

Sommaire

Introduction	3
Principales structures employées pour former un réseau	3
Les données	4
Le codage binaire.....	4
Numération binaire et décimal	5
A.N1 : conversion binaire en décimal	5
A.N2 : Convertissez le nombre 11111100 ₍₂₎ en base 10.....	5
L'addition en base 2	7
Conversion : base 10 en base 2	7
AN3 : Conversion de 163 ₍₁₀₎ en base 2 :	7
AN4 : Conversion de 128 ₍₁₀₎ en base 2 :	7
Qu'est ce qu'un Bit ?	8
Application : compteur électronique à circuits intégrés	8
Schéma (simplifié) de l'application :	9
état des niveaux logiques lors d'une première impulsion sur S.....	9
Fonctionnement simplifié de l'étage de commutation	9
Le codage hexadécimal : base 16	9
AN5 :	9
AN6 : convertissez le nombre binaire suivant en base 10 et 16.	9
Adressage IP :	10
Réseaux privés :	10
Equation logique :	11
Variable d'entrée et de sortie :	11
Fonctions logiques	11
Représentation des fonctions logiques :	12
Module logique du relais logique « ZELIO »	12
Intérêt des fonctions logiques :	12
Application simple en informatique :	12
Application en logique câblée :	13
Equation logique du fonctionnement de la pompe :	14
Application en logique programmée : relais logique ZELIO de Schneider	14

En langage « contact » :	17
Schéma de câblage du système de pompage:	18
Raccordement des entrées du relais ou de l'automate :	18
Raccordement des sorties du relais ou de l'automate :	18
Application : Complétez l'équipement de pompage	19
Les circuits intégrés logiques :	19
Les fonctions logiques se trouvent sous forme:	20
Logique séquentielle :	20
Schéma développé :	22
Etudions la dernière partie du cahier des charges :	22
Grafset fonctionnel de la station de pompage :	23
Solution programmée :	24
Symbolisation utilisée par le logiciel :	24
Simulation du fonctionnement du grafset:	25
Schéma électrique simplifié de l'application :	27
Quelques précisions sur le programme :	27
Capteur T.O.R, analogique ou numérique	29
Application de la mise en œuvre d'un capteur analogique sur la station de relevage:	29
Repérage du bornier de raccordement du capteur	30
Principe de fonctionnement du capteur	30
Caractéristiques des entrées analogiques du relais logique ZELIO	30
Câblage des entrées analogiques (document constructeur)	31
Schéma de câblage de l'entrée analogique	31
Programme de l'application	31
Configuration des entrées analogiques	32
Traitement de la boucle 0-20mA	33
Conclusion	36
Précision concernant les capteurs numériques (à faire)	36
Nécessité de mettre en œuvre des « bus » de terrain	36
Gestion centralisée	37
Gestion décentralisée	37

Introduction

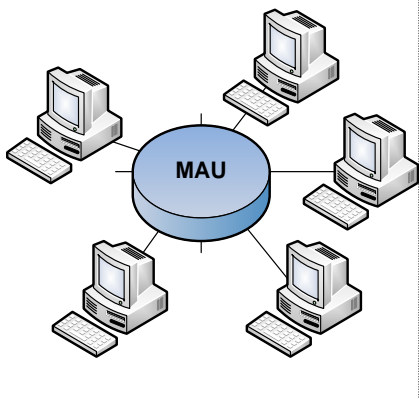
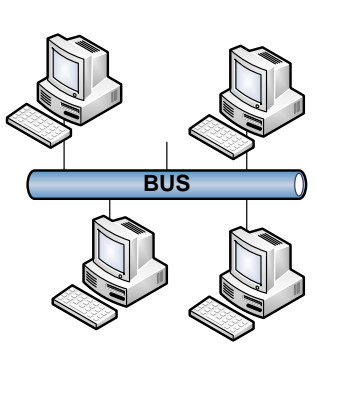
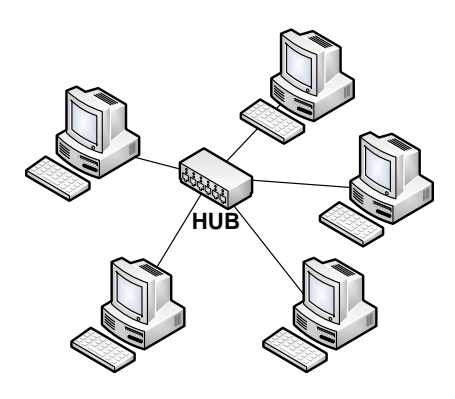
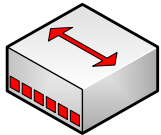
Lorsque nous surfons sur Internet pour chercher, acheter ou communiquer, nous savons que ces actions font intervenir d'autres machines qui sont **connectées** à la notre par l'intermédiaire de conducteurs et d'appareils spécifiques : l'**interconnexion** de l'ensemble des machines constitue un « **réseau** ».

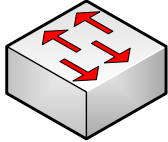
En étendant la notion de **réseau** à d'autres « **appareils** » que des ordinateurs : centrales de mesurages, automates, surveillance de BAES, de SSI, de réseau électrique ou de régime de neutre, on voit de suite l'intérêt que revêtent les réseaux dans la **collecte de données** ou la **commande de systèmes** tertiaires ou industriels. Toutes ces actions **effectuées à distance**, qui facilitent la gestion des systèmes par l'opérateur, demande de la part des techniciens qui les mettent en œuvre, une bonne connaissance de la **topologie** des réseaux :

- Au niveau de leur structure ou topologie physique : matériels entrant dans la composition des réseaux.
- Au niveau de leur structure ou topologie logique : paramétrage, réglage, programmation, adressage, configuration...etc.

Principales structures employées pour former un réseau

L'interconnexion de machines s'effectue suivant les schémas standardisés ci-dessous : (d'autres topologies de réseaux existent) .

Topologie en anneau	Topologie en bus	Topologie en étoile
		
<p>Anneau : Contrairement au schéma général, la boucle n'est pas physique car les ordinateurs sont raccordés à un répartiteur MAU (Multistation Access Unit) qui utilise une topologie logique Token Ring mise au point par IBM. La structure en anneau se justifie par le fait qu'une seule machine du réseau peut communiquer : lors de l'émission de données, la machine garde un jeton logique qu'elle relâche en fin d'émission</p>	<p>Bus : Chaque machine est connectée sur une même ligne (bus) : généralement un câble coaxial. Les dérivations sont réalisées à l'aide de connecteur de type BNC ou des connecteurs vampires (pincement du câble bus). Cette technologie obsolète en informatique car dépendante du bon fonctionnement de toutes les connexions, reste en revanche très répandue en</p>	<p>Etoile : Topologie la plus répandue : Toutes les machines sont connectées sur un concentrateur ou HUB qui répartit le flux d'informations présent sur son port d'entrée vers tous les ports de sortie: 16 ports = 1 entrée +16 machines connectables.</p>  <p>HUB</p> <p>Le HUB peut être remplacé par un commutateur ou SWITCH qui assure</p>



<p>lorsqu'elle récupère les infos qui ont été envoyées à chaque machine et qui lui reviennent, donnant à une autre machine la possibilité de capturer le jeton pour pouvoir à son tour communiquer sur la boucle...etc.</p> <p>Intérêt : évite la collision de données.</p>	<p>industrie : bus ASi, modbus...etc.</p>	<p>aussi le rôle de répartiteur sélectif car mais il aiguille les informations uniquement vers la machine concernée : donc vers 1 seul port.</p> <div style="text-align: center;">  <p>SWITCH</p> </div> <p>L'intervention sur un port de sortie d'un HUB ou d'un SWITCH est sans effet sur les autres ports, mais une panne du HUB affecte tous les ports.</p>
---	---	--

Les données



Problème : La transmission de données s'effectue principalement sous forme **BINAIRE**. Ainsi, le problème consiste à « coder » les informations (voix, données, image) en une succession d'impulsions électriques qui seront ensuite transportées à l'aide de conducteurs spécifiques : paire torsadée ou fibre.

Le codage binaire

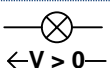
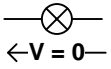
Le codage binaire repose sur le principe que les « systèmes » qui nous entourent ne peuvent prendre que 2 états, ainsi l'état d'un simple contact électrique peut s'exprimer comme ci-dessous :

contact	Etat physique	Etat logique
contact fermé		1
contact ouvert		0

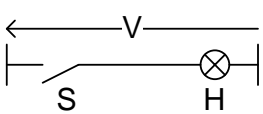
Le même raisonnement peut évidemment s'appliquer à un récepteur :

Lampe	Etat physique	Etat logique
allumée		1
éteinte		0

Les phases de fonctionnement du récepteur correspondent à son état électrique, par conséquent le tableau précédent peut aussi s'écrire :

Lampe	Etat physique	Etat logique
alimentée		1
Non alimentée		0

Ainsi le schéma ci-dessous peut être associé aux états « électriques » et « logiques » suivant :

Schéma	Tableau de vérité	Equation logique						
	<table border="1" style="margin: auto;"> <thead> <tr> <th>S</th> <th>H</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> </tr> </tbody> </table>	S	H	0	0	1	1	<p>Ce tableau peut être résumé par l'équation suivante :</p> <p style="font-weight: bold; font-size: 1.2em;">H = S</p>
S	H							
0	0							
1	1							

Numération binaire et décimal

On utilise naturellement la numération décimale (base 10), sauf lorsque l'on consulte sa montre (numération sexagésimale), en grande partie parce que nous avons 10 doigts. Ainsi, nous comptons en faisant des « paquets » de 10 et en ajoutant 1 à la colonne des dizaines pour « mémoriser » chaque « paquet » constitué.

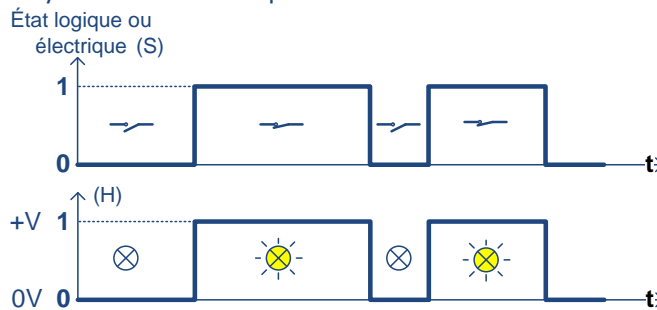
Ainsi, pour exprimer un nombre « X » dans une base de numération « b » constitué de « n chiffre », les « paquets » constitués obéissent à la relation ci-dessous:

$$X_b = a_{n-1} \cdot b^{n-1} + a_{n-2} \cdot b^{n-2} + \dots + a_0 \cdot b^0 + a_1 \cdot b^1 \quad 0 \leq a < b$$

Le nombre **6851** en **base 10** se décompose comme il suit:

$$6851_{(10)} = 6 \times 10^3 + 8 \times 10^2 + 5 \times 10^1 + 1 \times 10^0$$

Un réseau informatique n'a aucun intérêt à communiquer en base 10 alors qu'il n'a à sa disposition que l'énergie électrique pour effectuer des opérations. En revanche, on voit immédiatement ci-dessous, l'intérêt pour les systèmes informatiques et automatisés de communiquer en « binaire ».



La numération en base 2, ne fait appel qu'aux chiffres 0 et 1 : la conversion vers cette base s'effectue comme en décimal suivant la règle vue précédemment.

A.N1 : conversion binaire en décimal

La conversion du nombre **1100011**₍₂₎ en base 10 s'effectue comme ceci :

$$10100011_{(2)} = 1 \times 2^7 + 0 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 163$$

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0									
1	0	1	0	0	0	1	1									
128	+	0	+	32	+	0	+	0	+	0	+	2	+	1	=	163

A.N2 : Convertissez le nombre **11111100**₍₂₎ en base 10

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	
								=

AN3 : extrait épreuve E2 2011

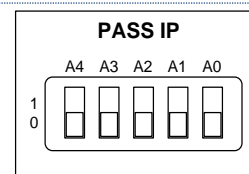
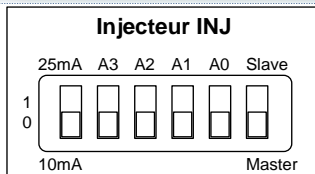
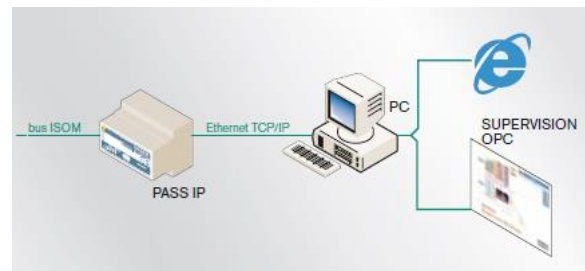
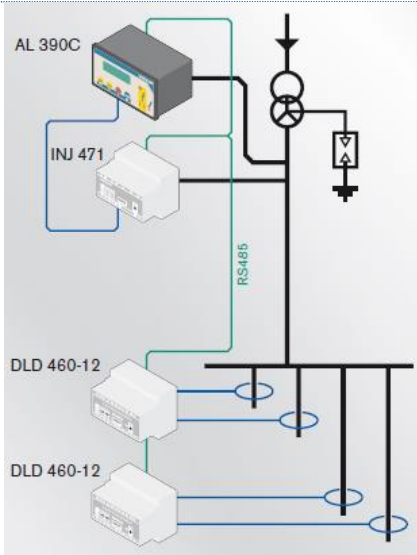
Une question de cette épreuve de communication consistait à régler l'adresse d'un injecteur de courant continu de marque SOCOMEC destiné à la surveillance d'un réseau électrique distribué en IT. Nous n'allons pas entrer dans le détail du raccordement électrique du schéma des liaisons à la terre, mais uniquement répondre à la question qui concerne l'aspect communicant de l'installation

- **Question C1 (sujet métropole 2011)**

A la mise en œuvre le paramétrage de l'interface IP (PASSIP) et de l'injecteur INJ se font par **micro-interrupteurs** situés en face avant. Le PASSIP sera défini en maître à l'adresse de **valeur 1** :

- L'injecteur sera donc esclave (Slave) avec comme adresse la **valeur 115**.
- Les impulsions seront de 10mA.
- Positionner sur les figures ci-dessous la position des micro-interrupteurs en noircissant la position 1 ou 0 de chaque micro-interrupteur de telle manière que les conditions ci-dessus soient respectées.

Ressources :



L'interrupteur de gauche configure le courant à 25mA ou à 10mA.
 L'interrupteur de droite configure le mode maître ou esclave.
 Les quatre interrupteurs du milieu servent à configurer une adresse binaire.
 A0 correspond à 2^0 , A1 à 2^1 , A2 à 2^2 et A3 à 2^3 .
 A cette adresse binaire, le constructeur demande de rajouter 110.
 l'adresse 110 est interdite et l'adresse maxi est 119.

Réponse :

$$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$$

$$0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 128 = 128$$

$$128_{(10)} = 1000000_{(2)}$$

Qu'est ce qu'un Bit ?

L'ensemble des « chiffres binaires » contenus dans le nombre $1000000_{(2)}$ [$128_{(10)}$] sont des « bits » : Le bit constitue la plus petite information **binaires** traitée par un système. En général, on manipule des groupes de bits, par exemple des « **octets** » qui contiennent **8 bits**.

