OFPPT

ROYAUME DU MAROC

مكتب التكوين المهني وإنعاش الشغل

Office de la Formation Professionnelle et de la Promotion du Travail

DIRECTION RECHERCHE ET INGÉNIERIE DE FORMATION

RÉSUMÉ THÉORIQUE & GUIDE DE TRAVAUX PRATIQUES

MODULE : 24 — UTILISATION DE L'AUTOMATE PROGRAMMABLE

SECTEUR : ÉLECTROTECHNIQUE

SPÉCIALITÉ : ÉLECTROMÉCANIQUE DES SYSTÈMES AUTOMATISÉS

NIVEAU : TECHNICIEN SPÉCIALISÉ

ANNEE 2003

Document élaboré par :

Nom et prénom	EFP	DR
ELKORNO NAIMA	CFF / ISIC	

Révision linguistique

-
-
-

Validation

-
-
-

SOMMAIRE

Module 8 : UTILISATION DE L'AUTOMATE PROGRAMMABLE

Code : **Durée : 75 h**

OBJECTIF OPÉRATIONNEL DE PREMIER NIVEAU
DE COMPORTEMENT

COMPORTEMENT ATTENDU

Pour démontrer sa compétence l'apprenti doit
utiliser un automate programmable
selon les conditions, les critères et les précisions qui suivent.

CONDITIONS D'ÉVALUATION

- À partir de directives.
- À l'aide :
  - de fiches techniques et du manuel d'utilisation d'un automate;
  - d'un programme en langage Grafcet ou en diagramme à échelons;
- Sur un automate programmable avec E/S " tout ou rien ".

CRITÈRES GÉNÉRAUX DE PERFORMANCE

- Utilisation appropriée de l'équipement informatique.
- Respect des méthodes et des conventions de programmation d'un automate.
- Respect des normes.

(à suivre)

OBJECTIF OPÉRATIONNEL DE PREMIER NIVEAU
DE COMPORTEMENT(suite)

PRÉCISIONS SUR LE COMPORTEMENT ATTENDU	CRITÈRES PARTICULIERS DE PERFORMANCE
A. Raccorder un automate.	- Repérage de l'information pertinence dans la documentation technique. - Localisation précise des points de raccordement - Câblage conforme au schéma de raccordement. - Reconnaissance précise des composants associés au matériel et des composants associés au logiciel.
B. Accéder aux fonctions d'un automate.	- Configuration précise de l'automate. - Utilisation appropriée du logiciel ou de la console de programmation dédiée. - Détermination juste du mode d'adressage.
C. Déceler des problèmes de fonctionnement d'un automatisme simple commandé par un automate.	- Respect de la procédure d'interrogation des E/S et des cases mémoire. - Indication juste des problèmes de fonctionnement.
D. Apporter des modifications mineures au programme d'un automate	- Modification conforme à la demande. - Respect de la procédure d'éducation. - Programmation précise des ajouts ou des retraits. - Respect de la procédure de sauvegarde.
E. Effectuer l'essai d'un automatisme simple commandé par un automate.	- Programmation fonctionnelle en simulation. - Système fonctionnel à la suite de la modification.

OBJECTIFS OPÉRATIONNELS DE SECOND NIVEAU

l'apprenti DOIT MAÎTRISER LES SAVOIRS, SAVOIR-FAIRE, SAVOIR PERCEVOIR OU SAVOIR ÊTRE JUGÉS PRÉALABLES AUX APPRENTISSAGES DIRECTEMENT REQUIS POUR L'ATTEINTE DE L'OBJECTIF DE PREMIER NIVEAU, TELS QUE :

Avant d'apprendre à raccorder un automate (A) :

1. Décrire l'architecture d'un automate.
2. Décrire les applications d'un automate.
3. Reconnaître les différents modules E/S.
4. Repérer un logiciel de programmation.

Avant d'apprendre à accéder aux fonctions d'un automate (B) :

5. Distinguer les langages de programmation d'un automate.
6. Décrire les principales instructions d'un automate.
7. Utiliser un logiciel de programmation.

