentation temporelle de la table de vérité :

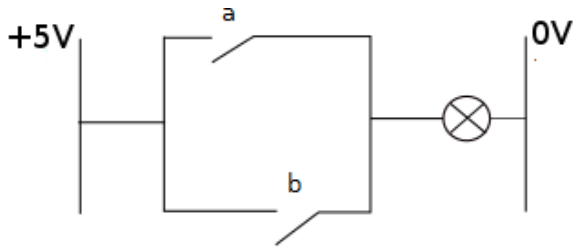


I.3.f) schéma électrique :

C’est une représentation sous forme de schéma développé :

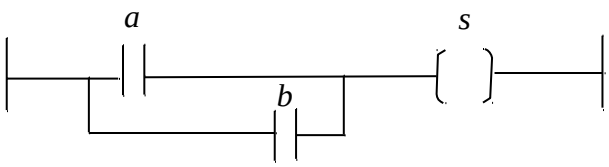
Ici la lampe s'allume si :

- on ferme a OU b.
- on ferme a ET b.

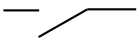
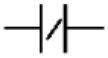


I.3.g) Ladder (langage pour automate programmable industriels) :

Les programmes sont créés à partir d'un logiciel puis téléverser vers un automate.



On représente les interrupteurs de la façon suivante :

	Symbole logique	Symbole électrique	Symbole électrique horizontal	Symbole électrique vertical	Symbole Ladder	Fonction recherchée
Normalement ouvert	a					Le courant passe si l'interrupteur est actionné.
Normalement fermé	$\bar{a}$ ou /a					Le courant passe si l'interrupteur est à l'état de repos.

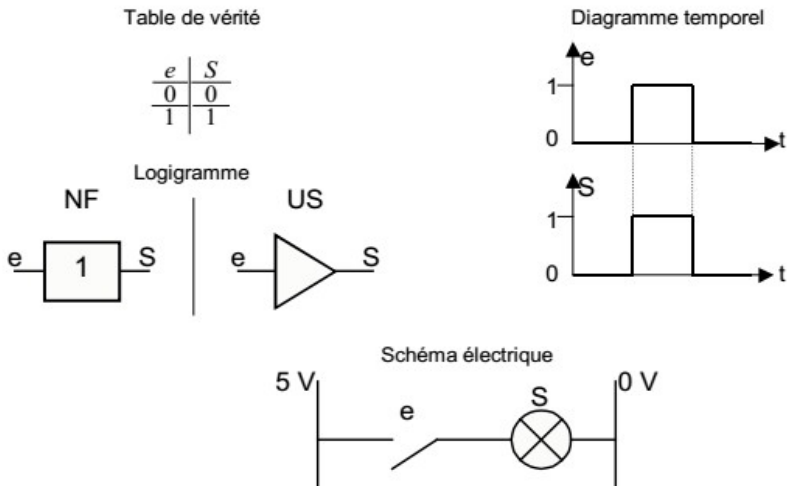
I.4) les quatre fonctions à une variable :

a	F_0	F_1	F_2	F_3
0	**0**	**0**	**1**	**1**
1	**0**	**1**	**0**	**1**

Ici, ***a*** peut prendre deux valeurs différentes (0, 1). Il existe donc $2^2=4$ fonctions dont le résultat correspond aux combinaisons des valeurs (0, 1) de la sortie.

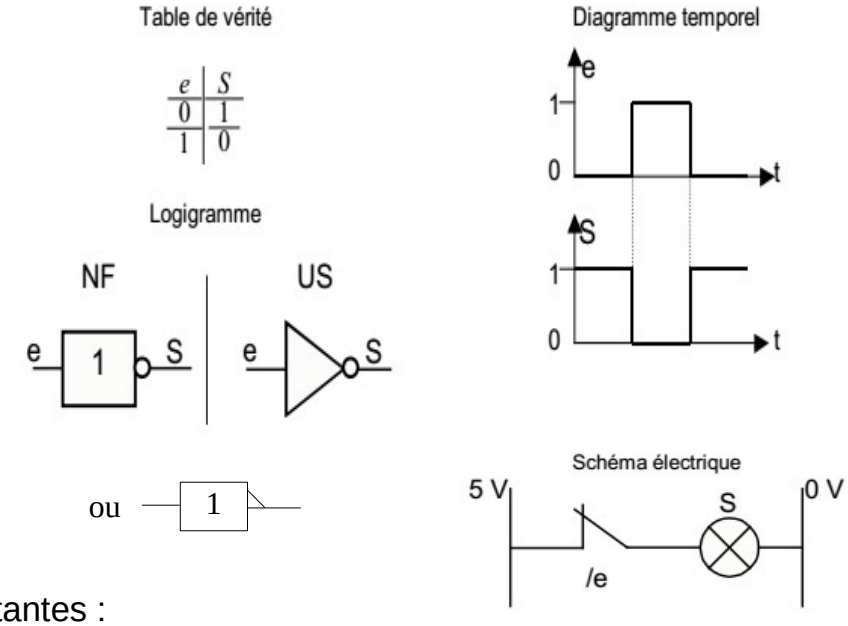
I.4.a) la fonction OUI

Elle recopie l'état de *a*



I.4.b) la fonction NON

Elle inverse l'état



I.4.c) les fonctions constantes :

Pour tout état de *a :*

- $F_0=0$ c'est la fonction NULLE.
- $F_3=1$ C'est la fonction UNITE.

I.5) les 16 fonctions à 2 variables :

b	a	F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Ici, a peut prendre deux valeurs différentes (0, 1). Il existe donc $2^4=16$ fonctions dont le résultat correspond aux combinaisons des valeurs (0, 1) de la sortie.

I.5.a) les fonctions constantes :

$F_0(b,a)=0$ $\quad$ $F_{15}(b,a)=1$

I.5.b) les fonctions à une variable :

$F_3(b,a)=b$ $\quad$ $F_{12}(b,a)=\overline{b}$

I.5.c) la fonction ET (produit logique) (commuative/associative):

$F_1(b,a)=b.a$

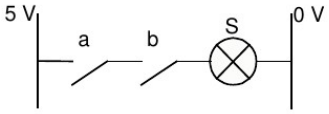


schéma électrique

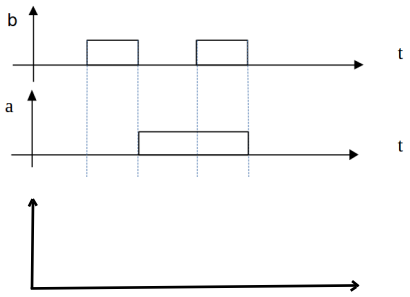
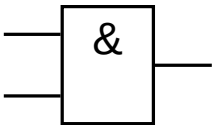


diagramme temporel



logigramme fr

logigramme US

b / a	0	1
0		
1		

Table de vérité

I.5.d) fonction NON ET (NAND) : (commutative / NON associative) :

$F_{14}(b,a)=\overline{b.a}$

schéma électrique

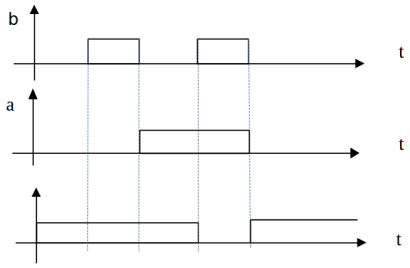
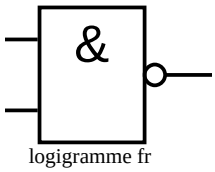


diagramme temporel



logigramme fr

logigramme US

b / a	0	1
0		
1		

I.5.e) la fonction OU (commutative / associative) $F_7 = b + a$

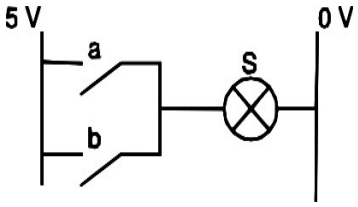


schéma électrique

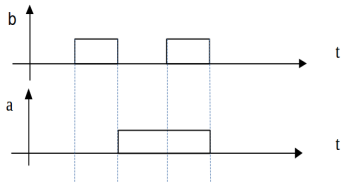
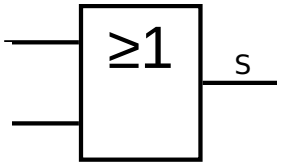


diagramme temporel



logigramme fr

logigramme US

b / a	0	1
0		
1		

I.5.f) la fonction NON OU (NI) (NOR): (commutative / NON associative) $F_{14} = \overline{b + a}$

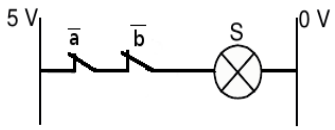


schéma électrique

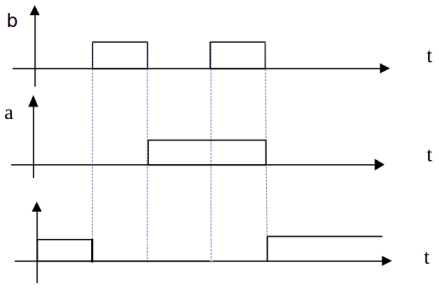
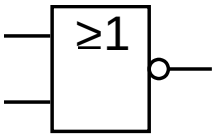


diagramme temporel



logigramme fr

logigramme US

b / a	0	1
0		
1		

I.5.g) la fonction OU EXCLUSIF (commutative / associative) $F_6(b,a) = b.\bar{a} + \bar{b}.a = a \oplus b$

schéma électrique

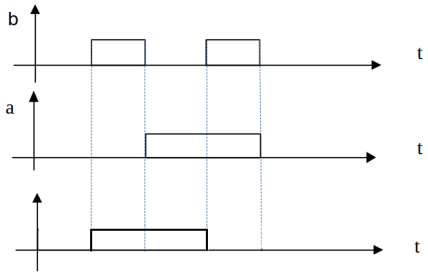
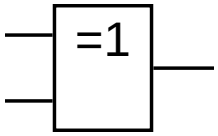
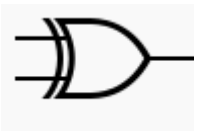


diagramme temporel



logigramme fr



logigramme US

b / a	0	1
0		
1		

I.5.h) la fonction EGALITE (NON OUEX) (commutative / associative)

$$F_9(b,a)=b\ a+\bar{b}.\bar{a}$$

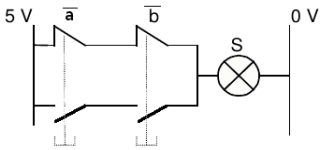


schéma électrique

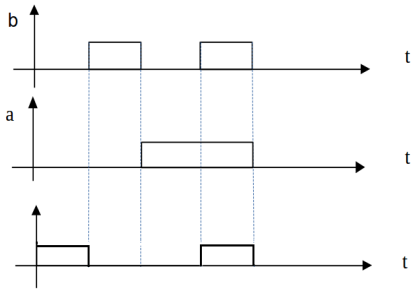
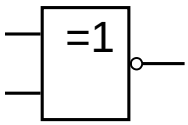
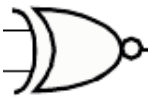


diagramme temporel



logigramme fr



logigramme US

b / a	0	1
0		
1		

II. propriétés des opérations logiques.

II.1) élément neutre.

Ce sont les éléments qui n'ont aucune incidence sur le résultat.

- Pour l'opération OU, il s'agit de 0 : a + 0 = a.
- Pour l'opération ET, il s'agit de 1 : a . 1 = a.

II.2) commutativité.

Propriété vraie pour toutes les opérations (également Nand et Nor).

$a+b=b+a \qquad a.b=b.a$

II.3) associativité.

$a+(b+c)=(a+b)+c$

II.4) distributivité

$a.(b+c)=a.b+a.c$
$a+(b.c)=(a+b).(a+c)$

II.5) idem potence

$a+a=a$ $\qquad\qquad$ $a.a=a$

Attention : $a \oplus a=0$ XOR
$a \downarrow a=\bar{a}$ NOR
$a/a=\bar{a}$ NAND

II.6) complémentarité.

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

II.7) involution.

$\bar{\bar{a}}=a$

II.8) produit logique avec le terme 0.

$a.0=0$

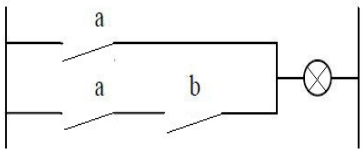
II.9) somme logique avec le terme 1.

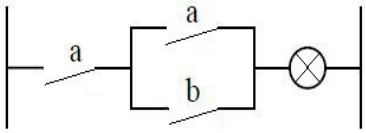
$a+b+c+1=1$

II.10) absorption.

$a+a.b=a$ **;** Le b est absorbé.

Vérification :





a . (a + b) = a

II.11) inclusion

Appelé également redondance.

$a.b+\bar{b}.c=a.b+\bar{b}.c+a.c$

II.12) exclusion

$a+\bar{a}.b=a+b$

III. dualité ou théorème de De MORGAN

Pour deux variables : $\overline{a+b}=\bar{a}.\bar{b}$

Pour trois variables : $\overline{a+b+c}=\bar{a}.\bar{b}.\bar{c}$ (idem pour n entrées)

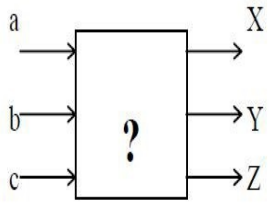
On a aussi $\overline{a.b}=\bar{a}+\bar{b}$

Remarque : Dual signifie complément.

IV. formes canoniques.

IV.1) table de vérité.

Rappel : Pour traduire l'énoncé d'une équation logique, on utilise une table de vérité. Exemple.



c	b	a	X	Y	Z
0	0	0	0	0	0
0	0	1	0	1	1
0	1	0	0	1	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	1	1
1	1	0	0	0	1
1	1	1	0	0	1

IV.2) définitions.

Minterme : C'est une intersection (fonction ET) où figure toutes les variables de la fonction.

Exemple : $\bar{a}.b.c$

Maxterme : C'est une réunion (fonction OU) où figure toutes les variables de la fonction.
Exemple : $a+\bar{b}+c$

IV.3) première forme canonique.

C'est la fonction écrite sous la forme d'une réunion de mintermes.

Exemple : $X=a.b.\bar{c}$

$Y=(a.\bar{b}.\bar{c})+(\bar{a}.b.\bar{c})+(a.\bar{b}.c)$

$Z=$

IV.4) deuxième forme canonique.

C'est la fonction écrite sous la forme d'une intersection de maxtermes. On obtient cette forme après manipulations.

Exemple pour Z :

$\bar{Z}=(\bar{a}.\bar{b}.\bar{c})+(\bar{a}.\bar{b}.c)$

$\bar{\bar{Z}}=$

$\bar{\bar{Z}}=$

$Z=(a+b+c).(a+b+\bar{c})$

Même chose pour Y :

$Y = a.\bar{b}.\bar{c} + \bar{a}.b.\bar{c} + a.\bar{b}.\bar{c}$

$\bar{Y} =$

$\bar{Y} =$

$Y =$

$\overline{\overline{Y}} = (\bar{a}+b+c).(a+\bar{b}+c).(\bar{a}+b+\bar{c})$

V. Simplifications algébriques.

Mise en facteur.

$X = \bar{a}.b.\bar{c} + \bar{a}.b.c + a.\bar{b}.c$ ($\bar{a}.b$) en facteur

$X = \bar{a}.b.(c + \bar{c}) + a.\bar{b}.c$

$X = \bar{a}.b + a.\bar{b}.c$

$Y = a.b.c + \bar{a}.b.c + \bar{a}.\bar{b}.c + a.b.\bar{c}$

$Y = a.b.(c + \bar{c}) + \bar{a}.c.(b + \bar{b})$

$Y = a.b + \bar{a}.c$

VI. simplification graphique

VI.1) matrice de karnaugh

VI.1.a) Présentation

L'objectif est ici d'obtenir l'équation logique qui relie la grandeur de sortie au nombre minimal de variables d'entrées. Cela permet de minimiser le nombre de portes logiques à utiliser dans un circuit électronique ou en programmation.

Cette simplification est effectuée par l'utilisation d'un tableau qui est une application de la théorie des ensembles.

Si n est le nombre de variable que comporte l'équation à résoudre, le tableau devra comporter 2^n cases.

Pour 2 variables, 4 cases :

b \ a	0	1
0	?	?
1	?	?

Pour 3 variables, 8 cases :

c \ ab	0 0	0 1	1 1	1 0	0 0
0	?	?	?	?	
1	?	?	?	?	

Pour passer d'une case à l'autre, une seule variable change d'état (code de Gray).

On note également qu'il doit exister une continuité dans le code appliqué, une seule variable doit changer d'état pour boucler le tableau.

VI.1.b) utilisation

Après avoir rempli le tableau des valeurs du système à simplifier, on effectue des regroupements.

ab \ cd	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	0	1	1	1
10	0	1	1	0

Ici, nous pouvons remarquer que le résultat est 1 quelque soit la valeur de c.

Ici, nous pouvons remarquer que le résultat est 1 quelque soit la valeur de b et c.

La simplification va être d'autant plus importante que le nombre de cases adjacentes de valeur 1 est important.

ab \ cd	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	0	1	1	1
10	0	1	1	0

Ici, nous pouvons remarquer que le résultat est 1 quelque soit la valeur de a et d.

Ici, nous pouvons remarquer que le résultat est 1 quelque soit la valeur de a, b et c.

La solution la plus simple de cet exemple est donc la somme de produits : $\underline{Y = d + b.c}$

Le tableau se simplifie selon les règles suivantes :

- Il faut regrouper uniquement les valeurs égales à 1 par groupe de puissance de 2.
- Le même 1 peut faire partie de plusieurs groupes différents. Les groupements peuvent être non adjacents mais symétriques (exercice 2). Si un état n'est pas défini, lui donner la valeur qui permet un regroupement.

VI.1.c) exemple :

Donner les simplifications des tableaux suivants :

ab \ c	**0 0**	**0 1**	**1 1**	**1 0**
0	**0**	**1**	**1**	**1**
1	**0**	**1**	**1**	**0**

abc \ d	000	001	011	010	110	111	101	100
0	1	0	0	1	1	0	0	1
1	1	0	0	1	1	1	1	1

VI.1.d) gestion des aléas

Lors de la simplification, si l'on ne peut pas effectuer un recoupement des groupes, un aléa de fonctionnement a lieu. Ce qui doit être évité bien évidemment, exemple :

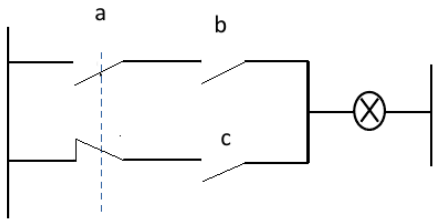
Soit la fonction $F = a \,.\, b \,.\, \bar{c} + a \,.\, b \,.\, c + \bar{a} \,.\, \bar{b} \,.\, c + \bar{a} \,.\, b \,.\, c$

Cette fonction peut se simplifier. Nous pouvons utiliser les règles algébriques vues au chapitre V, il est également possible d'utiliser le tableau de Karnaugh :

c \ ab	0 0	0 1	1 1	1 0
0	0	0	1	0
1	1	1	1	0

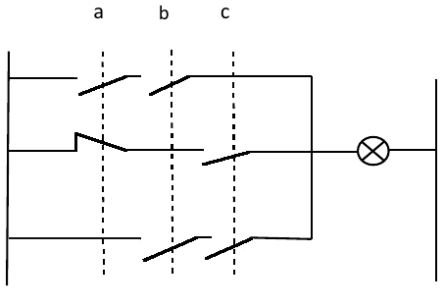
$F = a \,.\, b + \bar{a} \,.\, c$

L'aléa apparaît car $a + \bar{a} = 1$ n'est valable qu'en régime permanent. Dans la réalité, il existe une phase transitoire où a n'a pas encore basculé.



Pour éviter ce dysfonctionnement, il convient d'effectuer un regroupement supplémentaire :

c \ ab	0 0	0 1	1 1	1 0
0	0	0	1	0
1	1	1	1	0



L'aléa est ainsi supprimé.

I Représentations :

I-1 : Quelle représentation utilise-t'on pour représenter des fonctions logiques sous forme de schéma ?

a) la forme équationnelle b) la table de vérité c) le tableau de Karnaugh d) le logigramme

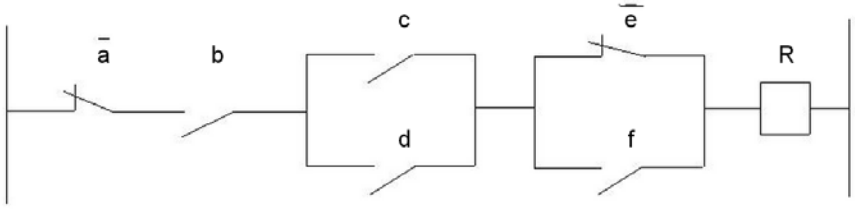
I-2 : Quelle est la fonction logique décrite de la manière suivante : dans sa table de vérité à trois variables (a, b et c) la colonne S ne possède qu'un seul 1, quand les trois variables sont à 0.

a) fonction OU b) fonction ET c) fonction NOR d) fonction NAND

I-3 : Quelle est la fonction décrite ici : dans sa table de vérité à trois variables (a, b et c), la colonne S ne possède qu'un seul 0, quand les trois variables sont à 1.

a) fonction OU b) fonction ET c) fonction NOR d) fonction NAND

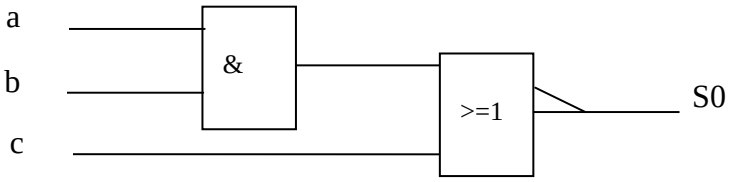
I-3 : Traduire le schéma à contacts sous forme d'équation logique :



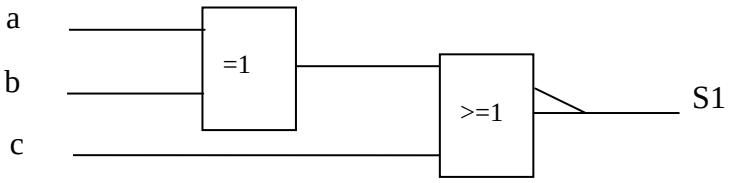
I-4 : On donne S = a + $\overline{b.c}$, représenter le schéma à contacts et le logigramme.

II Logigrammes

II-1) soient les logigrammes suivant:

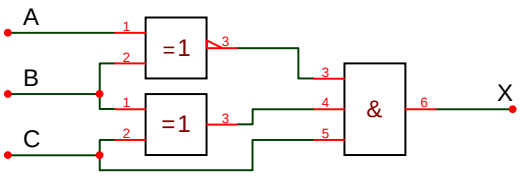


II-1-1 donner l'équation logique de S0 en fonction de a, b, c



II-1-2 donner l'équation logique de S1 en fonction de a, b, c

II-2) Vrai ou faux ?
On considère le logigramme suivant:



A) Pour que X= 0 il suffit que C = 0. -
B) Pour que X= 1 il faut que C = 1 et A et B soient identiques.
C) Pour que X = 0 il suffit que A et B soient identiques.
D) Pour que X = 0 il suffit que A et B soient différents. -
E) Pour que X = 1 il faut que C = 1, B = 0 et A = 1.

III Etude d'un convertisseur GRAY-BINAIRE naturel

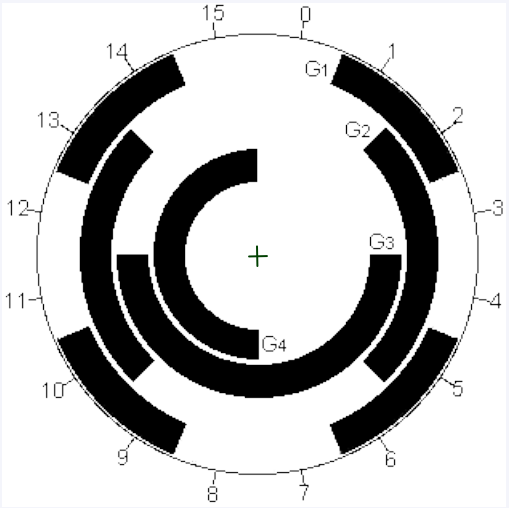
Le code Gray est fréquemment utilisé dans les capteurs angulaires ou de positionnement, mais aussi lorsque l'on désire une progression numérique binaire sans parasite transitoire.

Code binaire réfléchi ou code Gray : Le code Gray est un code construit de telle façon qu'à partir du chiffre 0 chaque nombre consécutif diffère du précédent immédiat d'un seul digit. En l'exprimant autrement nous pouvons également dire que l'on change un seul bit à la fois quand un nombre est augmenté d'une unité.

La correspondance du code Gray pour les nombres de 0 à 15, en binaire est donnée par le tableau suivant :

Code Gray				Nombre décimal	Code binaire pur			
G_4	G_3	G_2	G_1		B_4	B_3	B_2	B_1
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1
0	0	1	1	2	0	0	1	0
0	0	1	0	3	0	0	1	1
0	1	1	0	4	0	1	0	0
0	1	1	1	5	0	1	0	1
0	1	0	1	6	0	1	1	0
0	1	0	0	7	0	1	1	1
1	1	0	0	8	1	0	0	0
1	1	0	1	9	1	0	0	1
1	1	1	1	10	1	0	1	0
1	1	1	0	11	1	0	1	1
1	0	1	0	12	1	1	0	0
1	0	1	1	13	1	1	0	1
1	0	0	1	14	1	1	1	0
1	0	0	0	15	1	1	1	1

Exemple de roue codeuse Gray :



On code les 16 positions angulaires par 4 bits G4 … G1. La lecture s'effectue par 4 cellules photosensibles.

Le calculateur qui reçoit l'information issue de la roue codeuse ne comprend que le binaire naturel, il est donc nécessaire de réaliser une conversion code Gray (G4 … G1) → Code Binaire pur (B4 … B1)

III-1) Etablir les diagrammes de Karnaugh pour B1, B2, B3 et B4 à partir de G1, G2, G3, G4. (sur page suivante)

III-2) Etablir les équations logiques de B1, B2, B3 et B4 en fonction de G1, G2, G3, G4

III-3) Etablir le logigramme

III-4) Généraliser pour 8 bits :

diagramme de Karnaugh pour B4 : Equation de B4 :

G4 G3 \ G1 G2	0 0	1 0	1 1	0 1
0 0				
0 1				
1 1				
1 0				

diagramme de Karnaugh pour B3 : Equation de B3 :

G4 G3 \ G1 G2	0 0	1 0	1 1	0 1
0 0				
0 1				
1 1				
1 0				

diagramme de Karnaugh pour B2 : Equation de B2 :

G4 G3 \ G1 G2	0 0	1 0	1 1	0 1
0 0				
0 1				
1 1				
1 0				

diagramme de Karnaugh pour B1 : Equation de B1 :

G4 G3 \ G1 G2	0 0	1 0	1 1	0 1
0 0				
0 1				
1 1				
1 0				

Logigramme :

IV Extrait synthèse 2014

Le schéma de câblage électrique de la sortie S en fonction des variables d'entrée D, C, B, A est donné ci-dessous :

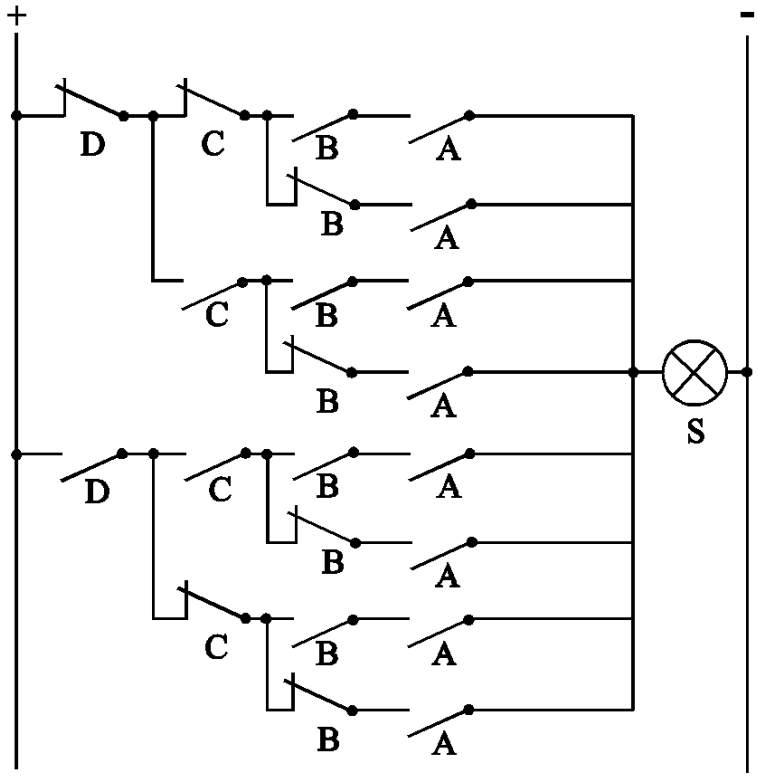


Schéma de câblage électrique de la sortie S

1. Déterminer l'équation logique de la sortie S en fonction des entrées D, C, B, A.

2. À l'aide d'un tableau de Karnaugh, simplifier l'équation logique de S.

3. De l'équation simplifiée de S que vous aurez trouvée, en déduire le nouveau schéma de câblage électrique de S.

METHODES DE RESOLUTION DES PROBLEMES DE LOGIQUE COMBINATOIRE SIMPLIFICATION DES FONCTIONS LOGIQUES

IV-1 GENERALITES

Un problème de logique combinatoire peut être représenté sous forme d'un tableau des valeurs dans lequel figurent les valeurs 0 et 1 que doit prendre la (ou les) sortie pour chaque combinaison possible des variables d'entrée.

Si certains cas de fonctionnement ne sont pas envisagés ou ne peuvent technologiquement exister, on notera l'absence de valeur affectée à la (ou les) grandeur de sortie par le symbole Ø (cas indifférent ou impossible).

Une représentation équivalente peut être obtenue à l'aide des matrices de Karnaugh.

IV-2 ETUDE D'UNE FONCTION LOGIQUE CONNAISSANT SON TABLEAU DES VALEURS OU SA MATRICE DE KARNAUGH

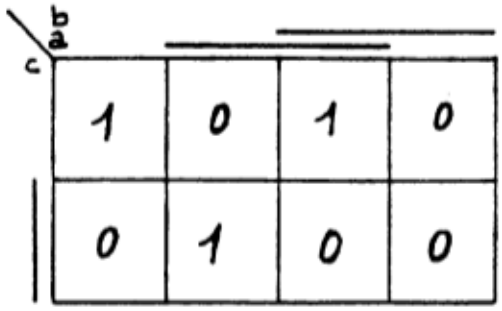
Il s'agit à partir de ce tableau ou de cette matrice de trouver l'équation algébrique de la fonction, d'en établir le logigramme correspondant ainsi que le schéma électrique équivalent.

IV-21 Forme canonique disjonctive de la fonction

Soit F(a,b,c) une fonction logique de 3 variables binaires indépendantes a, b, c, définie par le tableau des valeurs ou

la matrice de Karnaugh ci-dessous :

c	b	a	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0



On constate que la fonction F est égale à 1 pour 3 combinaisons distinctes des variables a,b,c.

Dans le premier cas :

F = 1 quand on a simultanément les valeurs : a = 0; b = 0; c = 0, ou ce qui revient au même quand : $\bar{a} = 1$; $\bar{b} = 1$; $\bar{c} = 1$.

En algèbre logique,ce cas est représenté par l'intersection canonique $\bar{a}.\bar{b}.\bar{c}$

De même le deuxième cas est représenté par $a.b.\bar{c}$,et le troisième par $a.\bar{b}.c$.

On a donc :

F = 1 quand $(\bar{a} = 1)$ET$(\bar{b} = 1)$ET$(\bar{c} = 1)$ OU $(a = 1)$ET$(b = 1)$ET$(\bar{c} = 1$

OU $(a = 1)$ET$(\bar{b} = 1)$ET$(c = 1)$

En algèbre logique binaire cette expression s'écrit :

$$F = \bar{a}.\bar{b}.\bar{c} + a.b.\bar{c} + a.\bar{b}.c$$

Cette équation prend quelle que soit la combinaison des variables binaires envisagée la même valeur 0 ou 1 que la fonction F définie par le tableau ou la matrice précédents.

Nota :

- La forme canonique disjonctive (somme de produits logiques) contient autant de termes qu'il y a de 1 dans le tableau ou la matrice.
- Chaque terme ou intersection canonique contient toutes les variables ou leurs compléments.

IV-22 Forme canonique conjonctive

A partir du tableau ou de la matrice on peut également faire la somme logique de toutes les intersections canoniques associées aux états pour lesquels la fonction F prend la valeur 0 ($\overline{F} = 1$).

