LES AUTOMATISMES

LE TRAITEMENT NUMERIQUE

GJC

Lycée L.RASCOL 10, Rue de la République
BP 218. 81012 ALBI CEDEX

SOMMAIRE

DONNEE NUMERIQUE

L'automate programmable possède, non seulement la possibilité de travail sur des bits, mais aussi sur des données numériques de type entiers et/ou réels à l'aide de variables mono éléments (%W, %D, %L) ou de tableaux.

1- Représentation binaire d'une donnée numérique

Le format de représentation binaire choisi est sur 16 bits (le plus courant sur les API) soit des puissance de 2 de 2^0 à 2^{15}

Octet de poids FORT								Octet de poids FAIBLE							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

2- Représentation des nombres signés

Les API de dernières générations savent travailler sur des nombres signés. La **valeur négative** s'obtient par le **complément a 2 de la valeur positive.**

Complément à 2 = complément a 1 + 1

Le complément à 2 est aussi appelé complément VRAI. Le complément à 1 ou complément restreint est le complément bit a bit.

Représentation valeur positive **629**

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

Représentation valeur négative **- 629**

- Valeur positive **629**

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

- Complément à 1 (complément bit à bit)

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

- Complément à 2 (complément à 1 + 1)

- 629

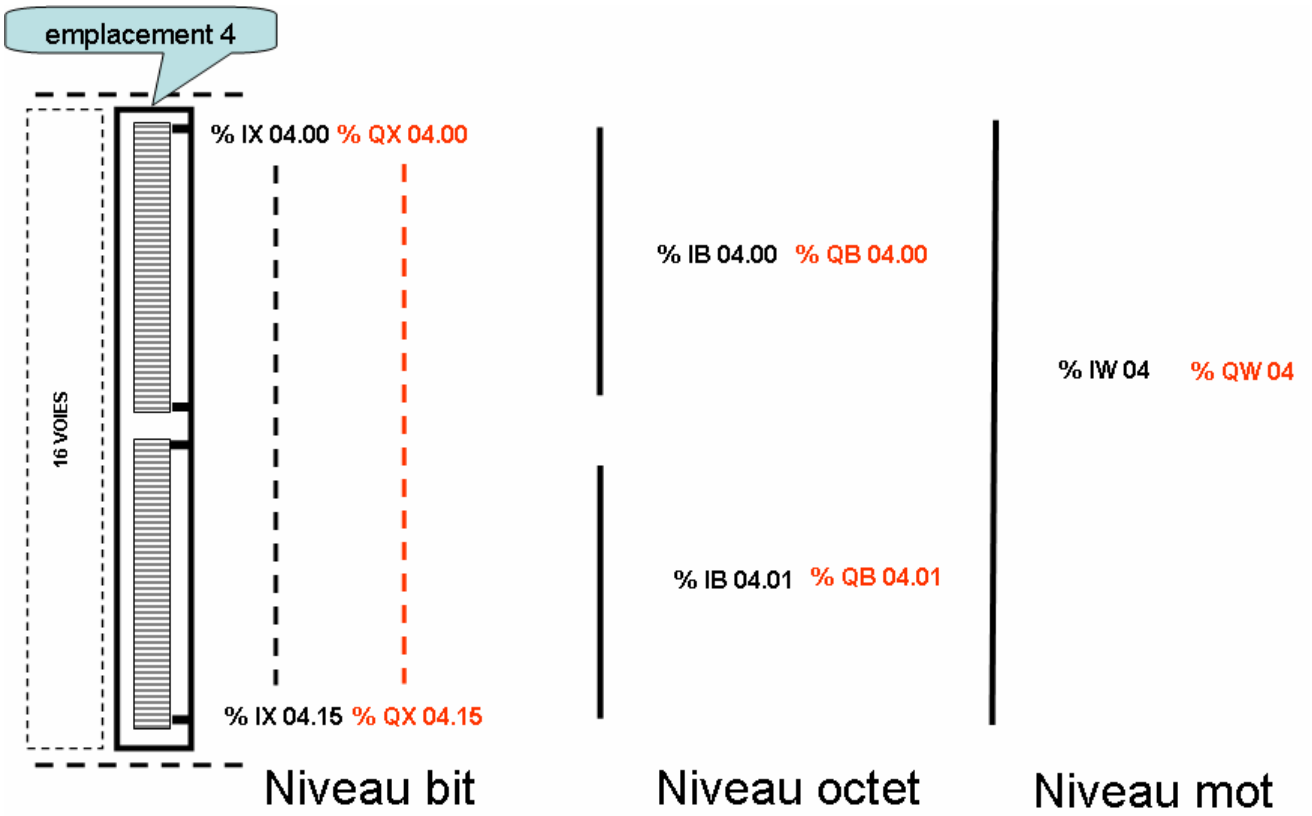
1	1	1	1	1	1	0	1	1	0	0	0	1	0	1	1

3- Acquisition ou affectation de valeurs numériques

Les automates programmables ne possèdent pas de coupleurs spécifiques pour le traitement numérique. La mémoire de donnée est caractérisée par des mots de 16 bits, que l'on peut utiliser sous forme éclatée (entrées/sorties TOR) ou sous forme groupée (byte ou mot).

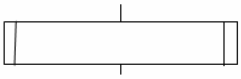
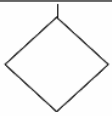
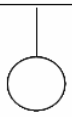
Exemple d'adressage : coupleur d'entrée ou de sortie a 16 voies

@ est caractérisée par : N° d'emplacement dans le rack 04
N° de voie d'E ou S 15



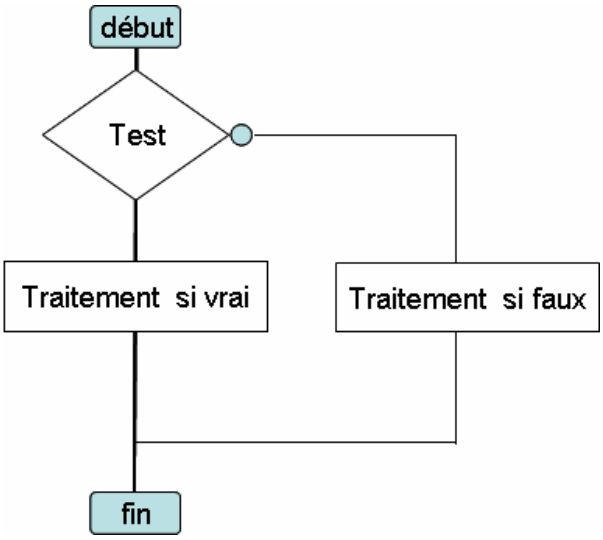
REPRESENTATION D'UN CALCUL NUMERIQUE

La représentation de la suite logique d'opérations nécessaires à la résolution d'un problème a base de données numériques se fait au moyen d'organigramme de programmation. La normalisation (norme NF Z 67-010) établit une série de symboles graphiques à utiliser dans les organigrammes de traitement de données.
Extrait de la norme.