Avant d'apprendre à déceler des problèmes de fonctionnement d'un automatisme simple commandé par un automate (C) :

8. Reconnaître l'état logique des E/S dans un programme par le mode dynamique (en ligne).

Avant d'apprendre à effectuer l'essai d'un automatisme simple commandé par un automate (E) :

9. Reconnaître les dangers potentiels.

PRÉSENTATION DU MODULE

Ce module de compétence générale est enseigné au troisième semestre du programme. Il est préalable aux apprentissages réalisés au module 29
(Système automatisé contrôlé par A P I) et nécessite comme pré requis les modules 18 (Logique combinatoire) et 19 (Logique séquentielle).

L'objectif de ce module est de faire acquérir les connaissances relatives à l'architecture d'un automate ,aux divers modes d'entrées et de sorties, aux langages de programmation, au mode de communication, ainsi qu'à la vérification de l'état logique des éléments en vue d'un diagnostic. Il vise donc à rendre l'apprenti apte à utiliser un automate programmable

La durée totale de ce module est de 75 H dont la théorie représente 38 % et la pratique 62% L'évaluation incluse.

Module24 UTILISATION DE L'AUTOMATE PROGRAMMABLE
RESUME THEORIQUE

I - Raccordement d'un automate programmable

I.1 - Introduction

Les automatismes sont réalisés en vue d'apporter des solutions à des problèmes de nature technique, économique ou humaine.

Eliminer les tâches dangereuses et pénibles, en faisant exécuter par la machine les tâches humaines complexes ou indésirables.

Améliorer la productivité en asservissant la machine à des critères de production, de rendement ou de qualité.

Piloter une production variable, en facilitant le passage d'une production à une autre.

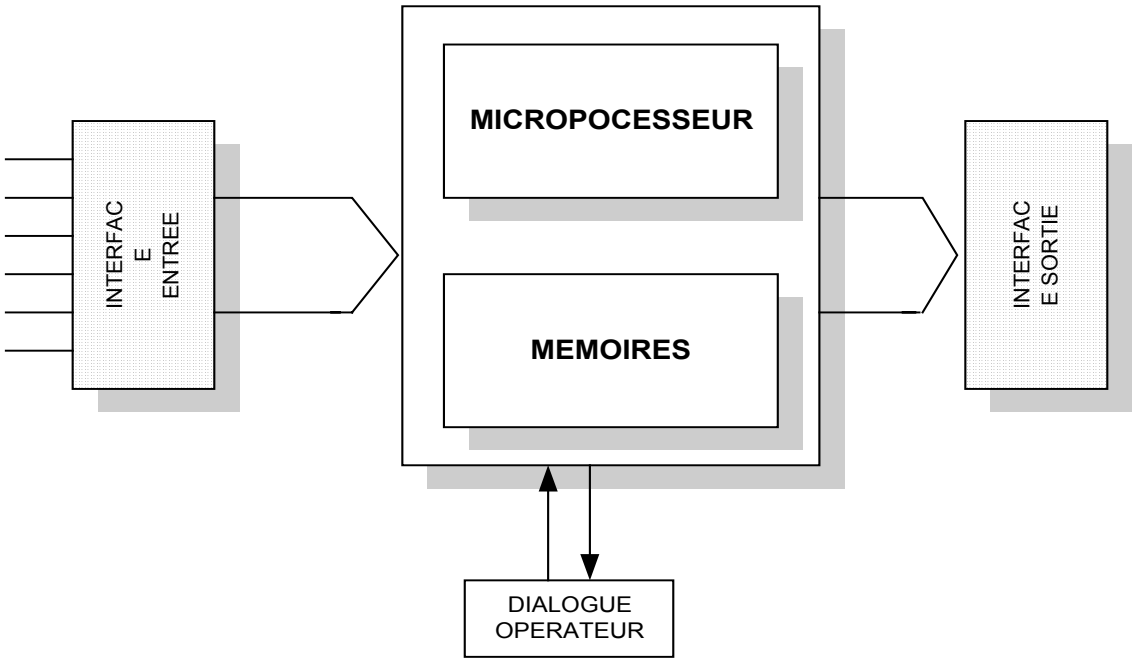
Renforcer la sécurité en surveillant et contrôlant les installations et machines.