On obtient alors :

$$\overline{F} = a.\overline{b}.\overline{c} + \overline{a}.b.\overline{c} + \overline{a}.\overline{b}.c + \overline{a}.b.c + a.b.c$$

C'est la forme canonique disjonctive du complément de la fonction F . En lui appliquant le théorème de De MORGAN, on trouve une expression sous la forme d'un produit de sommes qui est la forme canonique conjonctive de la fonction F.

$$\overline{\overline{F}} = \overline{a.\overline{b}.\overline{c} + \overline{a}.b.\overline{c} + \overline{a}.\overline{b}.c + \overline{a}.b.c + a.b.c}$$

$$\boxed{F = (\overline{a} + b + c).(a + \overline{b} + c).(a + b + \overline{c}).(a + \overline{b} + \overline{c}).(\overline{a} + \overline{b} + \overline{c})}$$

En développant cette expression on retrouve évidemment l'expression équivalente :

$$F = \overline{a}.\overline{b}.\overline{c} + a.b.\overline{c} + a.\overline{b}.c$$

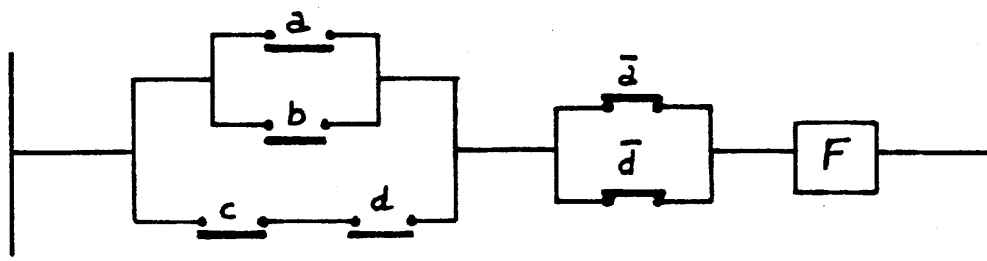
IV-3 ETUDE D'UNE FONCTION LOGIQUE CONNAISSANT SON SCHEMA ELECTRIQUE

Pour écrire l'équation algébrique représentant un schéma électrique, il suffit de se rappeler que, d'une part, les éléments placés en série sont représentés par un produit logique, et que d'autre part, les éléments de circuits montés en parallèles sont re présentés par la somme logique des variables associées à ces éléments.

Pour les relais, quels que soient les contacts associés, on doit

écrire : x = X et complémenter si nécessaire.

Exemple :



Le dipôle ci-dessus est représenté par l'expression logique:

$$F = [(a + b) + c.d].(\bar{a} + \bar{d})$$

$$F = (a + b + c.d).(\bar{a} + \bar{d})$$

$$F = a.\bar{a} + a.\bar{d} + \bar{a}.b + b.\bar{d} + \bar{a}.c.d + c.d.\bar{d}$$

$$F = 0 + a.\bar{d} + \bar{a}.b + b.\bar{d} + \bar{a}.c.d + 0$$

$$F = \bar{a}.(b + c.d) + \bar{d}.(a + b)$$

IV-4 SIMPLIFICATION DES FONCTIONS LOGIQUES

IV-41 Méthode algébrique

Les équations logiques suivent leslois algébriques pour :

a- Les mises en facteur

Soit $S = a.c + a.d + b.c + b.d$

$S = c.(a + b) + d.(a + b)$

$S = (a + b).(c + d)$

b- Les développementsde termes entre parenthèses

$S = (a + b).(c + d)$

$S = a.c + a.d + b.c + b.d$

Les simplifications peuvent également s'effectuer par :

c- Adjonction de termes déjà existants dans l'équation

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

Il est possible d'ajouter $\bar{a}.\bar{b}$ sans modifier l'équation

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

$S = \bar{a}.(b + \bar{b}) + \bar{b}.(a + \bar{a})$

i- Utilisation de la fonction complémentaire

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

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

$$\overline{S} = \overline{a}.\left[a + \overline{b}\right]$$

$$\overline{\overline{S}} = \overline{\overline{a}.(a + \overline{b})}$$

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

$$S = \overline{\overline{a}.\overline{\overline{b}}}$$

$$S = a + b$$

IV -411 Exercices à traiter

Simplifier les expressions suivantes :

I°- $(a + b).(c + \overline{d})$

2°- $(a.b + c).(\overline{b} + \overline{c})$

3°- $a.c + \overline{a}.b + \overline{b}$

4°- $e(a.b + c.d) + \overline{e}$

5°- $\overline{(a.\overline{c}.d + a.\overline{b}.\overline{c}).(a.\overline{c}.d + a.\overline{b}.\overline{c})} + \overline{a}.\overline{b}$

6°- $(a + \overline{b} + \overline{c}).(a + \overline{b}.c)$

7°- $a.b.\overline{c} + a.\overline{b}.\overline{c} + \overline{a}.b.\overline{c} + \overline{a}.\overline{b}.\overline{c}$

8°- $a.b + \overline{b}.c + \overline{(a + \overline{b} + \overline{c})} + \overline{a}.\overline{c}$

9°- $a.b + \overline{a}.\overline{b}.c + a.c + \overline{a}.b$

I0°- Complémenter l'expression $\overline{a}.\overline{c} + b + a.d$

Réponses :

I°- $a.c + a.\overline{d} + b.c + b.\overline{d}$

2°- $a.b.\overline{c} + \overline{b}.c$

3°- $\overline{a} + \overline{b} + c$

4°- $a.b + c.d + \overline{e}$

5°- $\overline{a}.\overline{b}$

6°- $a + \overline{b}.c$

7°- $\overline{c}$

8°- $\overline{a} + b + c$

9°- $b + c$

10°- $(a + c).\overline{b}.(\overline{a} + \overline{d})$

IV-42 Méthode graphique

IV-421 Propriétés du code binaire réfléchi

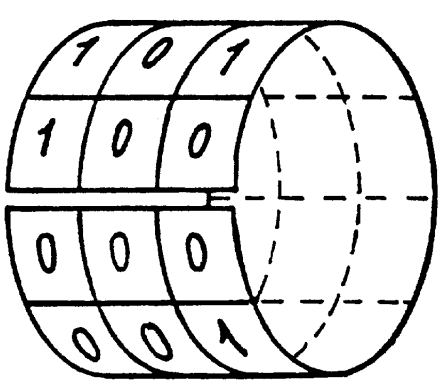
Si on considère le tableau ci-dessous formé à partir des huit combinaisons possibles des états de trois variables a,b,c, rangées selon l'ordre du code binaire réfléchi, on constate que :

Deux combinaisons adjacentes (consécutives), ou deux combinaisons disposées symétriquement par rapport à l'un des axes α , β , γ , passant par le milieu de l'intervalle correspondant à la valeur 1 de l'une des variables, ne diffèrent que par l'état de l'une de ces variables.

c	b	a	c b a	Intersection canonique
0	0	0		$\bar{a}.\bar{b}.\bar{c}$
0	0	1	1º axe de symétrie de a (α_1)	$a.\bar{b}.\bar{c}$
0	1	1		$a.b.\bar{c}$
0	1	0	1º axe de symétrie de b (β_1)	$\bar{a}.b.\bar{c}$
1	1	0		$\bar{a}.b.c$
1	1	1	2º axe de symétrie de a (α_2)	$a.b.c$
1	0	1		$a.\bar{b}.c$
1	0	0	1º axe de symétrie de c (γ_1)	$\bar{a}.\bar{b}.c$

Il faut également noter que le code binaire réfléchi étant cyclique, les combinaisons extrêmes du tableau sont aussi adjacentes.

Pour faciliter la compréhension, on peut imaginer ce tableau enroulé sur un cylindre d'axe horizontal; les combinaisons extrêmes sont alors adjacentes.



Exemples :

a- les deux combinaisons adjacentes (1,1,0) et (1,1,1) ne diffèrent que par l'état de la variable a.

b- les deux combinaisons symétriques (0,0,0) et (0,1,0) ne diffèrent que par l'état de la variable b.

IV-422 Principe de la simplification

Lorsqu'une fonction de N variables a la valeur 1 pour 2 combinaisons adjacentes ou symétriques, les deux intersections canoniques qui les représentent ne diffèrent que par une seule variable et peuvent se réduire à un terme unique ne contenant plus que (N-1) variables.

Exemples : Si une fonction a la valeur 1 pour les deux combinaisons adjacentes de l'exemple a précédent, la fonction est représentée pour ces deux termes par l'expression algèbrique :

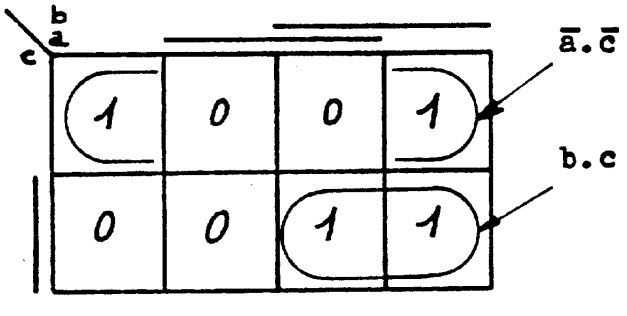
$$\bar{a}.b.c + a.b.c$$

soit $b.c\ (a + \bar{a}) = b.c.1$

ou encore : $b.c$

De même, si la fonction a la valeur 1 pour les deux combinaisons symétriques de l'exemple b, $\bar{a}.\bar{b}.\bar{c} + \bar{a}.b.\bar{c}$ peut être remplacé par $\bar{a}.\bar{c}$

Dans la représentation de Karnaugh - Zakrevski, le résultat des simplifications précédentes se traduit par le groupement des cases adjacentes ou symétriques.
Ceci est matérialisé en entourant le groupe ainsi formé par une ligne continue dans le cas où les cases sont jointives, par une ligne discontinue lorsque les cases sont séparées.



Pratiquement,l'expression algébrique représentant le groupement s'obtient en effectuant le produit des variables dont l'état ne change pas lorsqu'on considère les lignes ou les colonnes dont l'intersection forme le groupement.

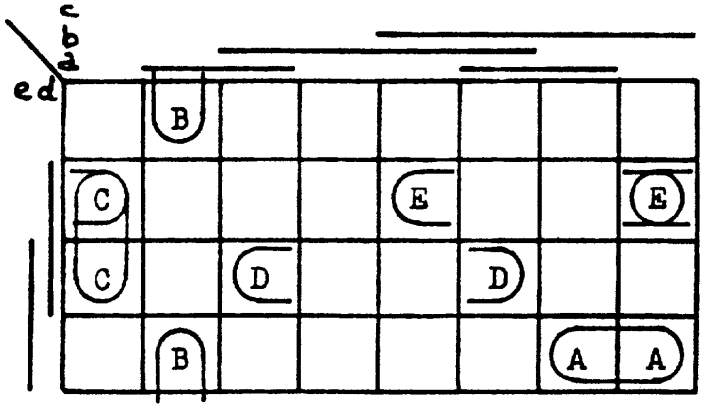
IV-423 Généralisation

D'une manière générale,si dans une matrice relative à N variables,on peut former un groupement de 2^n cases à l'état 1,possédant les propriétés d'adjacence ou de symétrie pour chacune des variables,on obtient pour ce groupement un terme unique ne contenant que (N - n) variables.
Il est alors évident que la représentation d'une fonction sera d'autant plus simple que les groupements formés seront plus grands.
Il faut noter à ce sujet qu'un même "1" peut être utilisé pour former des groupements différents (a + a = a).
Dans le cas où la matrice comporte peu de zéro,il est avantageux de déterminer la fonction complémentaire $\overline{F}$ en groupant selon les mêmes règles les cases de valeur 0 et en complémentant ensuite l'expression obtenue pour définir F.

Exemples :

Dans les exemples ci-dessous,les cases repérées A,B, C,D,.... sont des cases de valeur 1.

Groupement de 2 cases :



$F_A(a,b,c,d,e) = \overline{b}.c.\overline{d}.e$

$F_B(a,b,c,d,e) = a.\overline{b}.\overline{c}.\overline{d}$

$F_C(a,b,c,d,e) = \overline{a}.\overline{b}.\overline{c}.d$

$F_D(a,b,c,d,e) = a.b.e.d$

$F_E(a,b,c,d,e) = \overline{a}.c.d.\overline{e}$

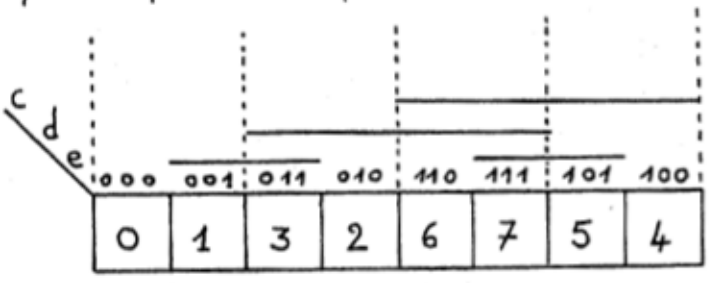
$F_{CE}(a,b,c,d,e) = \overline{a}.\overline{b}.d.\overline{e}$

Pour la simplification des "gros" tableaux de Karnaugh, il est bon de repérer les adjacences.

Qu'est-ce qu'une adjacence ?

Définition : Deux cases, colonnes ou lignes sont dites adjacentes si leurs numéros en repérage binaire ne diffèrent que par l'inversion (complémentation $0 \rightleftarrows 1$) d'un chiffre.

Avec le code binaire réfléchi, la recherche des adjacences est facilitée par la mise en évidence d'axes de symétrie partielle ; le repérage par traits facilite la mise en place de ces axes.

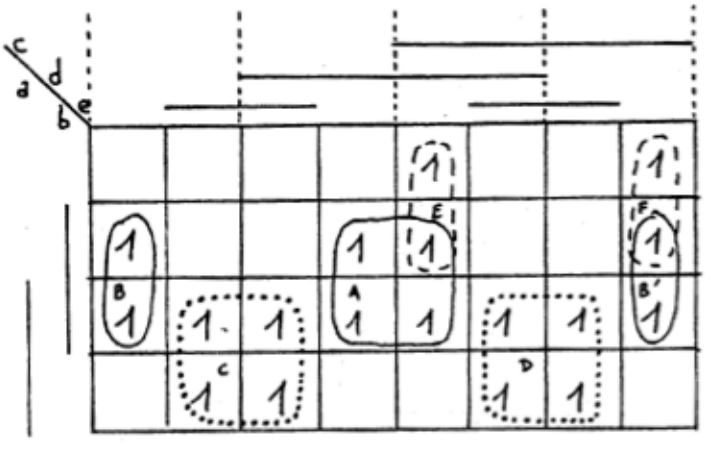


La case 0 est adjacente à 1, 2, 4
" 1 " 0, 3, 5
" 3 " 1, 2, 7

L'ASSOCIATION DE 2 CASES ADJACENTES EST UNE SIMPLIFICATION.

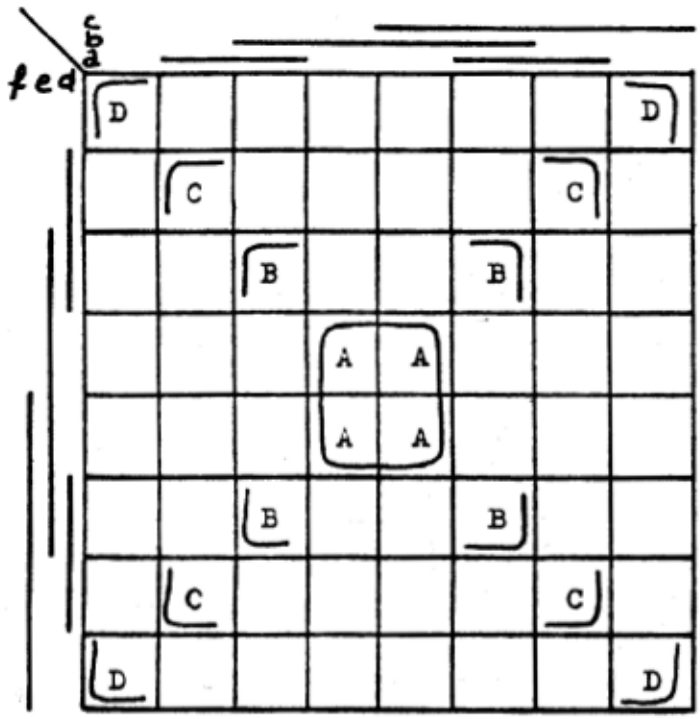
exemple : le groupement 3-7 représente $\bar{c}de + cde = de.(\bar{c}+c) = de$

exemple :

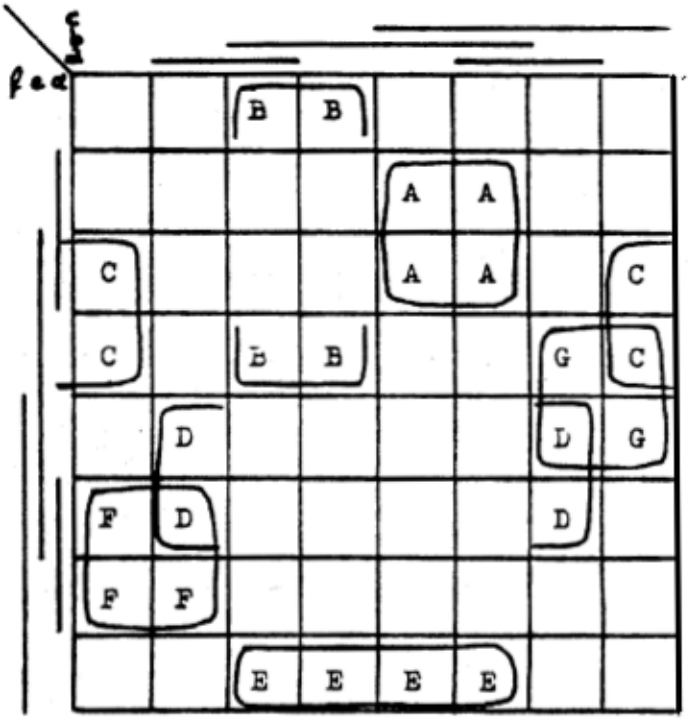


C+D a pour expression de
E+F " " $\bar{a}ce$
A+B+B' " " $b\bar{e}$

Groupements de 4 cases

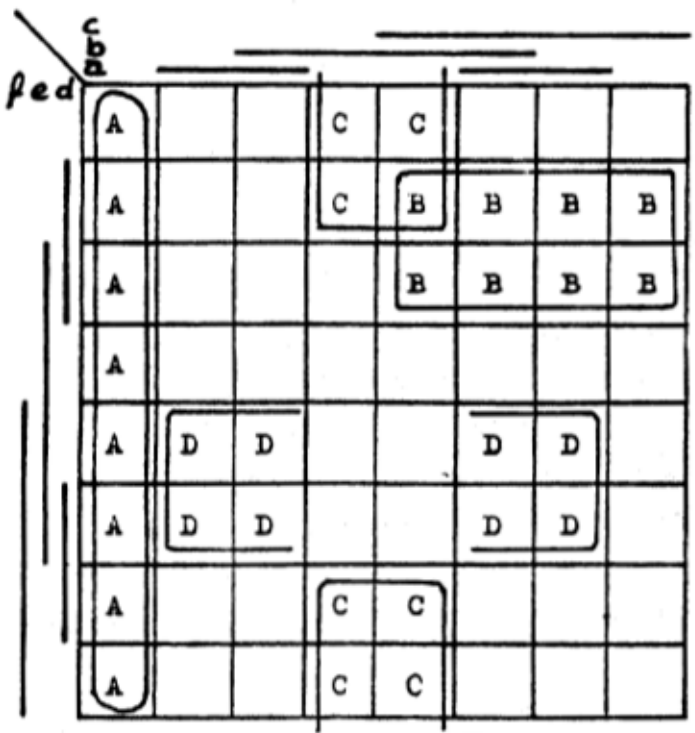


$F_A = \bar{a}.b.d.e$
$F_B = a.b.d.e$
$F_C = a.\bar{b}.d.\bar{e}$
$F_D = \bar{a}.\bar{b}.d.\bar{e}$

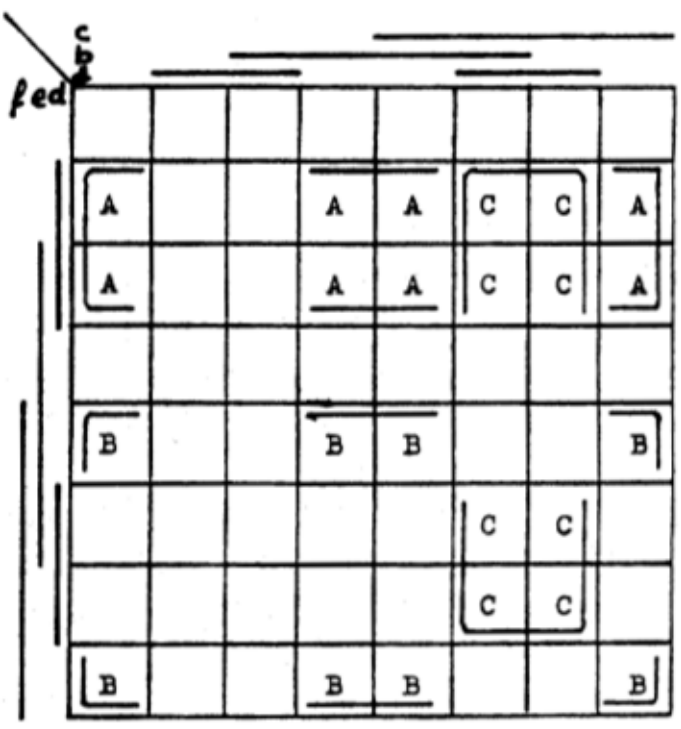


$F_A = b.c.d.\bar{f}$
$F_B = b.\bar{c}.\bar{d}.\bar{f}$
$F_C = \bar{a}.\bar{b}.e.\bar{f}$
$F_D = a.\bar{b}.e.f$
$F_E = b.\bar{d}.\bar{e}.f$
$F_{DF} = \bar{b}.\bar{c}.d.f$
$F_{CDG} = \bar{b}.c.\bar{d}.e$

Groupements de 8 cases

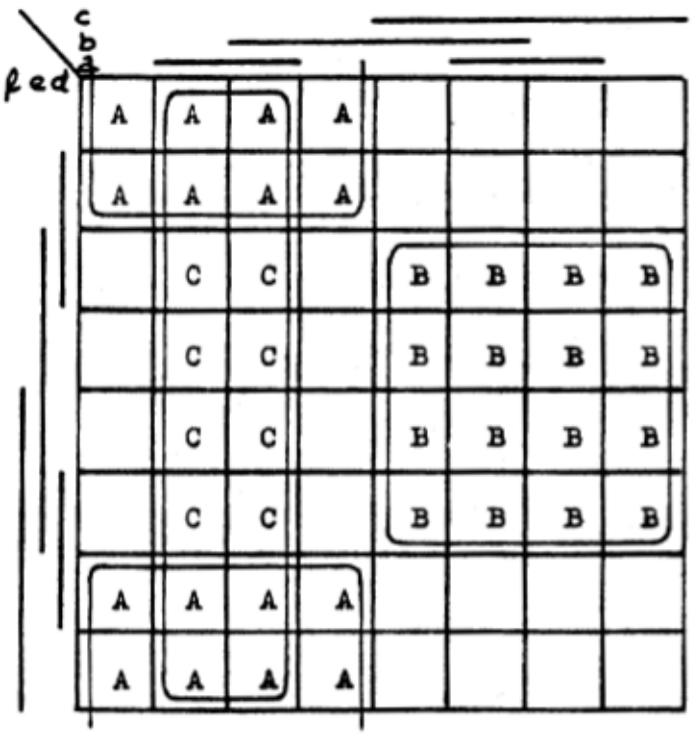


$F_A = \bar{a}.\bar{b}.\bar{c}$
$F_B = c.d.\bar{f}$
$F_{BC} = \bar{a}.b.\bar{e}$
$F_D = a.e.f$

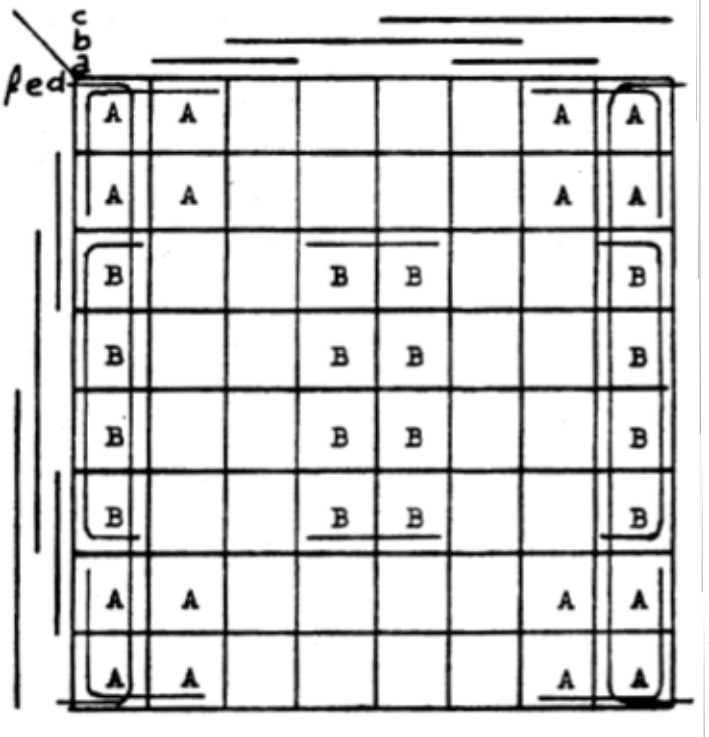


$F_A = \bar{a}.d.\bar{f}$
$F_B = \bar{a}.\bar{d}.f$
$F_C = a.c.d$

Groupements de 16 cases



$F_A = \overline{c}.\overline{e}$
$F_B = c.e$
$F_{AC} = a.\overline{c}$



$F_A = \overline{b}.\overline{e}$
$F_B = \overline{a}.e$
$F_{AB} = \overline{a}.\overline{b}$

Remarques :

1º - Comme nous venons de le voir, les possibilités de groupement sont extrêmement variées et leur validité n'est pas toujours évidente. Il est donc nécessaire de s'en assurer en vérifiant que pour une matrice de N variables, le produit logique correspondant à un groupement de 2^n cases comprend bien (N - n) variables.

Dans le cas d'un groupement incorrect, il manque toujours plus de variables que prévu (n).

2º - Dans le cas où la matrice est incomplète (cas indifférents ou impossibles), on affecte arbitrairement aux cases correspondantes la valeur 1 ou 0 selon l'intérêt qu'elles peuvent présenter en vue de la réalisation de groupements importants.

Exercices :

Simplification par la méthode graphique (matrices de KARNAUGH).
Le signe Ø indique une case indifférente ou impossible.

1

c \ b a				
	1	0	0	1
	0	0	1	1

2

c \ b a				
	1	1	Ø	1
	Ø	1	0	Ø

3

c \ b a				
	1	0	1	0
	1	0	0	1

4

d c \ b a				
	0	0	0	1
	1	0	0	1
	1	1	0	1
	1	1	0	0

5

d c \ b a				
	1	0	0	1
	1	1	0	0
	1	1	0	0
	1	0	1	1

6

d c \ b a				
	1	1	0	0
	0	0	1	0
	0	1	1	1
	1	1	0	0

7

d c \ b a				
	Ø	1	0	0
	1	1	0	1
	Ø	Ø	1	1
	Ø	Ø	Ø	0

8

d c \ b a				
	1	1	0	0
	Ø	0	0	1
	0	1	1	0
	1	1	Ø	0

9

d c \ b a				
	0	1	1	Ø
	1	0	Ø	1
	1	0	Ø	1
	0	1	1	Ø

10

d c \ b a				
	1	0	1	0
	0	0	1	0
	1	1	Ø	0
	0	0	1	1

11

d c \ b a				
	1	0	0	Ø
	1	0	Ø	1
	0	Ø	0	0
	Ø	0	1	1

12

d c \ b a				
	Ø	0	1	0
	Ø	Ø	1	1
	0	1	Ø	0
	0	0	1	0

13

cba / ed	00	01	11	10
00	0	0	1	0
01	1	1	0	0
11	∅	1	0	0
10	0	∅	∅	1

cba / ed	00	01	11	10
00	∅	0	1	1
01	0	∅	0	0
11	1	0	1	1
10	0	∅	∅	∅

14

cba / ed	00	01	11	10
00	1	0	1	1
01	0	1	0	0
11	0	1	∅	∅
10	1	∅	1	1

cba / ed	00	01	11	10
00	0	0	1	0
01	1	1	∅	0
11	0	1	0	∅
10	∅	0	0	1

15

cba / ed	00	01	11	10
00	∅	1	1	1
01	∅	1	∅	∅
11	0	0	1	0
10	0	0	1	∅

cba / ed	00	01	11	10
00	0	∅	1	0
01	∅	1	0	∅
11	0	1	0	1
10	0	1	0	1

16

cba / ed	00	01	11	10
00	1	∅	0	1
01	1	0	0	∅
11	0	0	0	1
10	1	0	0	∅

cba / ed	00	01	11	10
00	∅	0	1	∅
01	∅	∅	0	∅
11	0	1	1	0
10	∅	1	0	0

17

cba / ed	00	01	11	10
00	0	1	1	0
01	∅	1	1	∅
11	0	∅	∅	1
10	0	∅	1	1

cba / ed	00	01	11	10
00	0	1	1	0
01	∅	1	1	∅
11	1	0	1	1
10	0	∅	1	0

18

cba / ed	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1	1	1	1
10	0	1	0	0

cba / ed	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	1	1	1	1
10	0	1	0	0

19

cba / ed	00	01	11	10
00	0	1	1	0
01	0	0	1	0
11	1	0	1	0
10	1	1	1	0

cba / ed	00	01	11	10
00	0	1	1	0
01	0	0	∅	∅
11	0	1	0	0
10	1	1	1	1

20

cba / ed	00	01	11	10
00	1	0	0	1
01	1	0	1	1
11	∅	0	∅	1
10	1	0	∅	1

cba / ed	00	01	11	10
00	1	0	0	1
01	1	0	1	1
11	∅	0	∅	1
10	1	0	∅	1

Réponses des exercices de simplification graphique page suivante

Réponses des exercices de la page 53 :

1) $F = \bar{a}.\bar{c} + b.c$

2) $F = \bar{a} + \bar{b}$ ou $F = \bar{b} + \bar{c}$

3) $F = \bar{a}.(\bar{b} + c) + a.b.\bar{c}$

4) $F = \bar{a}.c + \bar{b}.d + \bar{a}.b.\bar{d}$

5) $F = \bar{b}.c + \bar{a}.\bar{c} + b.\bar{c}.d$

6) $F = \bar{b}.\bar{c} + a.b.c + c.d.(a + b)$

7) $F = \bar{b} + \bar{a}.c + c.d$

8) $F = a.d + \bar{b}.\bar{c} + \bar{a}.c.\bar{d}$

9) $F = a.\bar{c} + \bar{a}.c$

10) $F = a.b + \bar{b}.c.d + b.\bar{c}.d + \bar{a}.\bar{b}.\bar{c}.\bar{d}$

11) $F = \bar{a}.\bar{d} + b.\bar{c}.d$

12) $F = a.b + c.\bar{d} + a.c$

13) $F = \bar{b}.d + a.e + b.c.\bar{d}$

14) $F = \bar{a}.\bar{d} + a.\bar{b}.d + a.\bar{c}.d$

15) $F = \bar{c}.\bar{e} + a.b + a.\bar{d}.\bar{e} + \bar{a}.\bar{b}.c.e$

16) $F = \bar{a}.\bar{e} + \bar{a}.b + a.\bar{b}.c.d + \bar{a}.\bar{d}$

17) $F = a + d.\bar{e} + b.\bar{c}.e$

18) $F = a.\bar{b}.\bar{d} + a.\bar{c}.\bar{e} + \bar{c}.d.e + \bar{b}.d.e + \bar{a}.\bar{b}.c.\bar{e}$

19) $F = a.\bar{d} + \bar{a}.\bar{b}.e + a.b.\bar{c}$

20) $F = \bar{a} + b.d$

IV-5 ALEAS DE FONCTIONNEMENT DES SYSTEMES COMBINATOIRES

Un aléa est un risque de mauvais fonctionnement qui peut apparaitre lors d'une transition, c'est à dire lors du changement d'état d'une variable.

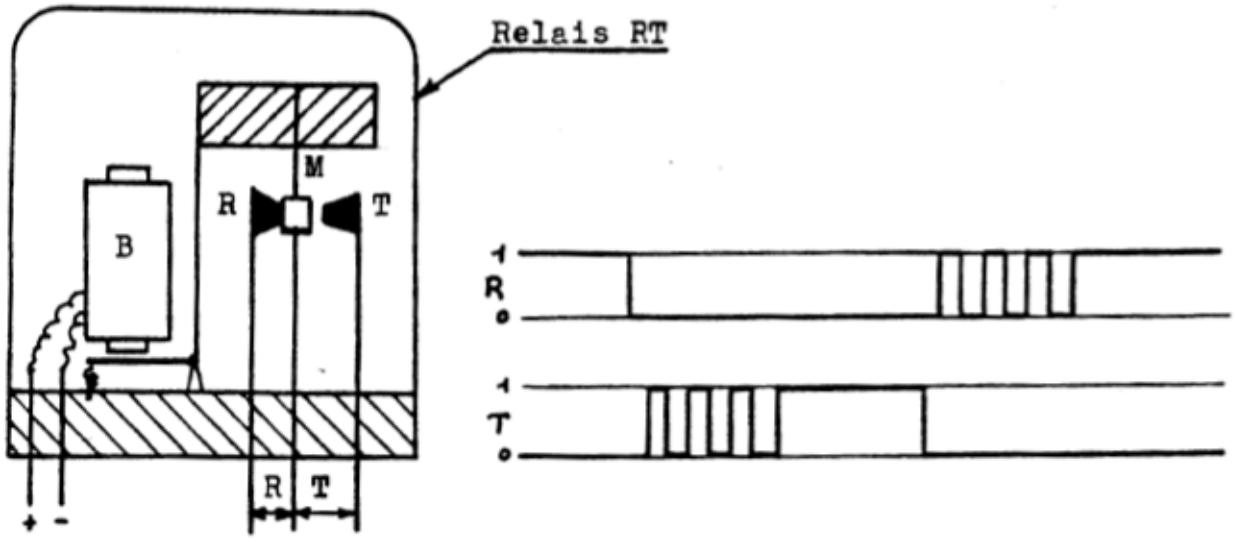
Les systèmes combinatoires peuvent présenter deux types d'aléas :

- Les aléas dynamiques;
- Les aléas de continuité ou technologiques.