La notion de bit conditionne le nombre maximal « d'informations » stockables par un système : ainsi, un octet de 8 bits dont chacun peut prendre la valeur 0 ou 1 permet de prendre 2^8 états : **n bits permettent de coder 2^n états.**

Application : compteur électronique à circuits intégrés



On désire visualiser sur un afficheur 7 segments, les chiffres de 1 à 8 à chaque impulsion sur un bouton poussoir (afficheur sportif par exemple). L'application sera réalisée à l'aide des circuits suivants :

Un compteur binaire	Décodeur	Afficheur 7 segments																																																																																																																																																																																				
Fonctionnement du compteur	Fonctionnement du décodeur																																																																																																																																																																																					
<table border="1"> <thead> <tr> <th></th> <th>Q_D</th> <th>Q_C</th> <th>Q_B</th> <th>Q_A</th> <th>g</th> </tr> </thead> <tbody> <tr><td>0</td><td>L</td><td>L</td><td>L</td><td>L</td><td>1</td></tr> <tr><td>1</td><td>L</td><td>L</td><td>L</td><td>H</td><td>1</td></tr> <tr><td>2</td><td>L</td><td>L</td><td>H</td><td>L</td><td>0</td></tr> <tr><td>3</td><td>L</td><td>L</td><td>H</td><td>H</td><td>0</td></tr> <tr><td>4</td><td>L</td><td>H</td><td>L</td><td>L</td><td>0</td></tr> <tr><td>5</td><td>L</td><td>H</td><td>L</td><td>H</td><td>0</td></tr> <tr><td>6</td><td>L</td><td>H</td><td>H</td><td>L</td><td>0</td></tr> <tr><td>7</td><td>L</td><td>H</td><td>H</td><td>H</td><td>1</td></tr> <tr><td>8</td><td>H</td><td>L</td><td>L</td><td>L</td><td>0</td></tr> </tbody> </table>		Q _D	Q _C	Q _B	Q _A	g	0	L	L	L	L	1	1	L	L	L	H	1	2	L	L	H	L	0	3	L	L	H	H	0	4	L	H	L	L	0	5	L	H	L	H	0	6	L	H	H	L	0	7	L	H	H	H	1	8	H	L	L	L	0	<table border="1"> <thead> <tr> <th></th> <th>D</th> <th>C</th> <th>B</th> <th>A</th> <th>a</th> <th>b</th> <th>c</th> <th>d</th> <th>e</th> <th>f</th> <th>g</th> </tr> </thead> <tbody> <tr><td>0</td><td>L</td><td>L</td><td>L</td><td>L</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr><td>1</td><td>L</td><td>L</td><td>L</td><td>H</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>2</td><td>L</td><td>L</td><td>H</td><td>L</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>3</td><td>L</td><td>L</td><td>H</td><td>H</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr> <tr><td>4</td><td>L</td><td>H</td><td>L</td><td>L</td><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>5</td><td>L</td><td>H</td><td>L</td><td>H</td><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>6</td><td>L</td><td>H</td><td>H</td><td>L</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>7</td><td>L</td><td>H</td><td>H</td><td>H</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>8</td><td>H</td><td>L</td><td>L</td><td>L</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr> </tbody> </table>			D	C	B	A	a	b	c	d	e	f	g	0	L	L	L	L	0	0	0	0	0	0	1	1	L	L	L	H	1	0	0	1	1	1	1	2	L	L	H	L	0	0	1	0	0	1	0	3	L	L	H	H	0	0	0	0	1	1	0	4	L	H	L	L	1	0	0	1	1	0	0	5	L	H	L	H	0	1	0	0	1	0	0	6	L	H	H	L	1	1	0	0	0	0	0	7	L	H	H	H	0	0	0	1	1	1	1	8	H	L	L	L	0	0	0	0	0	0	0
	Q _D	Q _C	Q _B	Q _A	g																																																																																																																																																																																	
0	L	L	L	L	1																																																																																																																																																																																	
1	L	L	L	H	1																																																																																																																																																																																	
2	L	L	H	L	0																																																																																																																																																																																	
3	L	L	H	H	0																																																																																																																																																																																	
4	L	H	L	L	0																																																																																																																																																																																	
5	L	H	L	H	0																																																																																																																																																																																	
6	L	H	H	L	0																																																																																																																																																																																	
7	L	H	H	H	1																																																																																																																																																																																	
8	H	L	L	L	0																																																																																																																																																																																	
	D	C	B	A	a	b	c	d	e	f	g																																																																																																																																																																											
0	L	L	L	L	0	0	0	0	0	0	1																																																																																																																																																																											
1	L	L	L	H	1	0	0	1	1	1	1																																																																																																																																																																											
2	L	L	H	L	0	0	1	0	0	1	0																																																																																																																																																																											
3	L	L	H	H	0	0	0	0	1	1	0																																																																																																																																																																											
4	L	H	L	L	1	0	0	1	1	0	0																																																																																																																																																																											
5	L	H	L	H	0	1	0	0	1	0	0																																																																																																																																																																											
6	L	H	H	L	1	1	0	0	0	0	0																																																																																																																																																																											
7	L	H	H	H	0	0	0	1	1	1	1																																																																																																																																																																											
8	H	L	L	L	0	0	0	0	0	0	0																																																																																																																																																																											
<p>L = low (niveau bas) : 0 (0V) H = high (niveau haut) : 1 (+5V)</p> <p>Chaque impulsion sur le bouton poussoir présente sur l'entrée input incrémente (ajoute 1 dans) le compteur suivant le codage binaire décrit ci-dessus.</p>	<p>L = low (niveau bas) : 0 (0V) H = high (niveau haut) : 1 (+5V)</p> <p>Seules, les 8 premières lignes du tableau de vérité sont représentées, mais les 4 entrées</p>		<p>L'afficheur est à anode commune : toutes les anodes sont reliées et le circuit intégré 7447 effectue l'alimentation de l'afficheur en reliant les Leds qui le composent au 0V.</p>																																																																																																																																																																																			

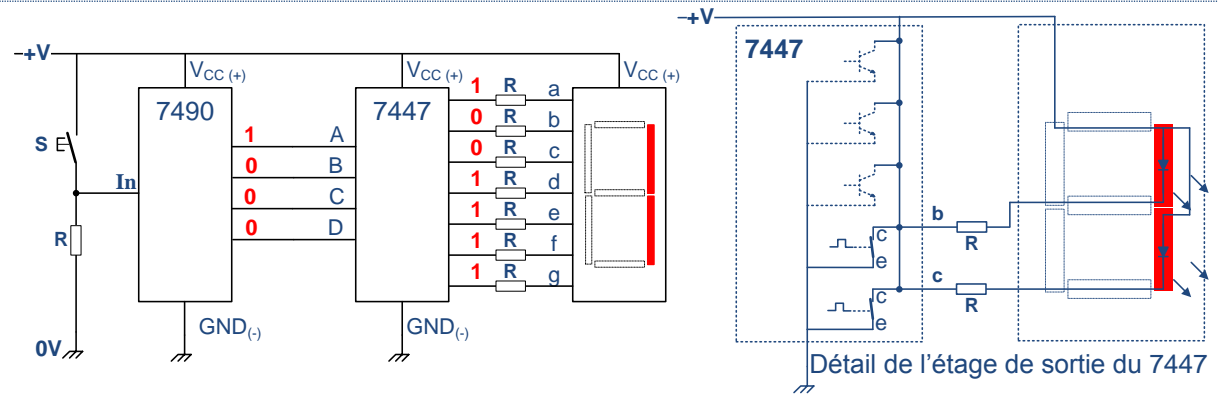
ABCD permettent de compter jusqu'à :

$$2^4 = 16$$

Le circuit relie la sortie à alimenter à la masse : le circuit est à collecteur ouvert.

Schéma (simplifié) de l'application :

état des niveaux logiques lors d'une première impulsion sur S

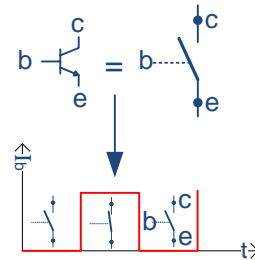


Fonctionnement simplifié de l'étage de commutation

Les sorties du 7447 sont à **collecteur ouvert** :

le niveau logique 0 en sortie du 7447 implique que le transistor de sortie soit fermé : $V_{CE} = 0$.

Le transistor en « commutation » possède un fonctionnement semblable au relais : la présence d'un courant de base très faible provoque la fermeture de la jonction CE. La commutation s'effectue au cœur du semi-conducteur sans mouvement (pas de contact) contrairement au relais.



Le codage hexadécimal : base 16

L'hexadécimal est un système de numération formé de 16 symboles alphanumériques : 10 chiffres + 6 lettres.

Le transcodage d'un code binaire en hexadécimal permet de simplifier l'écriture des codes binaires qui deviennent de plus en plus longs : les systèmes informatiques fonctionnent sous **16, 32 et 64 bits**.

AN5 :

$$1010\ 1110\ 0110\ 1011_{(2)} = 2^0 + 2^1 + 2^3 + 2^5 + 2^6 + 2^9 + 2^{10} + 2^{11} + 2^{13} + 2^{15} = 44651_{(10)} = \text{AE6B}_{(16)}$$

Décimal	44651			
Binaire	1010	1110	0110	1011
Hexadécimal	A	E	6	B

AN6 : convertissez le nombre binaire suivant en base 10 et 16.

$$1111\ 0011\ 1100\ 0011_{(2)} = \quad (10) = \quad = \quad (16)$$

Décimal				
Binaire				
Hexadécimal				

Correspondance entre binaire, décimal et hexadécimal																	
Base 10	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Base 2	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111	10000
Base 16	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10

Adressage IP :

La numération binaire est utilisée pour « numéroter » les ordinateurs connectés sur le réseau internet. Cette « adresse » contenant le numéro du réseau ainsi que le numéro de la machine est unique : l'**IPv4** est le protocole de codage des adresses **IP** sur 4 octets, il permet (théoriquement) de connecter 2^{32} postes en réseau (soit ≈ 4 milliards).

Sur internet, un ordinateur est repéré par son adresse **IP** codée sur **4 octets** ou **32 bits** suivant le modèle suivant : **chaque octet est séparé par un point**.

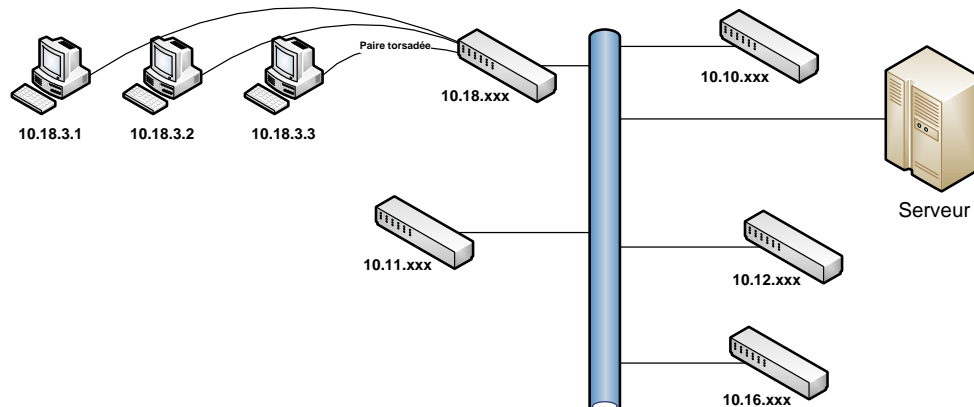
Réseau	X rouge : N° réseau				X bleu : N° postes
Classe A	0XXXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	1.0.0.0 à 126.0.0.0
Classe B	10XXXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	128.0.0.0 à 191.255.0.0
Classe C	110XXXXX	XXXXXXXX	XXXXXXXX	XXXXXXXX	192.0.0.0 à 223.255.255.0
IP	XXXXXXXX · XXXXXXXX · XXXXXXXX · XXXXXXXX				
octets	8bits	8bits	8bits	8bits	

Donnez le nombre maximal d'ordinateurs que l'on peut raccorder sur ce réseau.

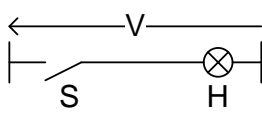
Classe	Décodage d'adresse IP (2)				Nb de postes	Adresse IP (10)
A	01111111	00000000	00000000	00000000	?	126.0.0.0
	00000001	00000000	00000000	00000000		1.0.0.0
B	10111111	11111111	00000000	00000000	?	191.255.0.0
	10000000	00000000	00000000	00000000		128.0.0.0
C	11011111	11111111	11111111	00000000	?	223.255.255.0
	11000000	00000000	00000000	00000000		192.0.0.0

Réseaux privés :

Dans chaque classe, quelques adresses sont réservées à la création de réseaux privés, ainsi en **classe A**, les adresses comprises entre **10.0.0.1** à **10.255.255.254** sont utilisables. Voici ci-dessous l'adressage IP d'ordinateurs appartenant à un réseau (simplifié) privé de **classe A**: Nous reviendrons ultérieurement sur la structure matérielle du réseau.



Equation logique :

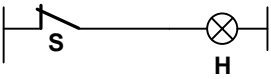
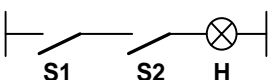
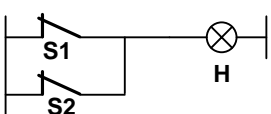
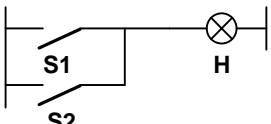
	<p>Comme en algèbre, une équation logique possède des « variables » : ici L et H pour l'exemple précédent : ces variables ne peuvent prendre que la valeur 0 ou 1.</p> <p>On remarque que l'équation logique, en établissant un lien entre les différentes variables d'un circuit, ne permet pas d'effectuer des calculs numériques, mais nous renseigne sur son fonctionnement lors de la modification de l'état d'une des variables.</p>
---	---

Variable d'entrée et de sortie :

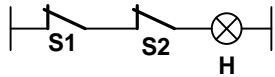
Ainsi, lorsque l'on écrit **S = H**, il faut comprendre qu'il y a égalité d'état entre la variable « **d'entrée** » (celle sur laquelle on agit) et la variable de « **sortie** » qui est le résultat de l'action sur la/les variables d'entrée.

Fonctions logiques

Il y a des combinaisons de variables d'entrée qui forment des fonctions **logiques** de base, dont la connaissance simplifie la compréhension des circuits électriques et des automatismes. Ces fonctions, auxquelles on fait souvent appel sont contenues dans le tableau ci-dessous.

Fonction	Schéma	Tableau de vérité	Equation logique															
NON (NOT)		<table border="1"> <thead> <tr> <th>S</th> <th>H</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </tbody> </table>	S	H	0	1	1	0	$H = \bar{S}$ <p>\bar{S} se lit: pas S</p>									
S	H																	
0	1																	
1	0																	
ET (AND)		<table border="1"> <thead> <tr> <th>S1</th> <th>S2</th> <th>H</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	S1	S2	H	0	0	0	0	1	0	1	0	0	1	1	1	$H = S1 \text{ ET } S2$ $H = S1 \cdot S2$
S1	S2	H																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
NON ET (NAND)		<table border="1"> <thead> <tr> <th>S1</th> <th>S2</th> <th>H</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table>	S1	S2	H	0	0	1	0	1	1	1	0	1	1	1	0	<p>Théorème de De Morgan</p> $H = \overline{S1 \cdot S2}$ $H = \bar{S1} + \bar{S2}$
S1	S2	H																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
OU (OR)		<table border="1"> <thead> <tr> <th>S1</th> <th>S2</th> <th>H</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table>	S1	S2	H	0	0	0	0	1	1	1	0	1	1	1	1	$H = S1 \text{ OU } S2$ $H = S1 + S2$
S1	S2	H																
0	0	0																
0	1	1																
1	0	1																
1	1	1																

NON OU
(NOR)



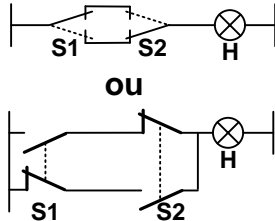
S1	S2	H
0	0	1
0	1	0
1	0	0
1	1	0

Théorème de
De Morgan

$$H = \overline{S1 + S2}$$

$$H = \overline{S1} \cdot \overline{S2}$$

OU
exclusif
(XOR)



S1	S2	H
0	0	0
0	1	1
1	0	1
1	1	0

$$H = S1 \cdot \overline{S2} + \overline{S1} \cdot S2$$

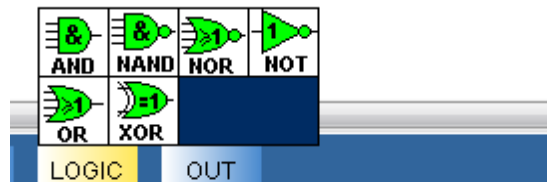
$$H = S1 \oplus S2$$

Représentation des fonctions logiques :

Fonction	Symbole ou logigramme		Fonction	Symbole ou logigramme	
	Américain	européen		Américain	européen
NON			ET		
OU			NOR		
NAND			XOR Ou exclusif		

Module logique du relais logique « ZELIO »

Note : les symboles **américains** sont très largement utilisés par les logiciels de programmation, de CAO, etc. Le « **relais logique** » ZELIO utilisé pour réaliser les applications contenues dans ce cours possède une bibliothèque « logique » permettant de programmer des automatismes directement à l'aide de blocs logiques.



Intérêt des fonctions logiques :

Application simple en informatique :

Propriété	Valeur
Suffixe DNS propre à la ...	
Description	Intel(R) Wireless WiFi Link 4965AGN
Adresse physique	00-1F-3B-2B-4E-DB
DHCP activé	Oui
IPv4 Adresse IP	192.168.1.2
IPv4 Masque de sous-ré...	255.255.255.0
Bail obtenu	lundi 6 février 2012 18:29:35
Bail expirant	lundi 13 février 2012 18:29:39
IPv4 Passerelle par défaut	192.168.1.1
IPv4 Serveur DHCP	192.168.1.1
IPv4 Serveurs DNS	192.168.1.1 192.168.1.1
IPv4 Serveur WINS	
NetBIOS sur TCP/IP act...	Oui
Adresse IPv6 locale de li...	fe80::5515:6256:1e92:4a15%12
IPv6 Passerelle par défaut	
IPv6 Serveur DNS	

En effectuant un **ET LOGIQUE** entre l'**IP** et le **masque de sous réseau**, on **sépare** l'adresse réseau de l'adresse machine qui sont tous les deux contenus dans l'IP. Cette action permet de créer artificiellement des sous-réseaux. Le masque, comme l'IP, est codé sur **4 octets**. Dans le cas ci-contre, l'adresse **IP** est : **192.168.1.2** et le masque : **255.255.255.0** (réseau de classe C).

Effectuons le **ET** à l'aide des valeurs binaires :

	11000000	10101000	00000001	00000010
ET	11111111	11111111	11111111	00000000
=	11000000	10101000	00000001	00000000
=	192	168	1	0

Le masque, comme son nom l'indique a caché le numéro du poste pour faire apparaître le numéro de réseau.