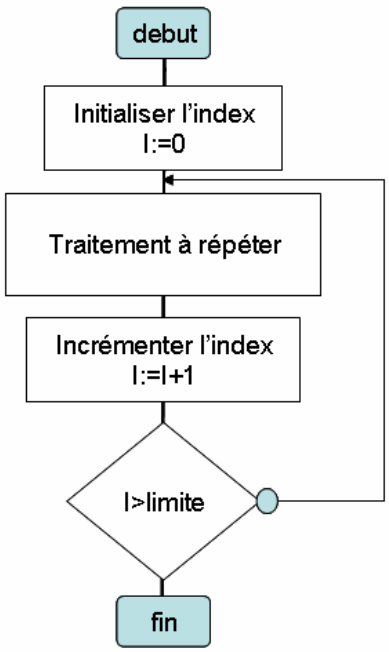
	**Début, fin** Début, fin ou interruption d'un organigramme, point de contrôle, etc
	**Symbole général de « traitement »** Opération ou groupe d'opérations portant sur des données.
	**Sous-programme** Portion de programme considérée comme une opération
	**Entrée-sortie** Entrée d'une information à traiter ou affectation d'une information.
	**Embranchement, test** Exploitation de variables impliquant un choix
	**Renvoi** Symbole utilisé pour assurer la continuité.

Les structures d'organigrammes les plus courantes sont :

Structure ALTERNATIVE



Structure REPETITIVE ou boucle contrôlée



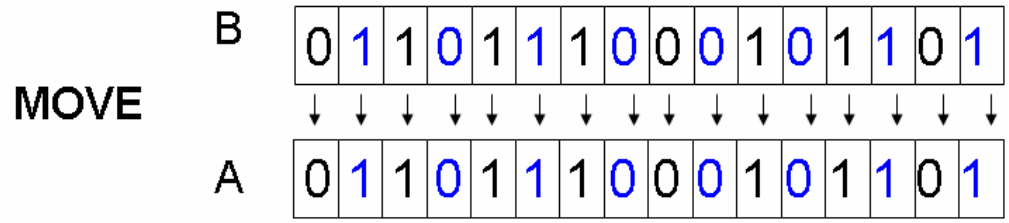
LES FONCTIONS

La norme des langages de programmation des API (NF 61131-3) défini les principales fonctions numériques utilisables dans le traitement de données (voir annexe page16).

a. La fonction de transfert MOVE

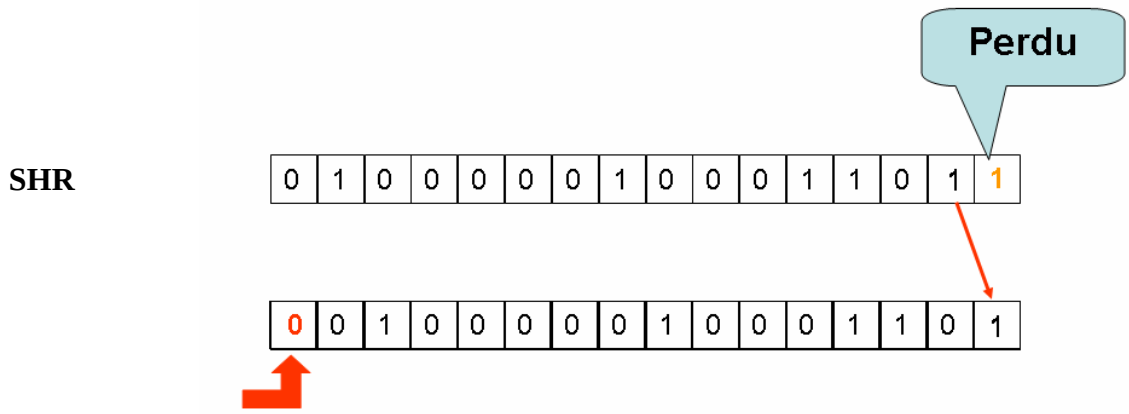
Cette fonction permet le transfert de valeurs d'une variable dans une autre, sans perdre la valeur de la variable initiale.
Exemple :

A := B ; transfère le contenu de la variable B dans la variable A sans perdre la valeur de B.

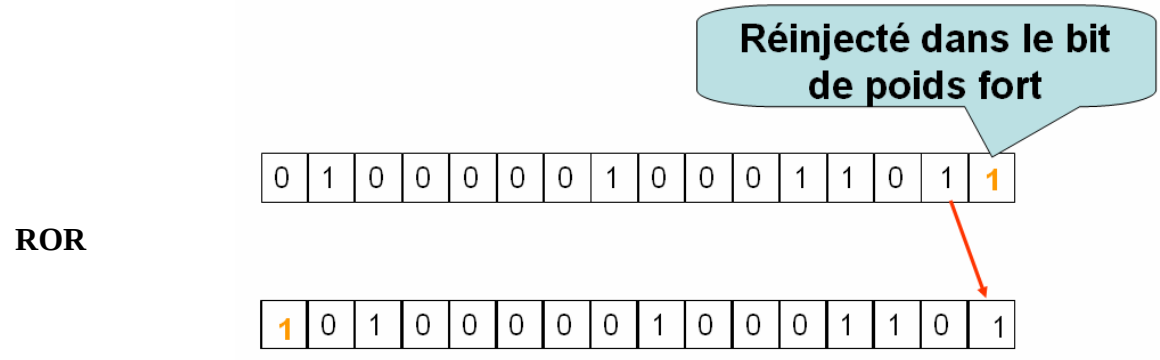


b. les fonctions de décalage

Décalage logique à droite ou à gauche

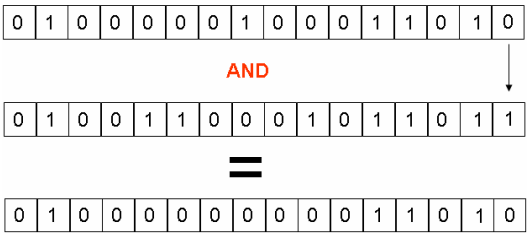


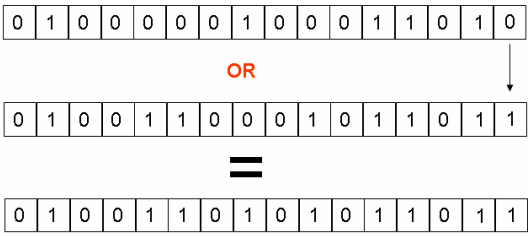
Décalage circulairee à droite ou à gauche



c. les fonctions logiques

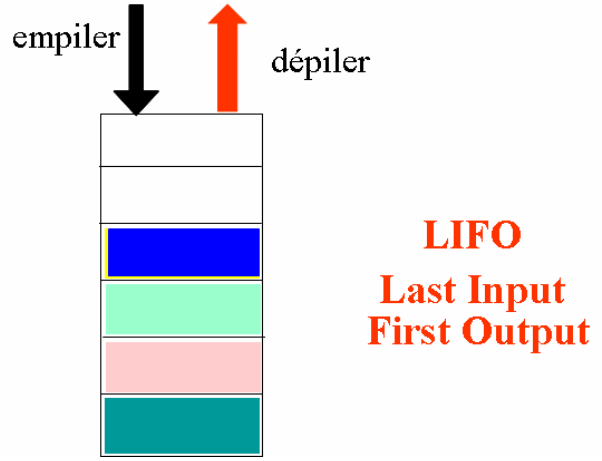
Les fonction logiques standards **ET** et **OU** sont applicables au données numériques.