IV-51 Aléa dynamique

L'aléa dynamique est dû à des rebondissements de contacts qui se traduisent par une succession de passages à l'état 0 et 1 de la variable correspondante, ce qui peut provoquer un fonctionnement incorrect de l'automatisme.

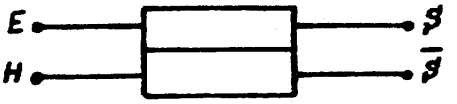
Par exemple, à l'excitation de la bobine B du relais ci-dessous ,le contact mobile M quitte le contact R (R = 0) et vient heurter le contact T qui rebondit sous le choc provoquant une succession d'états 0 et 1 pour finalement se stabiliser à l'état 1. Le fonctionnement est identique à la désexcitation de la bobine B, le contact R rebondit.

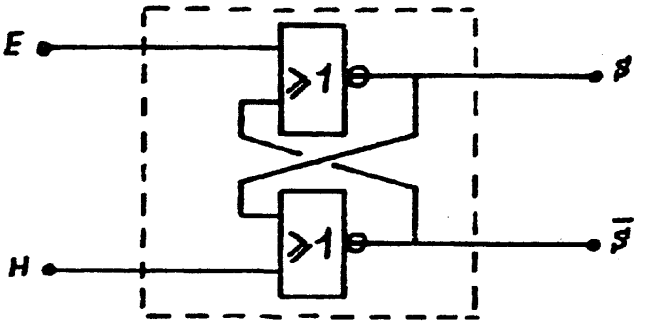


Pour éviter ces aléas dynamiques, il est nécessaire d'utiliser un circuit tel que sa sortie S prenne l'état 1 et y demeure lors qu'un état 1 même fugitif apparait à l'entrée.

On utilise généralement :

la mémoire bistable R-S





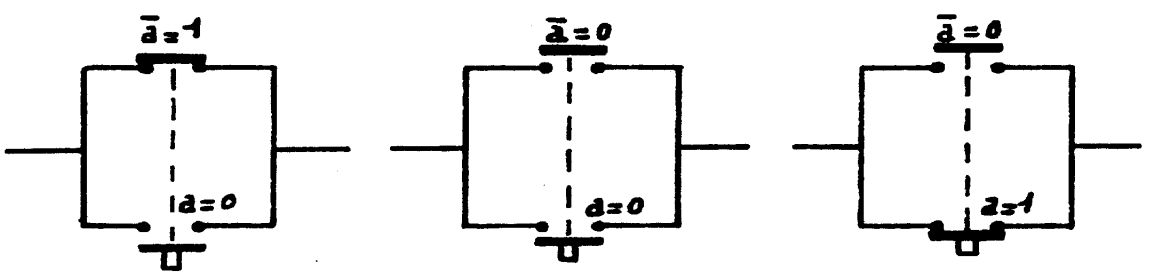
Anecdote :

Le codage numérique utilisé en informatique est un autre obstacle à la création de logiciels entièrement fiables. De nombreux systèmes physiques sont continus : leurs propriétés sont décrites par des paramètres qui peuvent prendre n'importe quelle valeur dans certaines gammes. Au contraire, la plus petite modification possible d'une variable numérique (c'est-à-dire le passage d'un bit de l'état 0 à l'état 1, ou inversement) correspond parfois à une variation notable de l'état du système. Un seul caractère erroné, dans le programme de commande d'une fusée *Atlas* qui transportait le premier vaisseau interplanétaire *Mariner 1*, a déréglé la trajectoire de la fusée et imposé sa destruction quelques instants après le lancement.

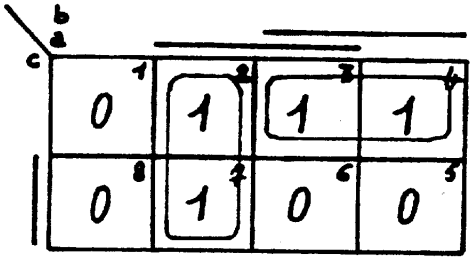
IN POUR LA SCIENCE JANVIER 93

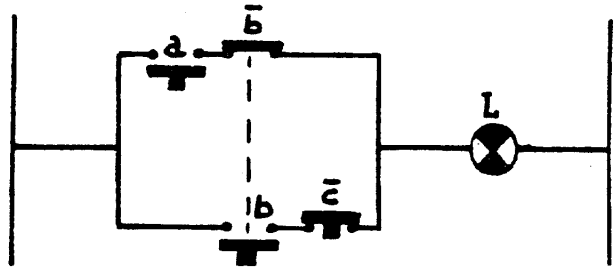
IV-52 Aléa de continuité ou technologique

Cet aléa est dû au fait que pour un inverseur réel, les deux états stables sont séparés par un état intermédiaire pour lequel on a : a = 0 et $\bar{a}$ = 0. La permanence de l'état 1 n'est donc pas assuré durant cet état.



Soit la fonction $L = a.\bar{b} + b.\bar{c}$ représentée par la matrice de Karnaugh ou le schéma électrique ci-dessous :





Si l'on considère la succession des états suivants (cases n° 2 et n° 3) :

$a.\bar{b} + b.\bar{c} = L$

case n°2 : $a = 1$; $b = 0$, $\bar{b} = 1$; $c = 0$, $\bar{c} = 1$ ⟶ $1.1 + 0.1 = 1$

Etat intermédiaire : $a = 1$; $b = 0$, $\bar{b} = 0$; $c = 0$, $\bar{c} = 1$ ⟶ $1.0 + 0.1 = 0$

case n°3 : $a = 1$; $b = 1$, $\bar{b} = 0$; $c = 0$, $\bar{c} = 1$ ⟶ $1.0 + 1.1 = 1$

On constate que l'état de la lampe L passe momentanément à l'état 0 lorsqu'on appuie sur le bouton-poussoir b.
La relation $b + \bar{b} = 1$ cesse donc d'être valable durant la transition.
Pour éliminer les aléas de ce type on peut utiliser l'un des procédés suivants :

a- Utiliser un terme redondant : dans l'exemple précédent,il est possible de supprimer l'aléa en masquant l'état 0 transi-

toire de la lampe L en ajoutant à l'expression $L = a.\bar{b} + b.\bar{c}$ le terme $a.\bar{c}$ qui est à l'état 1 pendant la transition.

$$a.\bar{b} + b.\bar{c} + a.\bar{c} = L$$

case n°2 $\begin{cases} a = 1 \\ b = 0 \ , \ \bar{b} = 1 \\ c = 0 \ , \ \bar{c} = 1 \end{cases}$ $1.1 + 0.1 + 1.1 = 1$

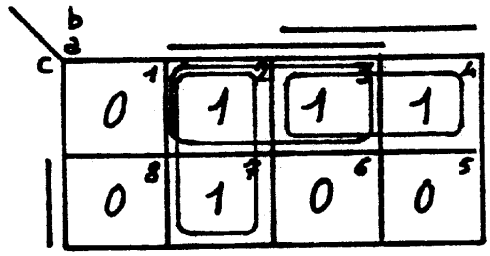
Etat intermédiaire $\begin{cases} a = 1 \\ \underline{b = 0 \ , \ \bar{b} = 0} \\ c = 0 \ , \ \bar{c} = 1 \end{cases}$ $1.0 + 0.1 + 1.1 = 1$

case n°3 $\begin{cases} a = 1 \\ b = 1 \ , \ \bar{b} = 0 \\ c = 0 \ , \ \bar{c} = 1 \end{cases}$ $1.0 + 1.1 + 1.1 = 1$

L'aléa a disparu sans modifier le fonctionnement de la lampe initialement défini par $L = a.\bar{b} + b.\bar{c}$,en effet :

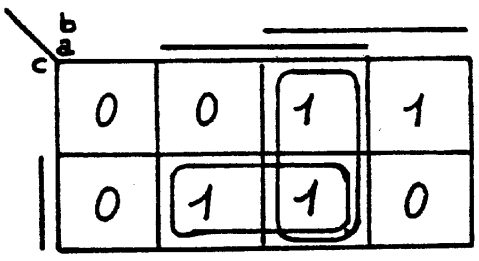
$L = a.\bar{b} + b.\bar{c} + a.\bar{c}$
$L = a.\bar{b} + b.\bar{c} + a.\bar{c}\ (b + \bar{b})$
$L = a.\bar{b} + b.\bar{c} + a.\bar{c}.b + a.\bar{c}.\bar{b}$
$L = a.\bar{b}\ (1 + \bar{c}) + b.\bar{c}\ (1 + a)$
$L = a.\bar{b} + b.\bar{c}$

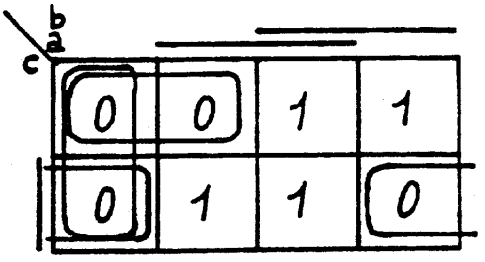
Graphiquement,pour supprimer l'aléa qui se produit lors du changement d'état de la variable b (passage de la case n°2 à la case n°3 ou réciproquement),il suffit que ces deux cases appartiennent au même groupement $a.\bar{c}$.



En règle générale(Huffman),pour qu'il n'y ait pas d'aléa technologique,il faut et il suffit,que le passage d'une case à une autre case de même valeur se produise à l'intérieur d'un même groupement (groupement de 1 ou groupement de 0).

Exemples :





$F = a.b + a.c$

$F = a.(b + c)$

$\overline{F} = \overline{a}.\overline{b} + \overline{a}.c + \overline{b}.\overline{c}$

$\overline{F} = \overline{a}.(\overline{b} + c) + \overline{b}.\overline{c}$

b- Concevoir des équations telles qu'elles ne contiennent que des variables a ou $\overline{a}$, b ou $\overline{b}$, c ou $\overline{c}$, mais pas les deux types de variables simultanément.

La fonction L peut par exemple s'écrire :

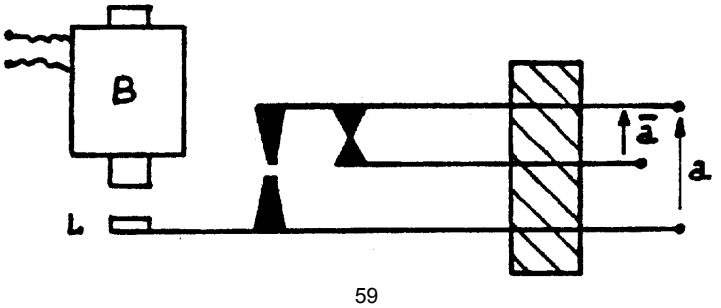
$L = a.\overline{b} + b.\overline{c} = \overline{\overline{a.\overline{b}} + b.\overline{c}}$

$L = \overline{\overline{a} + b + b.\overline{c}}$

Dans cette expression ne subsiste plus à la fois les variables b et $\overline{b}$,il n'y a donc plus d'aléa de continuité.

c- Utilisation de contact T.R

A l'excitation de la bobine B le déplacement de la lame mobile L provoque d'abord la fermeture du contact a,puis l'ouverture du contact $\overline{a}$ .Durant cette transition les états de a et de $\overline{a}$ étant simultanément égaux à 1 il n'y a plus d'aléa de continuité.



IV-6 CONVERSION DE LA LOGIQUE"ET,OU,NON"EN LOGIQUE"OU-NON" ET EN LOGIQUE"ET-NON"

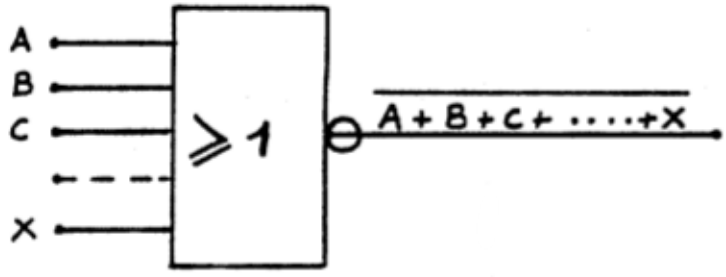
L'avantage des opérateurs logiques "OU-NON" et "ET-NON" réside dans le fait qu'une fonction logique quelconque peut toujours être exprimée en n'utilisant que l'un ou l'autre de ces opérateurs.Ainsi certains constructeurs profitent de cette particularité et réalisent des circuits qui ne comportent qu'un seul type d'opérateurs.
Il est donc nécessaire de savoir traduire en "logique OU-NON" et en "logique ET-NON" les fonctions usuellement écrites en "logique ET,OU,NON".

IV-61 Méthode algébrique

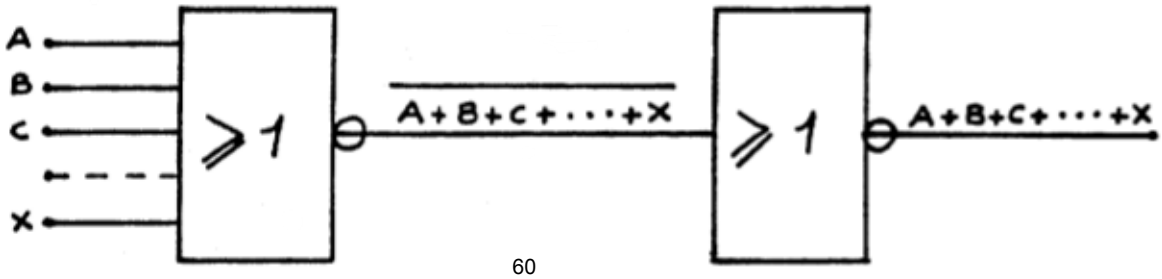
IV-611 Forme OU-NON

Pour obtenir la forme OU-NON d'une fonction logique,il suffit de complémenter deux fois les expressions qui n'ont pas la structure d'un complément de sommes puis éventuellement d'effectuer l'une des complémentations,ainsi que le montrent les différents exemples ci-dessous :

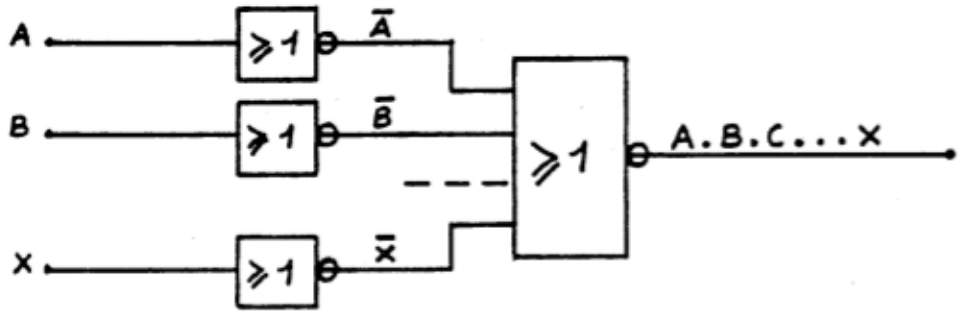
$\overline{A + B + C + \ldots\ldots + X}$



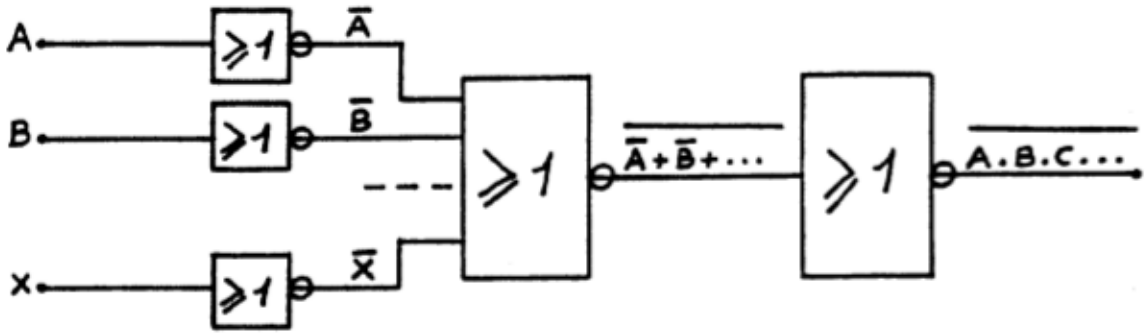
$$A + B + C + \ldots\ldots + X = \overline{\overline{A + B + C + \ldots\ldots + X}}$$



$$\overline{\overline{A \cdot B \cdot C \ldots\ldots X}} = \overline{A \cdot B \cdot C \ldots\ldots X} = \overline{\overline{A} + \overline{B} + \overline{C} + \ldots + \overline{X}}$$



$$\overline{A \cdot B \cdot C \ldots\ldots X} = \overline{A} + \overline{B} + \overline{C} + \ldots\ldots + \overline{X} = \overline{\overline{A} + \overline{B} + \overline{C} + \ldots\ldots + \overline{X}}$$

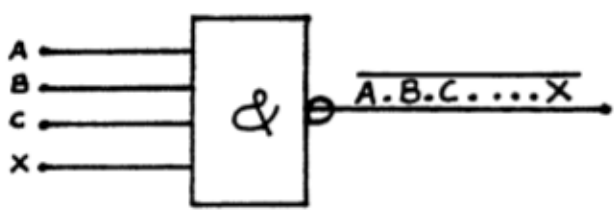


IV-612 Forme ET-NON

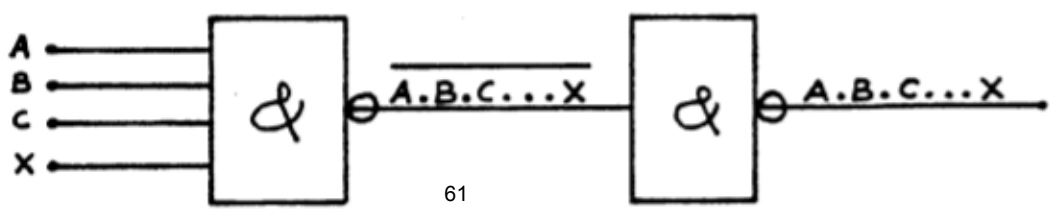
De même, pour établir la forme ET-NON d'une fonction logique, il suffit de complémenter deux fois les expressions qui n'ont pas la structure d'un complément de produits, puis éventuellement d'effectuer l'une de ces deux complémentations.

Exemples :

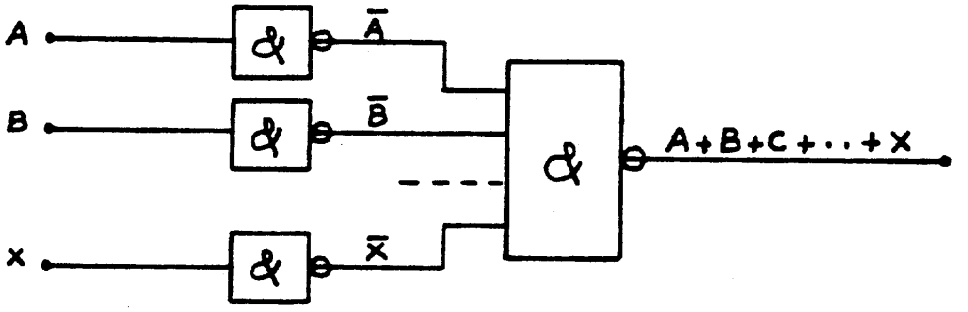
$$\overline{A \cdot B \cdot C \ldots\ldots X}$$



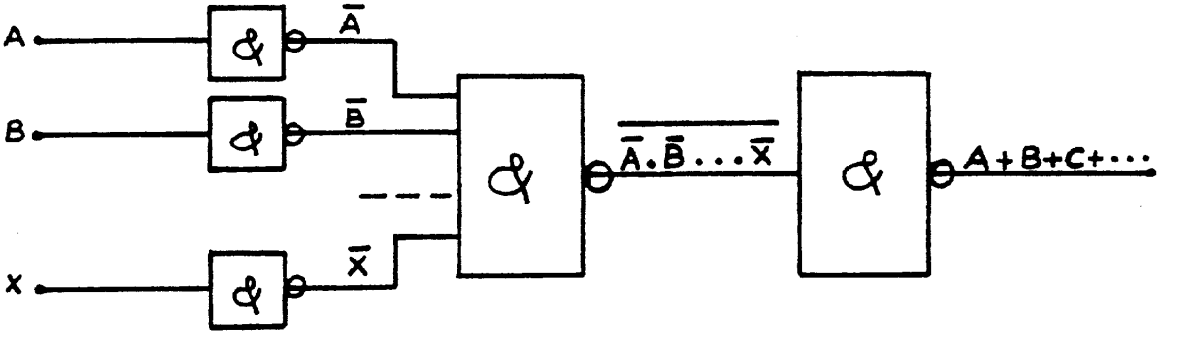
$$A \cdot B \cdot C \ldots\ldots X = \overline{\overline{A \cdot B \cdot C \ldots\ldots X}}$$



$$\overline{A + B + C + \ldots . X} = \overline{A + B + C + \ldots . + X} = \overline{A} \cdot \overline{B} \cdot \overline{C} \ldots \ldots \overline{X}$$

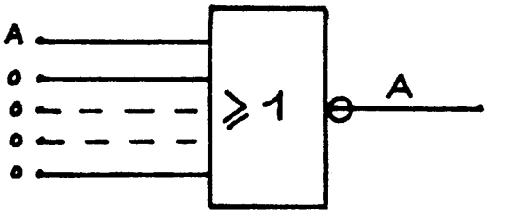


$$\overline{A + B + C + \ldots . X} = \overline{A} \cdot \overline{B} \cdot \overline{C} \ldots . \overline{X} = \overline{\overline{A} \cdot \overline{B} \cdot \overline{C} \ldots \ldots \overline{X}}$$

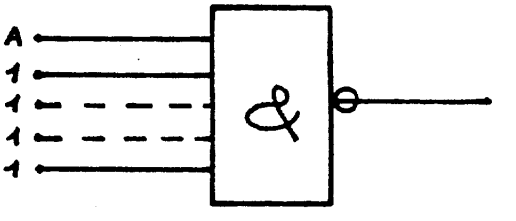


Remarque : Pour cabler à l'aide de modules OU-NON ou ET-NON à plusieurs entrées le complément $\overline{A}$ d'un terme unique A par exemple, il faut se souvenir que $\overline{A}$ peut s'écrire :

$$\overline{A} = \overline{A + 0 + 0 + \ldots .}$$



$$\overline{A} = \overline{A.1.1.1.\ldots\ldots}$$



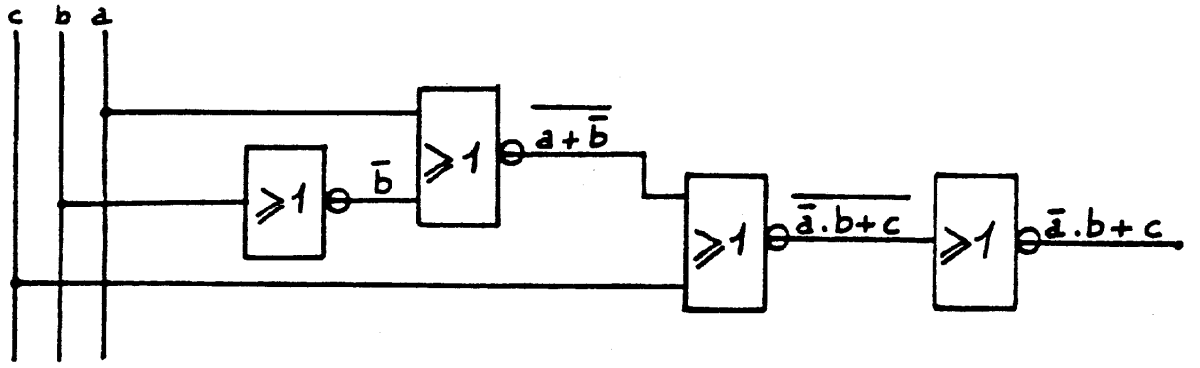
IV-613 Exemples

$F(a,b,c) = \overline{a}.b + c$

-Conversion en logique OU-NON :

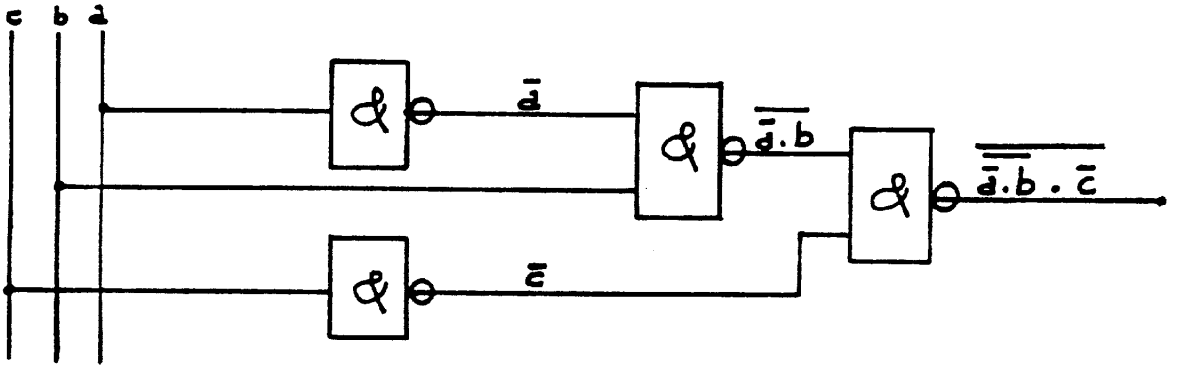
$$F(a,b,c) = \overline{\overline{a}.b + c} = \overline{a + \overline{b} + c}$$

d'où le logigramme suivant :



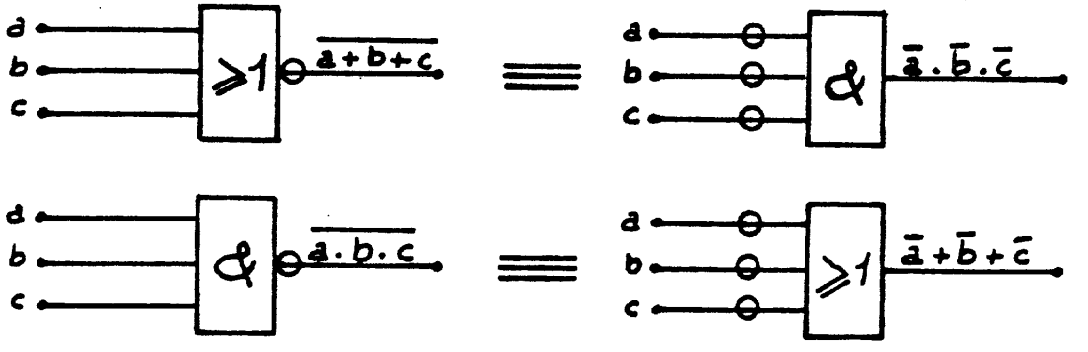
-Conversion en logique ET-NON :

$$F(a,b,c) = \bar{a}.b + c = \overline{\overline{\bar{a}.b + c}} = \overline{\overline{\bar{a}.b} \cdot \bar{c}}$$



IV-62 Méthode graphique :

Par application des théorèmes de De Morgan ($\overline{a + b} = \bar{a} \cdot \bar{b}$; $\overline{a \cdot b} = \bar{a} + \bar{b}$), les schémas ci-dessous sont équivalents :



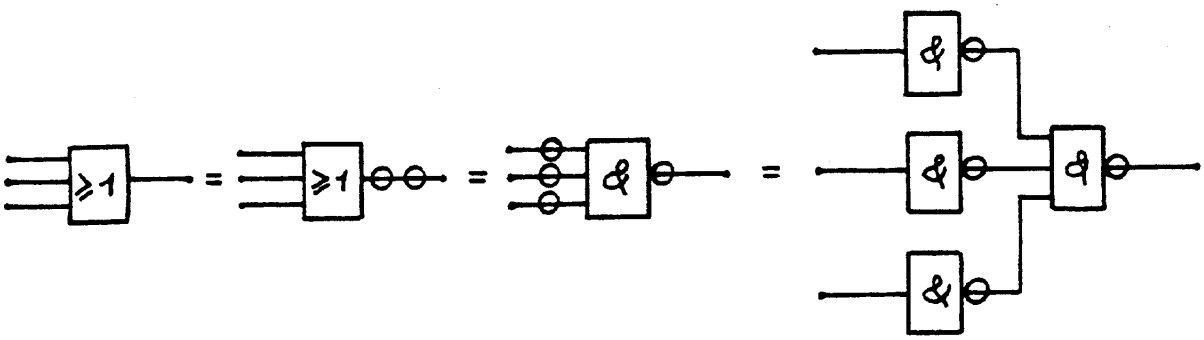
On constate, que l'application des théorèmes de De Morgan peut être assimilée graphiquement, au transfert d'un symbole de complémentation de la sortie sur les entrées, assorti de la conversion du symbole somme en celui du produit ou réciproquement.

La translation inverse est également possible.

Pour obtenir des fonctions OU-NON ou ET-NON à partir de fonctions ET ou de fonctions OU, il suffit donc d'introduire sur la sortie deux symboles de négation consécutifs, puis de transférer l'un d'eux à travers la fonction à transformer en appliquant la règle précédente.