Note : l'**IPv4** est le protocole de codage des adresses **IP** sur 4 octets, elle permet (théoriquement) de connecter 2^{32} postes en réseau (soit ≈ 4 milliards). Ce protocole, dépassé par l'explosion d'internet va être remplacé par l'**IPv6** : codage sur **16 octets**.

Application en logique câblée :

On oublie souvent qu'une bonne connaissance des fonctions logiques permet un apprentissage plus aisé du schéma électrique.

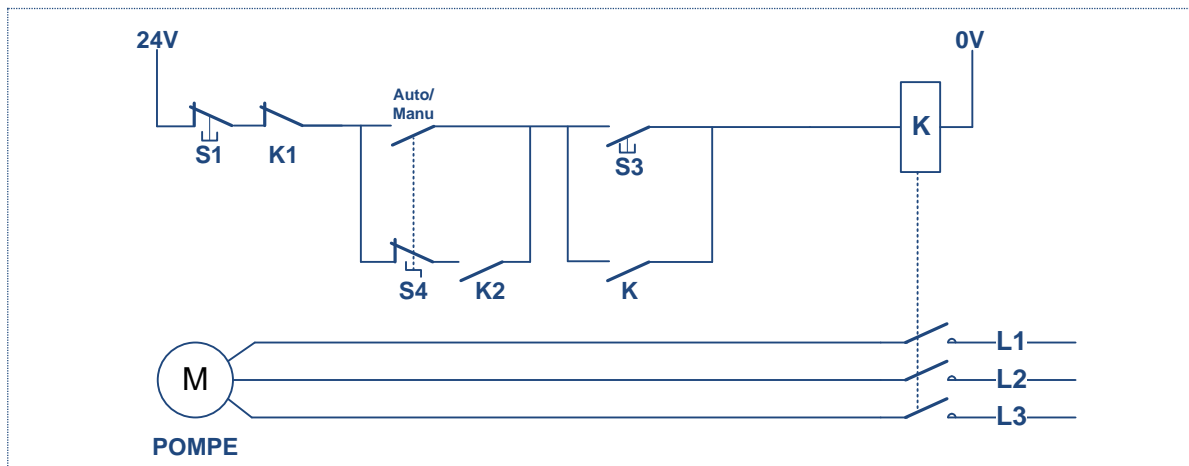
Exemple : Etude du fonctionnement d'une pompe.

Description du fonctionnement : (les appareils de protection ne sont pas représentés)

Si le mode manu est sélectionné: Le contacteur K est alimenté si une impulsion est donnée sur S3 et s'il n'y a pas de défaut.

Le mode auto désactive le mode manuel et provoque la mise sous tension de K si le cycle continu est validé et si une impulsion est donnée sur S3. Les conditions d'arrêt sont les mêmes qu'en mode manu. Une impulsion sur S1 provoque la mise hors tension de K dans les deux modes.

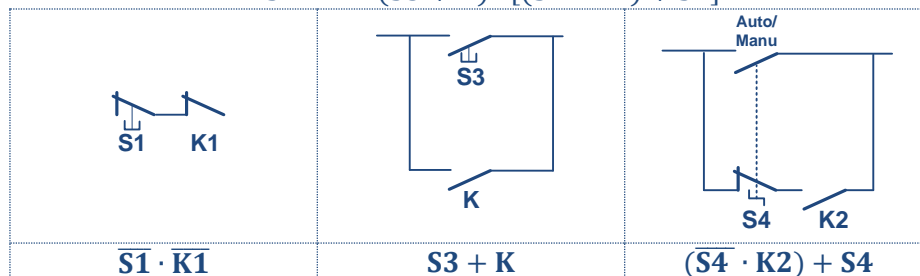
S1: arrêt	S4: Auto/Manu	S3: Marche	K1: absence défaut	K2: cycle continu



Equation logique du fonctionnement de la pompe:

Le fonctionnement décrit ci-dessus est réalisé par l'équation ci-dessous :

$$K = \overline{S1} \cdot \overline{K1} \cdot (S3 + K) \cdot [(\overline{S4} \cdot K2) + S4]$$



Application en logique programmée : relais logique ZELIO de Schneider

Les automates et les relais logiques **remplacent** la logique câblée par une **logique programmée**: Les câblages deviennent plus simples, plus fiables, modulables...etc. Ils sont d'un prix très abordables.








- En programmant les **blocs fonctions** (ET, OU, ...etc.) déjà en mémoire dans l'appareil (Zelio de Schneider).
- En programmant en langage **ladder** ou « contact » qui est une représentation « électrique » du programme (pratique pour les électriciens): dans le cas qui nous intéresse nous pourrions constater la similitude du programme et du schéma.


Principe : Le relais logique « Zelio » possède des **entrées** (nombre variable en fonction du modèle) sur lesquelles sont raccordées les BP, inters, capteurs, etc. : **variables d'entrées**.

Une seule sortie : le contacteur de mise sous tension de la pompe.


Entrées physiques

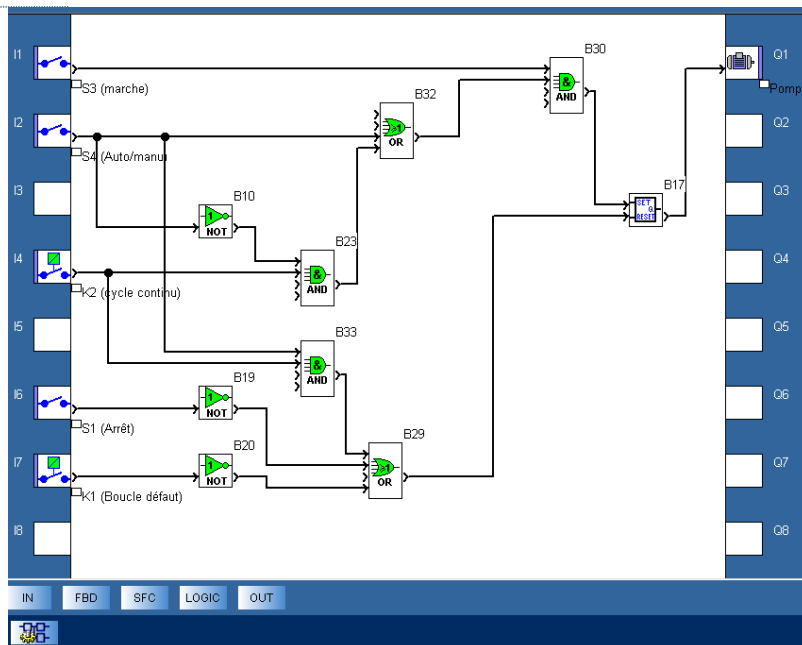
Entrée	No	Symbole	Fonction	Verrou	Paramètres	Commentaire
I1	B07		Contact	—	Pas de paramètres	S3 (marche)
I2	B04		Contact	—	Pas de paramètres	S4 (Auto/manu)
I4	B05		Relais	—	Pas de paramètres	K2 (cycle continu)
I6	B01		Contact	—	Pas de paramètres	S1 (Arrêt)
I7	B18		Relais	—	Pas de paramètres	K1 (Boucle défaut)

Sorties physiques

Sortie	No	Symbole	Fonction	Commentaire
Q1	B13		Moteur	Pompe

Fonctions paramétrables

No	Symbole	Fonction	Verrou	Rémanence	Paramètres	Commentaire
B17		Bascule RS	—	—	Priorité : RESET prioritaire	

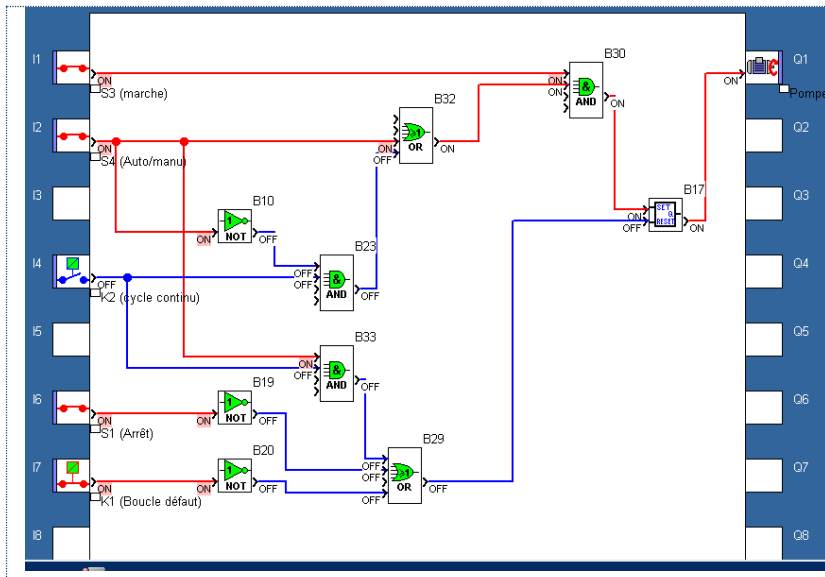


Logigramme de l'application :

Les entrées utilisées sont visibles sur le schéma à gauche et sont décrites dans le tableau « entrées physiques ».

La sortie unique Q1 est visible à droite.

Entre les entrées et les sorties se « dessine » le programme de l'application : **Un logigramme composé de fonctions NON, ET, OU et d'une bascule RS (mémoire).**

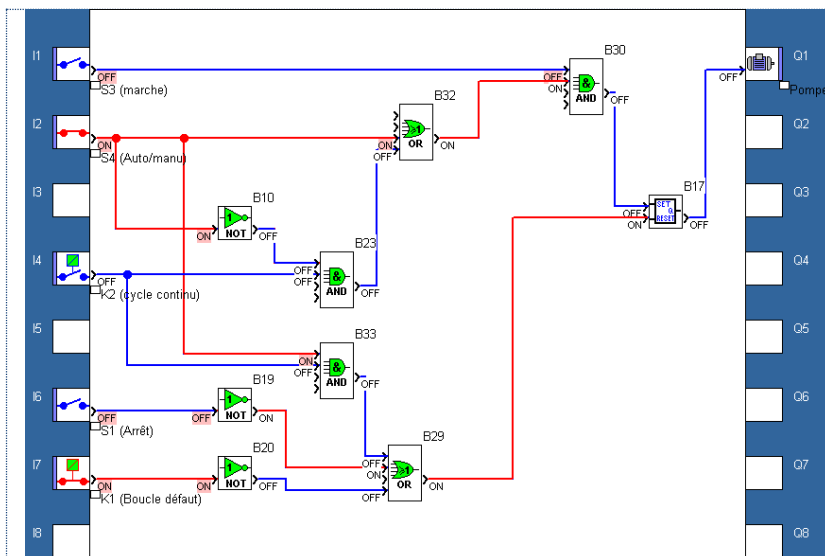


En mode manu :

S4 et S3 sont fermés (S3 va être relâché car BP). La bascule voit l'état **1** sur l'entrée SET : la sortie de B17 = 1.

Les entrées **16 et 17** sont raccordées aux contacts NC S1 et K1, donc les états 1 appliqués aux entrées B19 et B20 sont transformés en 0 par les NOT (NON).

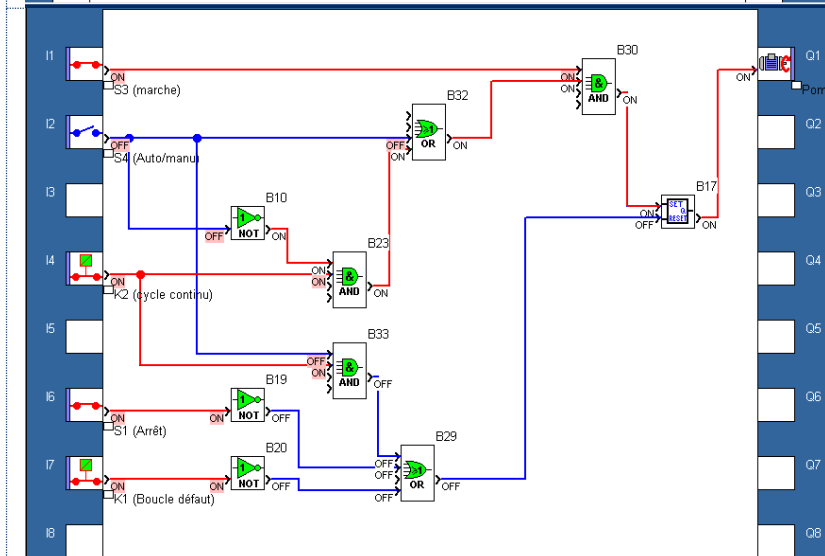
B29 (OU) = 0 + 0 = 0 donc sans effet sur le RESET de la bascule : le moteur fonctionne.



Un appui sur l'arrêt S1

provoque le basculement de B19 et de B29 : l'entrée RESET de la bascule se trouve à 1 et fait basculer sa sortie à 0. **Le moteur n'est plus alimenté.**

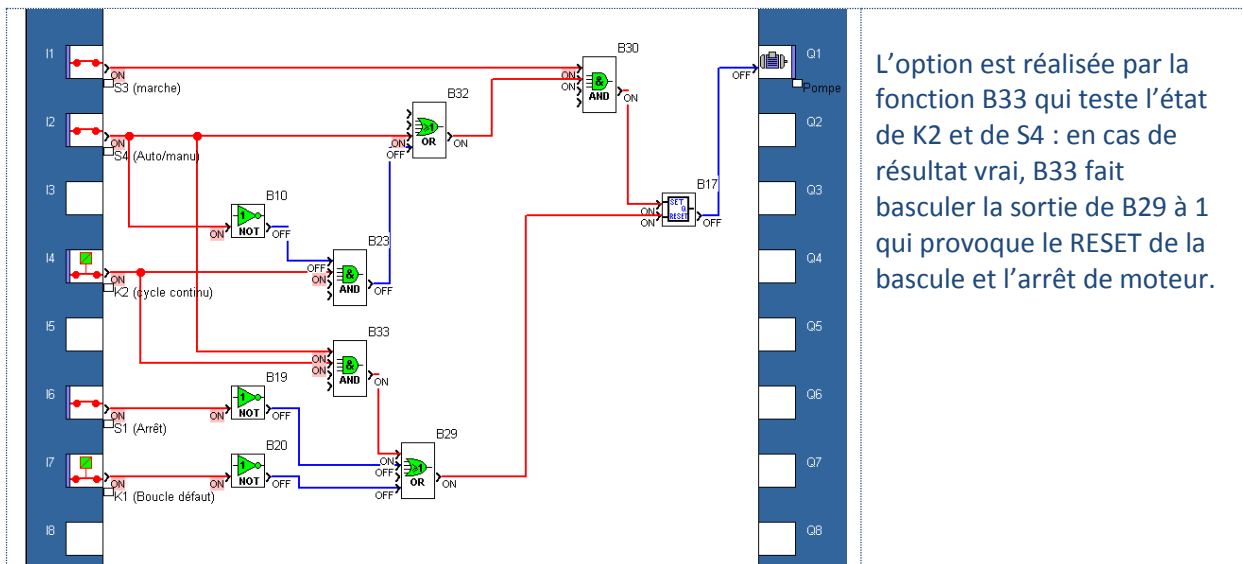
On observerait le même résultat en cas d'ouverture de K1 en cas de défaut de l'installation.



En mode auto :

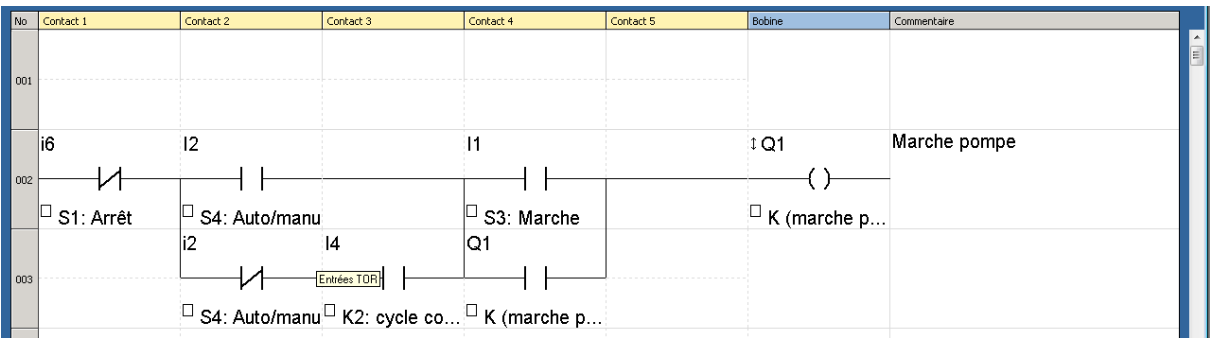
S4 relâché et K2 fermé provoquent le passage à 1 de B23 (ET), B32 (OU) et B30 (ET) : l'entrée SET de la bascule voit un état haut et bascule à 1. Le moteur est alimenté.

J'ai ajouté une petite option au programme : l'alimentation du moteur sera coupée en cas de basculement accidentel de l'installation en mode manu alors qu'on se trouve commuté en auto.