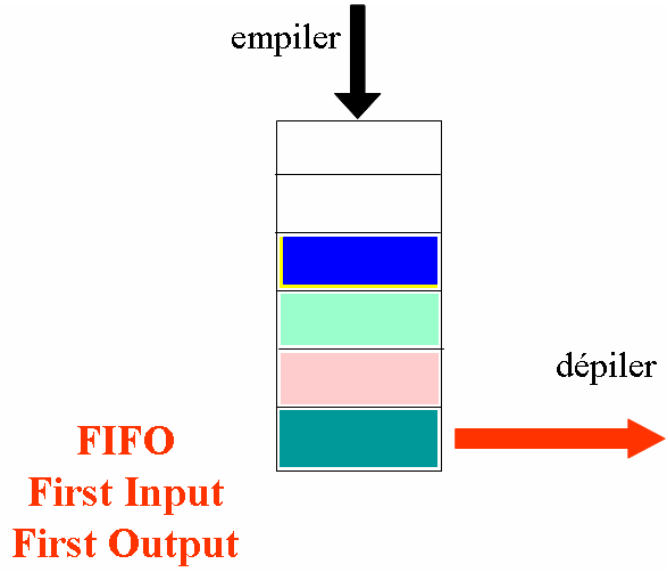




d. Fonctions complémentaires liées au registre d'instruction des API

- Fonctions de gestion de piles types FIFO ou LIFO.



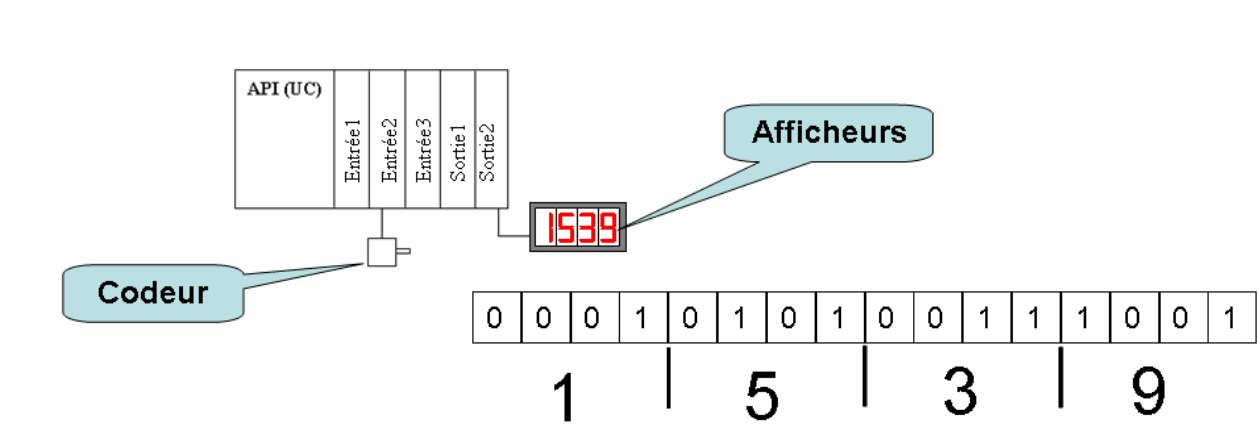


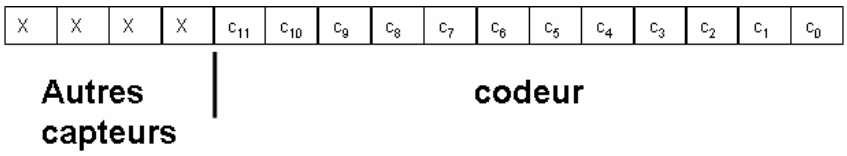
TRAITEMENT D'ENTREES ET DE SORTIES NUMERIQUES

Le traitement d'entrées et de sorties numériques, consiste essentiellement a acquérir ou à affecter des valeurs a partir de coupleurs d'entrées sorties de type tout ou rien.

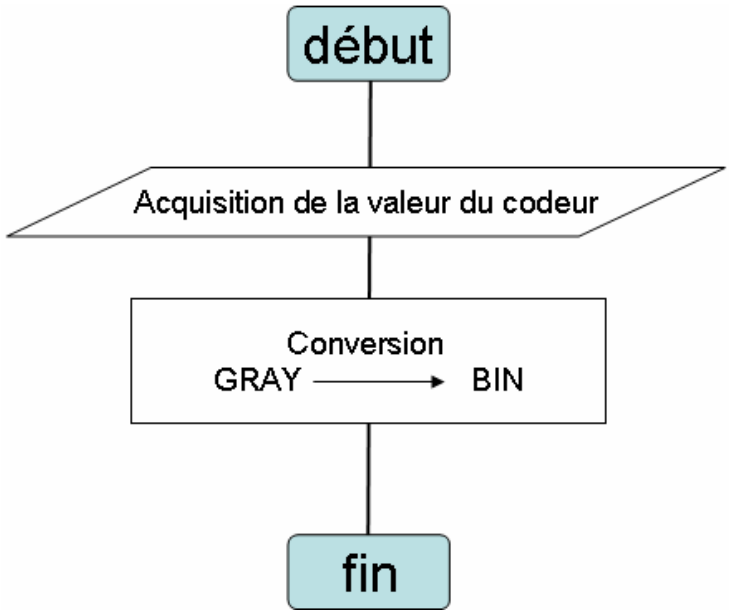
Exemple :

Le positionnement d'un mobile est réalisé à l'aide d'un codeur absolu (codage GRAY sur 12 bits) relié aux voies 0 à 11 d'une carte d'entrée tout ou rien de l'automate.
Les autres entrées (voies 12 à15) sont utilisées pour d'autres capteurs.
Le traitement de l'information par l'automate se fait en binaire.
La visualisation de la position du mobile se fait sur des afficheurs 7 segments (codage BCD) reliés à une carte de sortie tout ou rien.