Exemples :

Conversion de fonction OU en ET-NON :



Conversion de fonction ET en OU-NON :

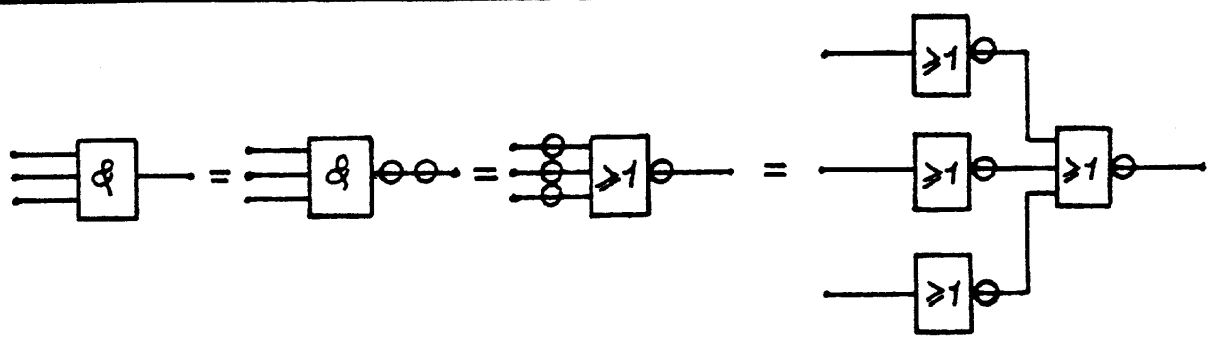
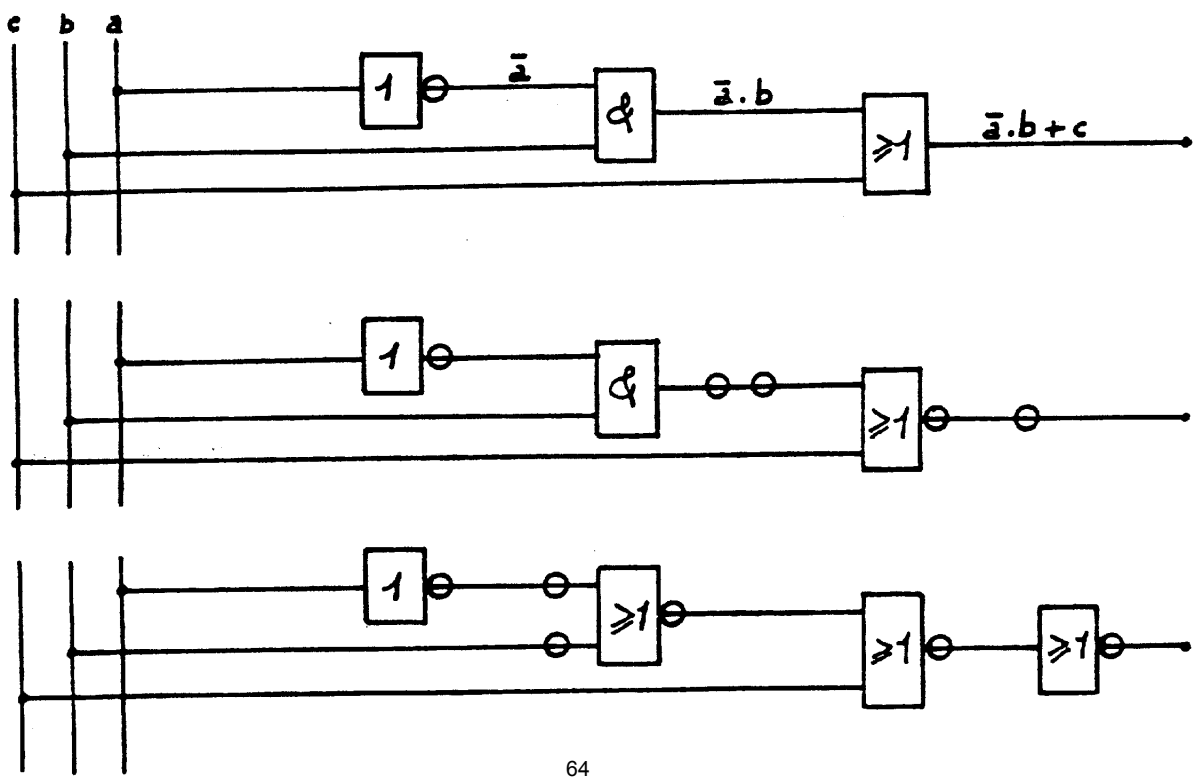


Schéma logique de la fonction $F = \bar{a}.b + c$ en OU-NON :



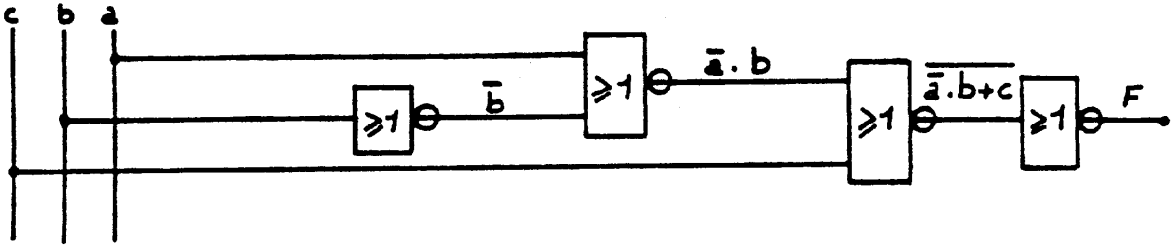
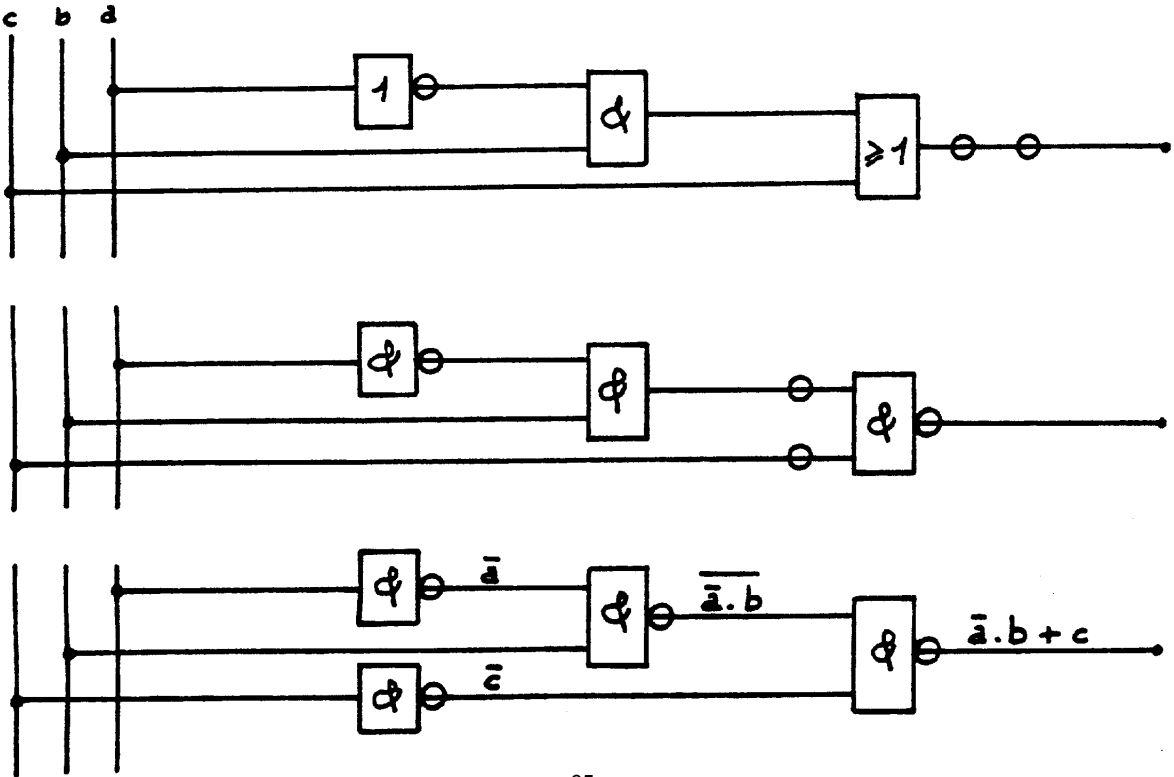
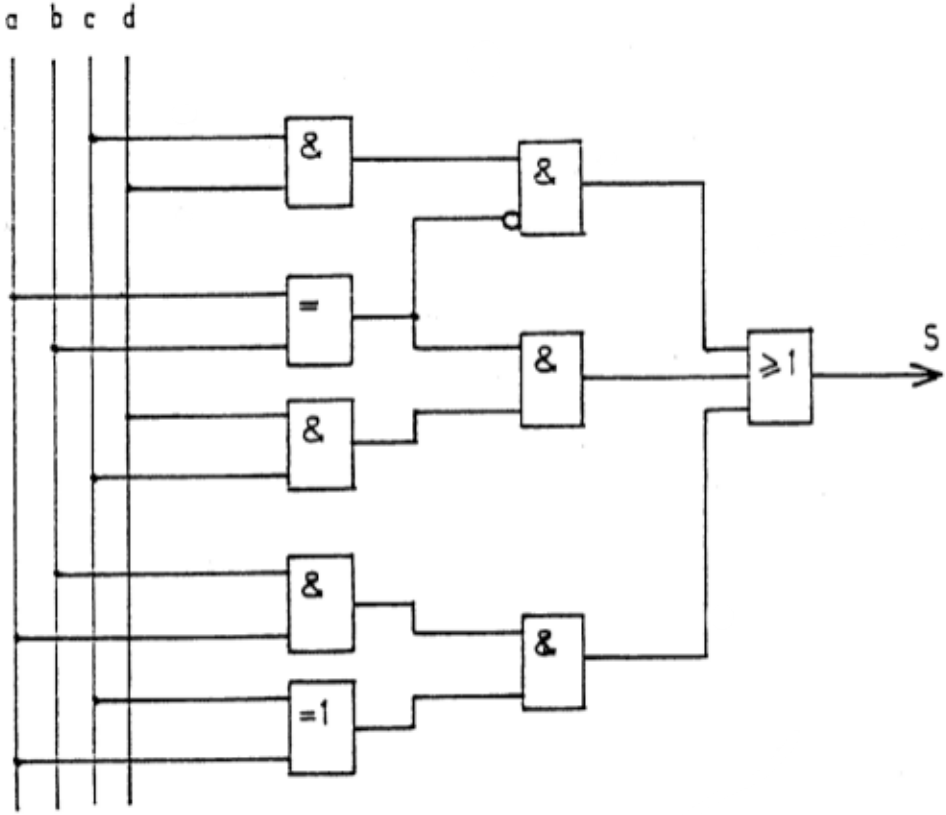


Schéma logique de la fonction $F = \bar{a}.b + c$ en ET-NON :



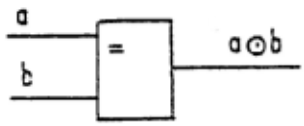
5e QUESTION (VALEUR = 3)

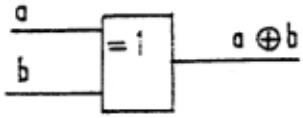
Soit le logigramme ci-dessous :



1. Déterminez l'expression de la fonction S (*a, b, c, d*).

   On rappelle :





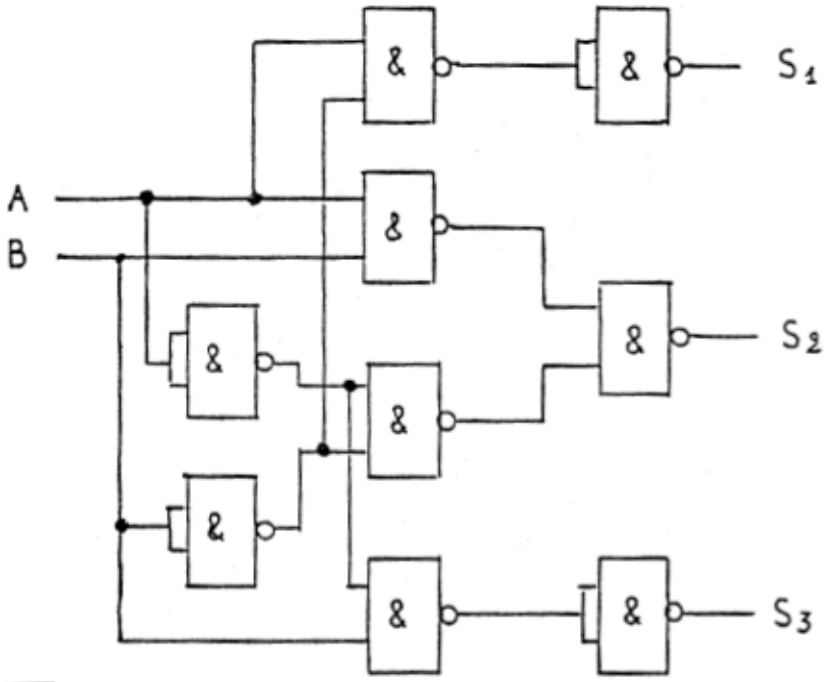
2. Simplifiez cette expression.

3. Établissez le logigramme de l'expression simplifiée en disposant des niveaux logiques 0 et 1 des variables *a, b, c, d* et en utilisant :

   a. uniquement des opérateurs NON ET (ET-NON) à 3 entrées;

   b. uniquement des opérateurs NI (OU-NON) à 3 entrées.

Le schéma ci-dessous est celui d'un comparateur logique (logique positive). A et B sont les entrées.



Les sorties S_1, S_2 et S_3 permettent de déterminer si $A = B$, $A > B$, $A < B$.

1° Que signifie pour A et B le fait que $S_1 = 1$?

2° Même question pour S_2 et S_3.

3° Vérifier qu'une des sorties et une seule est égale à 1 quels que soient les états de A et de B.

4e QUESTION (VALEUR = 5)

On considère le circuit de la figure 1 :

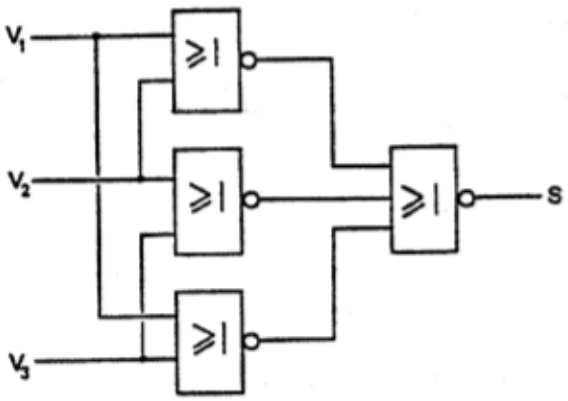


Figure 1

1° En dresser la table de vérité.

2° Un tel circuit est appelé seuil logique. Exprimer sa fonction par une phrase.

3° Son symbole étant celui de la figure 2, quelle est la valeur de m dans le cas de la figure 1 ?

4° Représenter un seuil logique à 4 entrées et où $m = 3$ en utilisant des fonctions logiques de votre choix ayant toutes le même nombre d'entrées, les entrées non utilisées étant mises à 0 ou 1.

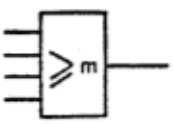


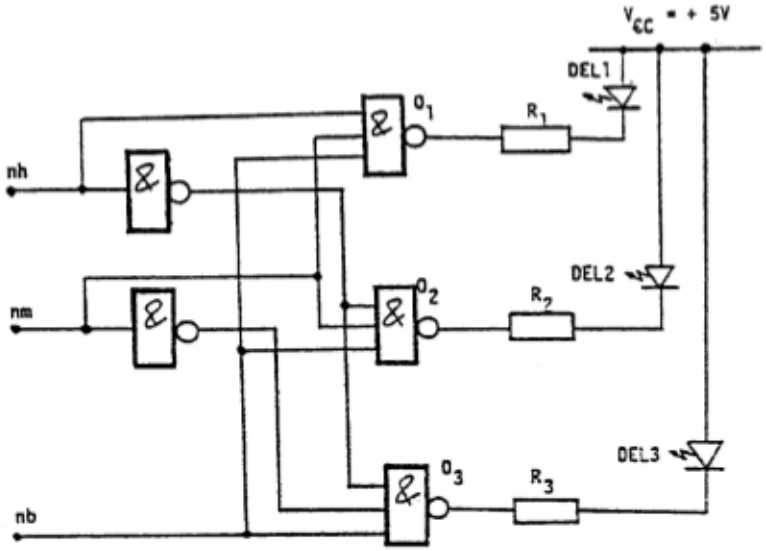
Figure 2

Le montage ci-dessous a pour but d'élaborer la signalisation du niveau d'un fluide dans une capacité.

Les informations n_h (niveau haut), n_m (niveau moyen) et n_b (niveau bas) sont issues de 3 capteurs positionnés judicieusement sur la capacité.

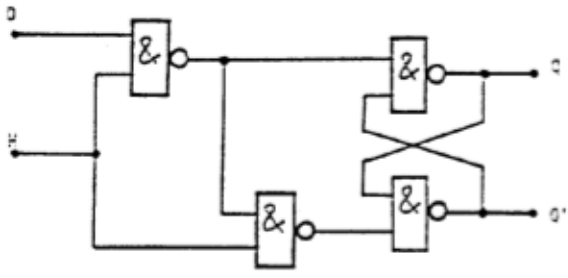
Le système de visualisation du niveau se compose de 3 diodes électroluminescentes DEL_1, DEL_2 et DEL_3 montées en séries avec les éléments résistifs limiteurs d'intensité R_1, R_2 et R_3.

Établir les équations logiques de commande des différentes DEL et expliquer le fonctionnement de l'installation.



Le logigramme ci-dessous représente une bascule D type « latch » ou « verrou ».

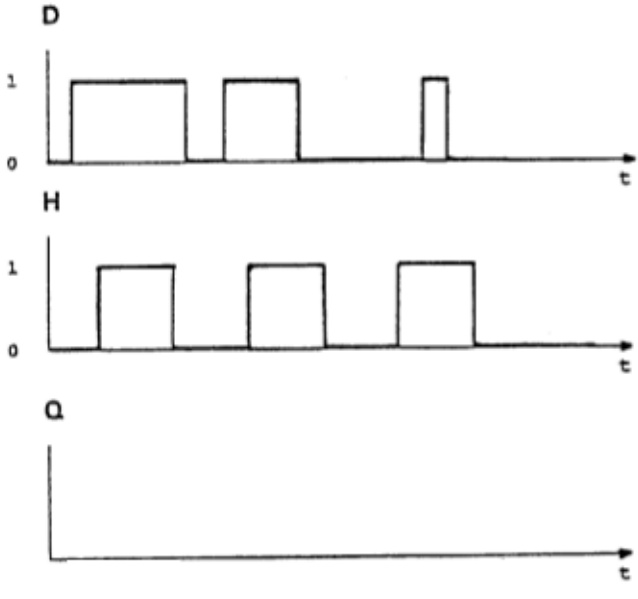
Elle comporte une entrée D, une horloge H et deux sorties Q et Q'.



1. Compléter le chronogramme présenté en annexe, en supposant Q = 0 à la date initiale.
2. Établir les expressions logiques de Q et de Q' en fonction des variables d'entrées D et H.
3. Compléter la table de vérité présentée en annexe et en déduire que :

$$Q' = \overline{Q}$$

ANNEXE



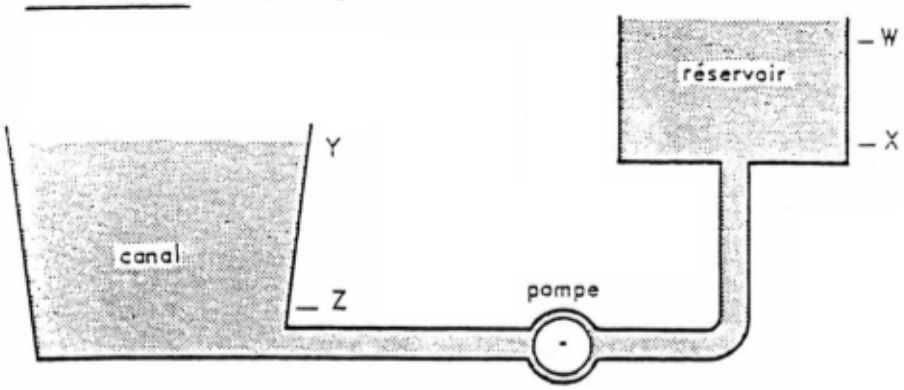
D	H	Q	Q'

Pour la bonne marche d'une réaction chimique, trois réservoirs doivent être :
- remplis ensemble de liquides
- ou vides ensemble

Si cette condition n'existe pas, le poste de contrôle doit être alerté d'urgence par une signalisation lumineuse

On peut convenir que le détecteur de niveau délivre les signaux :
- 1 réservoir plein
- 0 réservoir vide

▪ Problème n° 2 - Remplissage d'un réservoir



Capteurs : contacts ouverts si liquide en dessous du niveau

Niveau W : réservoir plein
» X : » vide
» Y : niveau de l'eau dans le canal = canal plein
» Z : » » » » » » = canal vide

Conditions à remplir :

A - Lorsque le niveau atteint W : arrêter la pompe
B - Lorsque le niveau descend en dessous de X : mettre la pompe en marche

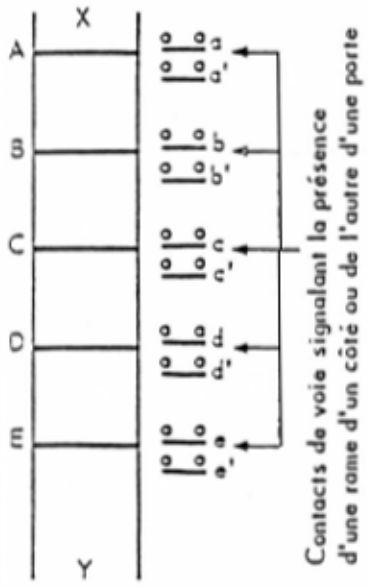
Sécurités :

1) Si le niveau tombe en dessous de Y tout en demeurant au-dessus de Z :

▪ si la pompe est en marche elle demeure en marche jusqu'à ce que le niveau atteigne W ; elle ne peut ensuite démarrer quel que soit le niveau dans le réservoir que lorsque le niveau dans le canal remonte au-dessus de Y
▪ si la pompe est arrêtée, elle ne peut démarrer que lorsque le niveau dans le canal remonte au-dessus de Y

2) Si le niveau tombe au-dessous de Z : arrêt de la pompe quel que soit son état.
Le redémarrage ne peut avoir lieu que lorsque le niveau dans le canal remonte au-dessus de Y.

▪ Problème n° 3 - Commande de SAS



Des rames de wagons circulent dans une galerie de mine.

Entre les points X et Y sont disposées 5 portes A, B, C, D, E.

Les rames s'arrêtent devant une porte fermée et démarrent dès que la porte s'ouvre.

Il s'agit de commander automatiquement la manœuvre des portes de manière à ce que deux portes consécutives ne puissent être ouvertes en même temps (SAS).

On supposera que chacune des portes, commandée électriquement, est fermée si l'électro de commande est à l'état 0 et ouverte si l'électro de commande est à l'état 1.

■ Problème n° 4 - Commande de lampes

On dispose de 3 interrupteurs A, B, C et de 3 lampes L, M, N; Les conditions de fonctionnement sont résumées c le tableau suivant :

A	B	C	L	M	N
0	0	0	0	0	0
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	1	1	0
1	1	0	1	0	0
1	0	1	0	1	0
0	1	1	1	0	1
1	1	1	1	0	0

■ Problème n° 5 - logique 2/3

Certaines sécurités, par exemple la chute d'une barre de contrôle dans un réacteur nucléaire, sont liées à 3 paramètres ; un fonctionnement optimal nécessite qu'au moins 2 sur 3 des paramètres soient réunis ; donner le schéma du circuit remplissant ces conditions.

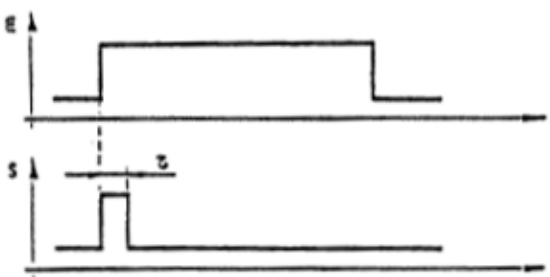
■ Problème n° 7

Réaliser un circuit OU EXCLUSIF (dilemne ou disjonction)

- avec fonctions ET, OU
- avec circuits NOR (OU)
- avec circuits NAND (ET)

■ Problème n° 8 - Réalisation de temporisateurs spéciaux

a) Le simulateur comporte un temporisateur qui diffère en sortie le signal d'attaque. Pour la réalisation de certa séquences, il est nécessaire de disposer d'un élément fournissant un état1 bref à l'apparition du signal d'atta (contact de passage à l'appel ou impulsion dérivée):



Réaliser le circuit donnant le signal S ?

■ Problème N° 9 - Détection de parité

Soient 4 informations : A, B, C et D. Une signalisation lumineuse doit être excitée seulement lorsqu'un nombre pair d'informations sont obtenues.

Autrement dit, on doit avoir :

S = 1 dans les cas suivants :

$A.B.\overline{C}.\overline{D}.$

$A.\overline{B}.C.\overline{D}.$

$A.\overline{B}.\overline{C}.D.$

$\overline{A}.\overline{B}.C.D.$

$\overline{A}.B.\overline{C}.D.$

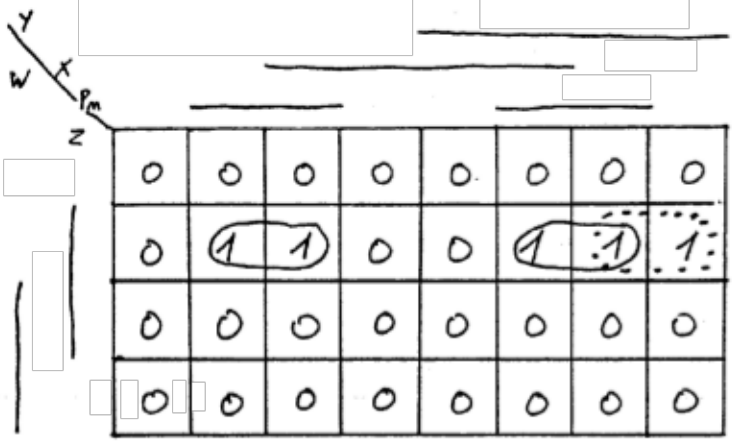
$\overline{A}.B.C.\overline{D}.$

$A.B.C.D.$

SOLUTIONS AUX PROBLEMES DE LOGIQUE COMBINATOIRE :

Pb N°2 remplissage d'un réservoir :

Utilisant l'état antérieur de la pompe, on remplit un tableau de Karnaugh avec W, X, Y, Z et P_m, dans les cases on met P_{m+1} :



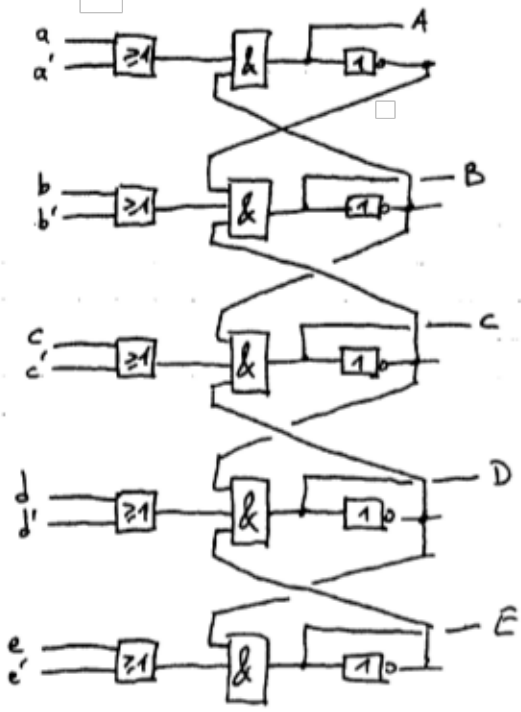
0	0	0	0	0	0	0	0
0	1	1	0	0	1	1	1
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

$$P_{m+1} = P_m . \bar{W} . Z + Y . \bar{X} . \bar{W} . Z$$

Pb N°3

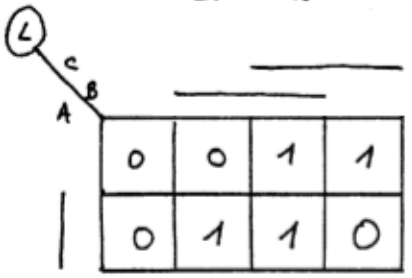
Considérons la porte D par exemples, D doit s'ouvrir si d ou d' passe à 1 et si E et C fermées soit

$$D = (d + d') . \bar{E} . \bar{C} \quad \text{etc...}$$

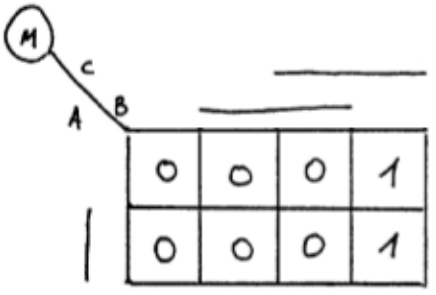


Pb N°4

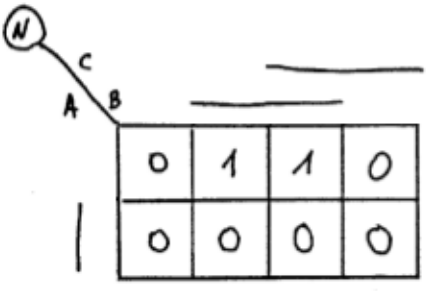
Il suffit de remplir trois tableaux de Karnaugh :



$L = AB + \bar{A}C$



$M = \bar{B}C$



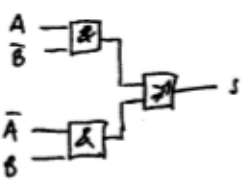
$N = \bar{A}B$

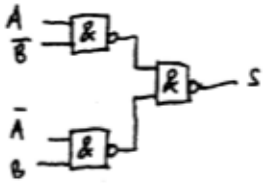
Pb N°5 $S = A.B + B.C + C.A + A.B.C = A.B + B.C + C.A$

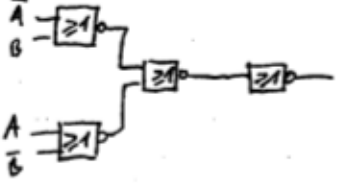
Pb N°7

A	B	S
0	0	0
1	0	1
0	1	1
1	1	0

$S = A.\bar{B} + \bar{A}.B$





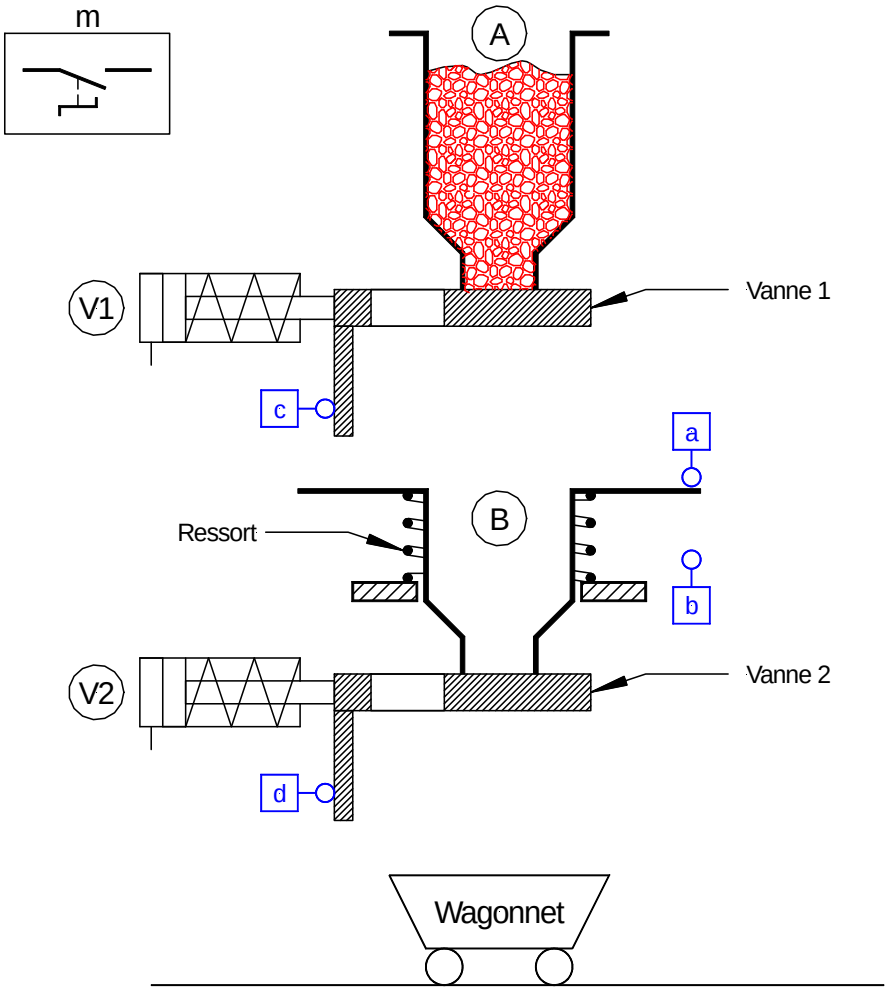


Pb N°1

$\bar{S} = A.B.C + \bar{A}.\bar{B}.\bar{C}$

2.2.1 Fonction Acquérir des informations, Traiter les informations, Commander la puissance.
3.2.1 Equations booléennes
4.1 Traitement combinatoire

Mise en situation



Ce système de dosage comporte :

- Une trémie (A) de stockage de sable.
- Une trémie (B) de pesage qui prend appui sur un ressort.
- Deux capteurs (a) et (b) détectant respectivement que la trémie B est vide ou pleine.
- Un capteur (c) actionné lorsque la vanne 1 est fermée.
- Un capteur (d) actionné lorsque la vanne 2 est fermée.
- Un interrupteur (m) pour la mise en marche.

Les vannes sont actionnées par des vérins SE et les pré-actionneurs des vérins sont MONOSTABLES.

Fonctionnement du système

Le système est représenté en position initiale : La trémie B est vide, le wagonnet est en place.

Lorsque l'ordre de marche est donné (m), la séquence de fonctionnement est la suivante :

1) Ouverture de la vanne 1
2) Le capteur (c) est libéré, le sable de la trémie (A) se déverse dans la trémie (B)
3) La trémie (B) comprime le ressort, le capteur (a) est alors libéré
4) Le capteur (b) est actionné lorsque le poids est atteint ; ce qui provoque la fermeture de la vanne 1
5) Le capteur (c) est à nouveau actionné et provoque l'ouverture de la vanne 2
6) Le capteur (d) est libéré, le sable de la trémie (B) se déverse dans le wagonnet
7) Au fur et à mesure du délestage de sa charge, la trémie (B) remonte et libère le capteur (b)
8) Le capteur (a) est actionné, ce qui provoque la fermeture de la vanne 2
9) Le capteur (d) est actionné (retour à la position initiale)

Travail demandé

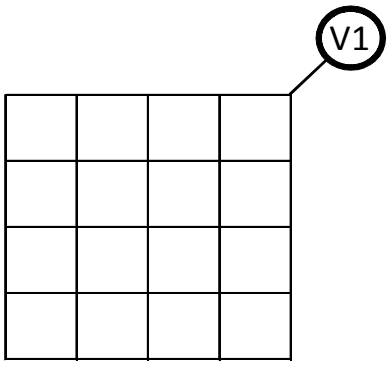
1) Etablir le tableau d'analyse en partant de la situation initiale donnée.
   Remarque :
   V1=1 ➔ Ouverture de la vanne 1 V1=0 ➔ Fermeture de la vanne 2
   V2=1 ➔ Ouverture de la vanne 2 V2=0 ➔ Fermeture de la vanne 2

2) Déterminer les équations de V1 et V2 par tableaux de Karnaugh.
   Vérifier votre résultat en utilisant l'application KARNAUGH.
3) Compléter le schéma de puissance Pneumatique et le schéma de commande électrique
4) Simuler le fonctionnement en utilisant le simulateur du Guide des Automatismes.