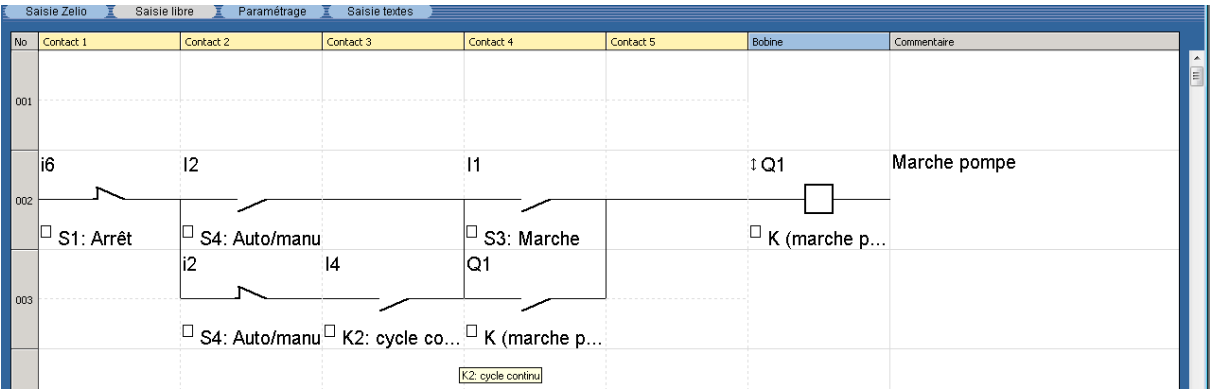
L'option est réalisée par la fonction B33 qui teste l'état de K2 et de S4 : en cas de résultat vrai, B33 fait basculer la sortie de B29 à 1 qui provoque le RESET de la bascule et l'arrêt de moteur.

- **Le LADDER**** ci-dessous reproduit les différents « contacts » comme sur un schéma électrique mais c'est pourtant un programme qui produira le même fonctionnement que le logigramme précédent.
****Ladder** : signifie échelle en américain de par la ressemblance du programme ladder avec un schéma électrique développé sur plusieurs lignes.



En langage « contact » :

Inutile d'en dire plus, le « programme » est explicite car il est la réplique exacte du schéma mais n'utilise aucun fil car les liaisons représentées sont logiques.



Note : L'application qui vient d'être étudiée entre dans la catégorie : **LOGIQUE COMBINATOIRE**, cela signifie que les actions effectuées sur les variables d'entrées produisent toujours le même résultat indépendamment de l'ordre dans lequel on les effectue.

Schéma de câblage du système de pompage:

Tous les programmes réalisés ci-dessus fonctionnent suivant un même schéma représenté ci-dessous. On voit le grand intérêt que revêtent ces relais programmables en termes de fonctionnalités et de simplicité de mise en œuvre.

Raccordement des entrées du relais ou de l'automate :

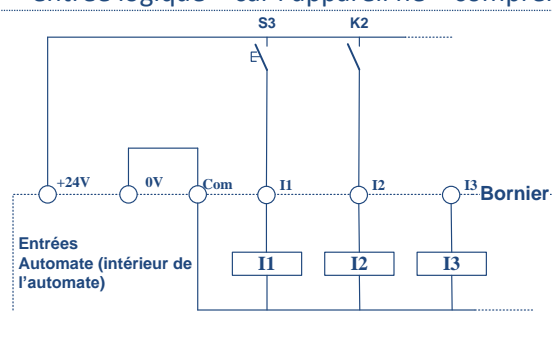
Un automate ou un relais logique possède des entrées sur lesquelles sont raccordées les variables d'entrées : BP, inters, capteur T.O.R, capteur analogique, etc.

Le principe retenu pour le câblage des entrées est universel : il consiste à relier l'entrée à un potentiel pour lequel elle est prévue de fonctionner. L'appareil que nous avons choisi pour cette application possède des entrées 230V :

Lorsque l'entrée se trouve soumise à 230V, l'état logique vu par l'automate est égal à 1.

Seul le principe d'alimentation des entrées décrit ci-dessus est généralisé à tous les automates car **les tensions d'entrées sont variables d'une marque à l'autre et doivent bien sur être respectées sous peine de détériorer l'appareil.**

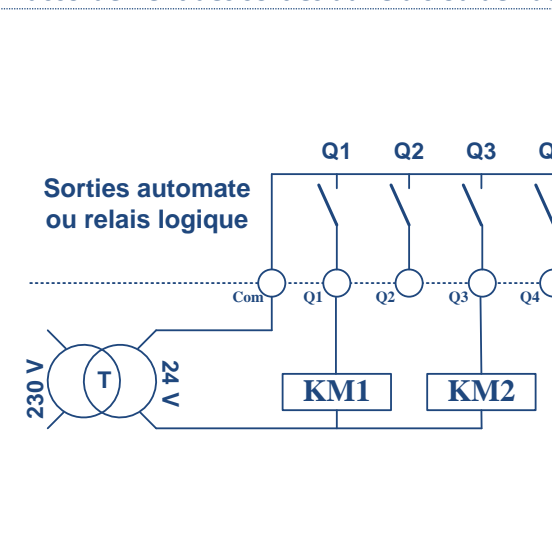
Les entrées de ce relais logique sont identifiées par l'acronyme **T.O.R : Tout Ou Rien**, cela signifie « entrée logique » car l'appareil ne « comprend » que **l'information 0 ou 1.**



Entrées Automate (intérieur de l'automate)

Sur les automates industriels, le principe ci-contre est en général adopté : Une alimentation interne ou externe permet de changer la polarisation des entrées afin de pouvoir raccorder tous types de capteurs électroniques : **NPN ou PNP**. On représente les entrées automate sous forme de bobine de relais car cela permet de mieux visualiser l'action de l'électricien, n'oublions pas que c'est faux du point de vue de la réalité.

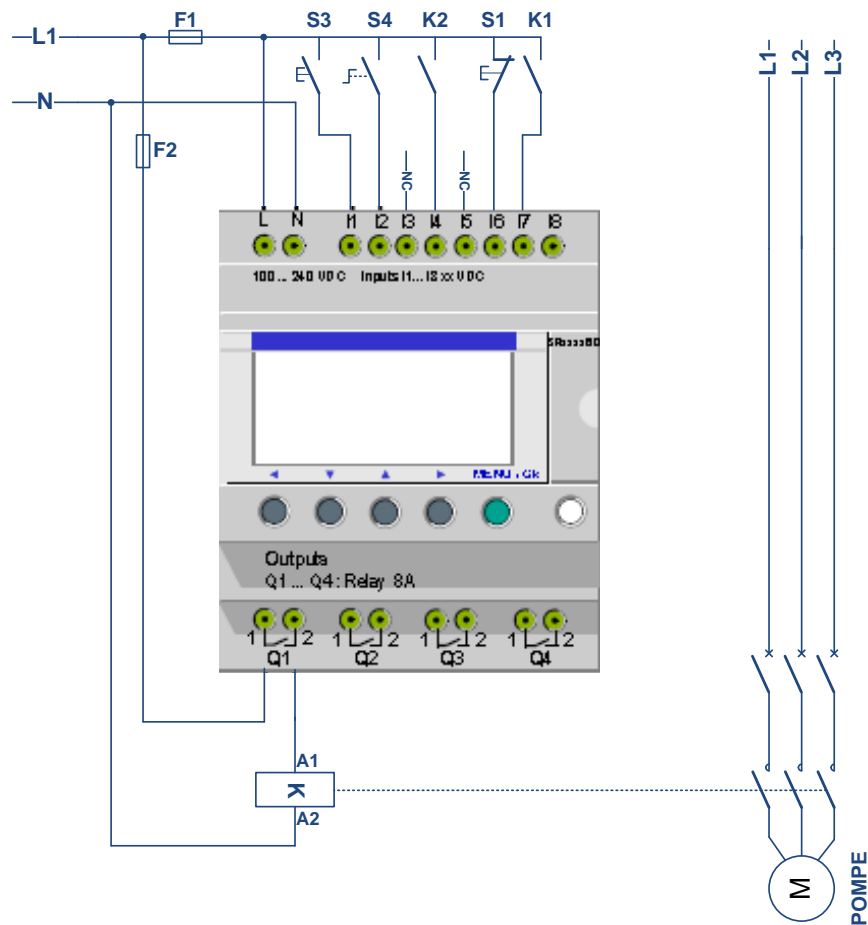
Raccordement des sorties du relais ou de l'automate :



Sorties automate ou relais logique

Les sorties sont des contacts « secs », c'est-à-dire non alimentés dont les caractéristiques de pouvoir de coupure sont données dans les documents constructeurs. L'alimentation des préactionneurs est généralement réalisée en TBT lorsque l'on utilise des automates programmables industriels : alimentation de bobine de contacteurs ou de relais. Les relais logiques sont aussi utilisés dans des applications tertiaires : gestion de chauffage, éclairage, etc., et peuvent donc, soit commander directement les sources, soit être relayés.

Capteurs analogiques et numériques : [lien](#)



Application : Complétez l'équipement de pompage

Insérez deux voyants de signalisation :

- Un voyant «Marche pompe»
 - Un voyant «Défaut pompe»
- a) Vous modifierez le schéma électrique du module **Zelio** ainsi qu'un des programmes au choix : logigramme ou langage contact ou **ladder**.
- b) Vous effectuerez les tests afin de valider le fonctionnement de votre installation.

Note importante: L'application ci-dessus n'est étudiée que du point de vue fonctionnement. Ne perdons pas de vue qu'automatiser un équipement (surtout s'il est industriel) ne se résume pas à le mettre « en marche », encore faut-il penser à gérer tous les **modes de marches et d'arrêts**: c'est le rôle du **GEMMA** non traité ici. En effet, peut-on imaginer de faire reposer l'arrêt complet d'un système uniquement sur un **ordre logique** sans le compléter par une **coupure électrique** d'un/des préactionneurs ?

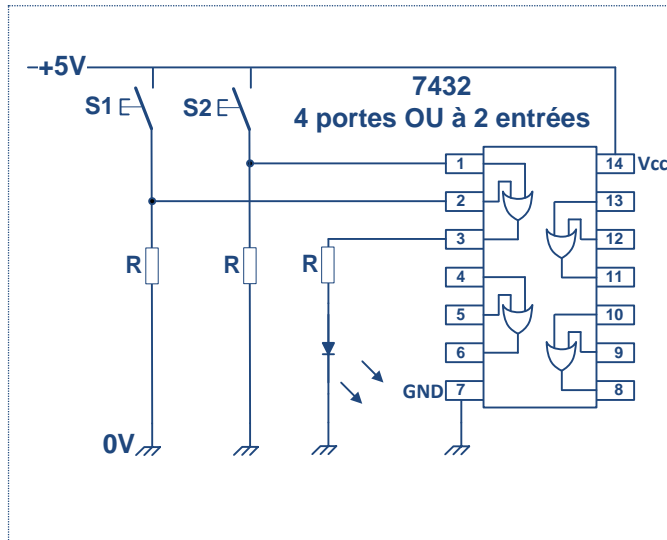
Les circuits intégrés logiques :



Nous venons de voir la place importante des fonctions logiques dans le monde de l'électricité. Il manque une application que je ne développerai pas mais que je donne à titre d'information : **Les circuits intégrés logiques.**

Les fonctions logiques se trouvent sous forme:

- Câblées : circuits électriques.
- Programmées : informatique ou automatisme (informatique industrielle)
- Intégrées dans des circuits logiques (idem forme câblée) : **voir ci-dessous**

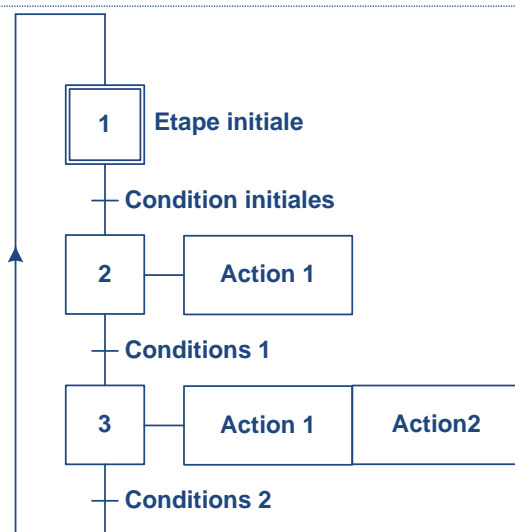


Bien qu'ils permettent de traiter les mêmes applications que les circuits logiques programmés, Les circuits intégrés logiques sont cantonnés à des applications plus légères car ils entrent dans la catégorie de la logique câblée et en possèdent tous les défauts. La réalisation d'une application doit être étudiée au cas par cas et nécessitent un matériel spécifique pour sa mise en œuvre : logiciel CAO pour l'étude du circuit imprimé, réalisation du circuit imprimé, mise en place et soudure des composants.

Logique séquentielle :

Nous venons de voir qu'en logique combinatoire, l'ordre dans lequel sont effectuées les actions sur les variables d'entrée n'a pas d'importance sur la/les variables de sortie : ainsi les commande marche et arrêt sont indépendantes l'une de l'autre, et du moment où elles sont effectuées.

En logique séquentielle, le système automatisé décrit un cycle composé d'actions qui s'effectuent dans un ordre préétabli qui ne peut pas être changé lors de son fonctionnement : ainsi le cycle de trois étapes représenté ci-dessous est figé (soit par programmation soit par le câblage).



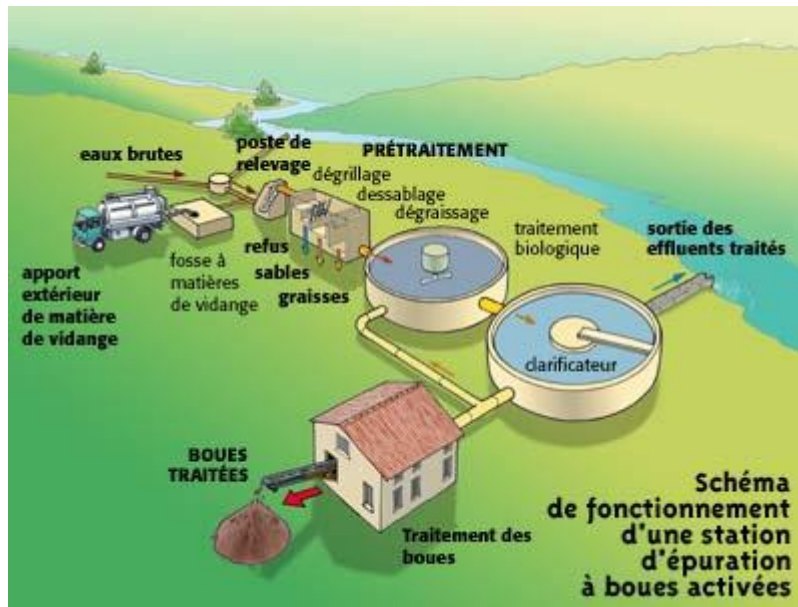
L'action 1 associée à **l'étape 2** ne pourra être lancée que si **l'étape initiale** est « active » et que les conditions initiales de départ de cycle réunies.

L'action 1 prend fin lorsque la condition 1 est vraie ($C1 = 1$) : l'étape 3 est désormais activée.

Cette représentation d'un cycle s'appelle le **GRAF CET**, elle permet non seulement décrire les séquences d'un cycle mais aussi d'effectuer sa programmation sur les automates actuels qui comprennent toutes les structures de ce langage.

Le **grafcet** ci-contre est linéaire mais d'autres

structures existent et seront développées plus loin dans le cours.



Source ADEME

Situation : Reprenons l'exemple précédent : Le système de pompage étudié est situé dans une station d'épuration (**Poste de relevage**): les pompes de relevage sont chargées de remonter les eaux usées stockées dans un réservoir jusqu'à une zone de dessablage chargée de retenir les sables qui ne sont pas dégradables. Les eaux usées sont amenées dans le réservoir par gravité. Le réservoir de stockage se situe à un niveau plus bas que les sites de traitement et de rejet dans le circuit de distribution : d'où le nom de pompe ou poste de relevage.

Situation du réservoir de stockage :

Le réservoir de stockage des eaux usées est équipé d'une pompe de relevage qui est immergée grâce aux rails de guidage prévus à cet effet.

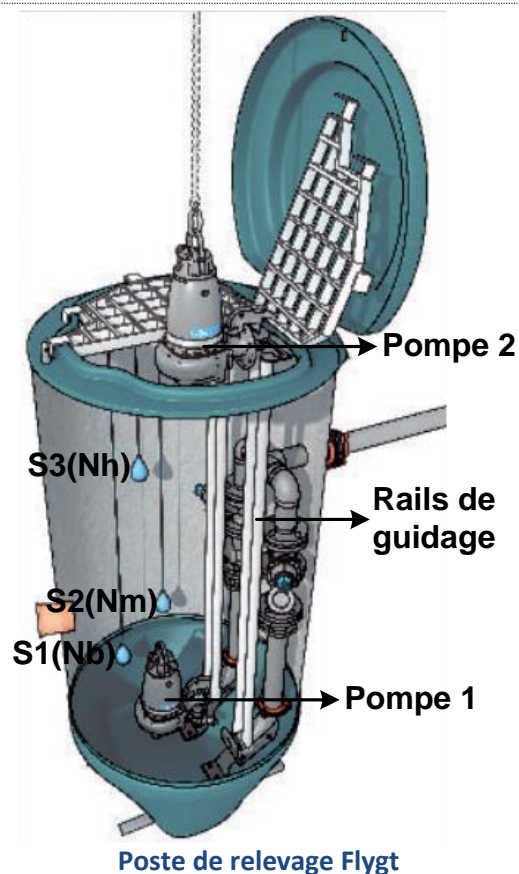
Détection du niveau :

Trois capteurs de niveau sont chargés de réguler le niveau d'eau dans le réservoir : **S1** (niveau bas), **S2** (niveau moyen), **S3** (niveau haut).