ACQUISITION D'UNE ENTREE NUMERIQUE **Mise en place de la fonction « ACQ_NUM »**

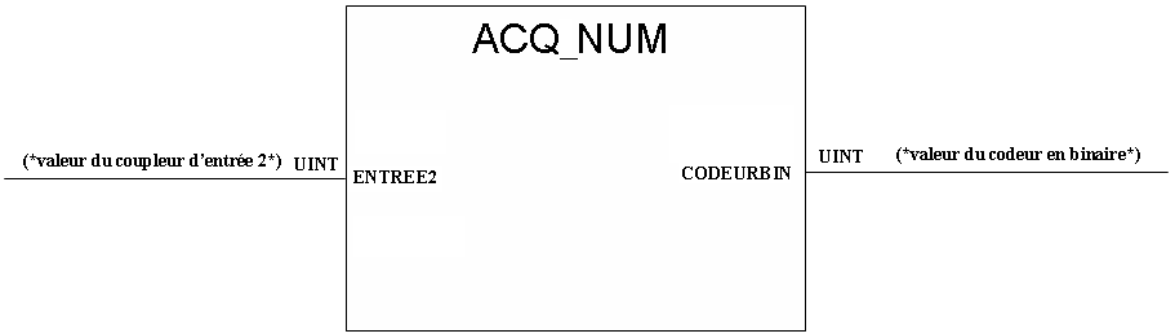


1) Présentation en langage ST (Littéral Structuré)

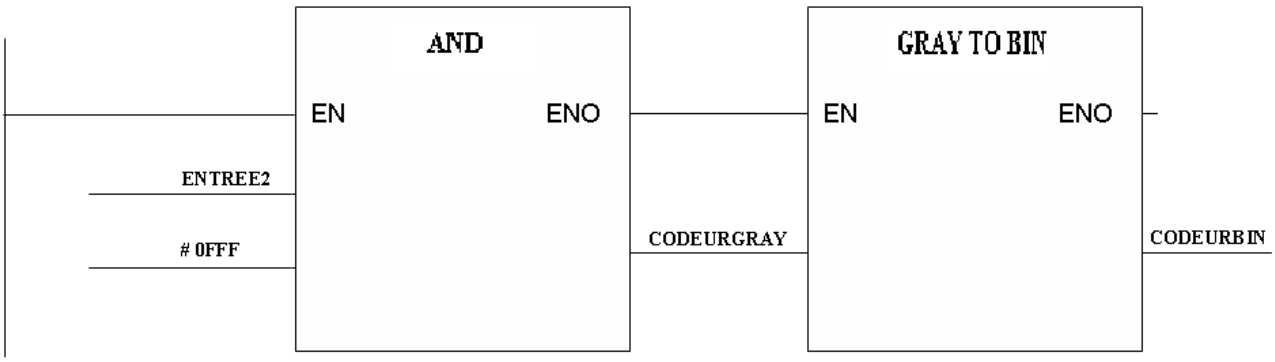
```
FONCTION ACQ_NUM

(*déclaration*)

VAR_INPUT
        ENTREE2 :               UINT ;   (*valeur du coupleur d'entrée 2*)
END_VAR

VAR
        CODEURGRAY              UINT ;  (*valeur du codeur en GRAY*)
END_VAR

VAR_OUTPUT
        CODEURBIN :             UINT ;  (*valeur du codeur en binaire*)
END_VAR


(*corps de la fonction*)

CODEURGRAY := ENTREE2  AND  # 0FFF ;
CODEURBIN :=  GRAY TO BIN CODEURGRAY ;


END_FONCTION
```

2) Présentation en langage LD (Ladder Diagram)

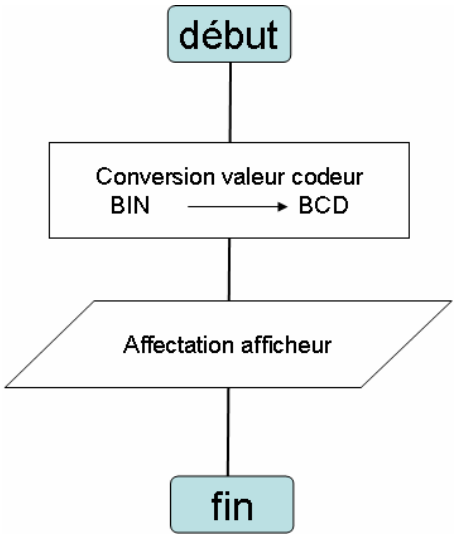
(*déclaration*)



(*corps de la fonction*)



AFFECTATION D'UNE SORTIE NUMERIQUE **Mise en place de la fonction « SOR_NUM »**

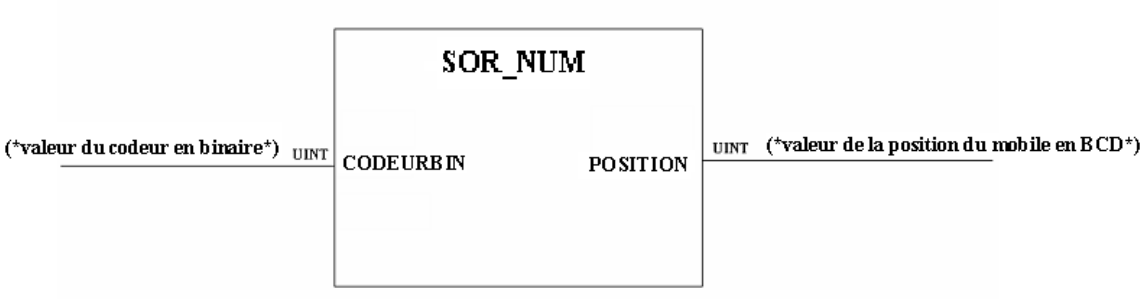


1) Présentation en langage ST (Littéral Structuré)

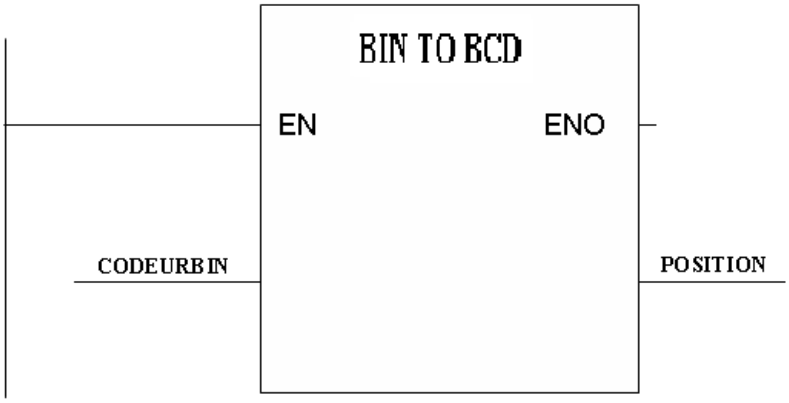
```
FONCTION SOR_NUM

(*déclaration*)

VAR_INPUT
        CODEURBIN :       UINT ;  (*valeur du codeur en binaire*)
END_VAR

VAR_OUTPUT
        POSITION :        UINT ;  (*valeur de la position du mobile en BCD*)
END_VAR



(*corps de la fonction*)

POSITION :=  BIN TO BCD CODEURBIN;



END_FONCTION
```

2) Présentation en langage LD (Ladder Diagram)

(*déclaration*)



(*corps de la fonction*)