/2 Tableau d'analyse

a	b	c	d	V1	V2
1	0	1	1	1	0

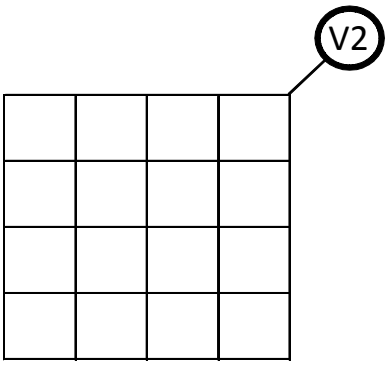
/2 Tableaux de Karnaugh



V1 =

/2

/2

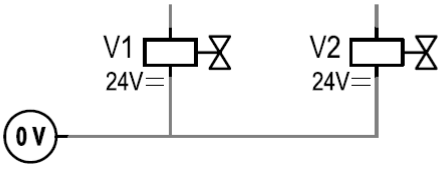


V2 =

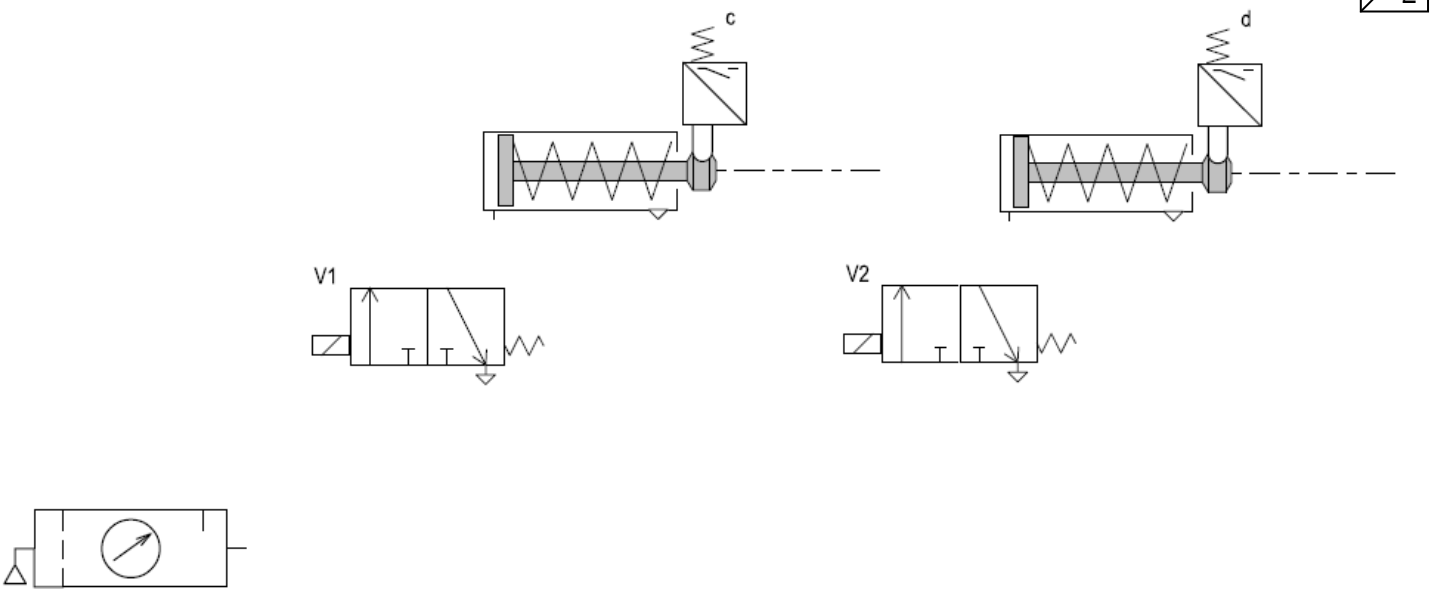
/2

Compléter le schéma de commande électrique et le schéma de puissance pneumatique.

2



2



Logique séquentielle

Sommaire

A / Présentation des automatismes séquentiels: 79

1. Structure d'un système automatisé 79
2. Le processus : 79
3. La partie commande : 79
4. Le poste de contrôle : 80
5. Exercices : 80
6. Les capteurs : 81
7. Exercices : 83
8. Les actionneurs : 83
9. Les pré-actionneurs : 85

B / Automate programmable industriel (API): 87

1. Description : 87
2. Fonctionnement cyclique : 88
3. Exercices : 88
4. Câblage des entrées / sorties. 88
5. Structure en réseau : 89
6. T.D. AUTOMATIQUE : Câblage du démarrage direct par automate d'un moteur triphasé 92
7. ANNEXE : Automate programmable S7-1200 93

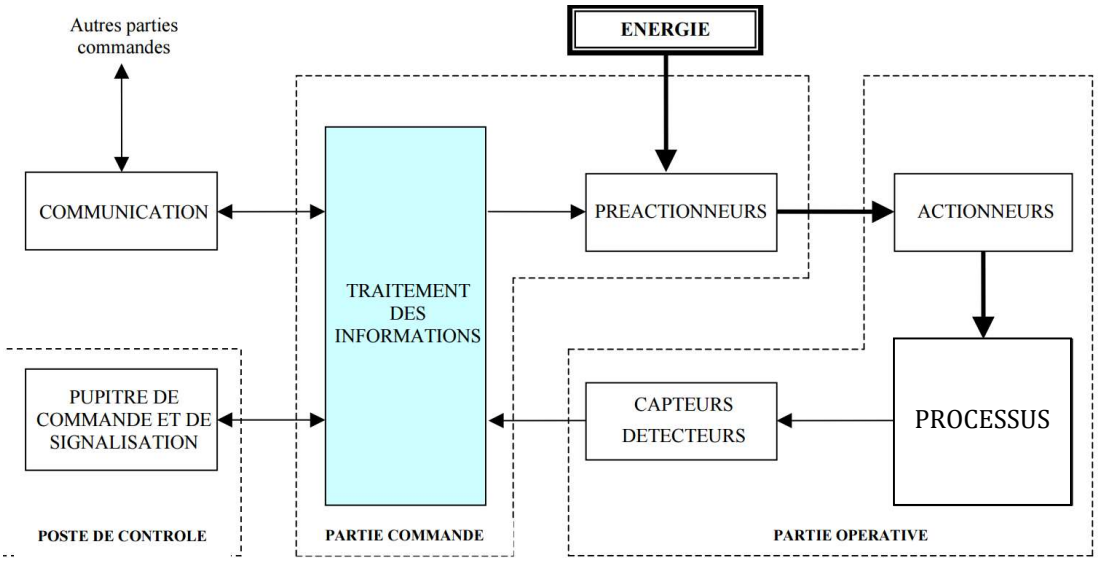
A / Présentation des automatismes séquentiels :

1. Structure d'un système automatisé

Un système automatisé se compose de deux parties qui coopèrent :

-une **partie opérative** constituée du processus à commander, des actionneurs qui agissent sur ce processus et des capteurs permettant de mesurer son état.

-une **partie commande** qui élabore les ordres pour les actionneurs en fonction des informations issues des capteurs et des consignes. Cette partie commande peut-être réalisée par des circuits câblés, ou par des dispositifs programmables (automates, calculateurs)



2. Le processus :

Un **processus** est un ensemble d'activités corrélées ou interactives qui transforme des éléments d'entrée en éléments de sortie (norme ISO 9000:2005).

Un système est le résultat d'un **processus** : par exemple le processus d'assèchement de cale comprend les pompes la tuyauterie et les vannes ...

3. La partie commande :

Elle réalise le **traitement des informations** en prenant en compte les informations renvoyées par les capteurs et en élaborant les commandes des actionneurs en fonction des ordres donnés par le poste de contrôle.

4. Le poste de contrôle :

Il réalise la **communication Homme ↔ Machine** par l'intermédiaire d'interrupteurs de commande, de LED et Voyants, d'écrans d'exploitation ...

Il existe **deux familles de l'automatique**, qui peuvent néanmoins cohabiter :
- L'automatique **analogique** pour la régulation (température d'une pièce …) non étudié ici
- L'automatique **LOGIQUE** ou Tout Ou Rien : combinatoire ou séquentielle. Les capteurs ou actionneurs sont de type TOR.

5. Exercices :

5-1 : Quel type de flux gère un circuit d'assèchement de cale ?

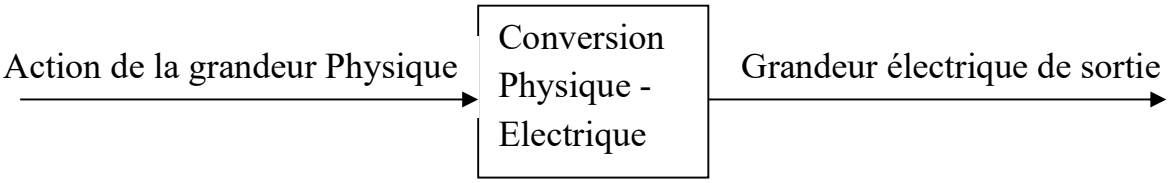
a) de la matière b) de l'énergie c) de l'information d) de la santé

5-2 : pour chacun des éléments ci-dessous, déterminer à quelle partie et sous partie ils appartiennent

Moteur électrique	Pupitre de commande et de contrôle	Partie commande
Joystick	Traitement des informations	
Détecteur de niveau	Processus à commander	
Vérin hydraulique	Actionneur	
Automate programmable	Capteur	Partie opérative
Écran	Actionneur	
Qualité de l'eau	Pupitre de commande et de contrôle	

6. Les capteurs :

Un capteur doit détecter une **grandeur physique** et transformer les variations de cette grandeur en tensions et courants **électriques** exploitables par l'unité de traitement



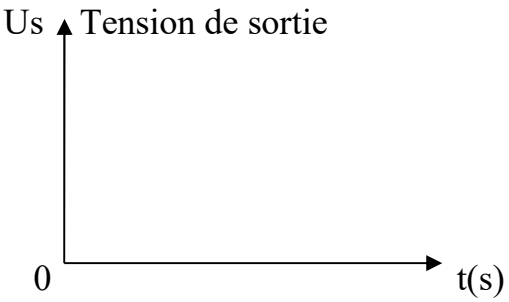
6-1 : Nature de l'information à détecter :

Force, température, débit, pression, déplacement, vitesse ...

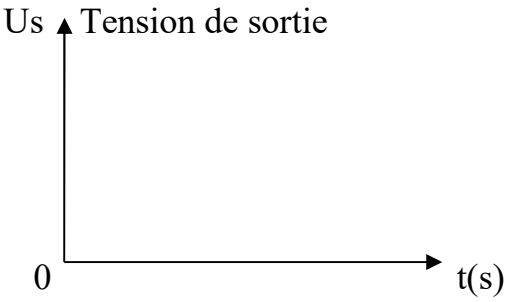
6-2 : Nature de l'information à délivrer :

Les informations à délivrer peuvent être sous forme

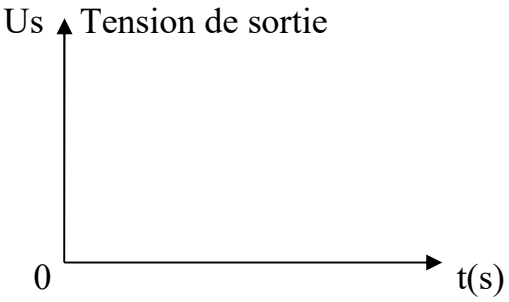
- **Tout ou Rien** (TOR) en général 0 et 5V



- **Analogique** : le signal de sortie est une tension (0 - 10 V) ou un courant (4 - 20 mA) qui donne l'image de la grandeur détectée



- **Numérique** : le signal numérique peut être sous forme binaire ou décimale ...

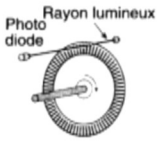
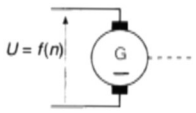
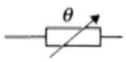


5-3 : Classification et principe de fonctionnement :

TABLEAU I. Classification des capteurs.

Repère	Désignation	Repère	Désignation
01	Déplacement	11	Torsion, couple
02	Allongement	12	Pression
03	Position linéaire et angulaire	13	Débit
04	Proximité	14	Densité
05	Épaisseur, diamètre	15	Température
06	Pesée	16	Niveau
07	Force	17	Viscosité
08	Temps, fréquence	18	Humidité des gaz et des solides
09	Vitesse, accélération	19	Analyse des liquides
10	Vibrations	20	Analyse des gaz

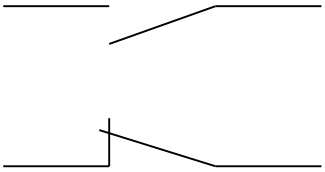
TABLEAU II. Principaux capteurs et détecteurs.

Grandeurs physiques	Désignation capteur et symbole	Principe de fonctionnement et caractéristiques	Applications
01 Déplacement	D Codeur optique	 Le mobile en déplacement entraîne un disque avec des parties opaques ou transparentes qui sont lues par une photodiode.	– Nombre d'impulsions par tour. – Détection du sens de déplacement. – Détecte également la position.
07 Force	F Jauge d'extensométrie	Une résistance est collée sur une pièce déformable soumise à la force à mesurer, la variation de longueur de la résistance est proportionnelle à la force.	L'extensométrie est utilisée pour les mesures de forces, de couple et de déformations.
08 Temps Fréquence	n n = fréquence de rotation	 Une génération à courant continu est entraînée par le système dont on veut connaître la vitesse ; sa tension de sortie est directement proportionnelle à la vitesse.	Asservissement de vitesse, dans les systèmes à vitesse variable
12 Pression	p Pressostat	L'air sous pression déforme une membrane qui actionne un contact, ou commande une variation de résistance.	Contrôle de la pression d'air, ou de la pression d'eau, d'huile.
13 Débit	qv Débimètre	Une turbine placée dans une canalisation tourne à une vitesse variable en fonction du débit de liquide, elle entraîne une dynamotachymétrique.	Contrôle du débit des pompes, maxi, mini et proportionnel.
15 Température	Ø Thermostat	  – Dispositif à bilame basé sur les dilatations différentes. – Thermistances (résistance variable en fonction de t°C). – Couples thermoélectriques f.é.m. fonction de la température.	Températures de 0 à 100° C Températures de –50 à +250° C Températures de 100 à 1 500° C

6-3 : Les capteurs TOR

Leur fonctionnement peut s'assimiler à celui d'un **interrupteur**

Normalement Ouvert (NO) ou contact à fermeture



Il délivre 0V non actionné et +5V (ou plus) lorsqu'il est activé

Normalement Fermé (NF) ou contact à ouverture

Il délivre 5V non actionné et 0 V lorsqu'il est activé

7. Exercices :

Quelle est la nature du signal délivré en sortie (TOR, Analogique, Numérique, temporisé) par un capteur de :

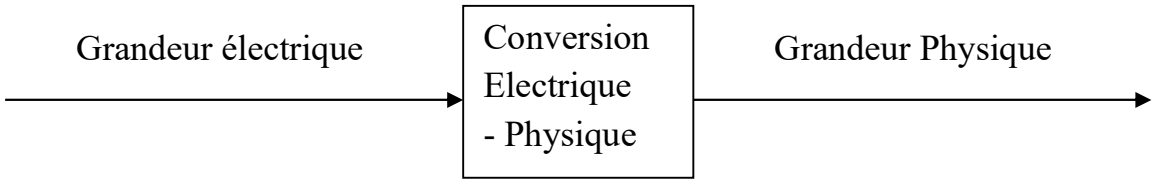
7.1) position angulaire code Gray :

7.2) Température :

7.3) interrupteur de position :

8. Les actionneurs :

Un actionneur doit restituer une **grandeur physique** correspondant à la grandeur **électrique** (tension et courant) qu'il a reçu de l'unité de traitement.

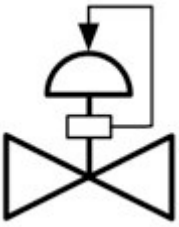


7-1 : Nature de l'information restituée :

Force, chaleur, froid, rotation, déplacement, ...

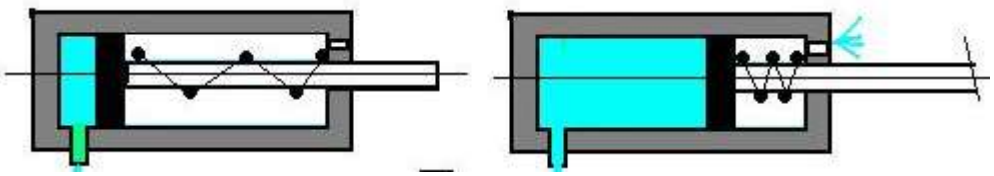
7-2 : Actionneurs TOR :

7.2.1 : Moteurs : soit à l'arrêt, soit en marche à vitesse constante, **Vannes** : soit ouvertes soit fermées, ...

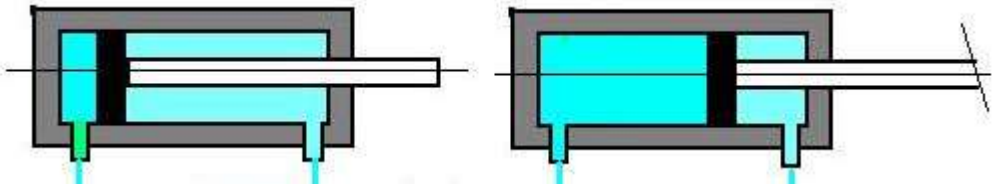
	Moteur à courant .....
	Moteur à courant .....
	Moteur à courant alternatif ..... à rotor en court-circuit
	Vanne . . . . .

7.2.2 : Vérin : soit rentré soit sorti

Simple effet : Il ne présente qu'un seul orifice d'admission de la pression, le retour en position de repos après coupure de la pression est assuré par un ressort



Double effet : Il présente 2 orifices d'admission de la pression, il peut recevoir la pression alternativement sur chaque face du piston pour rentrer ou sortir la tige.



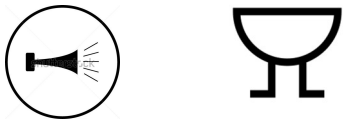
7.2.3) Organes de **signalisation :**

- Visuelle :

Lampe, LED (Allumée ou éteinte)

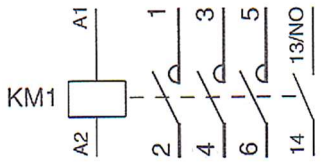
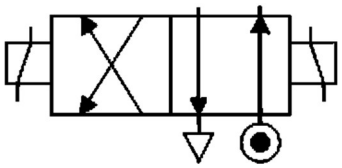
- Sonore :

Klaxon, Buzzer



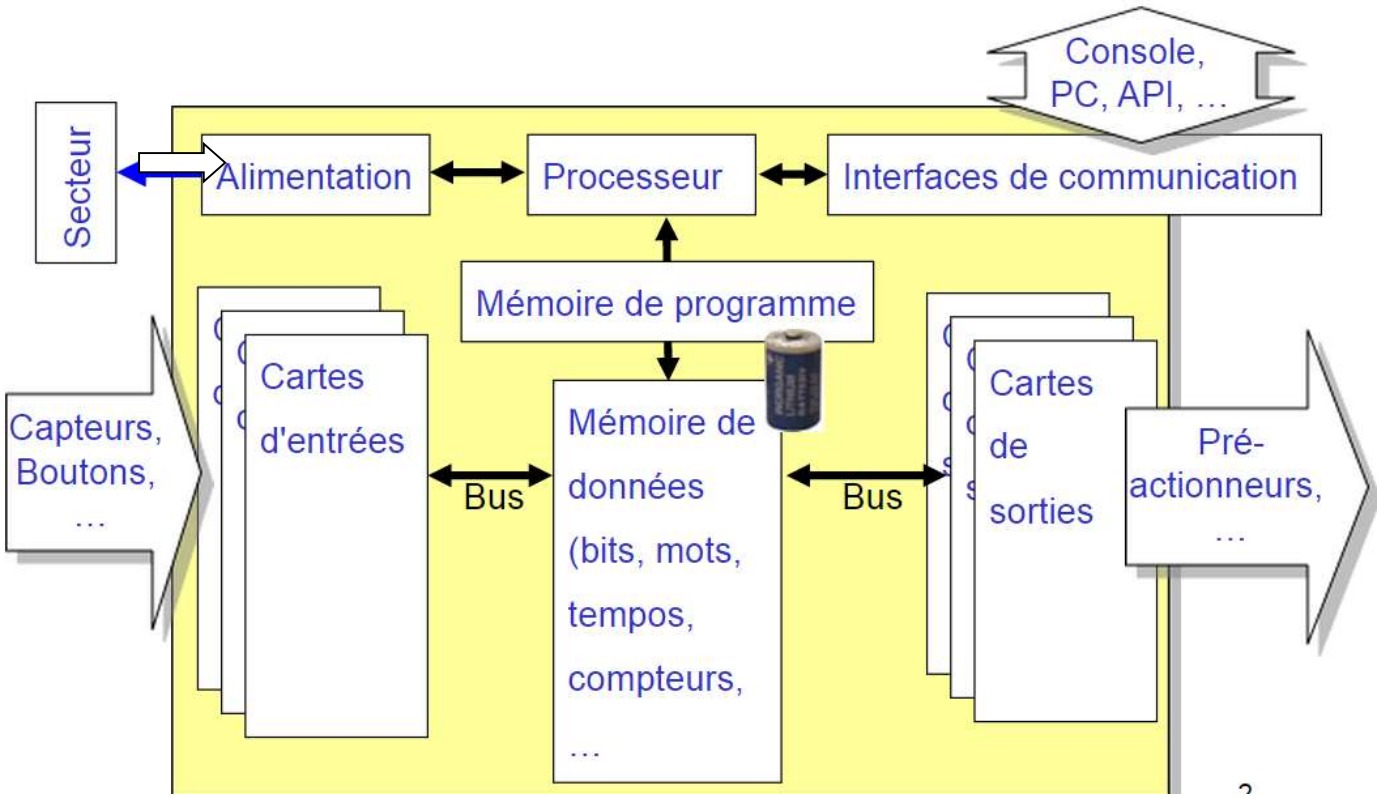
9. Les pré-actionneurs :

Ils assurent l'interface entre la partie commande (basse tension) et la partie Puissance.

	Relais contacteur
	Distributeur pour vérin et pour vanne

B / Automate programmable industriel (API):

1. Description:



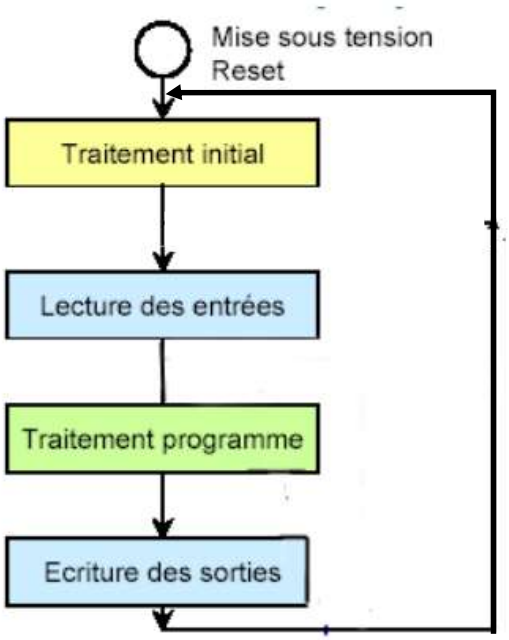
Le cœur de l'automate est un **processeur** qui exécute les instructions contenues dans la mémoire de programme

La **mémoire de programme** est non volatile (EEPROM) c'est à dire que son contenu ne s'efface pas à la disparition de l'alimentation

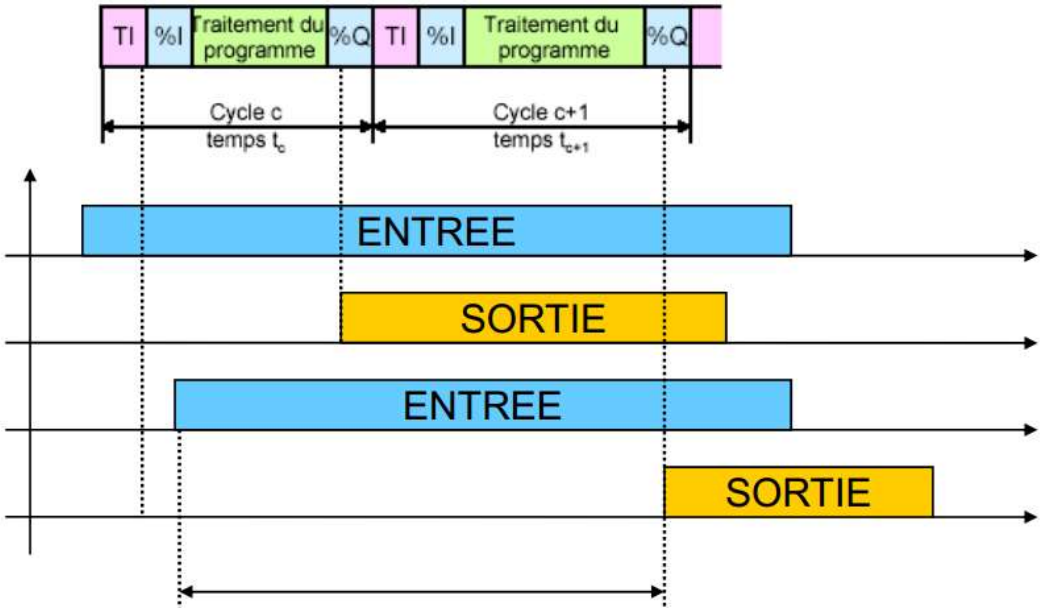
La **mémoire de données** contient les valeurs temporaires et intermédiaires. Elle est volatile (RAM).

2. Fonctionnement cyclique :

L'automate exécute son programme de manière cyclique



Temps de réponse :



Lorsque l'évolution d'une entrée doit activer une sortie, il peut s'écouler jusqu'à 2 x le temps de cycle avant que la sortie ne s'active.

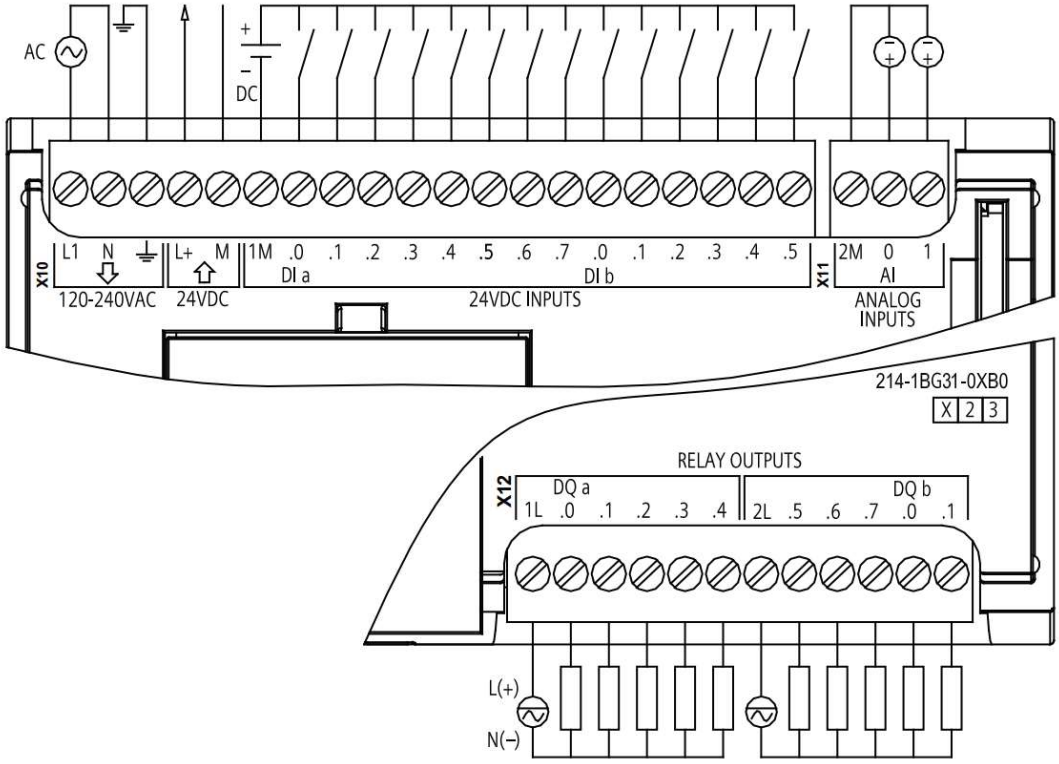
3. Exercices :

En utilisant l'extrait de documentation technique donné en annexe, préciser :

3-1 : Les automates de la série siemens s7 1200 sont de type monobloc : Vrai Faux
3-2 : Définition de A.P.I. ?
3-3 : Donner le nombre d'entrées et sorties TOR sur un CPU 1214C ainsi que le nombre d'Entrées et sorties analogiques.

4. Câblage des entrées / sorties.

Le schéma ci-dessous représente un exemple possible de câblage de l'automate siemens S1200.



1. Repérer sur le schéma :

- **L'alimentation** de l'automate.
- L'emplacement des **entrées TOR**. Combien l'automate en dispose-t-il ?
- L'emplacement des **entrées ANALOGIQUES**. Combien l'automate en dispose-t-il ?
- L'emplacement des **sorties TOR**. Combien l'automate en dispose-t-il ?
- **L'alimentation** pour les entrées. Est-elle utilisée dans notre exemple ?

2. L'automate fourni-t-il la tension nécessaire aux sorties ?

3. Les sorties doivent-elles toutes fonctionner sous la même tension d'alimentation ?

4. Dessiner les schémas simplifiés internes de l'automate :

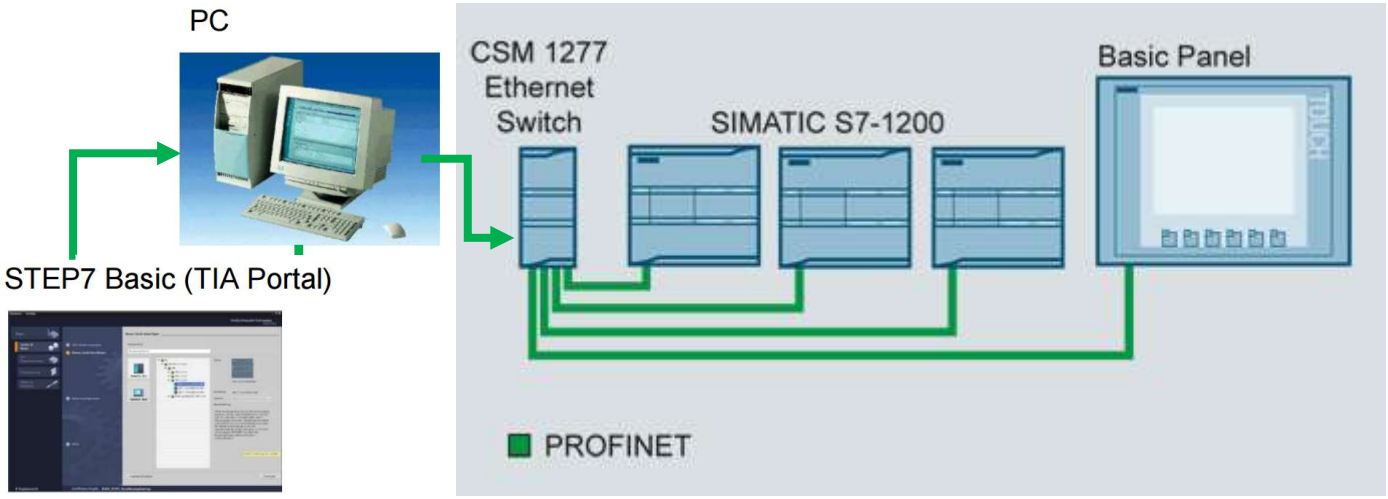
- Bornier X10 entre 1M et .2

- Bornier X12 entre 1L et .2

PS : certains capteurs possèdent 3 fils ou plus (nécessité d'alimentation séparée), il est important de se référer au mode d'emploi de l'automate pour leur branchement.

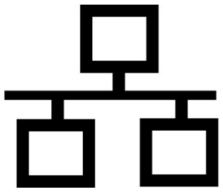
5. Structure en réseau :

L'automate S7 1200 communique avec le PC de programmation, l'interface HMI et éventuellement d'autres automates via un réseau Profinet (version industrielle d'Ethernet).



Sur un réseau tous les éléments sont connectés entre eux sur un câble unique.

Il est donc nécessaire de les identifier en leur attribuant une adresse unique :



On distinguera plusieurs types d'adresse :

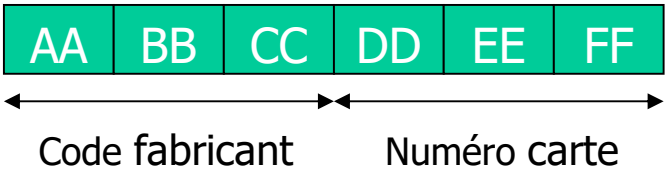
- Adresse physique
- Adresse réseau
- Numéro de port
- Nom, URL, adresse email

5.1) Adresse physique

Permet d'identifier une carte de communication sur le réseau

- Figée et intégrée dans la carte
- Unique et universelle
- Composée de 6 codes hexadécimaux (6x8=48 bits)

@ physique ou @ MAC :



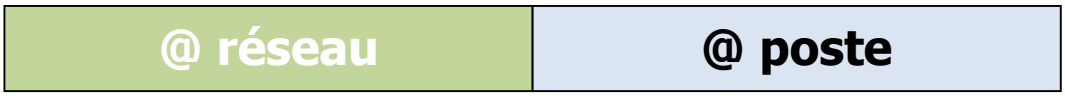
Problème : L'adresse physique MAC ne donne aucune indication sur la situation géographique du poste possédant la carte concernée.