Travail demandé :

On désire ajouter une deuxième pompe de relevage pour sécuriser et rendre l'équipement plus performant. De plus, il est prévu une permutation du fonctionnement des pompes pour éviter le non fonctionnement prolongé de l'une ou l'autre des machines.

Le fonctionnement en mode automatique de l'installation existante est décrit dans le graphe ci-dessous.



Fonctionnement de la pompe 1 : Fig.1

Le **grafcet** ci-contre, proche de l'exemple donné dans le cours, nous montre que la pompe travaille uniquement entre le niveau moyen et haut : ne perdons pas de vue que le but est de vider le réservoir.

Détail du fonctionnement :

A la fermeture de S2 la pompe 1 se met en fonctionnement, continue de pomper si le niveau de l'eau atteint S3 et ne sera mise hors tension qu'à l'ouverture de S2.

Ajout de la pompe 2 : Fig.2

On désire ajouter une **pompe 2** pour « aider » la **pompe 1** lorsque le niveau de l'eau dépasse le niveau du détecteur S3, pour cela, une deuxième action est ajoutée à l'étape 3 : les deux actions sont effectuées simultanément.

Conclusion : on ne voit pas pour l'instant l'intérêt de développer un outil comme le **grafcet** pour résoudre ce problème d'automatisme, car la solution **électrique ou LADDER** est relativement simple à mettre en œuvre comme il est montré ci-dessous.

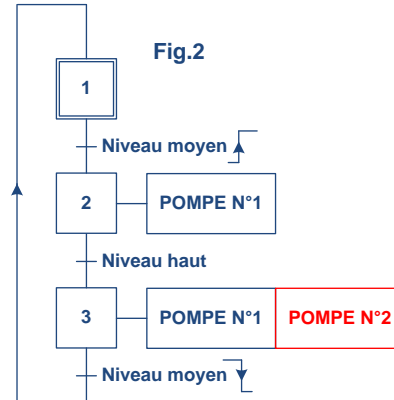
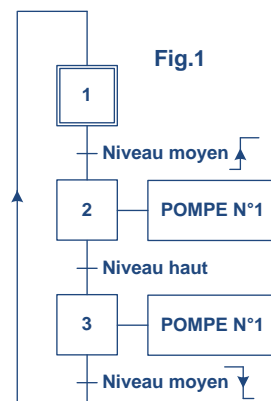
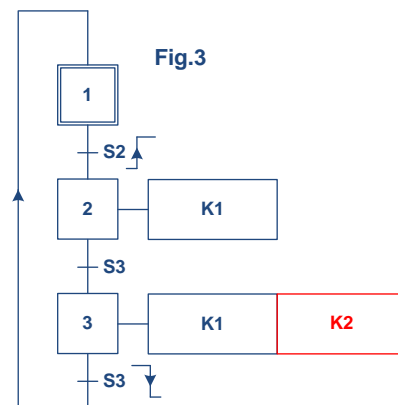
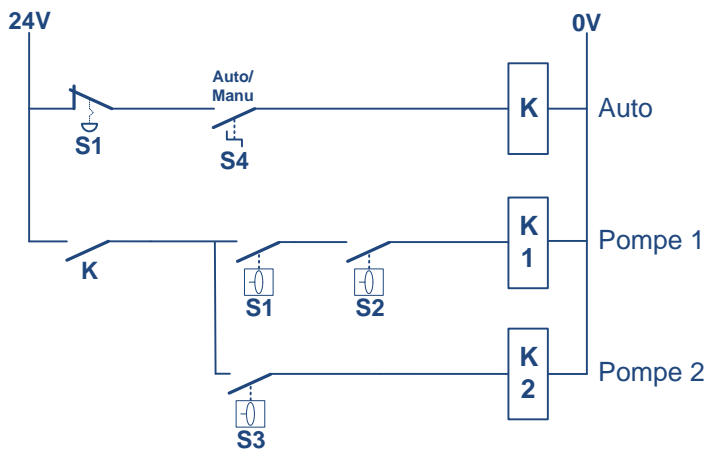
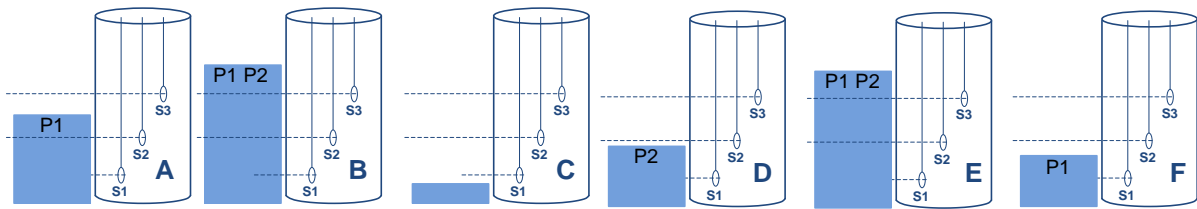


Schéma développé :

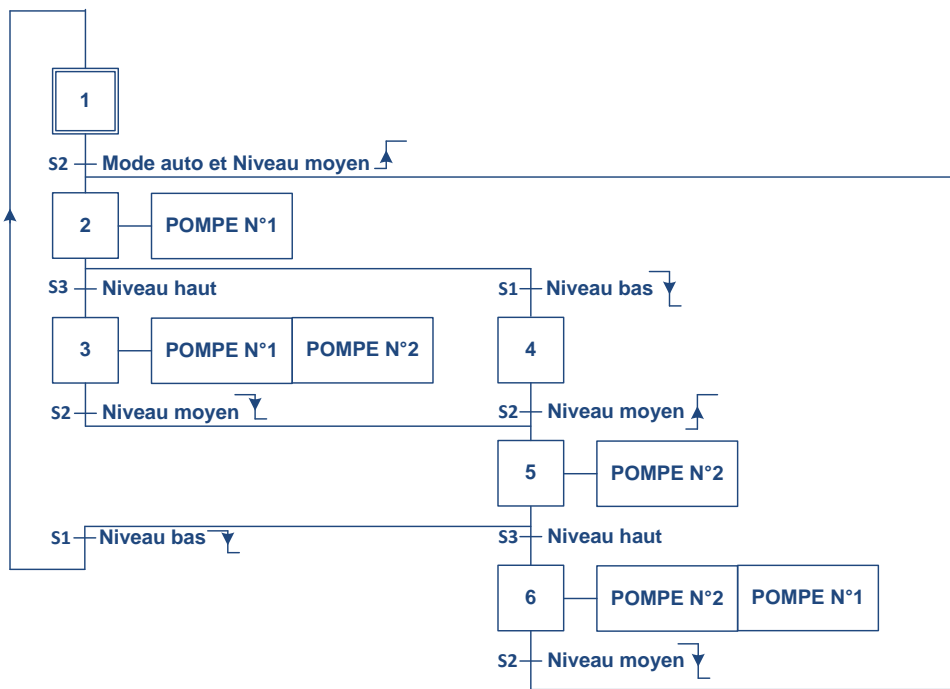


Etudions la dernière partie du cahier des charges :

Celui-ci prévoit la **permutation du fonctionnement des pompes** afin d'équilibrer leur temps de fonctionnement : **Fig.4**



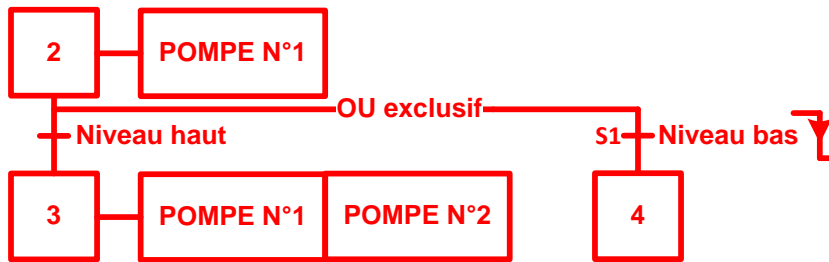
Grafcet fonctionnel de la station de pompage :



Décodons la structure du grafcet :

	<p>La pompe 1 démarre lorsque le niveau de l'eau passe S2 : Schéma A ; à ce moment deux solutions peuvent se produire :</p> <ol style="list-style-type: none"> 1) Soit le niveau de l'eau continue d'augmenter et atteint S3 : <u>évolution de l'étape 2 vers l'étape 3</u> <p style="text-align: right;">schéma B</p>
	<ol style="list-style-type: none"> 2) Soit le niveau baisse et passe sous S1: <u>évolution de l'étape 2 vers l'étape 4</u> <p style="text-align: right;">schéma C</p> <p>Lorsqu'une étape n'est associée à aucune action cela signifie qu'aucune « machine » ne fonctionne et que l'automatisme attend un « évènement » : Ici, que l'eau remonte jusqu'au niveau moyen S2.</p>

Le choix des 2 solutions possibles **schéma B** OU **C** constitue une **divergence en OU exclusif** : 1 seul choix est possible.



La divergence permet à l'automatisme de choisir entre deux cycles. Deux cycles peuvent aussi converger vers un seul : dans ce cas nous parlerons de **convergence** comme au point décrit ci-dessous.

Décodage : Lors de la descente du niveau d'eau (ouverture de S2), la pompe 2 est alimentée au lieu de la pompe 1 qui fonctionnait lors de l'étape 2 : la permutation a eu lieu.



Vous avez maintenant tous les éléments qui vous permettent le décodage du fonctionnement de la station de relevage.

L'équipement doit-il être réalisé en logique câblée ou programmée ?

Un coup d'œil au fonctionnement donné par le grafcet permet de proscrire la version câblée car seul le rôle de S3 est clair : mise en route de P1 et P2. En revanche, la permutation de P1 et P2 en logique câblée demeure obscure et demande une bonne expérience du schéma électrique pour être réalisée car elle dépend aussi des choix technologiques qui sont faits: je vous donnerai une solution possible.

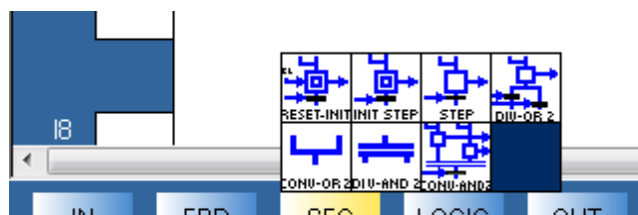
Solution programmée :

La solution proposée fait encore appel au module logique **Zelio** qui intègre aussi le langage grafcet : en situation réelle nous choisirions un « vrai » automate programmable plus « performant » du point de vue du nombre d'entrées et de sorties. D'autre part, l'automatisme serait rendu **communiquant** pour être surveillé et commandé à distance, nous verrons tout cela dans la deuxième partie.

Symbolisation utilisée par le logiciel :

Bien qu'elle diffère sensiblement d'un logiciel de programmation à l'autre, on reconnaît facilement la structure du **grafcet** car elle répond à une norme graphique et fonctionnelle universelle.

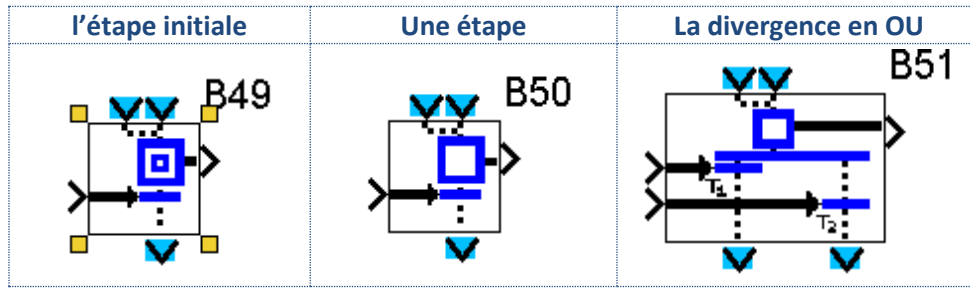
Ainsi, la bibliothèque de symboles du logiciel du progiciel de programmation des modules **Zelio** est la suivante :



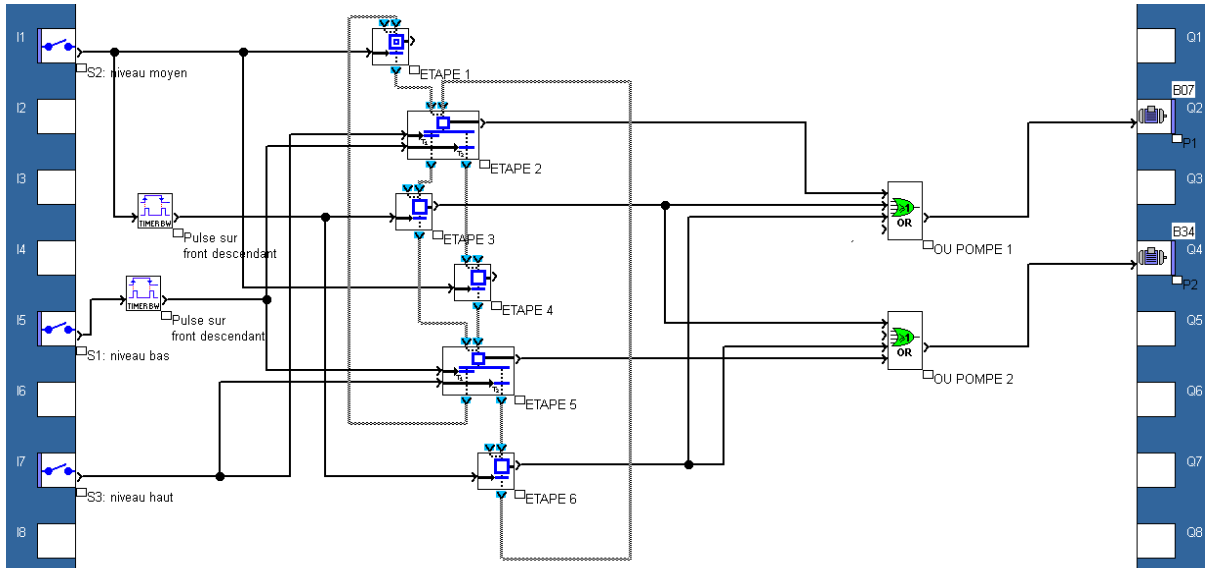
On reconnaît:

Marc Sanchez

<http://electrotechnique.fr>

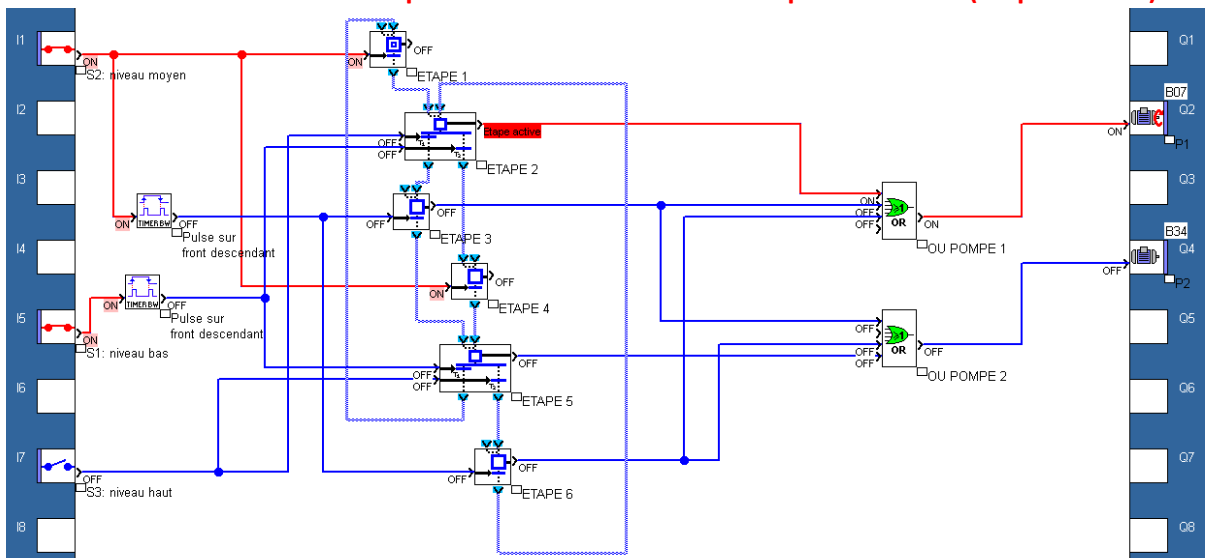


Le grafset fonctionnel programmé pour le module zelio est : [quelques précisions en fin du document](#)