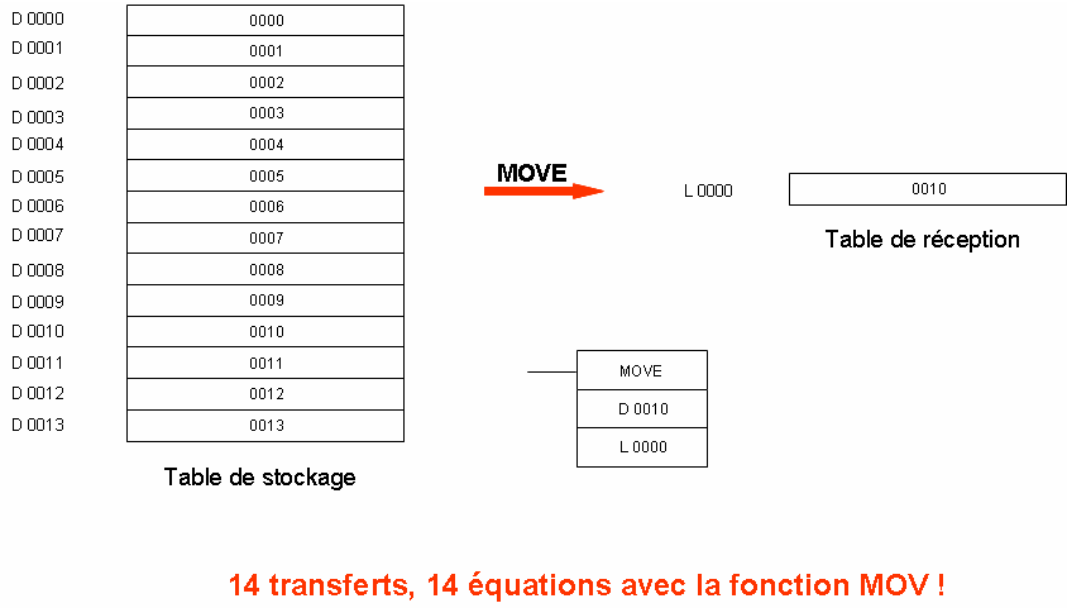
TRAVAIL SUR RECETTES

Cette technique de programmation utilise les possibilités de l'adressage indirect des API. Ceci permet de travailler avec des tables de mots pour de chargement de consignes.

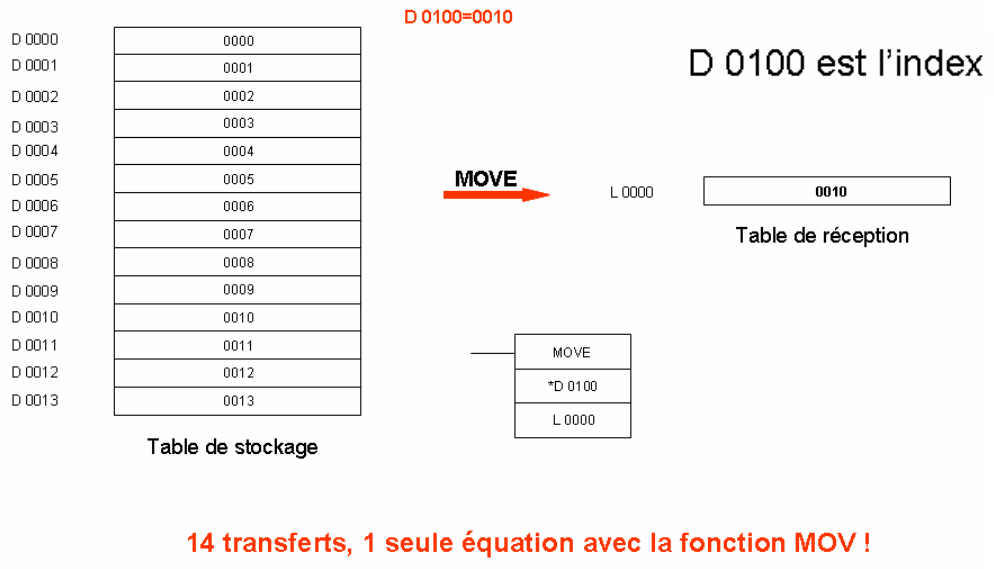
L'index est un mot entier toujours positif.

valeur finale = valeur du mot dont l'adresse est dans l'index

Cette technique permet d'extraire certaines valeurs d'une table de mots, ou la recherche d'une valeur parmi n.

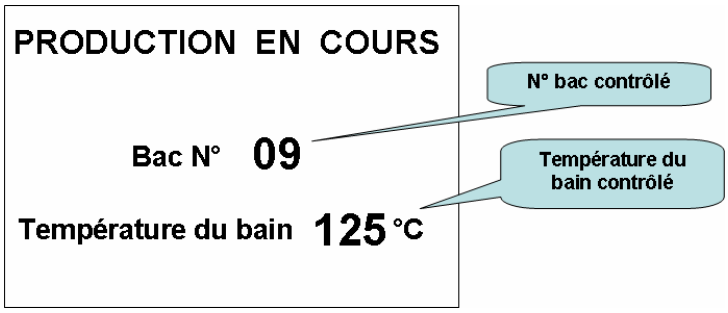


Travail en mode indexé

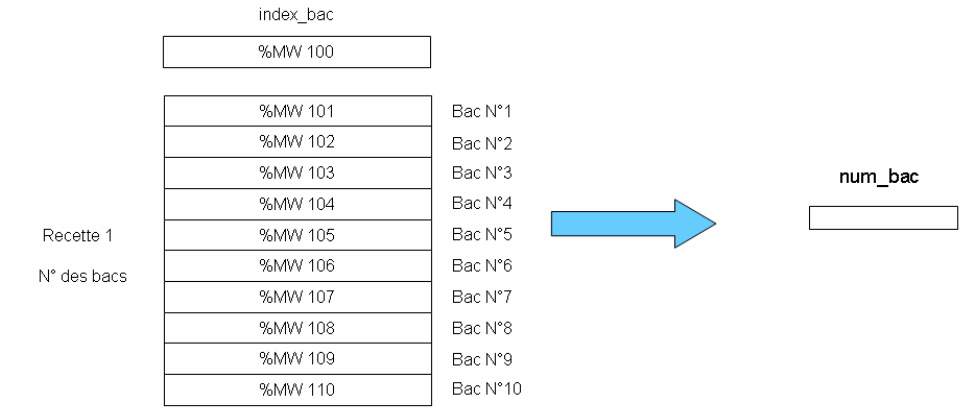


Exemple

On veut, dans une usine de traitement de surface afficher sur un seul écran du TDI (l'écran N°11) de manière successive (toutes les 10s) la température des10 bacs de traitement.