On distingue dans tout système automatisé la machine ou l'installation et la partie commande constituée par l'appareillage d'automatisme. Cette partie commande est assurée par des constituants répondant schématiquement à quatre fonctions de base :

- L'acquisition des données
- Le traitement des données
- La commande de puissance
- Le dialogue homme machine

I.2 - Structure fonctionnelle de l'automate :

L'automate programmable industriel est un appareil qui traite les informations selon un programme préétabli.
Son fonctionnement est basé sur l'emploi d'un micro processeur et de mémoires.



Structure d'un système de traitement

I.2.1 - Interface d'entrée

Elles permettent d'isoler électroniquement le circuit externe (saisie de l'information) du circuit de traitement.

I.2.2 - L' unité centrale:

cœur de l'automate, elle est constituée:

- d'un **processeu**r qui exécute le programme
- de **mémoires** qui, non seulement contiennent ce programme, mais aussi des informations de données (durée d'une temporisation, contenu d'un compteur)

Les types de mémoires :

Mémoires vives:

RAM – Random Access Memory (Mémoire à accès aléatoires)

Ce sont des mémoires volatiles.Lues et écrites par le processeur.

Mémoires mortes:
ROM – Read only memory
PROM – ROM programmable
NE PEUVENT PAS ETRE EFFACES
REPROM – effacement par UV
EEPROM – effacement électrique

I.2.3 - Interface de sortie

Elles permettent de commander les sorties toute ou rien (TOR) telle que : les contacteurs, les moteurs pas à pas, les électrovannes et ainsi des sorties analogiques (boucle de régulation débit température et variateur de vitesse.)

I.2.4 - Communication et dialogue

Elle est réalisée avec l'opérateur par un pupitre de dialogue ou par l'intermédiaire d'un ordinateur et avec les autres automates pour un réseau informatique local.

I.3 - Description des automates :

Il existe deux types d'automate programmable industriel:

- le type **monobloc**
- le type **modulaire**

I.3.1 - Automate Monobloc :

Le type **monobloc** possède généralement un nombre d'entrées et de sorties restreint et son jeu d'instructions ne peut être augmenté. Bien qu'il soit parfois possible d'ajouter des extensions d'entrées/sorties, le type monobloc a pour fonction de résoudre des automatismes simples faisant appel à une logique séquentielle et utilisant des informations tout-ou-rien.