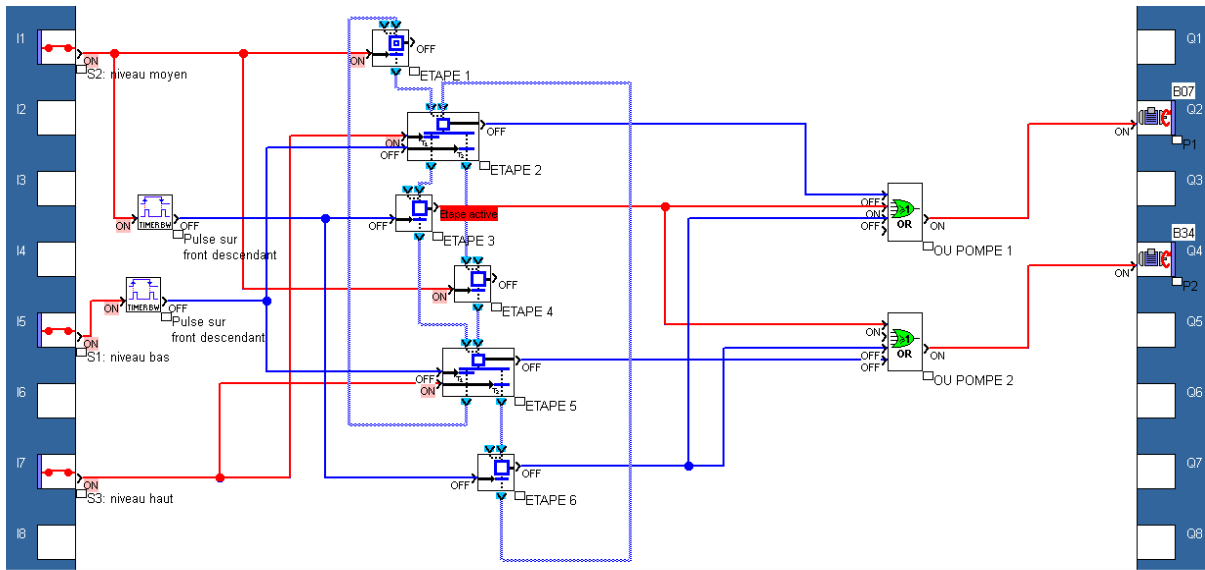


Simulation du fonctionnement du grafset:

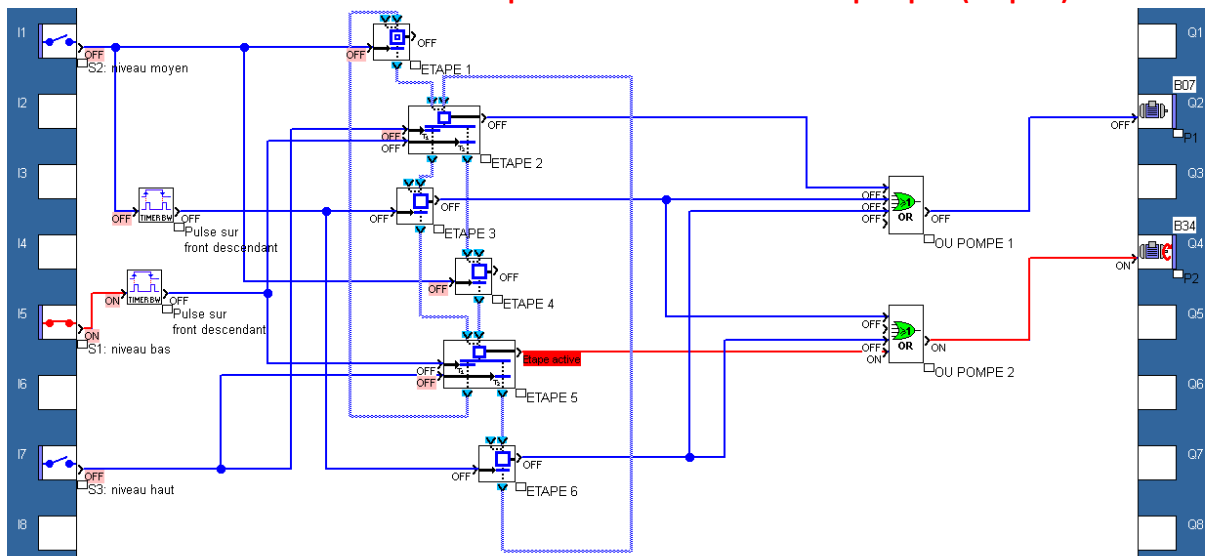
Niveau S1 et S2 atteints à la première mise en service : Pompe1 en service (étape 2 active)



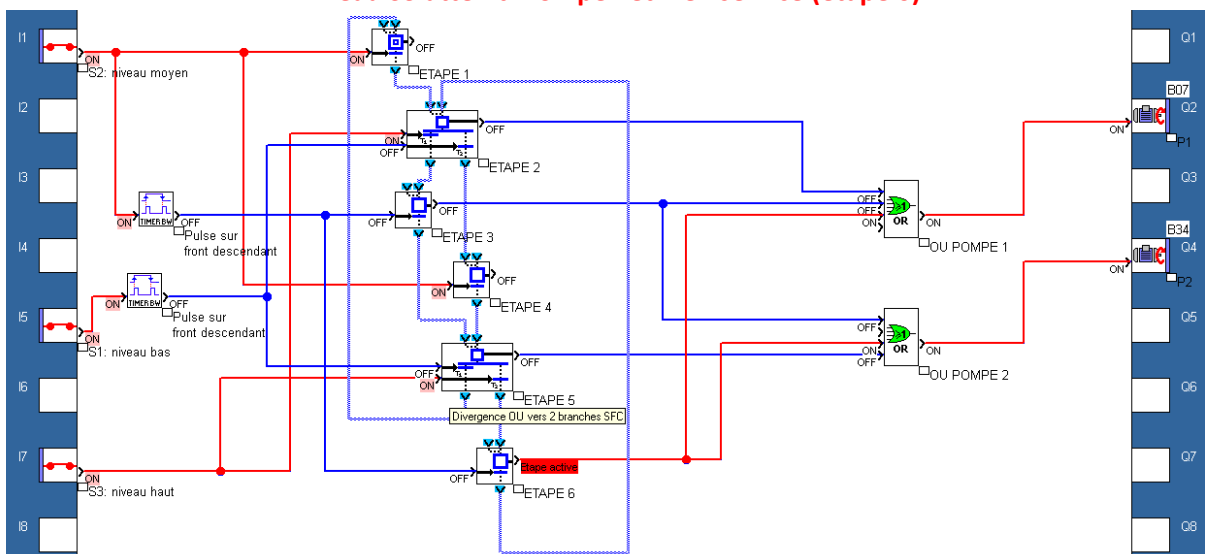
Niveau S3 atteint: Pompe1 et 2 en service (étape 3)



Niveau descend sous S2: Pompe2 en service et arrêt de la pompe1 (étape 5)



Niveau S3 atteint: Pompe1 et 2 en service (étape 6)



Niveau descend sous S2: Pompe1 en service et arrêt de la pompe2 (étape 2)

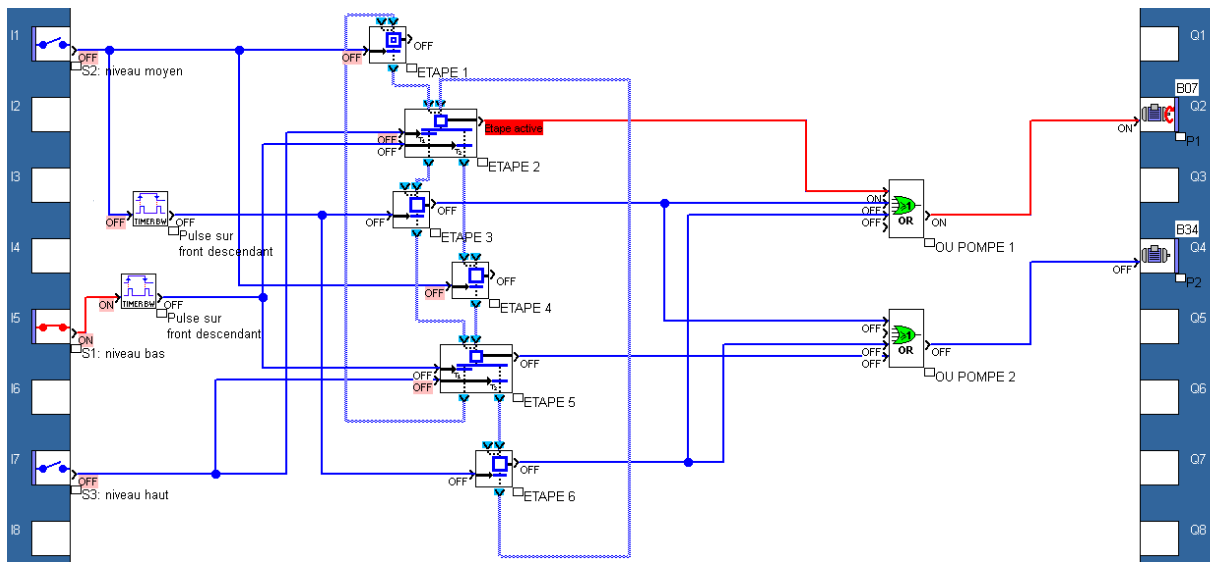
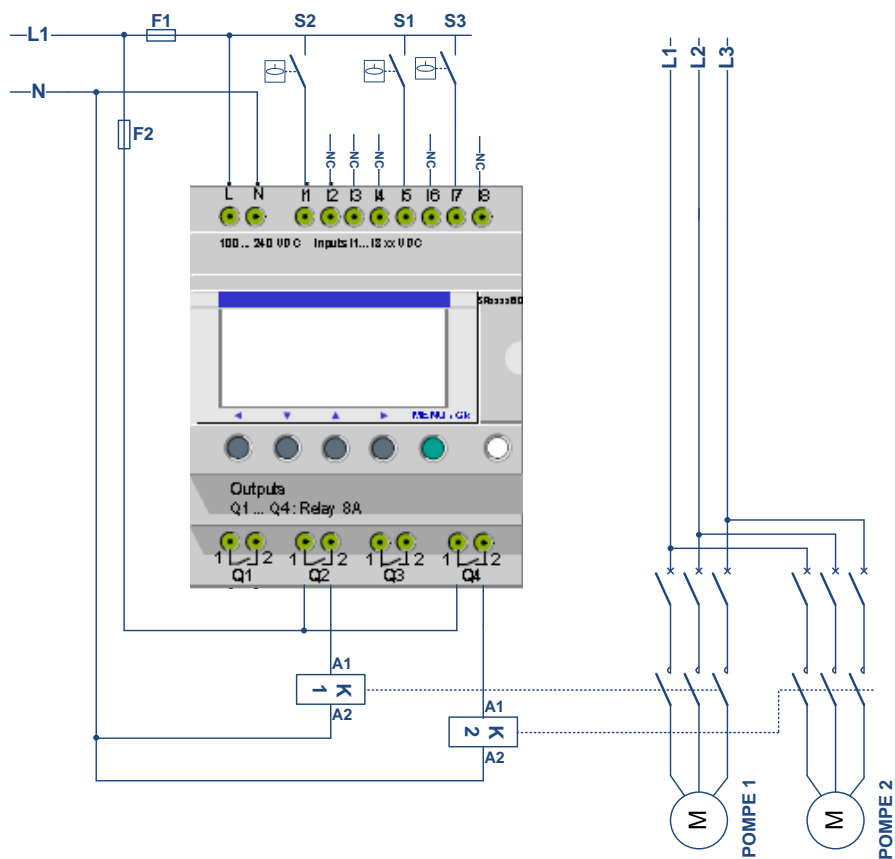


Schéma électrique simplifié de l'application :

Seuls les éléments qui entrent dans le fonctionnement de l'automatisme figurent dans le schéma.

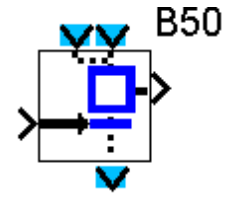


Quelques précisions sur le programme :

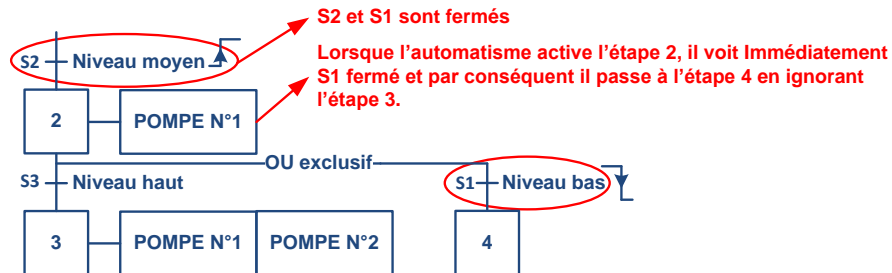
Détails graphiques des étapes	étape
-------------------------------	-------

Les étapes possèdent :

- 2 entrées hautes et une seule sortie basse pour effectuer les liaisons entre étapes.
- 1 entrée transition (gauche) pour raccorder les entrées.
- 1 sortie d'étape (droite) pour raccorder les actions : sorties.

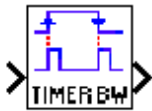


Une difficulté de cette étude est le point suivant :

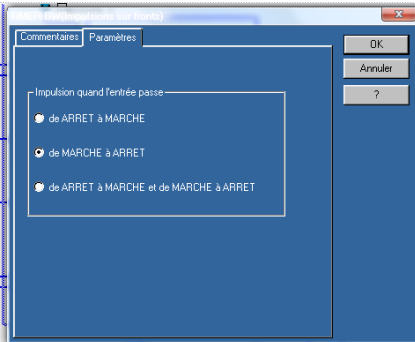


Pour contrer cette difficulté les développeurs ont trouvé une astuce de programmation qui permet de générer des « fronts », c'est-à-dire des impulsions dans des conditions spécifiques :

Dans le cas présent, **l'ouverture de S1 doit provoquer l'activation de l'étape 4 :**



Comme le symbole l'indique, ce module programmable est capable de générer une impulsion à la fermeture et/ou à l'ouverture du capteur qui est connecté sur son entrée : dans le cas qui nous intéresse, l'impulsion doit apparaître lors de l'ouverture de l'entrée.



Paramétrage de l'impulsion

Les portes logiques OU permettent l'alimentation des sorties en respectant les relations suivantes :

- **Pompe 1 = étape 2 + étape 3 + étape 6**
- **Pompe 2 = étape 3 + étape 5 + étape 6**

Précision sur le raccordement d'un relais logique ou d'un automate programmable :

Un automate ou un relais logique possède des entrées sur lesquelles sont raccordées les variables d'entrées : BP, inters, capteur T.O.R, capteur analogique, etc.

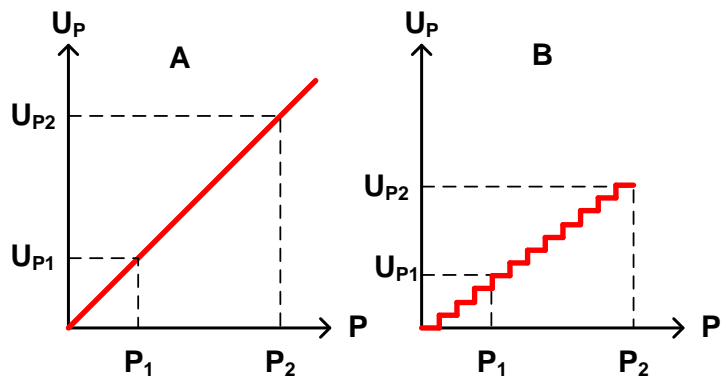
Le principe retenu pour le câblage des entrées est universel : il consiste à relier l'entrée à un potentiel pour lequel elle est prévue de fonctionner. L'appareil que nous avons choisi pour cette application possède des « entrées 230V » :

Lorsque l'entrée se trouve soumise à 230V, l'état logique correspondant est égal à 1.

Capteur T.O.R, analogique ou numérique

Il existe des **capteurs analogiques ou numériques**: ou l'information donnée à l'automate varie avec la grandeur physique que le capteur d'entrée est chargé de contrôler.

Exemple : un capteur de pression dont l'image **P** est renvoyée à l'automate sous forme d'une tension analogique (**A**) ou numérique (**B**).



Application de la mise en œuvre d'un capteur analogique sur la station de relavage:

Données constructeur Vega : <http://www.vega.com>

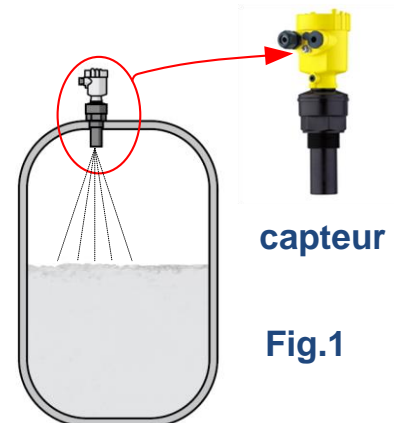
But :

Nous allons remplacer les détecteurs de niveau électromécanique par le détecteur de niveau par **ultrasons** : **VEGASON 62 (Fig.1)** tout en conservant le fonctionnement décrit précédemment ainsi que le relais logique ZELIO.