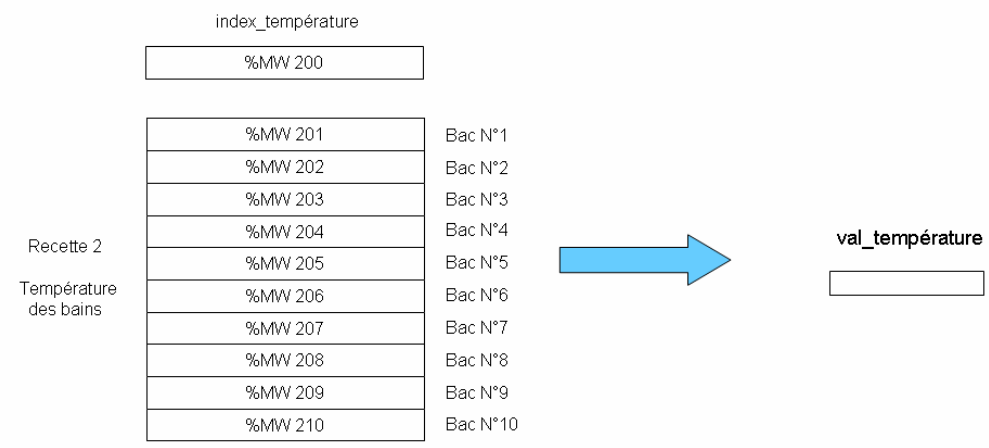


num_écran : UINT; (*numéro de l'écran en cours*)
num_bac : UINT, (*numéro du bac contrôlé*)
val_température : UINT; (*température du bain contrôlé*)



Sélection du bac et envoi de son N° vers la variable « num_bac » pour affichage sur le TDI.

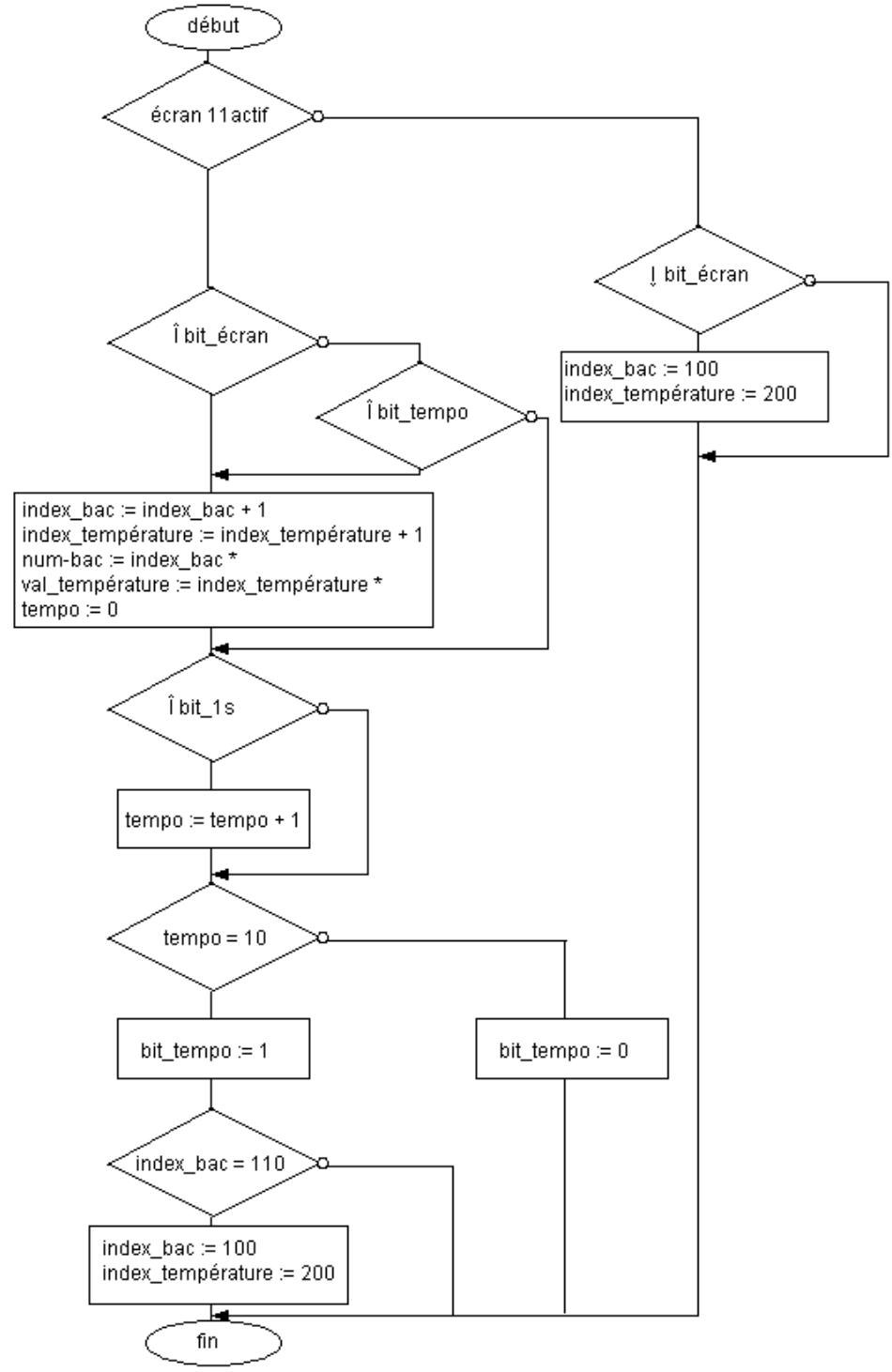
« utilisation de index_bac »



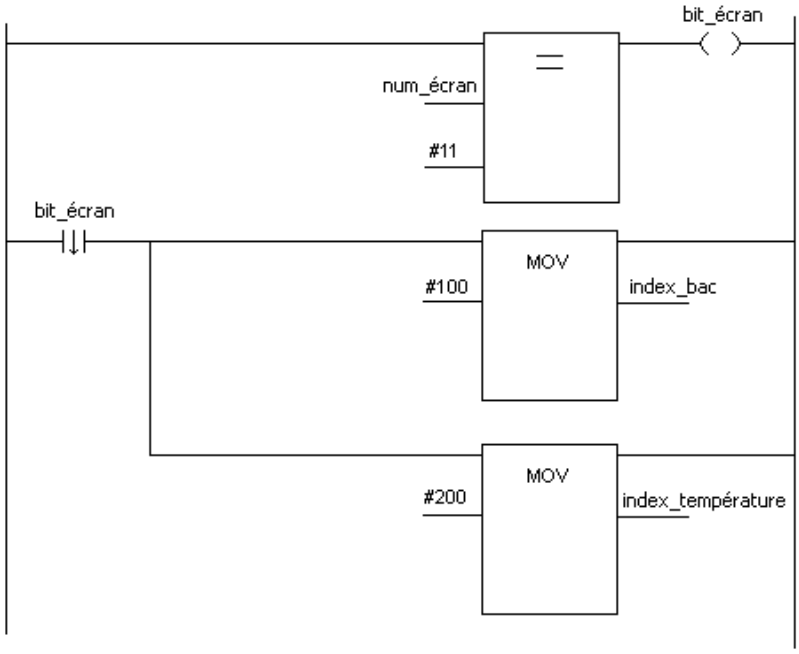
Sélection du bac et envoi de la température de son bain vers la variable « val_température » pour affichage sur le TDI.