Le bloc d'information étant propagé sur l'ensemble du réseau pour atteindre le poste concerné il faut utiliser une adresse réseau pour aiguiller le bloc d'information sur le réseau concerné

5.2) Adresse réseau

Identifier de manière unique un poste sur un réseau

Cette adresse logique est composée de 2 champs :



5.3) Les protocoles TCP-IP

Qu'est-ce qu'un protocole ?

Un code de conduite permettant la coordination harmonieuse des comportements (Le code de la route);

Une langue qui partagée permet de communiquer (le français ou l'anglais).

TCP-IP est une suite de code ou de langue qui permettent aux ordinateurs du monde entier d'entrer en communication les uns avec les autres et d'échanger de l'information… quel que soit leur fabricant, leur modèle ou leur système d'exploitation.

TCP Transmission Control Protocol **IP** Internet Protocol

Profinet utilise le protocole TCP IP pour identifier les automates et les HMI sur le réseau.

5.4) Adresse IP

Une adresse IP est une adresse 32 bits notée sous forme de 4 nombres décimaux entiers variant de 0 à 255 séparés par des points. On distingue en fait deux parties dans l'adresse IP:

Une partie des nombres à gauche désigne le réseau (on l'appelle netID)
Les nombres de droite désignent les machines de ce réseau (on l'appelle host-ID)
Ex d'adresse : 192.168.0.1

Pour distinguer le netID de l'host-ID on utilise un Masque de sous réseau :
Le masque de sous-réseau est composé de 4 nombres décimaux variant aussi de 0 à 255, séparés par un point. Par exemple, « 255.255.255.0 »

L'adresse réseau est le résultat d'un ET logique entre l'adresse IP et le masque de sous réseau.
Seules les machines possédant la même adresse réseau (netID) peuvent communiquer entre elles. Elles sont distinguées les unes des autres par leur adresse hôte (host-ID)

5.5) Exercice :

5.5.1: Quelle est l'adresse réseau d'un API ayant comme Adresse IP 192.168.0.1 et comme masque de sous réseau 255.255.255.0 ?

5.5.2 : Quelle est son adresse hôte ?

5.5.3 : Peut il communiquer avec une HMI d'adresse IP 192.168.1.2 ? (Même masque).

5.5.4 : Proposer une adresse IP compatible pour l'HMI pour pouvoir échanger des données avec l'API, ainsi que pour l'ordinateur.

5.5.6 : Avec combien d'autres hôtes l'API pourrait-il communiquer sachant que l'on n'attribue pas à des hôtes les adresses réservées "0" (adresse de base du réseau) ni "255" (adresse de broadcast).

6. T.D. AUTOMATIQUE : Câblage du démarrage direct par automate d'un moteur triphasé

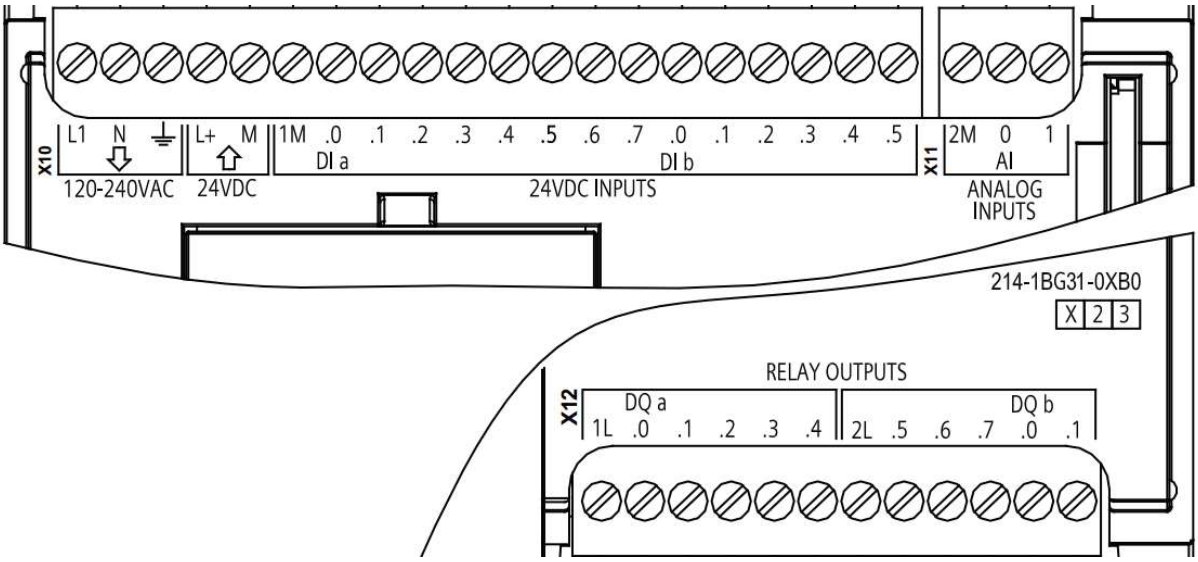
On souhaite réaliser le circuit de démarrage d'un moteur triphasé par commande Marche Arrêt.

On dispose d'un automate ref 214-1BG31-0XB0, un interrupteur Marche NO, Arrêt NF, d'un voyant 24V signalant la mise en Marche, un relais K1 24V 1RT, un moteur triphasé 3x400V, un contacteur de puissance triphasé K2.

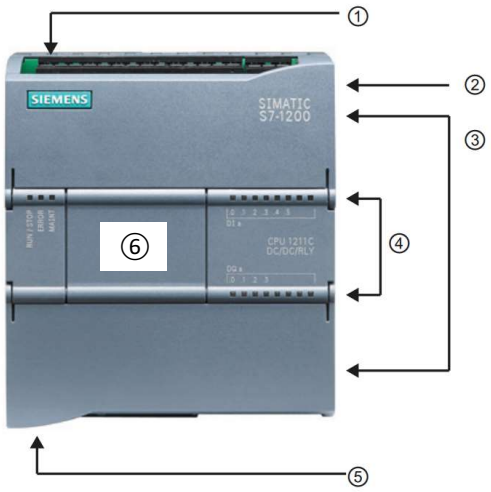
Les alimentations disponibles sont : monophasé 240V et triphasé 3x400V

1) En utilisant les symboles normalisés de chacun des éléments, réaliser le câblage de l'ensemble (schémas de commande et de puissance)

2) Ecrire le Ladder de commande correspondant.



7. ANNEXE : Automate programmable S7-1200



① Prise d'alimentation

② Logement pour carte mémoire sous le volet supérieur

③ Connecteurs amovibles pour le câblage utilisateur (derrière les volets)

④ DEL d'état pour les E/S intégrées

⑤ Connecteur PROFINET (sur la face inférieure de la CPU)

⑥ Emplacement pour carte "Signal Board" permettant l'ajout d'E/S TOR ou Analogiques supplémentaires

Tableau 1- 1 Comparaison des modèles de CPU

Caractéristique		CPU 1211C	CPU 1212C	CPU 1214C	CPU 1215C
Dimensions (mm)		90 x 100 x 75	90 x 100 x 75	110 x 100 x 75	130 x 100 x 75
Mémoire utilisateur	de travail	30 Ko	50 Ko	75 Ko	100 Ko
	de chargement	1 Mo	1 Mo	4 Mo	4 Mo
	rémanente	10 Ko	10 Ko	10 Ko	10 Ko
E/S intégrées locales	TOR	6 entrées/4 sorties	8 entrées/6 sorties	14 entrées/10 sorties	14 entrées/10 sorties
	Analogiques	2 entrées	2 entrées	2 entrées	2 entrées/2 sorties
Taille de la mémoire image	Entrées (I)	1024 octets	1024 octets	1024 octets	1024 octets
	Sorties (Q)	1024 octets	1024 octets	1024 octets	1024 octets
Mémentos (M)		4096 octets	4096 octets	8192 octets	8192 octets
Modules d'entrées-sorties (SM) pour extension		Aucun	2	8	8
Signal Board (SB), Battery Board (BB) ou Communication Board (CB)		1	1	1	1
Module de communication (CM) (extension vers la gauche)		3	3	3	3
Compteurs rapides	Total	3 intégrés, 5 avec SB	4 intégrés, 6 avec SB	6	6
	Monophase	3 à 100 kHz SB : 2 à 30 kHz	3 à 100 kHz 1 à 30 kHz SB : 2 à 30 kHz	3 à 100 kHz 3 à 30 kHz	3 à 100 kHz 3 à 30 kHz
	Quadrature de phase	3 à 80 kHz SB : 2 à 20 kHz	3 à 80 kHz 1 à 20 kHz SB : 2 à 20 kHz	3 à 80 kHz 3 à 20 kHz	3 à 80 kHz 3 à 20 kHz
Sorties d'impulsions[1]		4	4	4	4
Carte mémoire		Carte mémoire SIMATIC (facultative)			
Durée de conservation de l'horloge temps réel		20 jours, typ. / 12 jours min. à 40 degrés C (supercondensateur sans maintenance)			
PROFINET		1 port de communication Ethernet			2 ports de communication Ethernet

Langages de programmation grafcet

Sommaire

Langages de programmation 97

1. Automate programmable industriel (API): 97
2. Objets manipulés 97

3.1) Repérage des entrées et des sorties: 97

3.2) Adressage des variables Siemens: 98

3. Le langage à contact ou Ladder 98

3.1) Représentation des éléments principaux: 99

3.2) Exercice : 100

3.3) Fonctions de temporisation: 101

3.3.1 Retard à l'enclenchement : Mode TON 102

3.3.2: Retard au déclenchement : Mode TOF 102

3.3.3: Monostable : Mode TP 102

3.4) Exercice : 103

3.5) Fonctions de comptage : 104

3.6) Fonctions avancées : 106

3.6.1 Comparateurs 106

4. Le GRAFCET : 107

4.1 Définition du GRAFCET : 107

4.2. Le point de vue : 107

Point de vue système (procédé et processus) (Niveau 1) 107

Point de vue partie opérative 107

Point de vue partie commande (Niveau 2) 108

4.3. Constitution 109

4.4 Règles d'évolution du Grafcet 109

4.5 Actions associées aux étapes 110

4.5.1 Ordre continu 110

4.5.2 Ordre conditionnel 110

4.5.3 Ordre de mémorisation de l'action 110

4.6 Transitions 110

4.7 Séquences de base 111

4.7.1: Le Grafcet linéaire 111

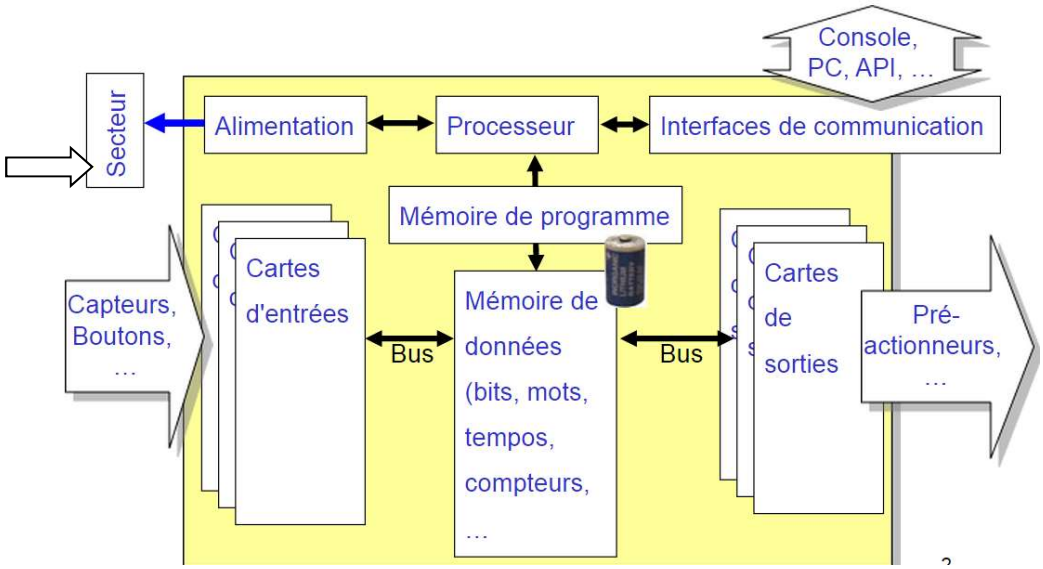
4.7.2 Divergence et convergence en "OU". 111

4.7.3 Divergence et convergence en "ET" 112

4.8 Exercices : Indiquer les erreurs réalisées dans les grafcets ci dessous et proposer les corrections à apporter : ..113
5. Le GRAFCET évolué : 114
5.1 : Les temporisations : 114
Syntaxe d'utilisation : 114
Illustration : 114
5.2 : Les compteurs : 115
Syntaxe d'utilisation : 115
Illustration : 115
5.3 : Grafcet partitionnés : 116
Syntaxe d'utilisation : 116
Illustration : 116
Analyse : 116
5.4 Exercices : 117
Marche 117
↑X12 117
synthèse 122
annexe fonctionnement de l'API 124

Langages de programmation

1. Automate programmable industriel (API):



L'automate nécessite une programmation dans un langage adapté.

Les langages de programmation sont littéraux : langage littéral structuré ou graphiques : Ladder, Grafcet.

La programmation consiste à définir les relations entre les entrées (capteurs) et les sorties (Pré-actionneurs, actionneurs ...)

2. Objets manipulés

3.1) Repérage des entrées et des sorties:

Le repérage ou adressage, c'est le repère correspondant à l'emplacement de chaque entrée et sortie ainsi son adresse en mémoire où est stockée l'image de son état 0 ou 1, cela permet d'utiliser plusieurs fois l'entrée ou la sortie dans le programme.

Un automate ayant 14 entrées et 10 sorties seront aux adresses suivantes :

Entrées : % I0.0 ; %I0.1 ; %I0.2 ; ... ; %I0.7 ; %I1.0 ; %I1.1 ; ..... ; %I1.5

% : donnée binaire (bit) I : Input 1 : N° du module .n : N° de la voie

Sorties : %Q0.0 ; %Q0.1 ; %Q0.2 ; ... ; %Q0.7 ; %Q1.0 ; %Q1.1

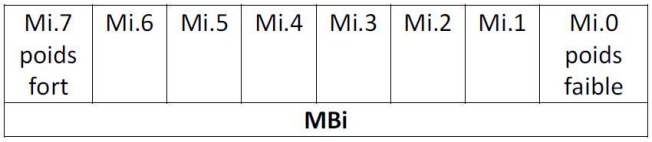
Q : Output

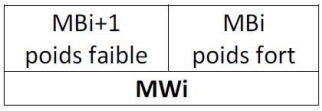
3.2) Adressage des variables Siemens:

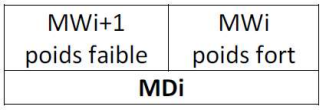
- Bits internes %M0.0 à %M255.7 dépendants des mots suivants
- Octets internes : ensemble de 8 bits %MBi
- Mots internes : ensemble de 16 bits %MWi
- Mots doubles : ensemble de 32 bits %MDi

Principe de l'adressage des mots interne

Poids fort et poids faible :







s : (**attention aux chevauchements des mots**)

3. Le langage à contact ou Ladder

Le langage à contact est adapté à la programmation de traitements logiques, il utilise le schéma développé.

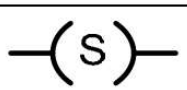
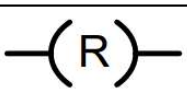
Nous retrouvons : La fonction ET en utilisant des contacts en série
La fonction OU en utilisant des contacts en parallèle

3.1) Représentation des éléments principaux:

Entrées :

Graphe	Désignation	Fonction	Schéma à contact
	Contact à fermeture	contact passant quand il est actionné	
	Contact à ouverture	contact passant quand il n'est pas actionné	
	connexion horizontale	permet de relier les éléments action série	
	connexion verticale	permet de relier les éléments action en parallèle	
	Impulsion Positive		
	Impulsion Négative		

Sorties :

Graphe	Désignation	Fonction	Schéma à contact
	bobine directe	la sortie prend la valeur du résultat logique	
	bobine inverse	la sortie prend la valeur inverse du résultat logique	
	bobine d'enclenchement	le bit interne est mis à 1 et garde cet état	
	bobine déclenchement	le bit interne est mis à 0 et garde cet état	

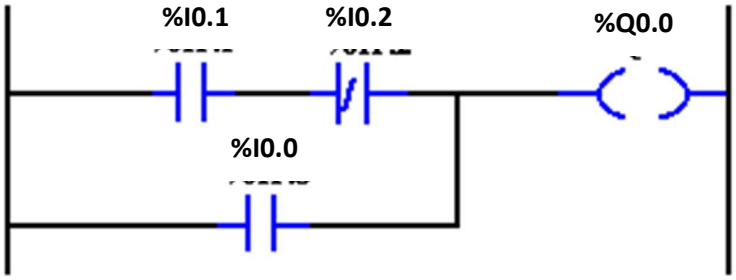
Un bit interne étant une mémoire logique prenant la valeur 0 ou 1

Une bobine d'enclenchement S « set » et bobine de déclenchement R « reset » correspondent à un bistable RS (bascule RS).

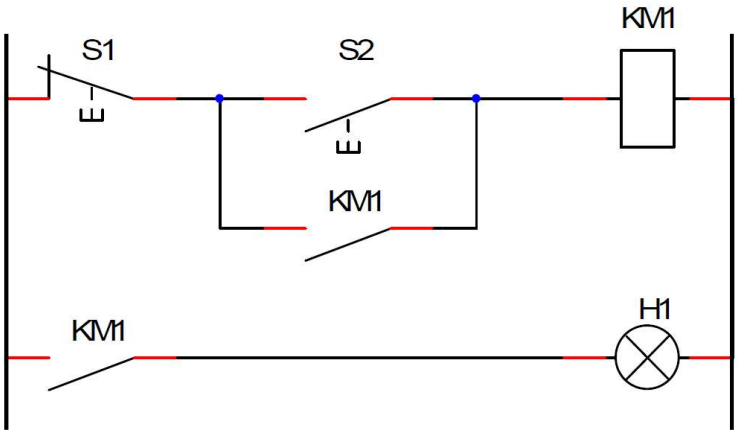
Le réseau à contact s'inscrit entre deux barres verticales représentant la tension d'alimentation.

3.2) Exercice :

3.2.1 Déterminer l'équation de %Q0.0 :



3.2.2 : Soit le schéma à contact ci contre :



a) Combien d'Entrées et de Sorties le schéma comporte t'il ?

b) Proposer une affectation de bits pour chacune des entrées et sorties

c) Donner l'équation de KM1. S'agit t'il d'un problème combinatoire ou séquentiel ? Justifier.

d) Dessiner le schéma Ladder correspondant au schéma à contact.

e) Proposer un second Ladder utilisant des bobines d'enclenchement de déclenchement respectant la priorité du schéma à contact.

f) Donner le schéma normalisé correspondant.

3.3) Fonctions de temporisation:

Ce sont des blocs configurables nommés IEC_Timer_i pour réaliser des retards, des timers, des impulsions précises ...

Le nombre de temporisations qu'il est possible d'utiliser dans le programme utilisateur est limité uniquement par la quantité de mémoire dans la CPU.

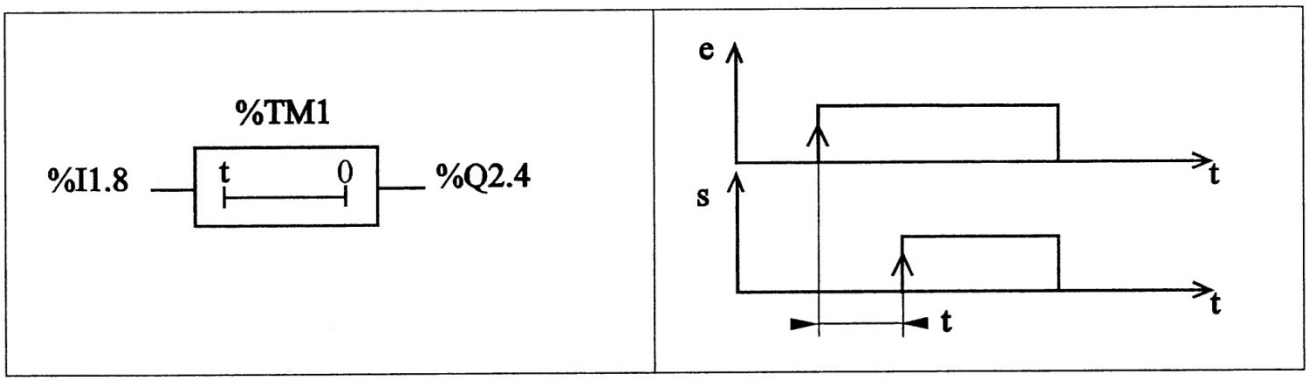
Chaque temporisation utilise une structure de DB de 16 octets de type de données IEC_Timer pour conserver les données de la temporisation, cette structure étant indiquée au-dessus de la boîte ou de la bobine.

Le programme de développement STEP 7 crée automatiquement le DB lorsque l'on insère le bloc.

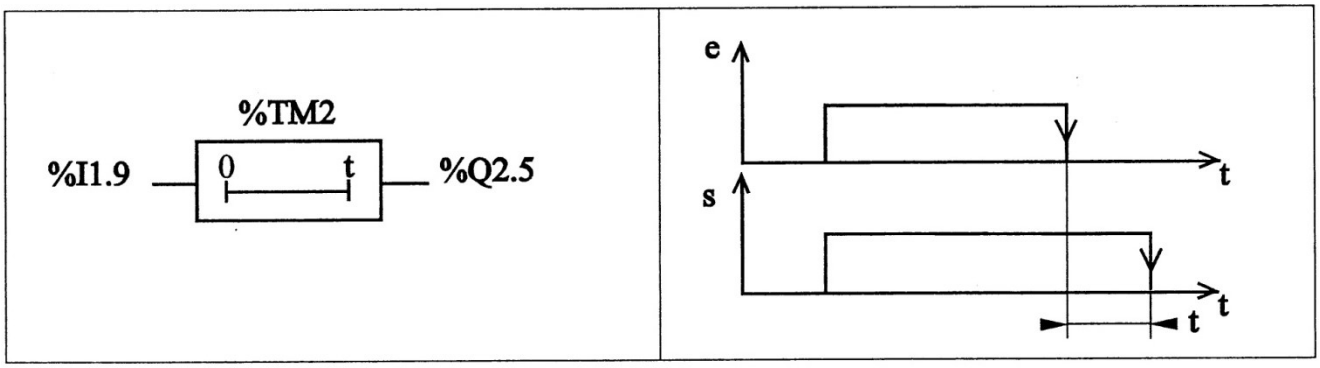
Principaux blocs de temporisation :

Boîtes CONT/LOG	Bobines CONT	Description
IEC_Timer_0 TP Time IN Q PT ET	TP_DB TP "PRESET_Tag"	La temporisation TP génère une impulsion de durée prédéfinie.
IEC_Timer_1 TON Time IN Q PT ET	TON_DB TON "PRESET_Tag"	La temporisation TON met la sortie Q à 1 après un temps de retard prédéfini.
IEC_Timer_2 TOF Time IN Q PT ET	TOF_DB TOF "PRESET_Tag"	La temporisation TOF met la sortie Q à 0 après un temps de retard prédéfini.

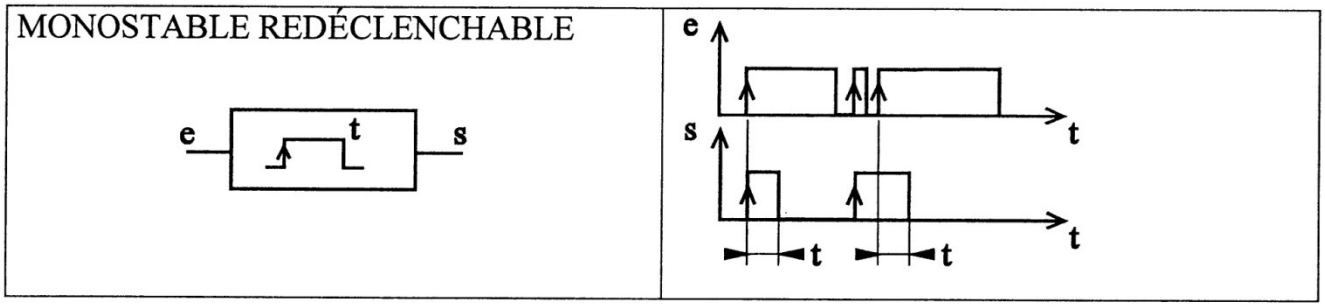
3.3.1 Retard à l'enclenchement : Mode TON

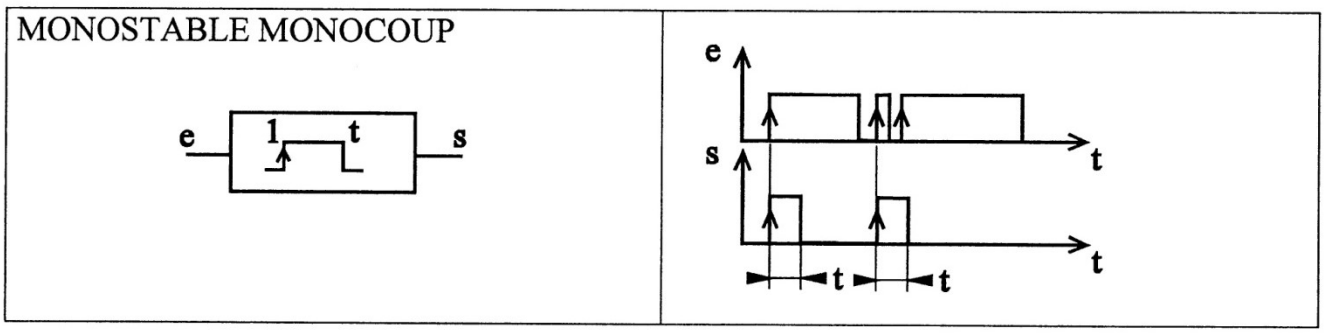


3.3.2: Retard au déclenchement : Mode TOF



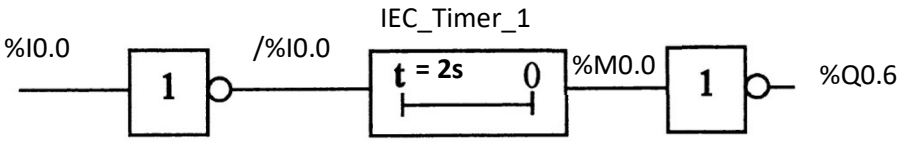
3.3.3: Monostable : Mode TP

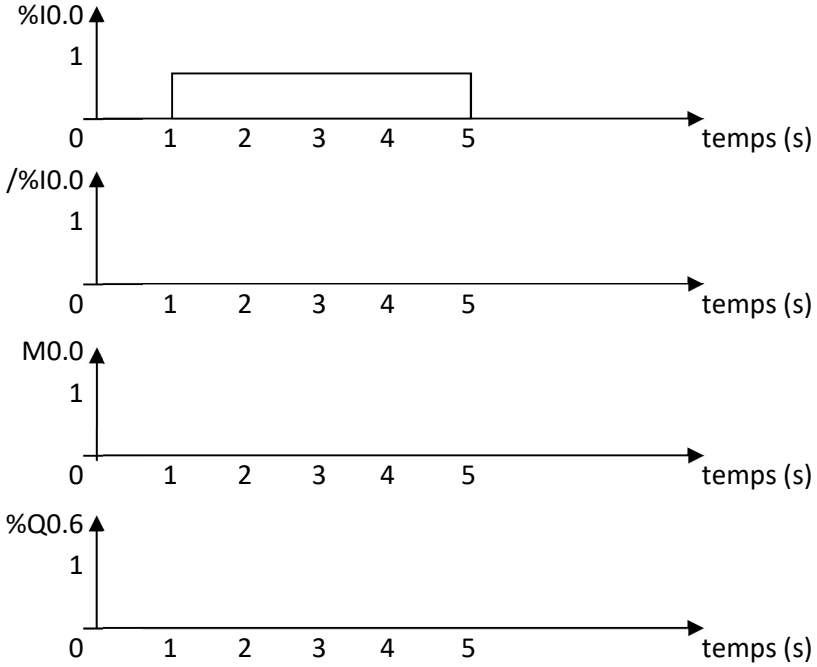




3.4) Exercice :

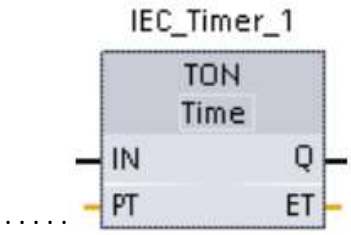
3.4.1 a) Compléter le chronogramme donné pour le montage ci contre :



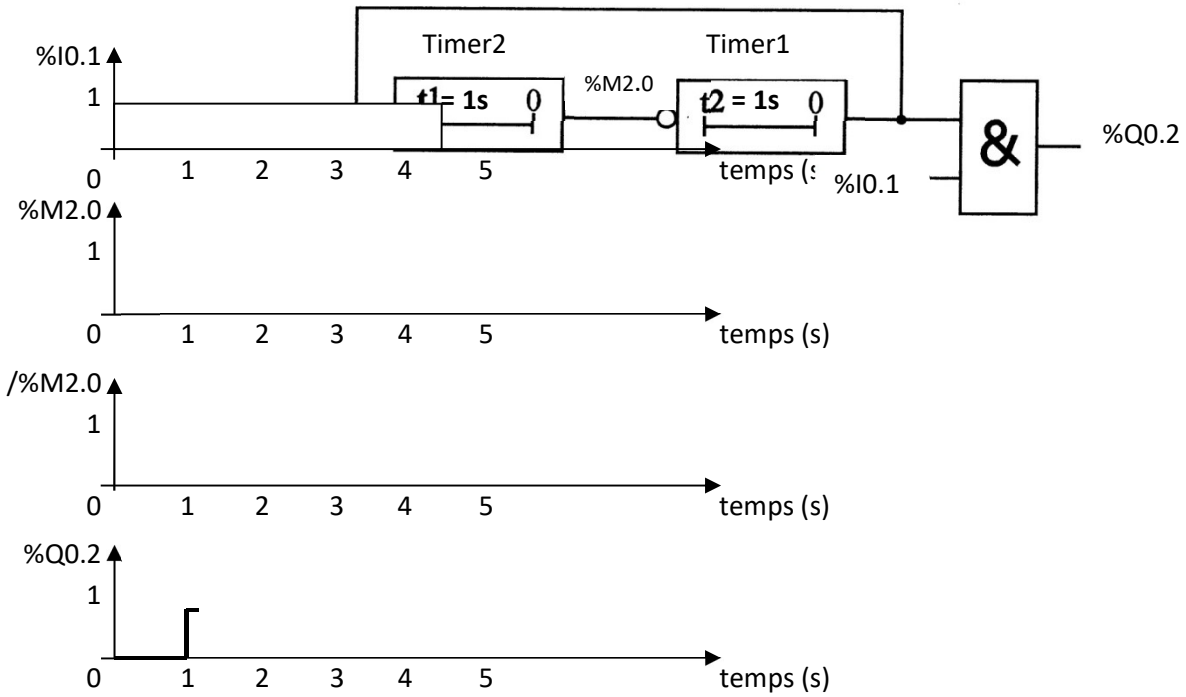


b) Quelle est la fonction réalisée ?

c) Donner le schéma ladder complet:



3.4.2 a) Compléter le chronogramme donné pour le montage ci contre :



b) Quelle est la fonction réalisée ?

3.5) Fonctions de comptage :

CONT/LOG	Description
"Counter name" CTU SInt CU Q R CV PV	Utilisez les compteurs pour compter des événements de programme internes et des événements de processus externes. Chaque compteur utilise une structure sauvegardée dans un bloc de données afin de conserver les données du compteur. Vous affectez le bloc de données lors du placement du compteur dans l'éditeur. • CTU est un compteur de comptage. • CTD est un compteur de décomptage. • CTUD est un compteur de comptage et de décomptage.
"Counter name" CTD SInt CD Q LOAD CV PV	
"Counter name" CTUD SInt CU QU CD QD R CV LOAD PV	

Paramètre	Type de données[1]	Description
CU, CD	Bool	Compter ou décompter, d'une valeur de 1
R (CTU, CTUD)	Bool	Remise à zéro de la valeur de comptage
LD (CTD, CTUD)	Bool	Commande de chargement pour la valeur prédéfinie
PV	SInt, Int, DInt, USInt, UInt, UDInt	Valeur de comptage prédéfinie
Q, QU	Bool	Vrai si CV >= PV
QD	Bool	Vrai si CV <= 0
CV	SInt, Int, DInt, USInt, UInt, UDInt	Valeur de comptage en cours

La plage numérique des valeurs de comptage dépend du type de données sélectionnée :

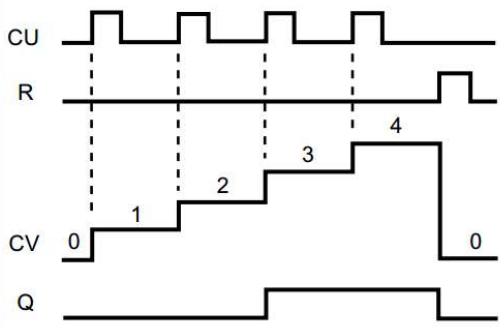
- Si la valeur de comptage est un entier non signé, vous pouvez décompter jusqu'à zéro ou compter jusqu'à la limite de plage.