Principe :

Lors de la mesure par ultrasons sans contact, le capteur envoie des impulsions ultrasonores vers le produit pour être en suite réfléchies par la surface du produit.

La durée entre l'émission et la réception des signaux est proportionnelle au niveau dans la cuve.



Domaines d'application :

Le **VEGASON 62** convient à la mesure de niveau des liquides ou des produits en vrac dans des petites cuves. Les applications typiques sont la mesure des liquides dans des cuves de stockage, la mesure de débit sur canaux ouverts. La mesure de produits en vrac est également possible.

Avantages

Mesure de niveau continue sans contact.
Mesure indépendante des caractéristiques du produit.
Sonde de température intégrée pour la correction du temps de propagation.

Caractéristiques techniques du VEGASON 62

Plage de mesure liquides: 0,4 ... 8 m
solides en vrac: 0,4 ... 3,5 m
Raccord process : G2A en PVDF
Température process : - 40 ... +80 °C
Pression process : -0,2 ... +2 bar (-20 ... +200 kPa)
Sécurité fonctionnelle : jusqu'à SIL2

Repérage du bornier de raccordement du capteur

Caractéristiques électriques du capteur : [lien caractéristiques détaillées](#)

3

Alimentation tension

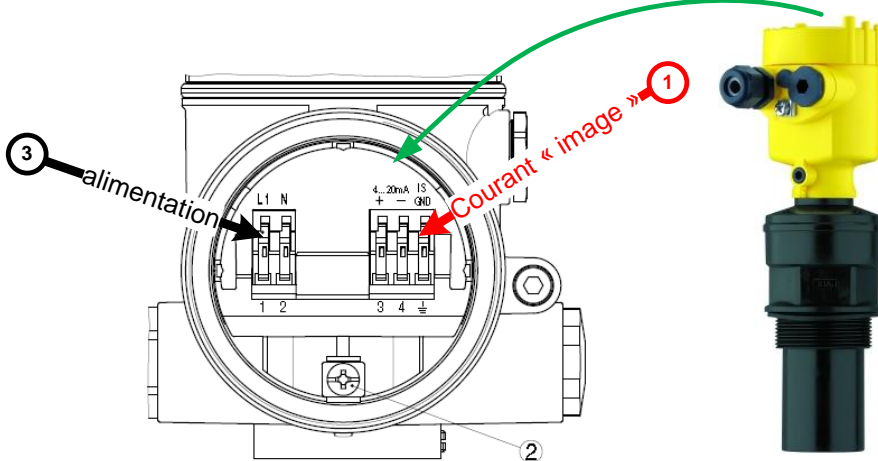
Tension de service

Appareil non Ex et appareil Ex-d

20 ... 72 V DC, 20 ... 253 V AC, 50/60 Hz

Consommation

maxi. 4 VA; 2,1 W



1

Grandeur de sortie

Signal de sortie

4 ... 20 mA/HART (actif)

Valeurs de sortie HART

Valeur HART (Primary Value)

Distance au niveau

Valeur HART (Secondary Value)

Température

Valeur HART (3rd Value)

Distance au niveau - calibrée

Résolution

1,6 μ A

Signal défaut sortie courant (réglable)

valeur mA inchangée, 20,5 mA, 22 mA, < 3,6 mA

Limitation de courant

22 mA

Charge

< 500 Ω ⁴⁾

Atténuation (63 % de la grandeur d'entrée)

0 ... 999 s, réglable

Recommandation NAMUR satisfaite

NE 43

Principe de fonctionnement du capteur

Le capteur convertit le niveau du liquide dans la cuve en un courant dont la valeur i évolue comme il suit :

Niveau atteint	niveau mini	niveau maxi	$N_{\min} < \text{niveau} < N_{\max}$
Courant de sortie du capteur	$i = 4\text{mA}$	$i = 20\text{mA}$	$4\text{mA} < i < 20\text{mA}$

Caractéristiques des entrées analogiques du relais logique ZELIO

Note : le capteur étudié utilise aussi la technologie **HART** qui est abordée en fin du cours.

Caractéristiques des entrées analogiques (entrées IH, IJ et Pt)				
Entrées analogiques	Utilisation	0-10 V	0-20mA	Pt100
	Entrées affectables	IH et IJ	IH et IJ	IJ
	Gamme d'entrée	0...10 Vdc	0...20 mA	-25 °C...125 °C
	Impédance d'entrée	1 K	246	-
	Valeur maximale sans destruction	30 V	30 mA	-
	Valeur du LSB	9,8 mV	20 µA	0,15 °C
	Type d'entrée	Mode commun		Sonde pt100 - IEC 751 3 fils
Conversion	Résolution	10 bits		
	Temps de conversion	Temps de cycle module		
	Précision	à 25 °C	± 1 %	± 1,5 °C
		à 55 °C	± 1 %	± 1,5 °C
	Répétabilité	à 25 °C	< ± 1 %	< ± 0,3 °C
isolement	Voie analogique et alimentation	Aucun		
Distance de câblage		m	10 maximum, avec câble blindé (capteur non isolé)	
Protection	Contre les inversions des bornes		Pas de prise en compte de la commande	

Câblage des entrées analogiques (document constructeur)

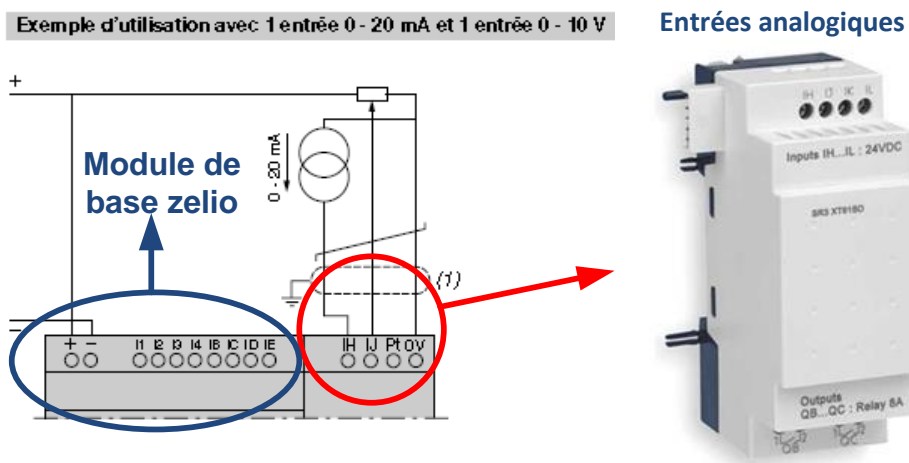
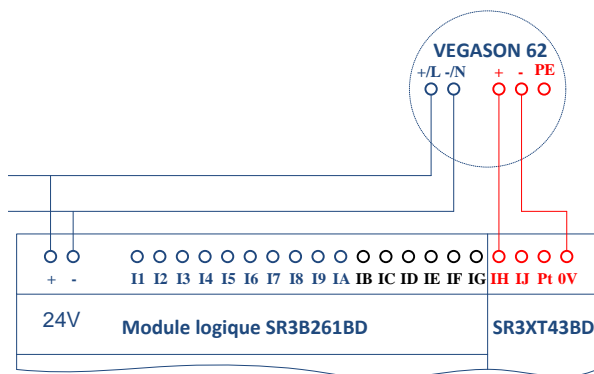


Schéma de câblage de l'entrée analogique



- I1 à IA** : entrées T.O.R
- IB à IG** : entrées T.O.R et analogique 0 – 10V
- IH et IJ** : entrées analogique 0 -10V et 0 -20mA
- Pt** : entrée sonde Pt100
- 0V** : commun entrées analogiques

Note : Les entrées T.O.R mises au +24V verront un 1 logique (voir doc constructeur).

Les appareils de protection ne sont pas représentés.

Le schéma de l'alimentation des contacteurs d'alimentation des pompes reste identique :

- Q9 alimente P1
- QA alimente P2

Programme de l'application

Marc Sanchez

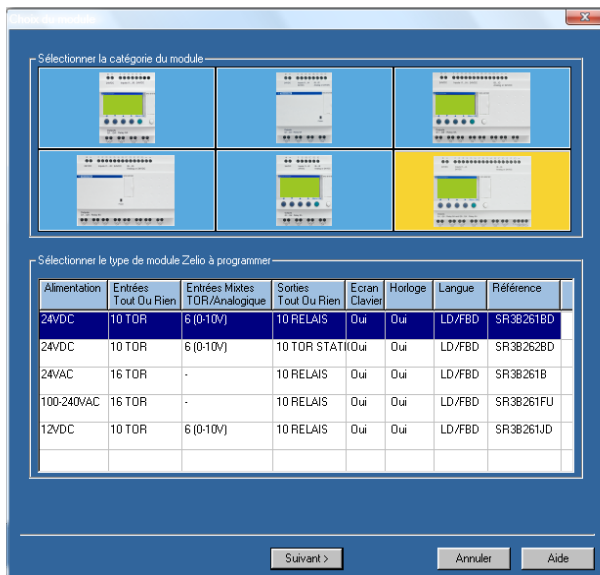
<http://electrotechnique.fr>

Précisions concernant les choix technologiques: Le programme décrit ci-dessous est une solution possible au problème posé. Ainsi, le **grafcet** fonctionnel étudié précédemment est conservé pour éviter d'ajouter des difficultés supplémentaires inutiles, afin de se focaliser uniquement sur le traitement de la **boucle de courant**.

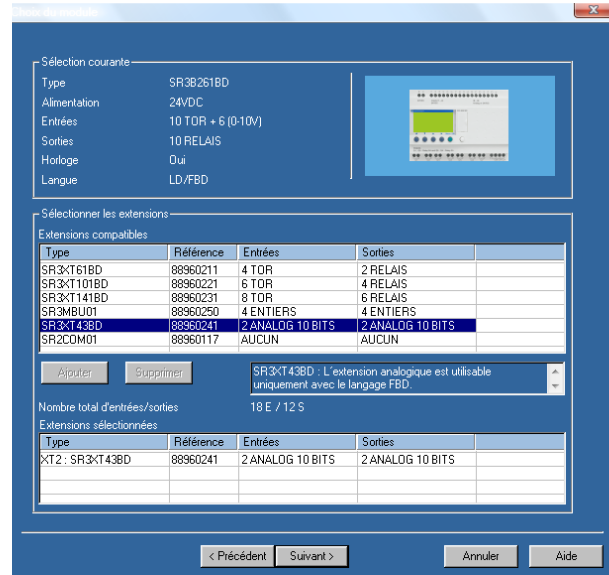
Les possibilités offertes par les capteurs analogiques (ou numériques), couplés aux calculateurs que sont les automates ou relais logiques, permettent d'effectuer des réglages d'une grande finesse des équipements automatisés : régulation de température, de débit, ou asservissement de vitesse...etc. Dans le cas présent, l'intérêt du capteur analogique réside surtout dans sa simplicité de mise en œuvre par rapport aux capteurs électromécaniques. D'autre part, rappelons que le relais logique est choisi ici pour l'approche très pédagogique qu'il offre à l'apprentissage des langages d'automatisme. Car son faible nombre d'entrées ne lui permet pas de gérer un système automatique complet.

Le module d'entrées analogiques ne peut être associé qu'à un module de base fonctionnant sous une tension **24V DC**.

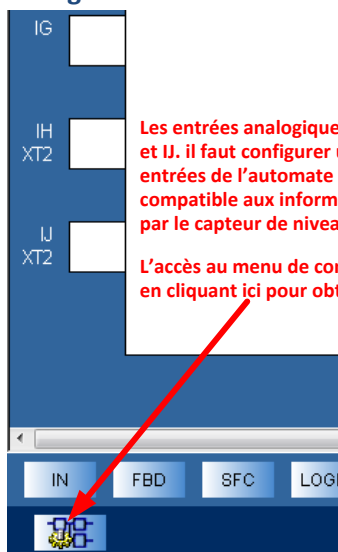
Choix du module de base



Choix du module d'entrées analogiques

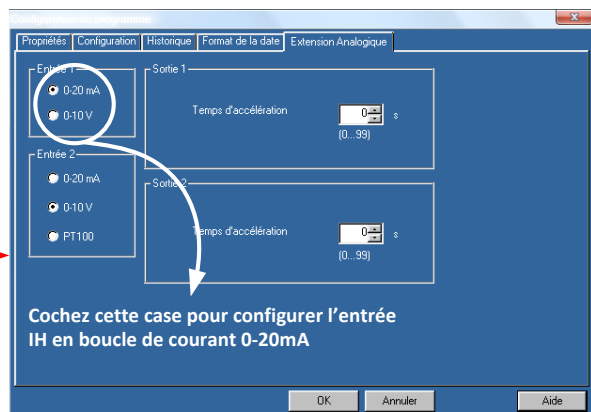


Configuration des entrées analogiques

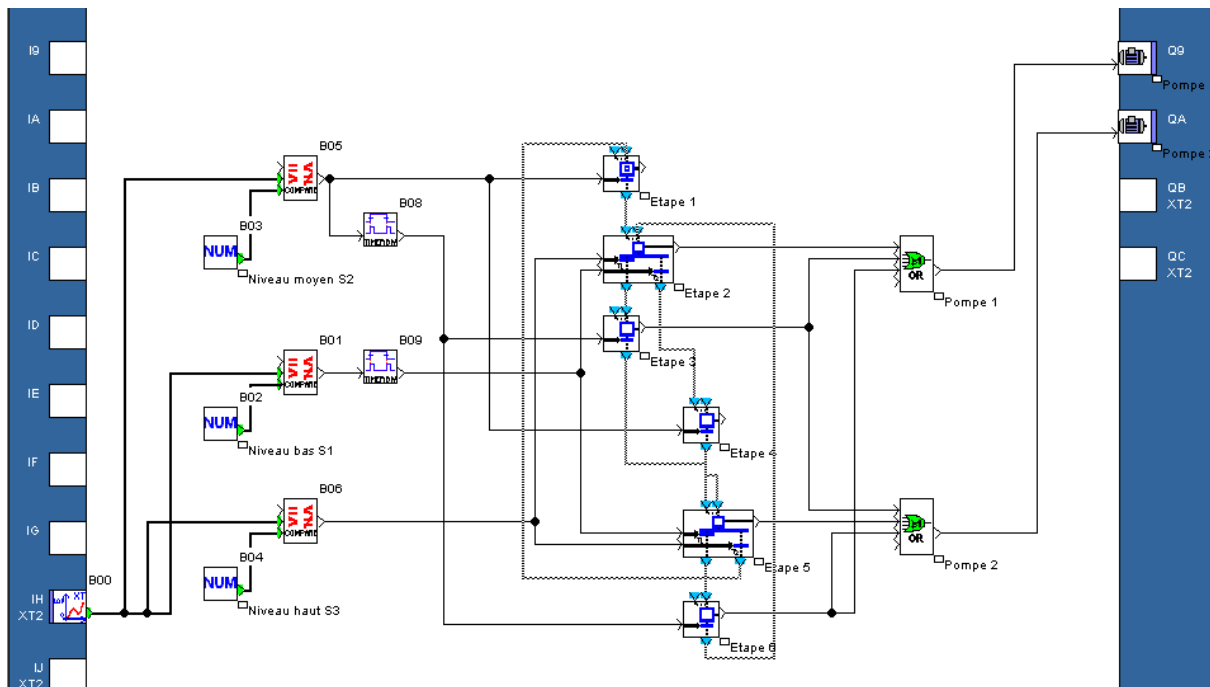


Les entrées analogiques sont repérées IH et IJ. Il faut configurer une des deux entrées de l'automate pour la rendre compatible aux informations transmises par le capteur de niveau.