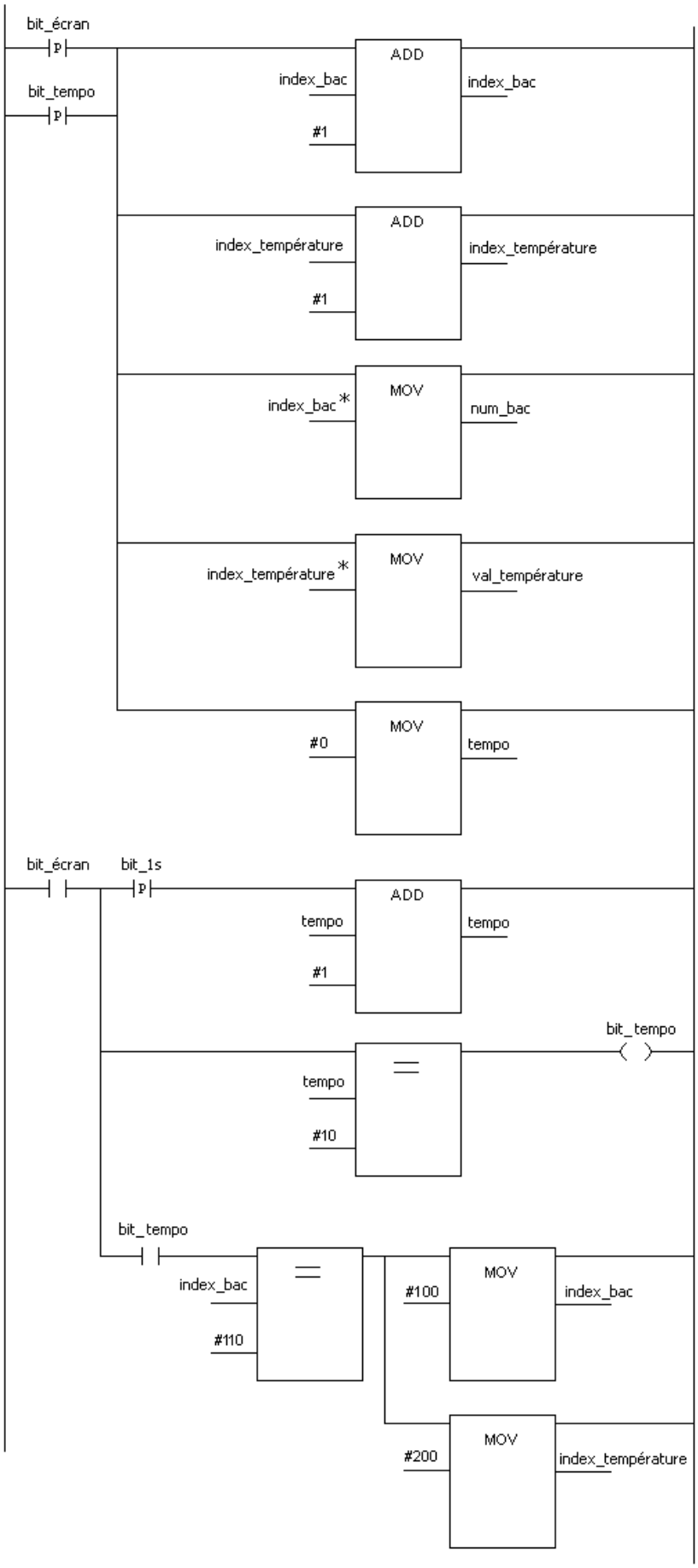
« utilisation de index_température »

Affichage cyclique des données sur l'écran N°11 du TDI



Programme


bit_écran
=
num_écran
#11
bit_écran
MOV
#100
index_bac
MOV
#200
index_température

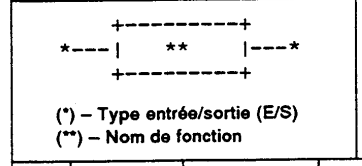

bit_écran
bit_tempo
ADD
index_bac
#1
index_bac
ADD
index_température
#1
index_température
MOV
index_bac*
num_bac
MOV
index_température*
val_température
MOV
#0
tempo
bit_écran
bit_1s
ADD
tempo
#1
tempo
bit_tempo
tempo
#10
bit_tempo
index_bac
#110
MOV
#100
index_bac
MOV
#200
index_température

ANNEXE Traitement Numérique

Norme 61131

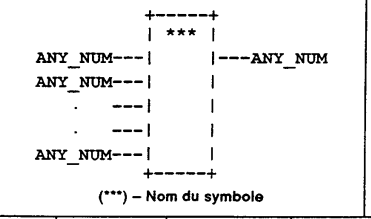
Fonctions standards

La présente annexe donne des définitions de fonctions communes à tous les langages de programmation d'automates programmables.
Une fonction standard, spécifiée comme extensible, a un nombre d'entrée variable. Le nombre maximal d'entrées est un paramètre propre à l'application concernée.
Fonctions numériques : Fonctions numériques standard a une seule variable

Forme graphique	Exemple d'application
	A := SIN(B) ; (Langage ST – voir 3.3)

N°	Nom de fonction	Type d'E/S	Description
	Fonctions générales		
1	ABS	ANY_NUM	Valeur absolue
2	SQRT	ANY_REAL	Racine carrée
	Fonctions logarithmiques		
3	LN	ANY_REAL	Logarithme naturel
4	LOG	ANY_REAL	Logarithme en base 10
5	EXP	ANY_REAL	Exponentielle naturelle
	Fonctions trigonométriques		
6	SIN	ANY_REAL	Sinus d'entrée en radians
7	COS	ANY_REAL	Cosinus d'entrée en radians
8	TAN	ANY_REAL	Tangente d'entrée en radians
9	ASIN	ANY_REAL	Arc sinus principal
10	ACOS	ANY_REAL	Arc cosinus principal
11	ATAN	ANY_REAL	Arc tangente principal

Fonctions arithmétiques :

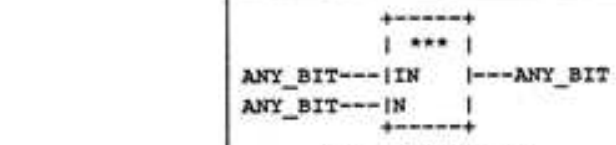
Forme graphique	Exemple d'application
	A := ADD(B,C,D) ; ou A := B+C+D ;

N°	Nom	Symbole (note 1)	Description (notes 2 et 8)	
	Fonctions arithmétiques extensibles			
12	ADD	+	OUT := IN1 + IN2 +...+ INn	
13	MUL	*	OUT := IN1 * IN2 *...* INn	
	Fonctions arithmétiques inextensibles			
14	SUB	–	OUT := IN1 – IN2	
15	DIV	/	OUT := IN1 / IN2	(note 5)
16	MOD		OUT := IN1 modulo IN2	(note 3)
17	EXPT	**	Exponentiation: OUT := $IN1^{IN2}$	(note 4)
18	MOVE	:=	OUT := IN	(note 9)

Fonctions de conversion de type :

N°	Forme graphique	Exemple d'application
1		A := INT_TO_REAL(B) ;