- Si la valeur de comptage est un entier signé, vous pouvez décompter jusqu'à la limite entière négative et compter jusqu'à la limite entière positive.

Le nombre de compteurs qu'il est possible d'utiliser dans le programme utilisateur est limité uniquement par la quantité de mémoire dans la CPU.

Les compteurs utilisent la quantité de mémoire suivante :
- Un compteur utilise 3 octets pour les types de données 8bits.
- Un compteur utilise 6 octets pour les types de données 16 bits
- Un compteur utilise 12 octets pour les types de données 32 bits

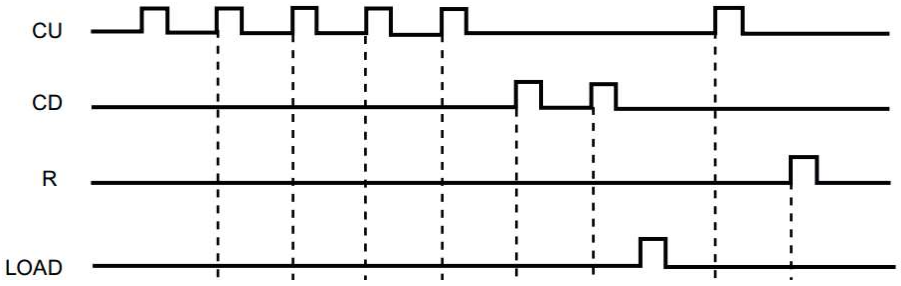
Exemple d'utilisation en mode comptage :

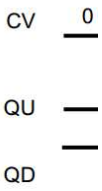
Compteur	Fonctionnement
Le compteur CTU incrémente de 1 lorsque la valeur du paramètre CU passe de 0 à 1. Le chronogramme illustre le fonctionnement d'un compteur CTU avec une valeur de comptage entière non signée (PV = 3). • Si la valeur du paramètre CV (valeur de comptage en cours) est supérieure ou égale à la valeur du paramètre PV (valeur prédéfinie), le paramètre de sortie Q du compteur est égal à 1. • Si la valeur du paramètre de réinitialisation R passe de 0 à 1, la valeur de comptage en cours est remise à 0.	

Exemple d'application :

Un compteur CTUD est programmé à PV= 4 et reçoit les signaux CU, CD, R et LOAD donnés ci dessous:

Compléter les chronogrammes des signaux CV, QU et QD.





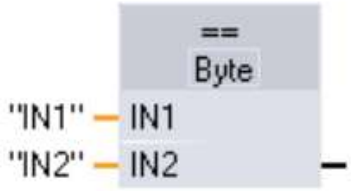
3.6) Fonctions avancées :

3.6.1 Comparateurs

Ils permettent de réaliser des comparaisons entre expressions numériques "IN1" et "IN2"

Lorsque la comparaison est vraie, la sortie de la boîte de comparaison est VRAIE.

Boite de COMPARAISON



Type de relation	La comparaison est vraie si ...
=	IN1 est égal à IN2
<>	IN1 est différent de IN2
>=	IN1 est supérieur ou égal à IN2
<=	IN1 est inférieur ou égal à IN2
>	IN1 est supérieur à IN2
<	IN1 est inférieur à IN2

4. Le GRAFCET :

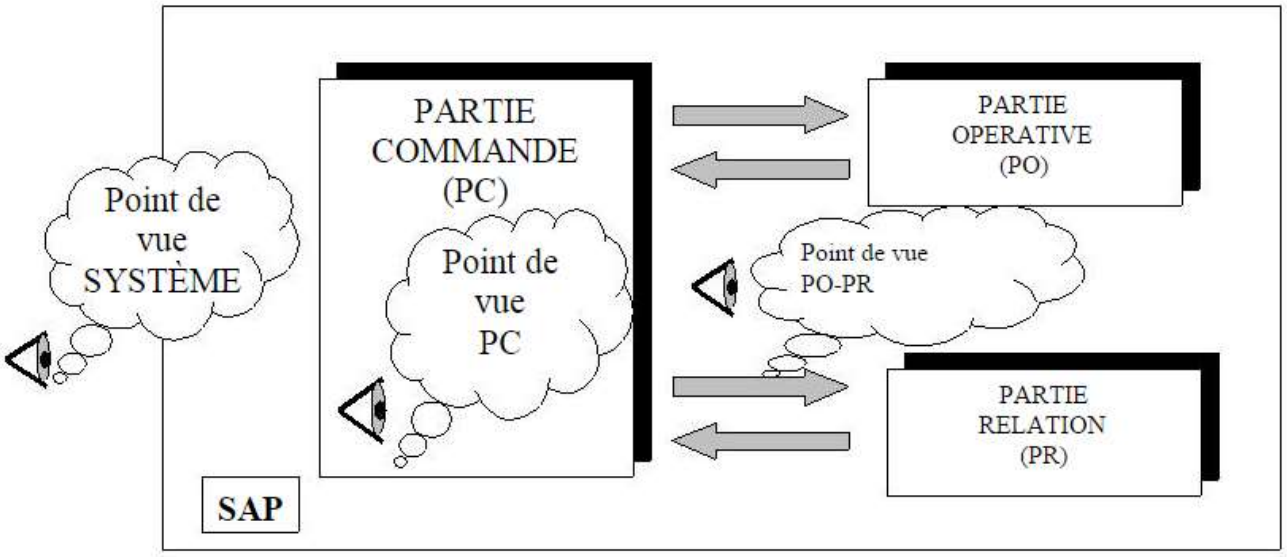
4.1 Définition du GRAFCET :

L'AFCET (Association Française pour la Cybernétique Economique et Technique) et l' ADEPA (Agence nationale pour le DEveloppement de la Production Automatisée) ont mis au point et développé une représentation graphique qui traduit, sans ambiguïté, l'é volution du cycle d'un automatisme séquentiel.

Ce diagramme fonctionnel: le GRAFCET (Graphe Fonctionnel de Commande, Etapes Transitions) permet de décrire les comportements attendus de l'automatisme en imposa nt une démarche rigoureuse, évitant ainsi les incohérences dans le fonctionnement.

4.2. Le point de vue :

Il caractérise la situation de l'observateur décrivant le système automatisé :



Point de vue système (procédé et processus) (Niveau 1)

C'est la description faite par un observateur se situant d'un point de vue externe au SAP.
Le point de vue système décrit le comportement du système vis à vis du produit.il
permet le dialogue entre le client et le concepteur pour la spécification du système automatisé
La notation retenue est **littérale**.

Point de vue partie opérative

Décrit les actions produites par les actionneurs à partir des informations acquises par les capteurs.
Il permet le dialogue entre le concepteur de la partie opérative et le concepteur de la partie com mande.

Point de vue partie commande (Niveau 2)

Cette description prend en compte les choix technologiques de l'ensemble des échanges entre PO et PC.
C'est le point de vue du réalisateur de la Partie Commande
La notation retenue à ce niveau est la notation **symbolique** utilisant les repères du dossier techni que.

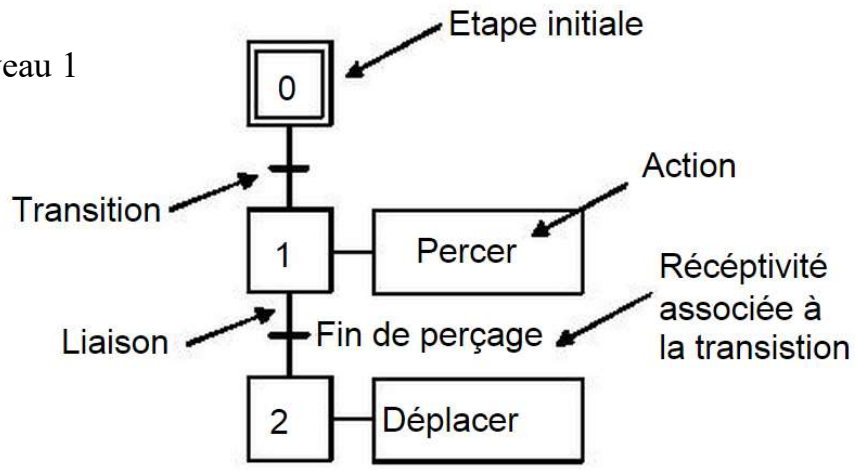
4.3. Constitution

Il est constitué d'un ensemble :

- d'éléments graphiques de base comprenant : les étapes, les transitions, les liaisons orienté es.
- d'une interprétation traduisant le comportement de la partie commande vis-à-vis de ses entrées et de ses sorties, et caractérisée par les réceptivités associées aux transiti ons et les actions associées aux étapes.

- de 5 règles d'évolution définissant formellement le comportement dynamique de la partie commande.

Grafcet niveau 1



- **Etape initiale :** représente une étape qui est active au début du fonctionnement.Elle se différencie de l'étape en doublant les côtés du carré.
- **Transition** : la transition est représentée par un trait horizontal

- **Réceptivité** : les conditions de réceptivité sont inscrites à droite de la transition

- **Etape** : chaque étape est représentée par un carré repéré numériquement

- **Action**(s) : elles sont décrites littéralement ou symboliquement à l'intérieur d'un ou plusieurs rectangles reliés par un trait à la partie droite de l'étape.
- **Liaisons orientées** : indique le sens du parcours.

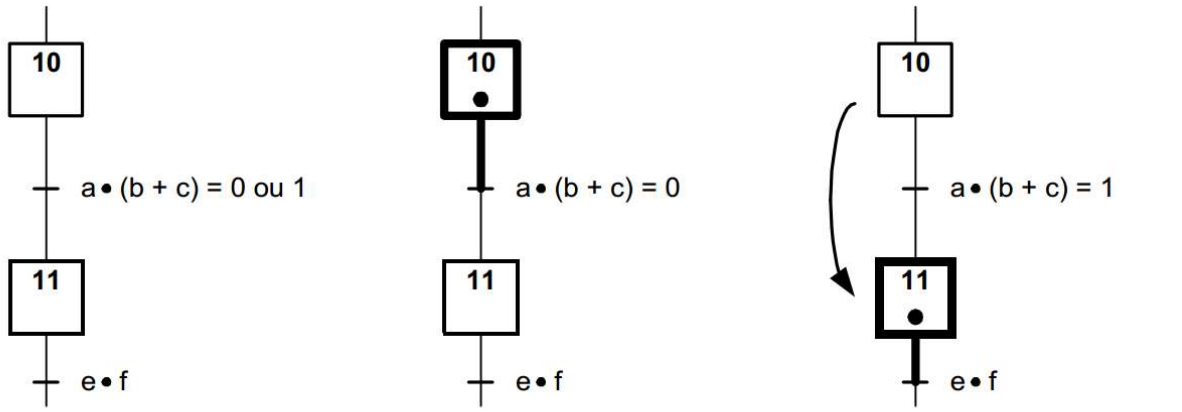
4.4 Règles d'évolution du Grafcet

Règle 1 : L'initialisation précise les étapes actives au début du fonctionnement. Elles sont activées inconditionnellement et repérées sur le GRAFCET en doublant les côtés des symboles correspondants.

Règle 2 : Une transition est soit validée soit non validée. Elle est validée lorsque toutes les étapes immédiatement précédentes sont activées. Elle ne peut être franchie que:
- lorsqu'elle est validée
- et que la réceptivité associée à la transition est vraie.

La transition est alors obligatoirement franchie.

illustration :



Transition non validée
La transistion 10-11 est non validée, l'étape 10 étant inactive

Transition validée
La transistion 10-11 est validée, l'étape 10 étant active, mais ne peut être franchie car la réceptivité a • (b + c) = 0

Transition franchie
La transistion 10-11 est franchie car la réceptivité a • (b + c) = 1
L'étape 11 est active

Règle 3 : Le franchissement d'une transition entraîne l'activation de toutes les étapes immédiatement suivantes et la désactivation de toutes les étapes immédiatement précédentes.

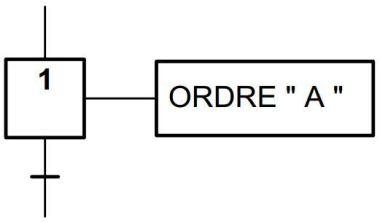
Règle 4 : Plusieurs transitions simultanément franchissables sont simultanément franchies.

Règle 5 : Si au cours du fonctionnement, une même étape doit être désactivée et activée simultanément, elle reste activée. L'activation doit être prioritaire sur la désactivation au niveau d'une même étape.

4.5 Actions associées aux étapes.

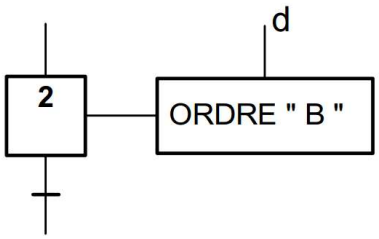
4.5.1 Ordre continu

L'ordre est émis de façon continue tant que l'étape à laquelle il est associé est active



4.5.2 Ordre conditionnel

L'ordre est émis lorsqu'en plus de l'activité de l'étape à laquelle il est associé, une condition logique spécifiée doit être satisfaite.



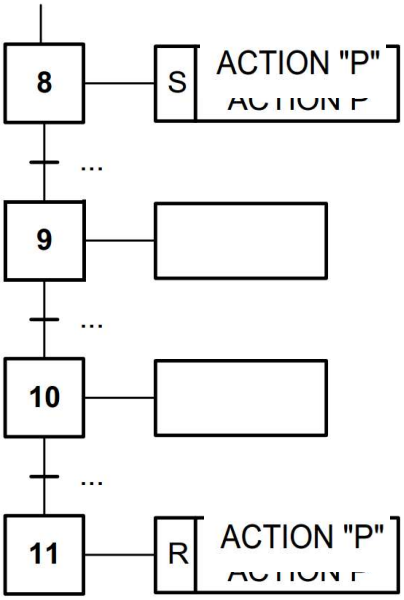
4.5.3 Ordre de mémorisation de l'action

Ces deux ordres, de mémorisation et d'effacement permettent de réduire les répétitions.

- ordre de début d'action , noté : "S" (Set)

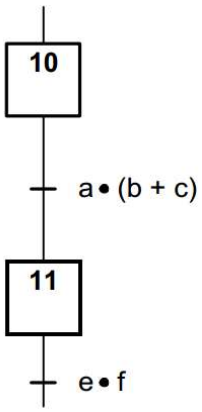
- ordre de fin d'action, noté : "R" (reset)

Ici l'action "P" ...



4.6 Transitions

Les **transitions** indiquent les possibilités d'évolution entre étapes. On associe à chaque transition une réceptivité.



La **réceptivité** est écrite sous forme de proposition logique, c'est une information simple ou une fonction combinatoire d'informations extérieures (capteur, compteur, etc …)

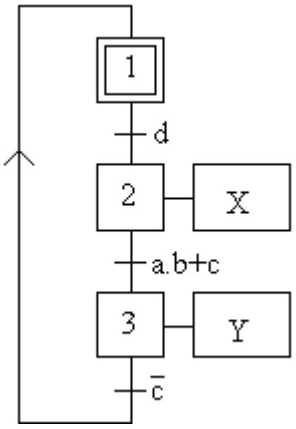
Les **liaisons** indiquent l'évolution de l'état du grafcet.

Les liaisons sont horizontales ou verticales.

4.7 Séquences de base

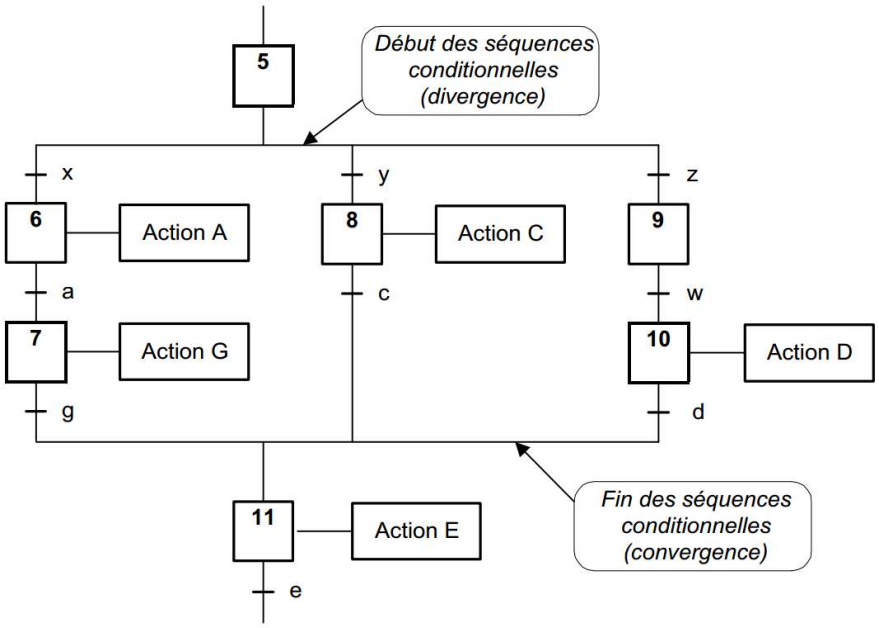
4.7.1: Le Grafcet linéaire

C'est une succession d'étapes et de transitions



4.7.2 Divergence et convergence en "OU".

Choix conditionnel entre plusieurs séquences.



Règles :

- Dans une séparation en OU les transitions se placent **après** la divergence et **avant** la convergence.

- Après une divergence en OU, on trouve une convergence en OU.

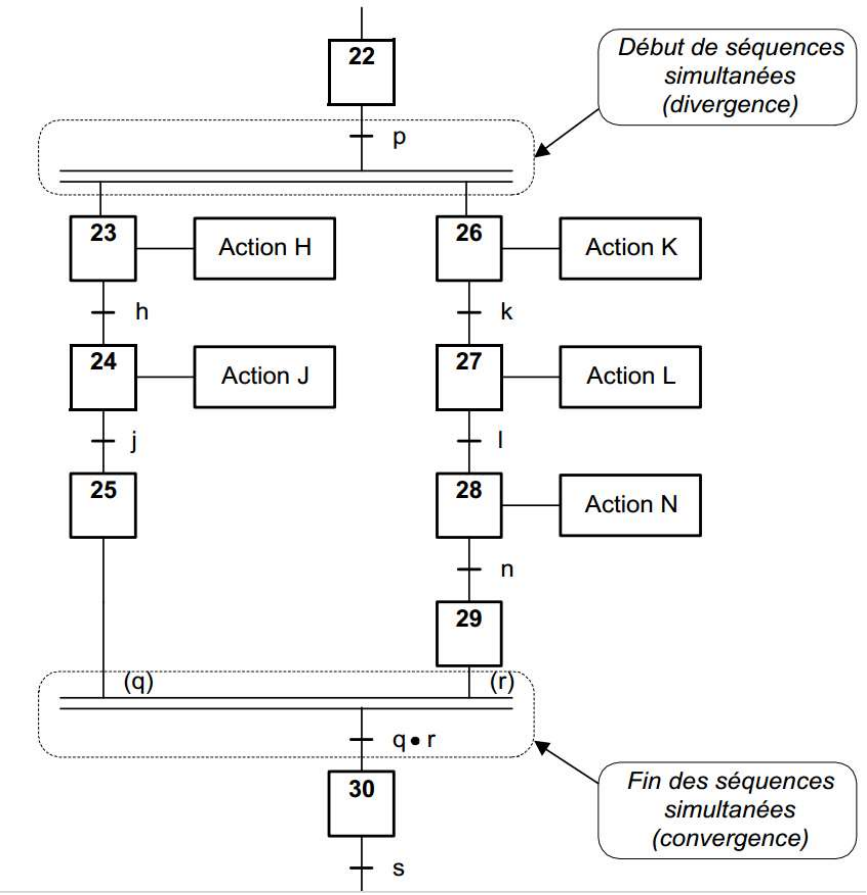
- Le nombre de branches peut-être supérieur ou égal à 2.

- La convergence de toutes les branches ne se fait pas obligatoirement au même endroit.

4.7.3 Divergence et convergence en "ET"

Un GRAFCET peut comporter plusieurs séquences s'exécutant simultanément mais dont les évolutions des étapes actives dans chaque branche restent indépendantes.

Pour représenter ces fonctionnements simultanés, une transition UNIQUE et **deux traits parallèles** indiquent le début et la fin des séquences, c'est-à-dire l'activation simultanée des branches.



Règles :

- Dans une séparation en ET **la** transition se placent **avant** la divergence et **après** la convergence.

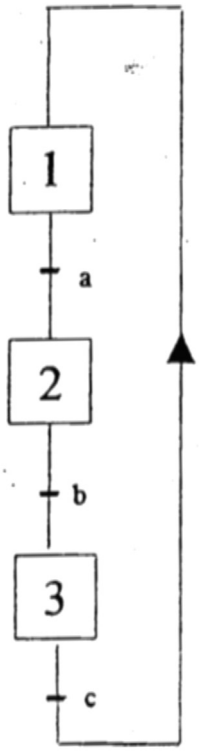
- Après une divergence en ET, on trouve une convergence en ET.

- Le nombre de branches parallèles peut-être supérieur à 2.

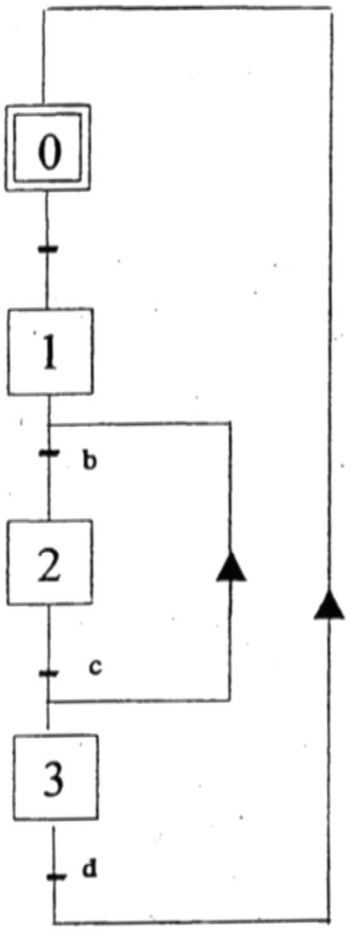
- La réceptivité associée à la convergence peut-être de la forme = 1. Dans ce cas la transition est franchie dès que toutes les étapes finales sont actives .

4.8 Exercices : Indiquer les erreurs réalisées dans les grafcets ci dessous et proposer les corrections à apporter :

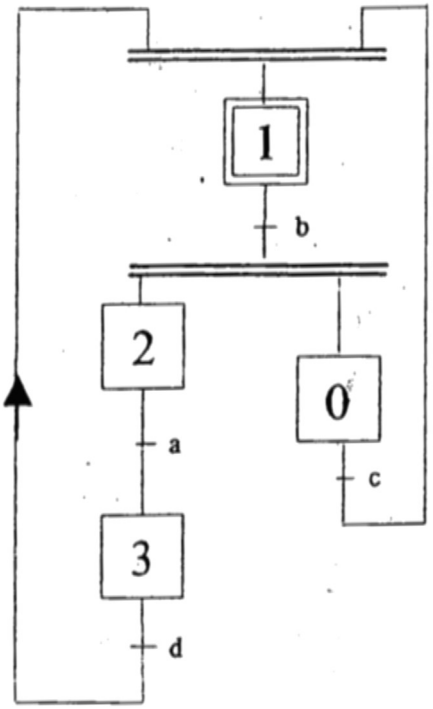
GRAFCET 1



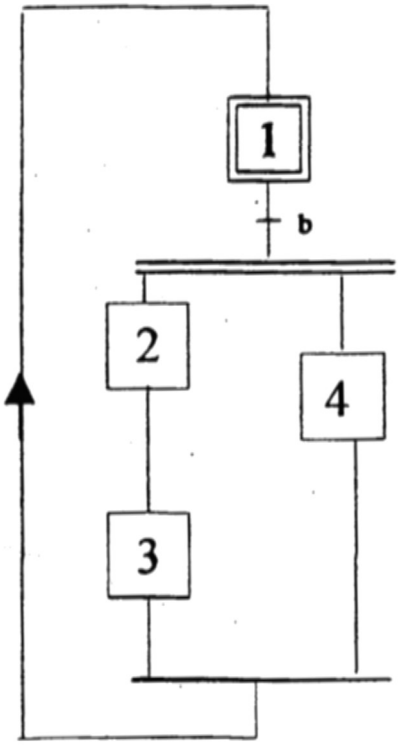
GRAFCET 2



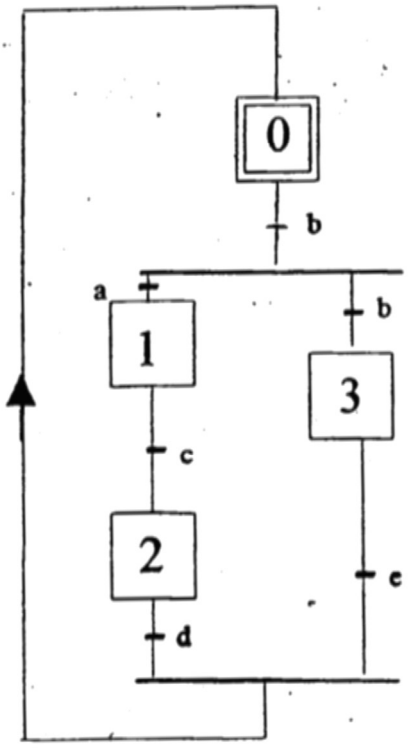
GRAFCET 3



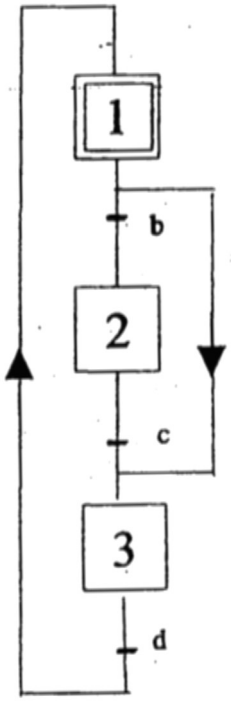
GRAFCET 4



GRAFCET5



GRAFCET6



5. Le GRAFCET évolué :

5.1 : Les temporisations :

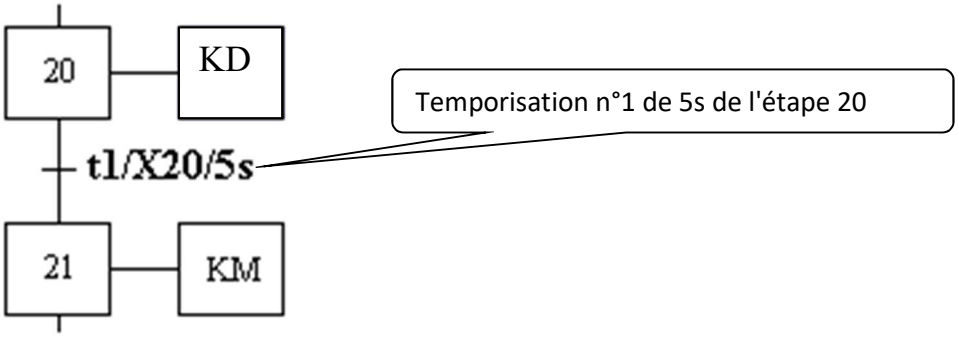
Une temporisation est une réceptivité particulière qui permet d'imposer un temps d'attente avant de franchir la transition et quitter l'étape à laquelle elle est associée (par exemple faire fonctionner un actionneur durant 5s)

Syntaxe d'utilisation :

Choix d'un N° de Temporisateur : — %Txx avec 0<= xx <= 9999

Définition de la durée : 5s ou 5000 ms ou 50 (dixième de sec par défaut)

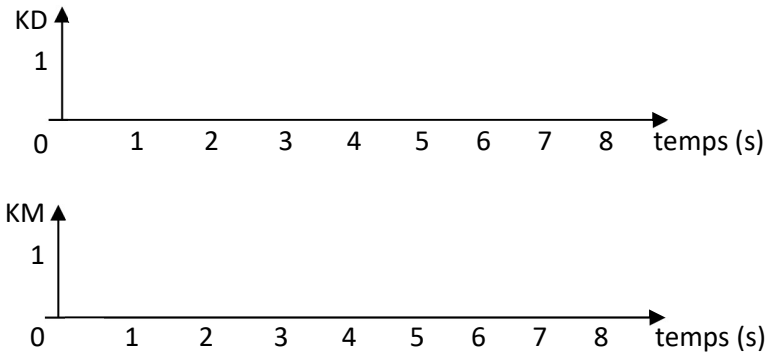
Définition de l'étape concernée : Xnn avec nn = N° de l'étape temporisée



La transition 20 - 21 est franchie lorsque la temporisation n° 1 (t1), démarrée à l'étape 20 est écoulée, soit au bout de 5s.

"X20" signifie : Etape 20 active

Illustration :



5.2 : Les compteurs :

Un compteur est une **action** particulière qui permet de comptabiliser le nombre de fois qu'un évènement s'est produit (par ex le nombre de passage par une étape spécifique).

Syntaxe d'utilisation :

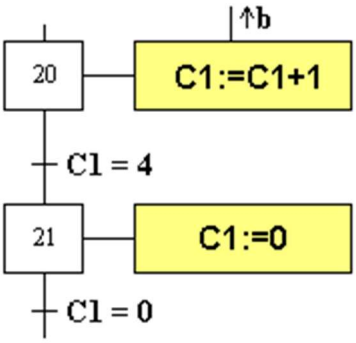
Choix d'un N° de compteur : %Cxx avec 0<= xx <= 9999

Initialisation à zéro d'un compteur : Cxx := 0

Incrémentation d'un compteur : Cxx := Cxx + 1

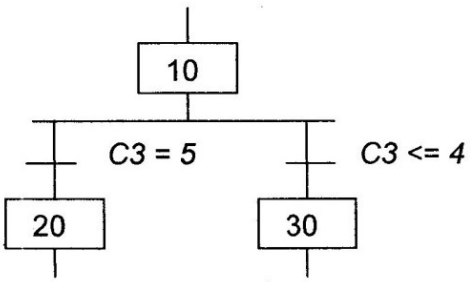
Décrémentation d'un compteur : Cxx := Cxx - 1

Illustration :



La transition 20 - 21 est franchie lorsque le contenu du compteur C1 est égal à 4.
Le compteur est incrémenté sur front montant du signal b.
Il est mis à zéro à l'étape 21.

Les compteurs peuvent être utilisés comme valeur de test en entrée ou sortie des divergences:



Dans l'exemple ci contre :

- La branche de l'étape 30 sera parcourue 5 fois (de %C3 = 0 à % C3 = 4)

- La branche de l'étape 20 sera parcourue après la sixième décision (%C3 = 5)

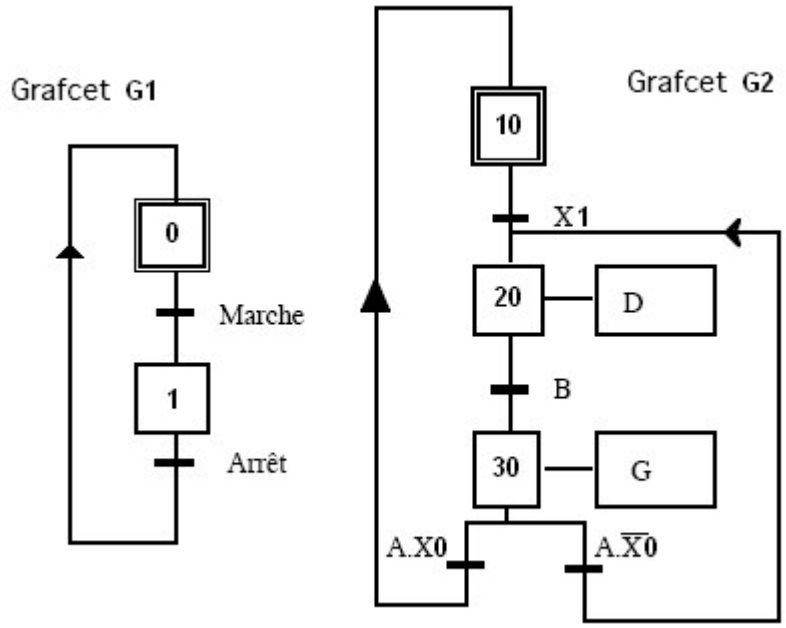
5.3 : Grafcet partitionnés :

Pour simplifier l'écriture, la lisibilité et la maintenance d'un grafcet volumineux, il est judicieux de partager un grand Grafcet en plusieurs petits grafcets indépendants ayant chacun une sous tâche spécifique à exécuter. La synchronisation entre Grafcets est assurée par des réceptivités particulières liées à l'état d'une étape spécifique (bit d'étape Xnn)

Syntaxe d'utilisation :

Définition de l'étape concernée : Xnn avec nn = N° de l'étape concernée

Illustration :



Analyse :

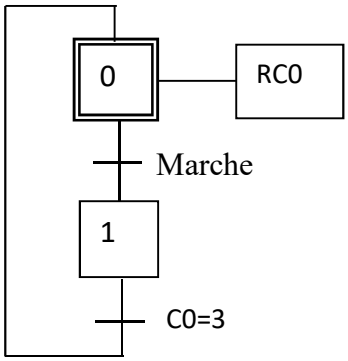
Le grafcet G2 ne démarre que si X1est Vrai, c'est à dire que l'étape 1 du Grafcet G1 est active, c'est à dire après une action sur Marche.