L'accès au menu de configuration se fait en cliquant ici pour obtenir l'écran suivant



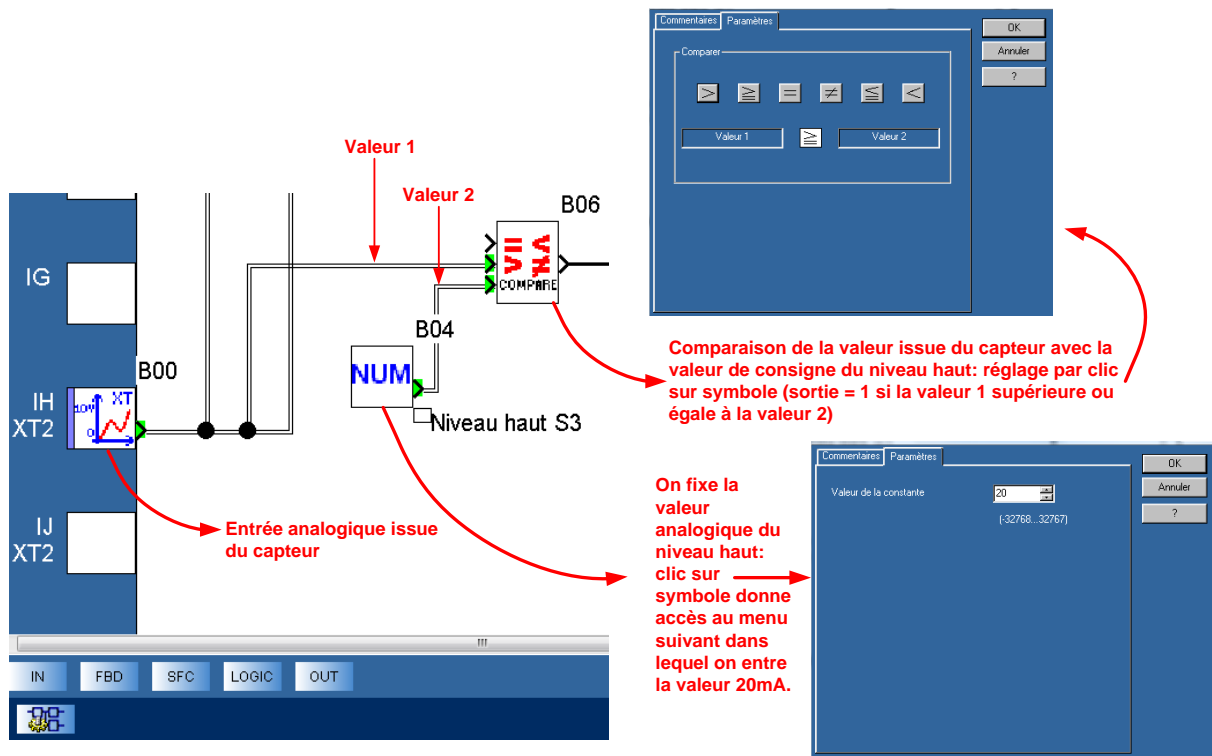
Désormais, un seul capteur est nécessaire pour faire fonctionner le cycle de pompage. La programmation de l'application utilise la même structure de grafset étudiée précédemment.



Traitement de la boucle 0-20mA

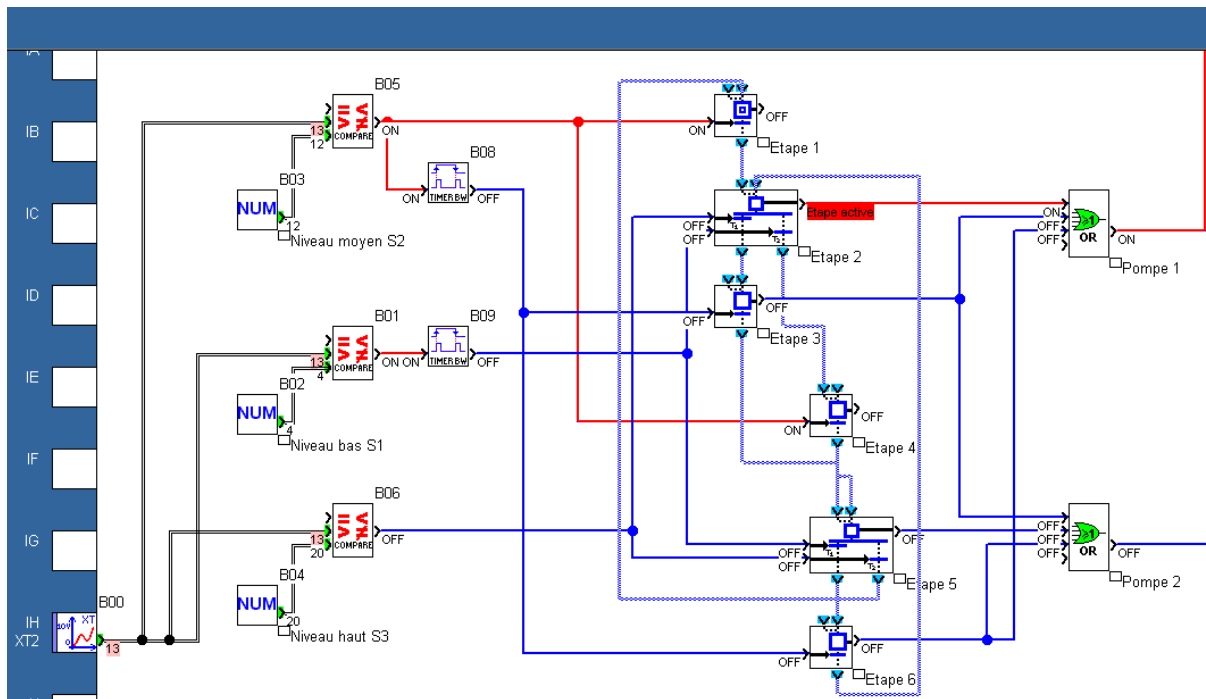
Explication de la fonction des blocs **B00**, **B04** et **B06** qui effectuent le traitement du niveau haut. Les autres niveaux reprennent la même structure.

- Réglage du niveau bas : NUM (bas) = 4
- Réglage du niveau moyen : NUM (moyen) = 12
- Réglage du niveau haut : NUM (haut) = 20

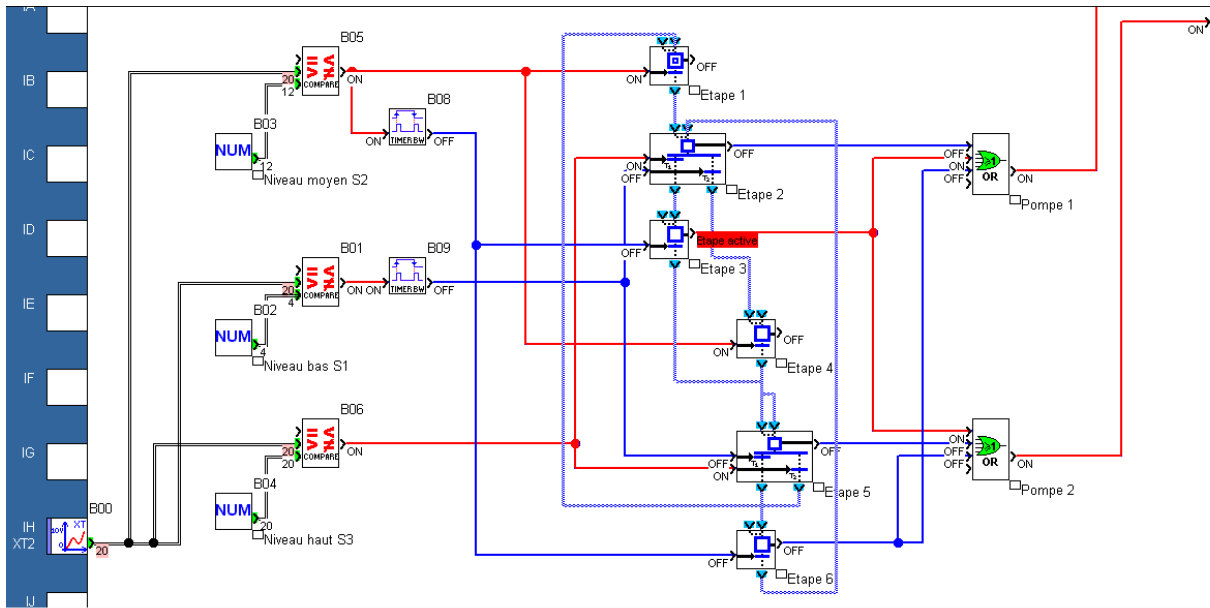


Visualisation du fonctionnement

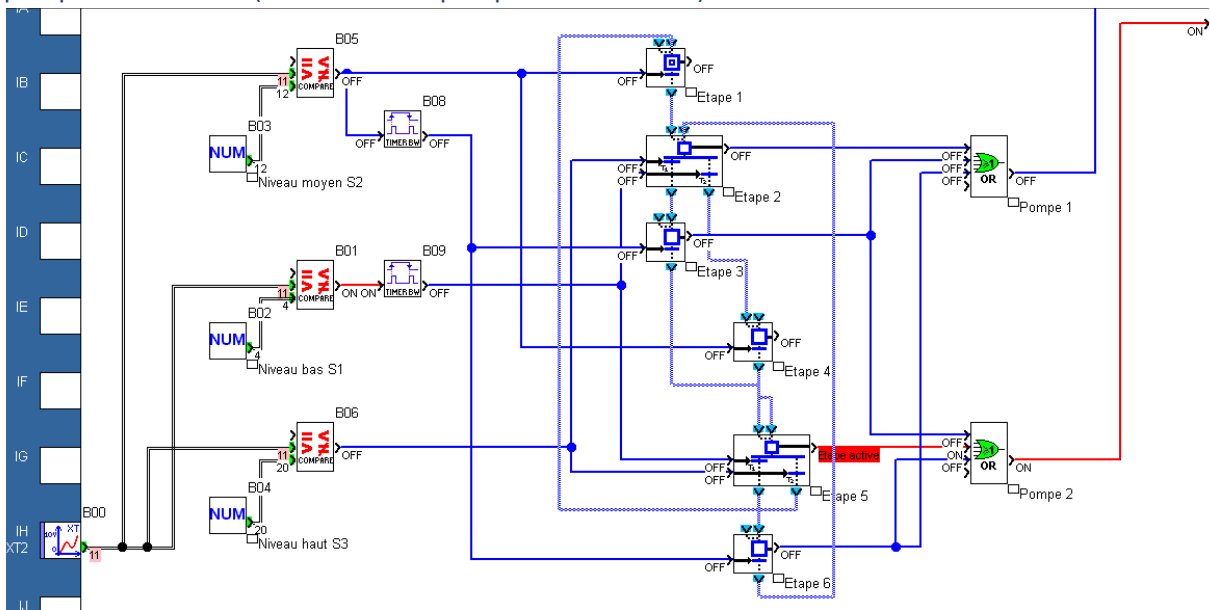
Le niveau moyen est dépassé : le capteur renvoie la valeur courante 13 (en B00) qui provoque le basculement des comparateurs B01 et B05. L'étape 2 est active et lance le fonctionnement de la pompe P1.



L'eau atteint le niveau : la valeur 20 est renvoyée au relais logique et provoque le basculement de B06. L'activation de l'étape 3 provoque le fonctionnement des deux pompes.

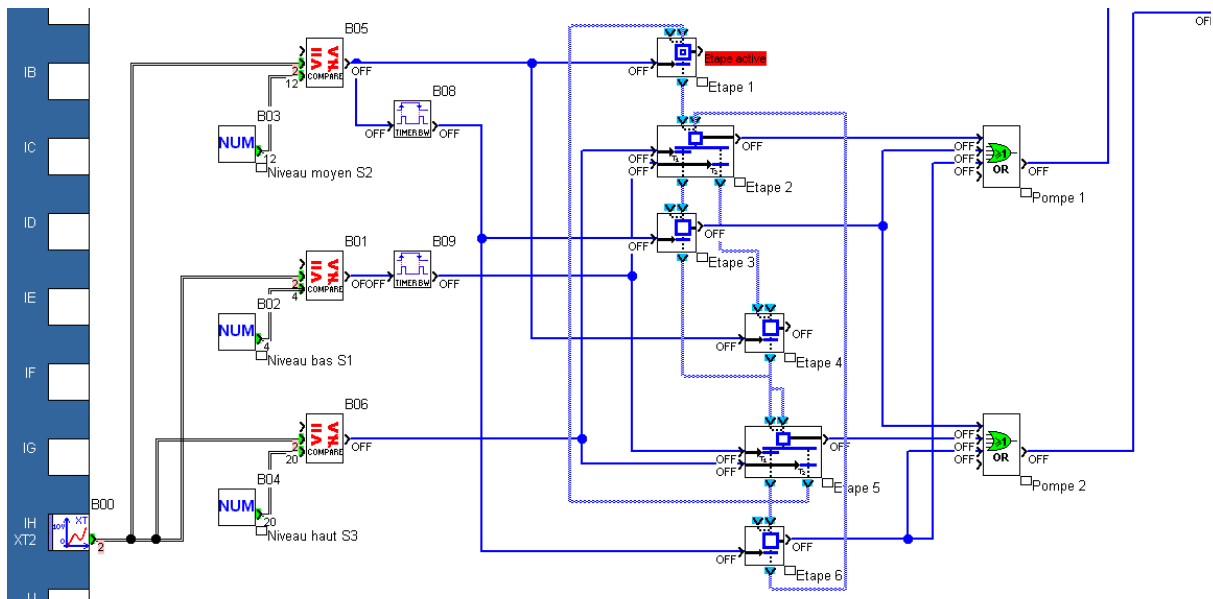


Le niveau de l'eau passe sous le niveau moyen (B00 = 11): passage de l'étape 3 à l'étape 5, seule la pompe 2 fonctionne (l'inversion des pompes fonctionnent).



Lorsque le niveau descend en dessous du niveau mini (B00 = 2) l'automatisme se retrouve à l'étape initiale : les deux pompes sont à l'arrêt.

Note : La création du front montant est réalisée en suivant la même méthode étudiée précédemment : on crée un front montant sur « ouverture » du capteur.

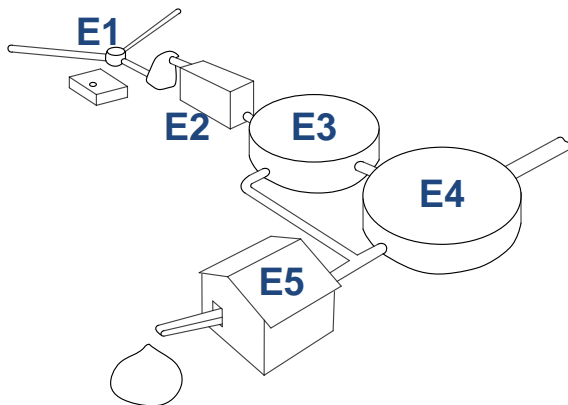


Conclusion

L'emploi d'un capteur analogique élimine les interventions de maintenance (réglage, nettoyage, remplacement...etc.) qu'on devait effectuer sur les capteurs de niveau électromécaniques situés dans le bassin. La partie « mécanique » de la maintenance s'est simplifiée au détriment de la maintenance « électrique ».

Précision concernant les capteurs numériques (à faire)

Nécessité de mettre en œuvre des « bus » de terrain



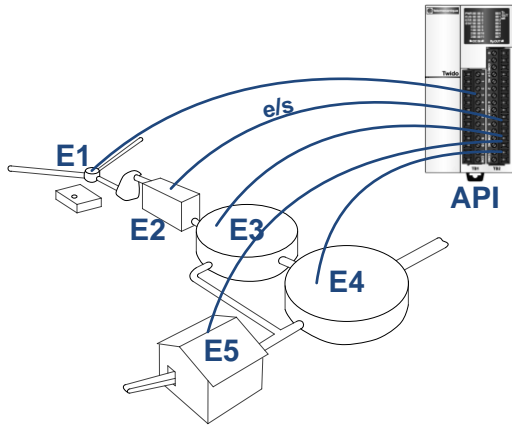
Reprenons la station d'épuration pour illustrer un problème majeur que l'on rencontre dans les installations actuelles.

Une installation ci-contre est constituée de plusieurs entités « E » : poste de relevage, dessablage, ...etc. Il est utile de faire communiquer ces entités entre elles, ou bien de centraliser les données (niveau, débit, ...etc.) afin de régler, mesurer, commander, analyser, surveiller, et, le cas échéant d'intervenir en cas de besoin sur l'une d'elles.

Pour rendre une installation **communicante**, il faut utiliser un « bus de terrain » dont quelques uns sont listés ci-dessous:

- Profibus
- ASI
- CANopen
- Modbus
- KNX
- Etc.

Gestion centralisée



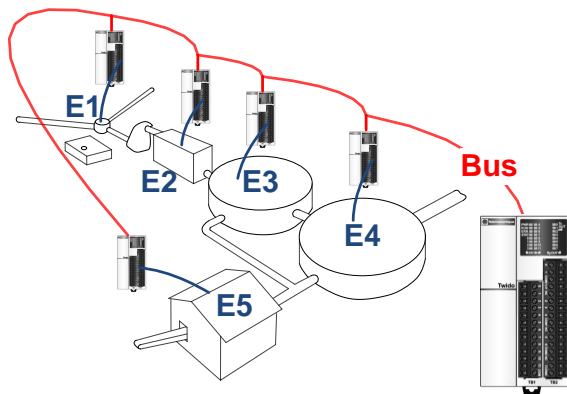
La gestion des entités est :

soit centralisée : un automate gère les entrées et sorties de l'ensemble du site.

Problèmes:

- l'installation complexe à mettre en œuvre.
- Installation peu évolutive.
- Un dysfonctionnement de l'API (automate programmable industriel) altère toute l'installation.
- Ce type d'installation obsolète est à réserver à des petites unités de production.

Gestion décentralisée



Soit décentralisée : chaque entité est gérée par un automate mis en réseau avec tous les autres à travers **un bus de terrain**. La fragmentation de l'installation en sous-ensembles favorise leur exploitation et leur évolutivité. D'autre part, le bus numérique permet de suivre l'état des processus en cours, augmentant la réactivité des équipes de maintenance en cas de nécessité. Ici aussi, l'utilisation de bus de terrain simplifie la mise en œuvre de l'installation au détriment de son exploitation qui demande une main d'œuvre qualifiée.

Voici ci-dessous une illustration du bus AS-I (Actuators Sensors Interface) où un câble de couleur jaune véhicule simultanément l'énergie et les informations. On remarque immédiatement l'intérêt du Bus dans la diminution du volume de conducteur utilisé dans l'installation.