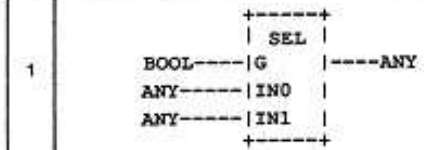
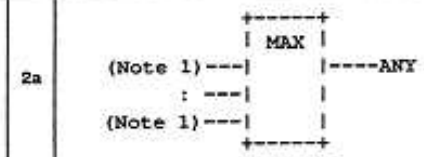
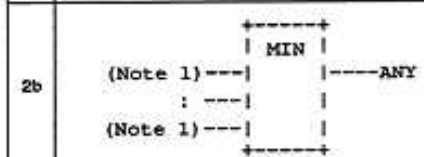
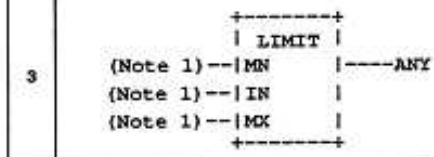
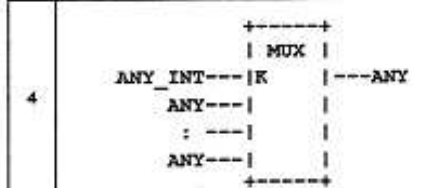
Fonction de décalage binaire :

Forme graphique	Exemple d'application
	A := SHL(IN := B, N := 5) ; (Langage ST – voir 3.3)

N°	Nom	Description
1	SHL	OUT := IN Décalage à gauche de N bits, remplissage de zéros à droite
2	SHR	OUT := IN Décalage à droite de N bits, remplissage de zéros à gauche
3	ROR	OUT := IN Rotation à droite de N bits, circulaire
4	ROL	OUT := IN Rotation à gauche de N bits, circulaire

NOTE – La notation "OUT" se rapporte à la sortie de la fonction.

Fonctions de sélection :

N°	Forme graphique	Explication/exemple
1		Sélection binaire: OUT := IN0 si G = 0 OUT := IN1 si G = 1 Exemple: A := SEL(G := 0, IN0 := X, IN1 := 5) ;
2a		Fonction extensible au maximum: OUT := MAX(IN1,IN2,...,INn) Exemple: A := MAX(B,C,D) ;
2b		Fonction extensible au minimum: OUT := MIN(IN1,IN2,...,INn) Exemple: A := MIN(B,C,D) ;
3		Limiteur: OUT := MIN(MAX(IN,MN),MX) Exemple: A := LIMIT(IN := B, MN := 0, MX := 5) ;
4		Multiplexeur extensible: Sélection d'une entrée parmi "N" entrées en fonction de K d'entrée Exemple: A :=MUX(K := 0, IN0 :=B, IN1 := C, IN2 := D) ; devrait avoir le même effet que: A := B ;

Fonctions de comparaison

Forme graphique	Exemple d'application
(Note 1) --| *** |---BOOL : --| | (Note 1) --| | (***) – Nom ou symbole	A := GT(B,C,D) ; ou A := (B>C) & (C>D) ;

N°	Nom	Symbole	Description
5	GT	>	Séquence décroissante: OUT := (IN1>IN2) & (IN2>IN3) & ... & (INn-1 > INn)
6	GE	>=	Séquence monotone: OUT := (IN1>=IN2) & (IN2>=IN3) & ... & (INn-1 >= INn)
7	EQ	=	Egalité: OUT := (IN1=IN2) & (IN2=IN3) & ... & (INn-1 = INn)
8	LE	<=	Séquence monotone: OUT := (IN1<=IN2) & (IN2<=IN3) & ... & (INn-1 <= INn)
9	LT	<	Séquence croissante: OUT := (IN1<IN2) & (IN2<IN3) & ... & (INn-1 < INn)
10	NE	<>	Inégalité (inextensible) OUT := (IN1 <> IN2)

Fonctions booléennes

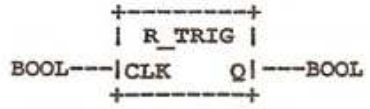
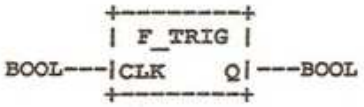
Forme graphique	Exemple d'application
ANY_BIT---| *** |---ANY_BIT ANY_BIT---| | . ---| | . ---| | ANY_BIT---| | (***) – Nom ou symbole	A := AND(B,C,D) ; ou A := B & C & D ;

N°	Nom	Symbole	Description
5	AND	& (note 1)	OUT := IN1 & IN2 & ... & INn
6	OR	>=1 (note 2)	OUT := IN1 OR IN2 OR ... OR INn
7	XOR	=2k+1 (note 2)	OUT := IN1 XOR IN2 XOR ... XOR INn
8	NOT		OUT := NOT IN1 (note 4)

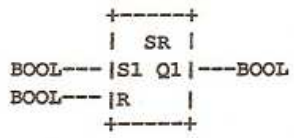
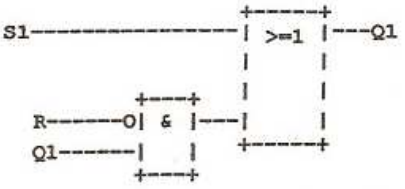
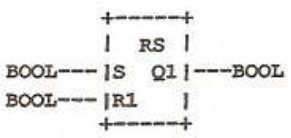
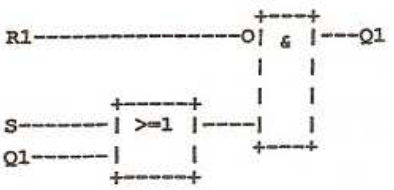
Blocs fonctionnels temporisateurs

N°	Description	Forme graphique
1	***est: TP (impulsion)	+-------+ | *** | BOOL---|IN Q|---BOOL TIME---|PT ET|---TIME +-------+
2	TON (Enclenchement)	
3	TOF (Déclenchement)	

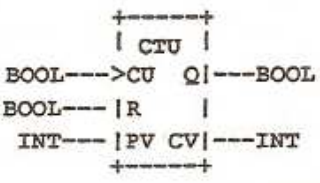
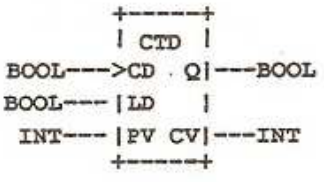
Blocs fonctionnels détection de front

N°	Forme graphique	N°	Forme graphique
1	Détecteur de front montant	2	Détecteur de front descendant
			

Blocs fonctionnels bistables

1	Bloc fonctionnel bistable (forcé dominant)	
		
2	Bloc fonctionnel bistable (réinitialisé dominant)	
		

Blocs fonctionnels compteurs

1	Compteur (comptage)	
		IF R THEN CV := 0 ; ELSIF CU AND (CV < PVmax) THEN CV := CV+1 ; END_IF ; Q := (CV >= PV) ;
2	Compteur (décomptage)	
		IF LD THEN CV := PV ; ELSIF CD AND (CV > PVmin) THEN CV := CV-1 ; END_IF ; Q := (CV <= 0) ;