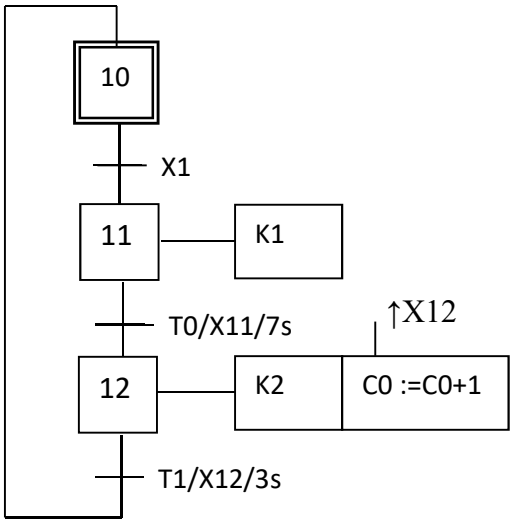
Les action D et G associées aux étapes 20 et 30 seront répétées (liaison orientée de droite) tant que le Grafcet G1 ne sera pas revenu en étape 0 (lors d'une action sur Arrêt).

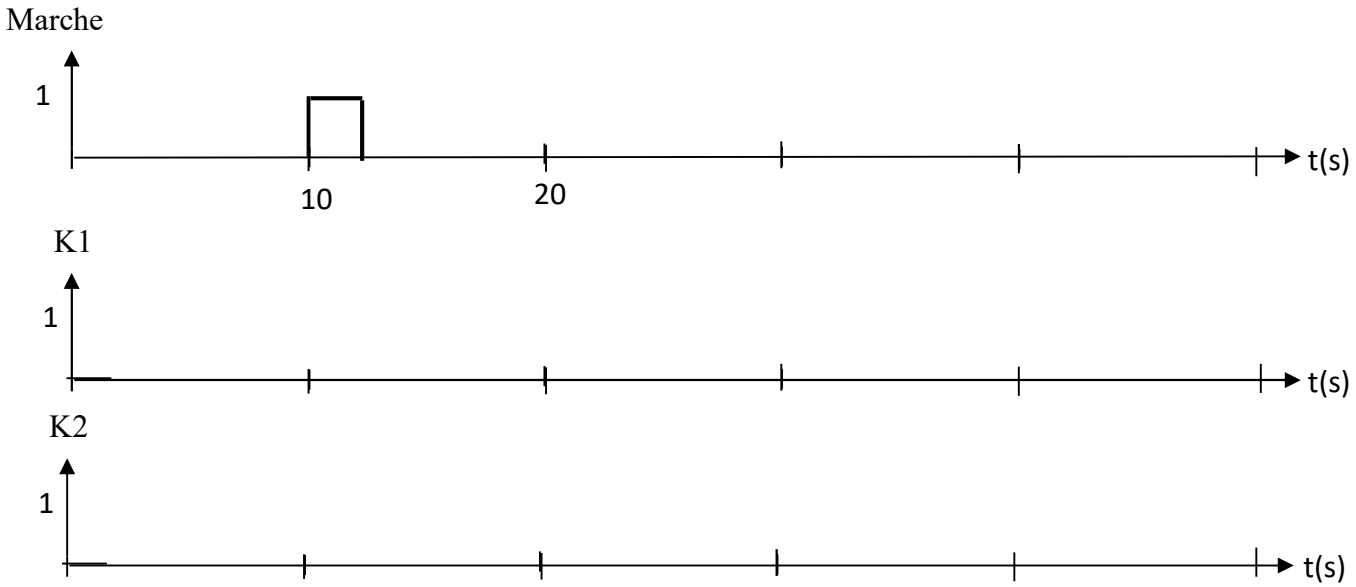
Le grafcet G2 reviendra à l'étape intitiale après que le grafcet G1 soit revenu lui aussi en étape initiale (X0) et que la réceptivité "A" sera Vraie.

5.4 Exercices : Représentation temporelle d'un GRAFCET:

On considère le grafcet suivant : Compléter les chronogrammes de K1 et K2





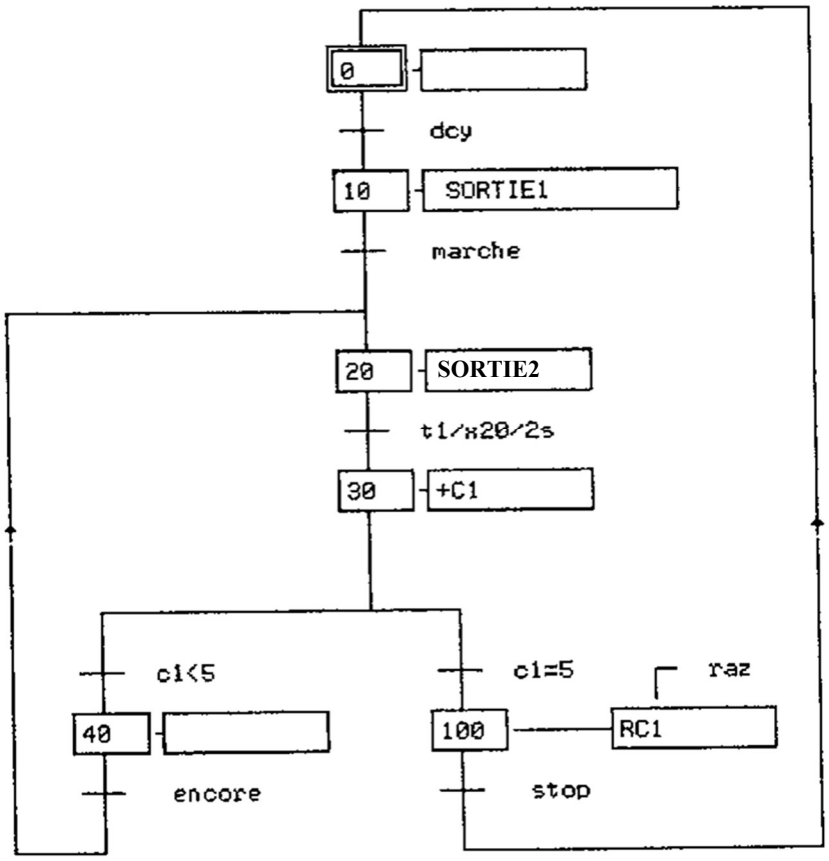


Fonction réalisée ?

Caractéristiques du signal K1 ?

Caractéristiques du signal K2 ?

EXERCICE 1 : Soit le grafcet ci-contre :



1.1) Quelle est le type de la réceptivité associée à la transition entre les étapes 20 et 30 ?

1.2) Quelle est l'action réalisée à l'étape 30 ?

1.3) Quel est le type de …vergence entre les étapes 30, 40 et 100 ?

1.4) Quel est le type de …vergence entre les étapes 40, 10 et 20 ?

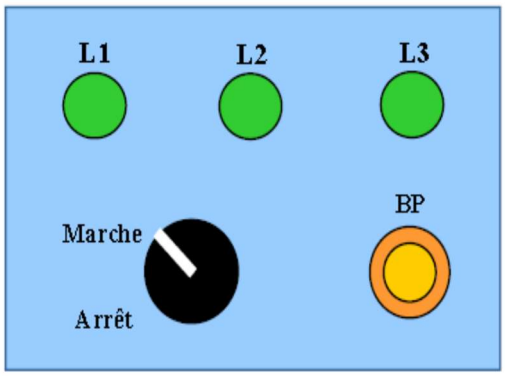
1.5) Quelle est l'action réalisée à l'étape 100 ? A quelles conditions sera t'elle réalisée ?

1.6) On appui sur dcy : quelle sortie est active ?

1.7) On appui sur marche : quelle sortie est active ? Combien de temps ?

1.8) On laisse « encore » activé : quelle sortie est active ? Combien de temps ?

EXERCICE 2 : Pupitre de commande



Lorsque le système est en marche:

• 1er appui sur BP →L1 est allumée
• 2ème appui sur BP → L1 et L2 sont allumées
• 3ème appui sur BP → L1, L2 et L3 sont allumées
• 4ème appui sur BP → L1, L2 et L3 sont éteintes

A tout moment, l'utilisateur peut demander l'arrêt du système (L1, L2 et L3 sont éteintes).

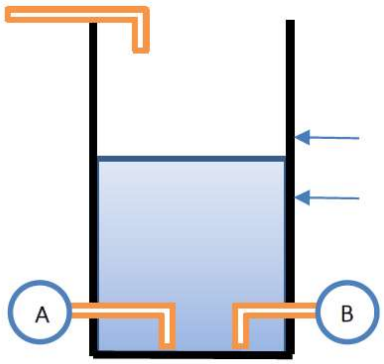
- Le bouton BP utilisé est un bouton poussoir. Il faudra détecter le passage de l'état inactif (niveau logique 0 à l'état actif (niveau logique 1).
- Le commutateur Marche / Arrêt est à 2 positions stables.

2-1) Dessiner le grafcet partie commande.

2.2) Proposer une solution avec un grafcet partie commande, partitionné, gérant chaque lampe.

EXERCICE 3 : Compléter un grafcet

La surveillance automatique du vidage, par deux pompes A et B, d'une cuve est réalisée par un automate. Deux capteurs délivrent une information avec un cycle d'hystérésis :



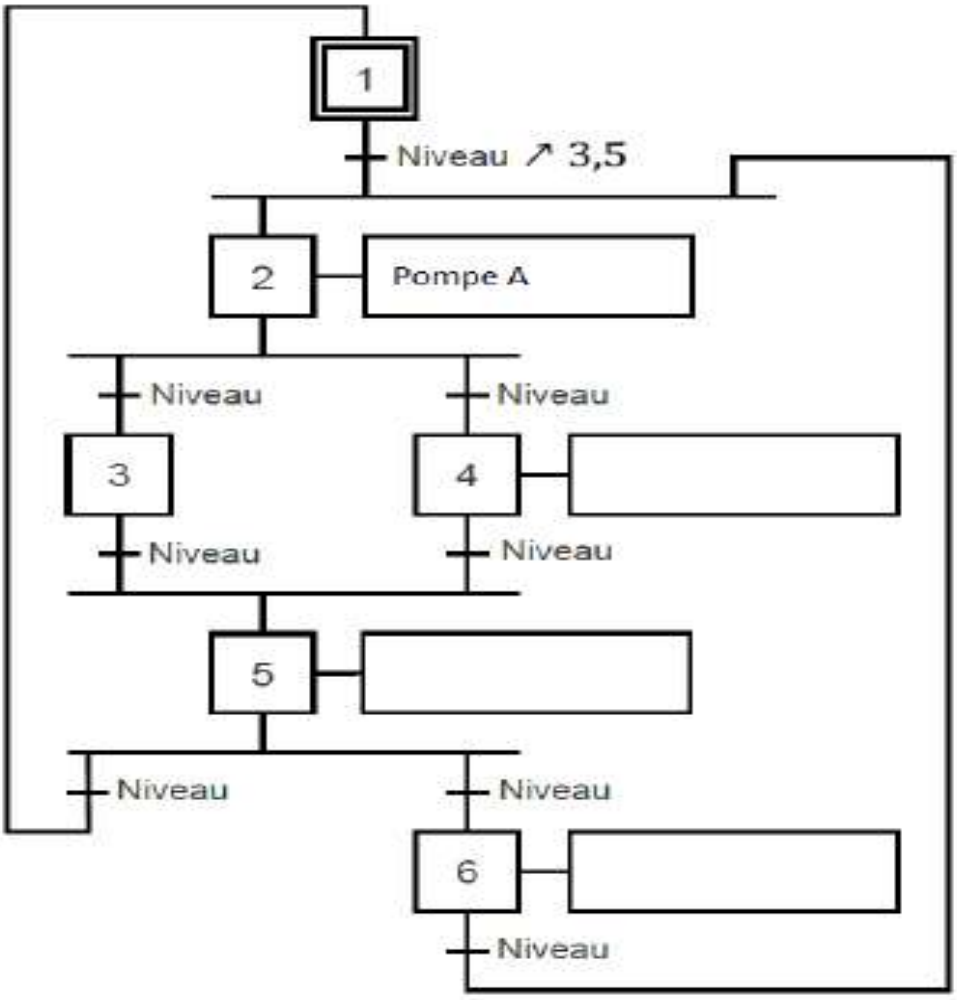
► Capteur de niveau haut
- Activé lorsque le niveau passe au-dessus 5,5m (↗5,5)
- Désactivé lorsque le niveau passe en dessous de 5m (↘5)

► Capteur de niveau moyen
- Activé pour Niveau passe au-dessus 3,5m(↗3,5)
- Désactivé lorsque le niveau passe en dessous de 3m(↘3)

- Le fonctionnement des pompes A ou B est alterné.
- Une seule pompe se met en route au-dessus du niveau moyen.
- Les deux sont actionnées au-dessus du niveau haut.
- Aucune pompe n'est activée en dessous du niveau moyen.

La gestion du démarrage ou arrêt de l'installation ou des alarmes ne sont pas abordées.

3) Compléter le grafcet suivant avec les réceptivités venant des capteurs de niveau et les actions sur les pompes A et B.


1
Niveau ↗ 3,5
2
Pompe A
Niveau
Niveau
3
4
Niveau
Niveau
5
Niveau
Niveau
6
Niveau

L1 **Synthèse d'AUTOMATIQUE** **2016-2017**
Semestre S1 **(Durée : 1 heure 00 min)**

1ère QUESTION (valeur = 5)

Numération :

1) Quel est le complément à 2 en binaire de la valeur 0b 1100 0110 ;
2) Convertir en décimal la valeur 0x 15CE ;
3) Effectuer la soustraction binaire en posant le calcul 0b 1011 1100 – 0011 0101 ;
4) Quel est le résultat de la multiplication par le base de 0b 0100 0101 ;
5) Quelle est la plus grande valeur possible en BCD.

2ème QUESTION (valeur = 8)

Réalisation d'un automate :

L'étude porte sur la réalisation de la commande d'un monte charge. Ce moteur est commandé par trois boutons poussoirs (monostable) :

M pour la monter la charge
D pour descendre la charge
ST pour l'arrêt du monte charge

Lorsque le monte charge est en montée ou en descente, il faut nécessairement passer en ST pour pouvoir inverser la course. En d'autres termes, l'action du bouton M lorsque le monte charge est en D n'a aucune influence et réciproquement, D n'a pas d'influence lorsque le moteur est en M.

Le ST est prioritaire sur toutes les autres actions.

Le bouton ST est prioritaire sur M ou D en cas d'appui simultané.

1) Le problème est-il séquentiel ou combinatoire, justifier.
2) Établir la table de vérité pour la montée et pour la descente de la charge.
3) Dresser le tableau de Karnaugh pour la montée et pour la descente et en déduire les équations logiques des sorties.
4) Établir le logigramme de la commande de montée et de descente du monte charge.

3eme QUESTION (valeur = 7)

Compression des images, étude d'un sondeur de plaisance ;

Une image numérique est une image constituée d'un nombre fini de points codant l'information lumineuse dans une gamme finie de longueur d'onde.

- *Point* = pixel (picture element)
- *Définition :* dimension de l'image en pixel
- *Résolution :* nombre de pixel sur une unité de surface [ppi, ppp]
- Profondeur : nombre de valeurs possibles de la gamme

première partie : taille d'image brute

Illustration 1: Image sondeur 256x256 n&b

1) Calculer le nombre de bits nécessaires pour coder une ligne de cette image ? en déduire la taille d'une ligne en octets lorsque les pixels sont organisés par groupe de 8 bits.

2) En déduire la taille de l'image complète en octets et koctets.

deuxième partie : compression de l'image

Pour réduire l'espace nécessaire au stockage de l'image on réalise la compression de données.
La compression peut être destructive, il y a alors perte d'information, maîtrisée , ou non destructive, pas de perte d'information. On parle alors de compactage. Lors de l'étude sera employée une compression non destructive dite RLE (Run-Length Encoding)
Principe : Si une donnée d apparaît n fois consécutivement dans le flux d'entrée, remplacer les n occurrences par la paire "n d".
Pour une image en noir et blanc : deux couleurs possibles : N (0) & B (1)

- ex: 000001111011000000001111 ...

On suppose que la couleur de départ est le noir. On compresse alors uniquement en indiquant le nombre de pixels Noirs ou Blancs successifs
ex : 5 pixels noirs (PN) « 0 » ; 4 pixels blancs (PB) « 1 »; 1 PN « 0 » ; 2 PB « 1 » ; 7 PN « 0 » ; 4 PB « 1 » ;
donne : 0x05 0x04 0x01 0x02 0x07 0x04

L'analyse de la ligne 30 de l'image du sondeur donne les valeurs suivantes :

Valeur du bit	0	1	0	1	0	1	0	1	0	1	0	1
Nombre d'occurrences successives	7	12	8	13	30	8	5	26	4	6	11	126

3) Donner la suite hexadécimale de la ligne compressée
4) Donner le taux de compression Tc
avec Tc = Nombre d'octets de la ligne compressée / Nombre d'octets de la ligne non compressée
5) En considérant que Tc est le même sur toutes les lignes, donner la taille en Koctets de l'image complète compressée.

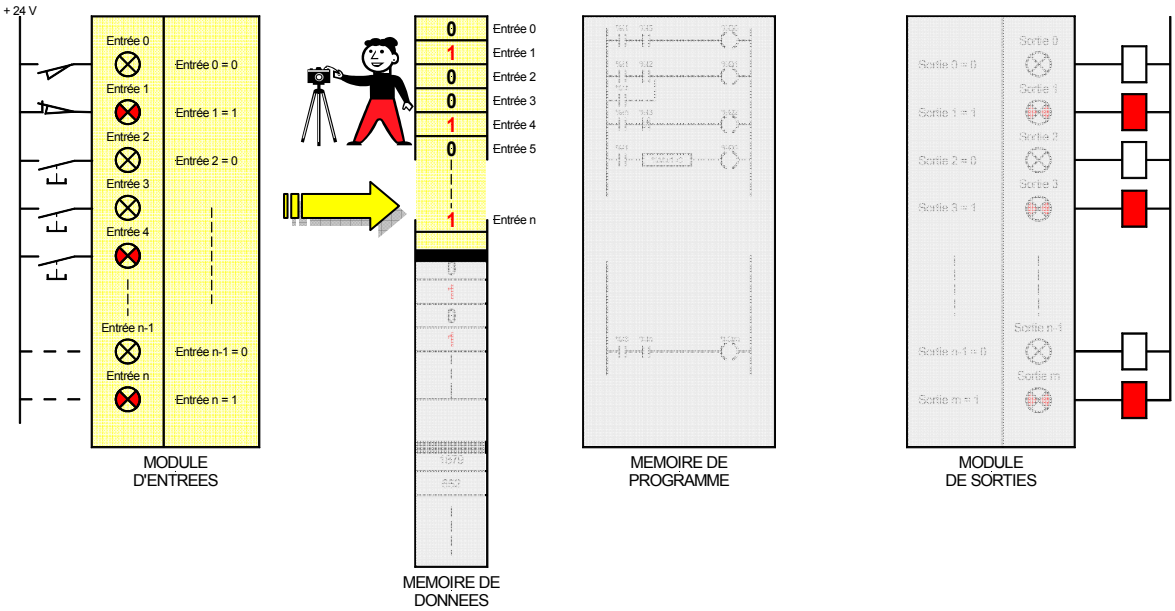
Annexe
Fonctionnement d'un A.P.I.

Lorsque l'API est en fonctionnement, trois phases se succèdent :

PHASE 1	PHOTOGRAPHIE DES ENTREES

Durant cette phase qui dure quelques micro-secondes :

- les entrées sont « photographiées » et leurs états logiques sont stockés dans une zone spécifique de la mémoire de donnée.
- Le programme n'est pas scruté.
- Les sorties ne sont pas mises à jour.

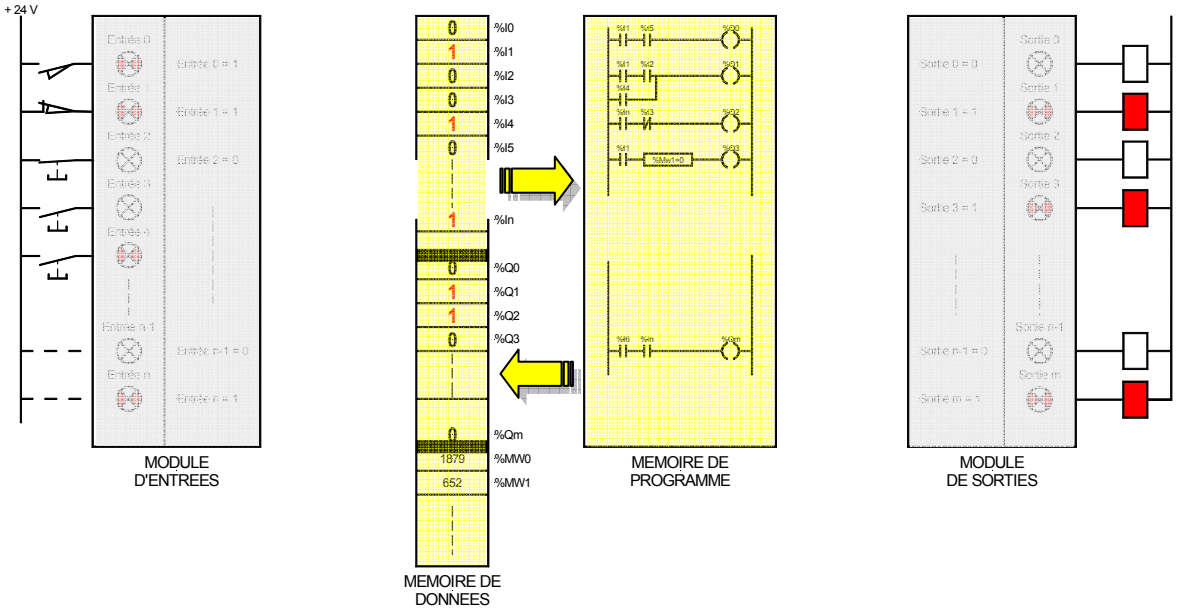


PHASE 2 EXECUTION DU PROGRAMME

Durant cette phase qui dure quelques milli-secondes :

- Les instructions de programme sont exécutées une à une. Si l'état d'une entrée doit être lu par le programme, c'est la valeur stockée dans la mémoire de données qui est utilisée.
- Le programme détermine l'état des sorties et stocke ces valeurs dans une zone de la mémoire de données réservée aux sorties.
- Les entrées ne sont pas scrutées.
- Les sorties ne sont pas mises à jour.

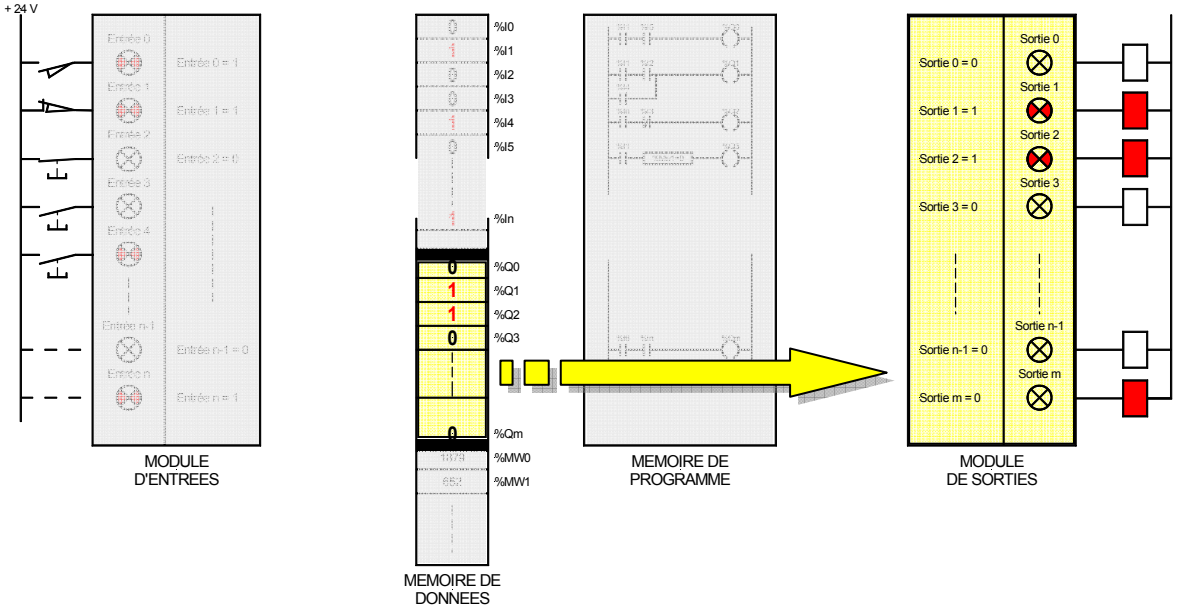
Notez que pendant cette phase, seules la mémoire de données et la mémoire programme sont mises à contribution. Si une entrée change d'état sur le module d'entrées, l'API ne « voit » pas ce changement.

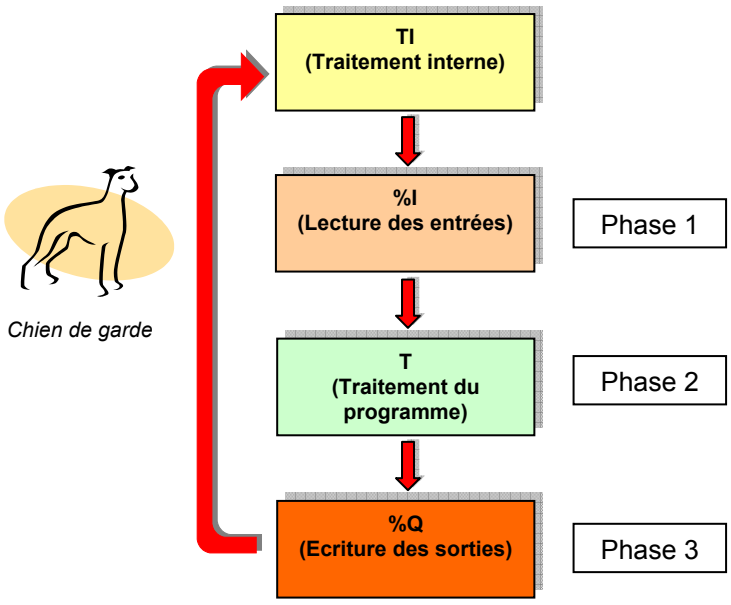


PHASE 3 MISE À JOUR DES SORTIES

Durant cette phase qui dure quelques micro-secondes :

- Les états des sorties mémorisés précédemment dans la mémoire de données sont reportés sur le module de sorties.
- Les entrées ne sont pas scrutées.
- Le programme n'est pas exécuté.





L'enchaînement des trois phases se répète sans cesse de façon cyclique lorsque l'API est en fonctionnement :

1. Lecture des entrées (%I)
2. Traitement du programme (T)
3. Ecriture des sorties (%Q)

Avant chaque cycle l'API effectue des traitements internes afin de vérifier ses circuits et les sollicitations extérieures.

Le temps de cycle de l'ordre de quelques milli-secondes est surveillé par un circuit électronique appelé « Chien de garde ».

Si pour une raison quelconque le temps de cycle mesuré par le chien de garde est supérieur au temps de cycle maxi configuré, l'API signale le défaut arrête le traitement.

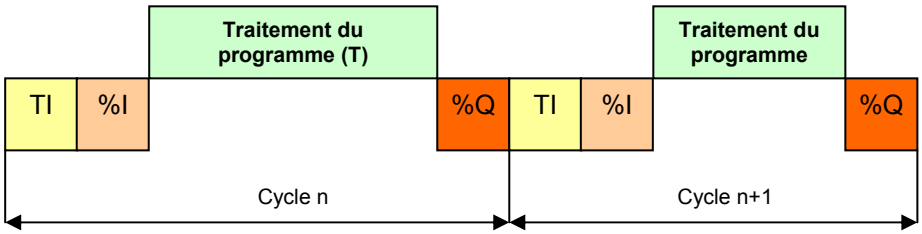
Structure logicielle MONOTACHE

Le programme d'une application monotâche est associé à une seule tâche utilisateur la tâche maître MAST.

L'exécution de la tâche maître peut être :

- cyclique
- ou périodique

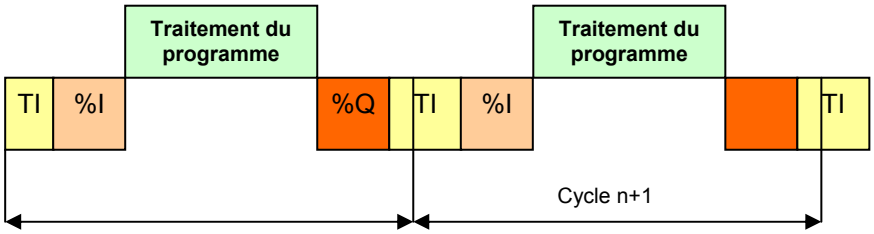
1 Exécution CYCLIQUE



Repère	Phase	Description
TI	Traitement Interne	Le système réalise implicitement la surveillance de l'automate (gestion des bits et mots système, mise à jour des valeurs courantes de l'horodateur, mise à jour des voyants d'état, détection des passages RUN/STOP, ...) et le traitement des requêtes en provenance du terminal (modifications et animation).
%I	Acquisition des Entrées	Ecriture en mémoire de l'état des informations présentes sur les entrées des modules TOR et métier associées à la tâche.
T	Traitement du programme	Exécution du programme application, écrit par l'utilisateur.
%Q	Mise à jour des Sorties	Ecriture des bits ou des mots de sorties associés aux modules TOR et métier associés à la tâche selon l'état défini par le programme application.

La durée d'un cycle ne doit pas dépasser le temps réglé pour le *chien de garde*.

2 Exécution PERIODIQUE

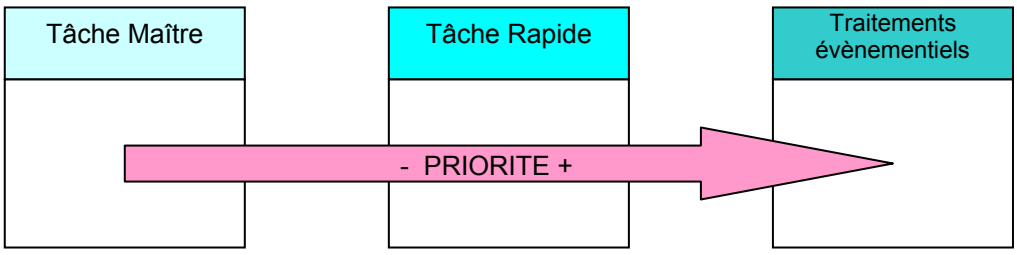


Dans ce mode de fonctionnement, l'acquisition des entrées, le traitement du programme application et la mise à jour des sorties s'effectuent de façon périodique selon un temps défini en configuration (de 1 à 255 ms).
En début de cycle automate, un temporisateur dont la valeur courante est initialisée à la période définie en configuration, commence à décompter.
Le cycle automate doit se terminer avant l'expiration de ce temporisateur qui relance un nouveau cycle.

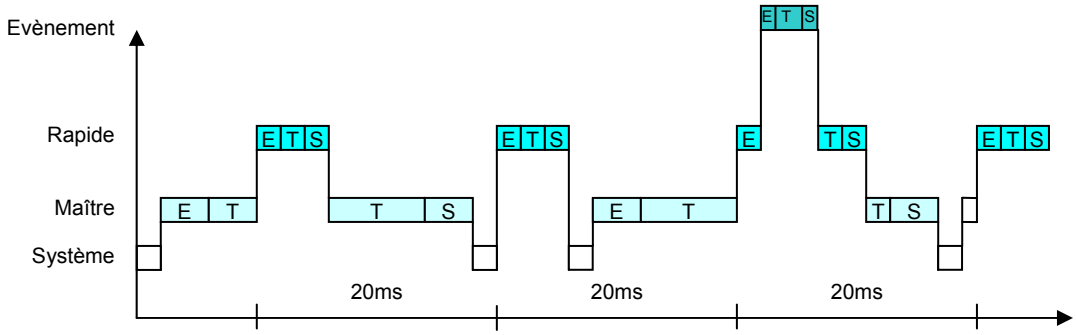
Le processeur effectue dans l'ordre le traitement interne, l'acquisition des entrées, le traitement du programme application et la mise à jour des sorties.

- Si la période n'est pas encore terminée, le processeur complète son cycle de fonctionnement jusqu'à la fin de la période par du traitement interne.
- Si le temps de fonctionnement devient supérieur à celui affecté à la période, l'automate signale un débordement de période par la mise à l'état 1 du bit système %S19 de la tâche, le traitement se poursuit et est exécuté dans sa totalité (il ne doit pas dépasser néanmoins le temps limite du chien de garde). Le cycle suivant est enchaîné après l'écriture implicite des sorties du cycle en cours.

2 Structure logicielle MULTI-TACHES



La tâche maître est par défaut active.
La tâche rapide est par défaut active si elle est programmée.
Le traitement événementiel est activé lors d'apparition de l'événement qui lui a été associé.



Légende **:**
E : acquisition des entrées
T : traitement du programme
S : mise à jour des sorties