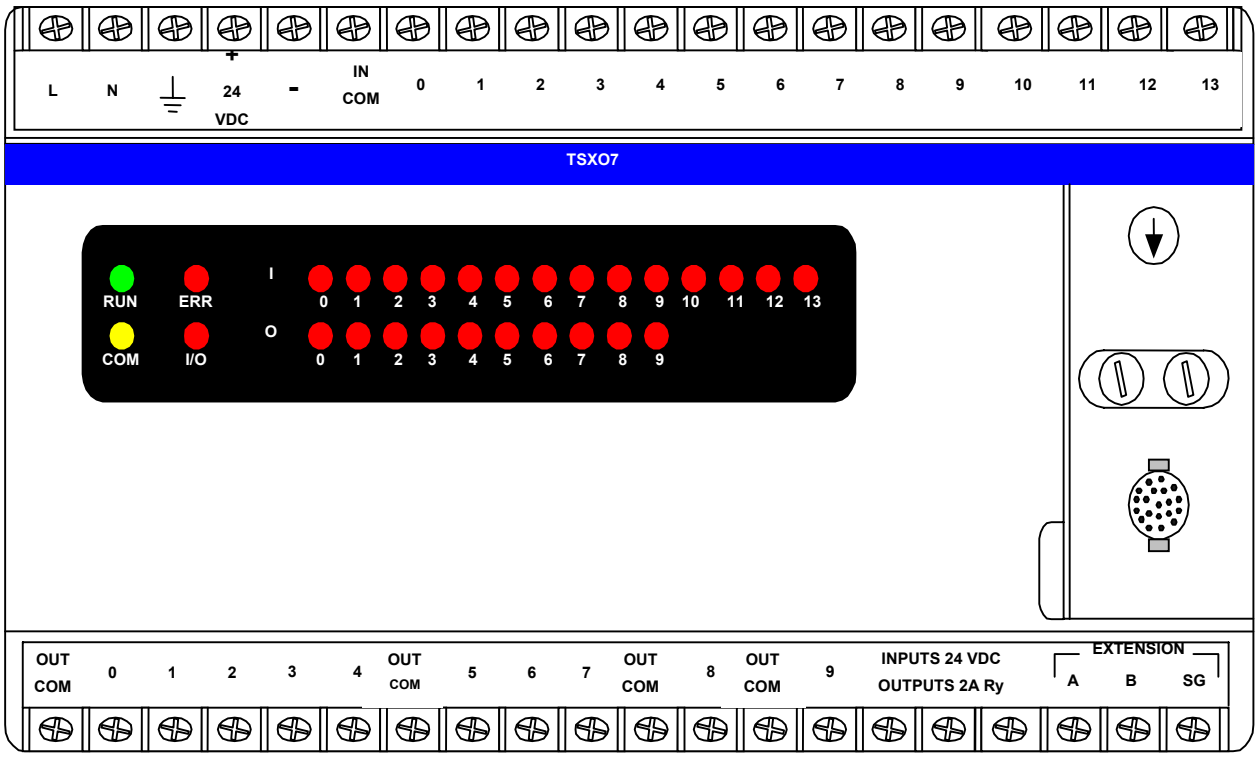


FIGURE 1 :Automate monobloc TSX Nano

Exemple 1 : automate monobloc

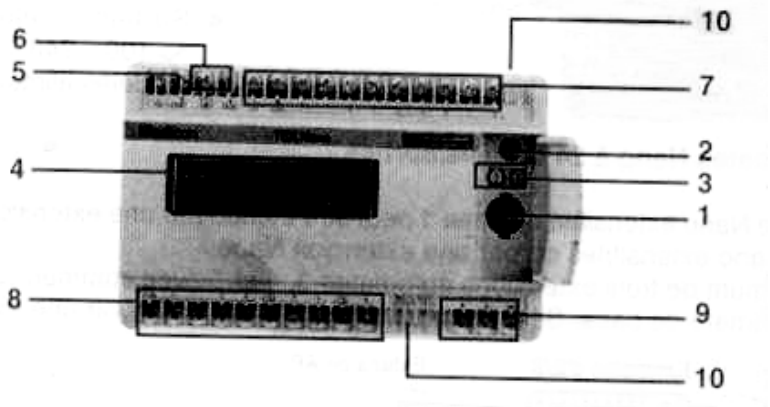


Fig. 2 automate monobloc

1- Une prise (1) pour raccordement du terminal de programmation .
2- Un sélecteur pour codage de la fonction base / extension.
3- Deux points de réglage analogique.
4- Une visualisation :
- Des entrées 0 à 8 ou 0 à 13 et sorties 0 à 6 ou 0 à 9,
- De l'état automate (RUN, ERR, COM, I/O).
5- Un raccordement de l'alimentation secteur
6- Une alimentation capteurs (=24V/150mA) sur modèles alimentés en ~ 100…240V.
7- Un raccordement des capteurs d'entrées.
8- Un raccordement des préactionneurs de sorties.
9- Un raccordement extension (extension d'entrées /sorties et / ou extension automate) ou raccordement Modbus esclave
10- Un cache amovible pour protection des borniers à vis.

I.3.2 - Automate Modulaire:

Par ailleurs, le type **modulaire** est adaptable à toutes situations. Selon le besoin, des modules d'entrées/sorties analogiques sont disponibles en plus de modules spécialisés tels: PID, BASIC et Langage C, etc. La modularité des API permet un dépannage rapide et une plus grande flexibilité. La figure 3 présente un automate modulaire.

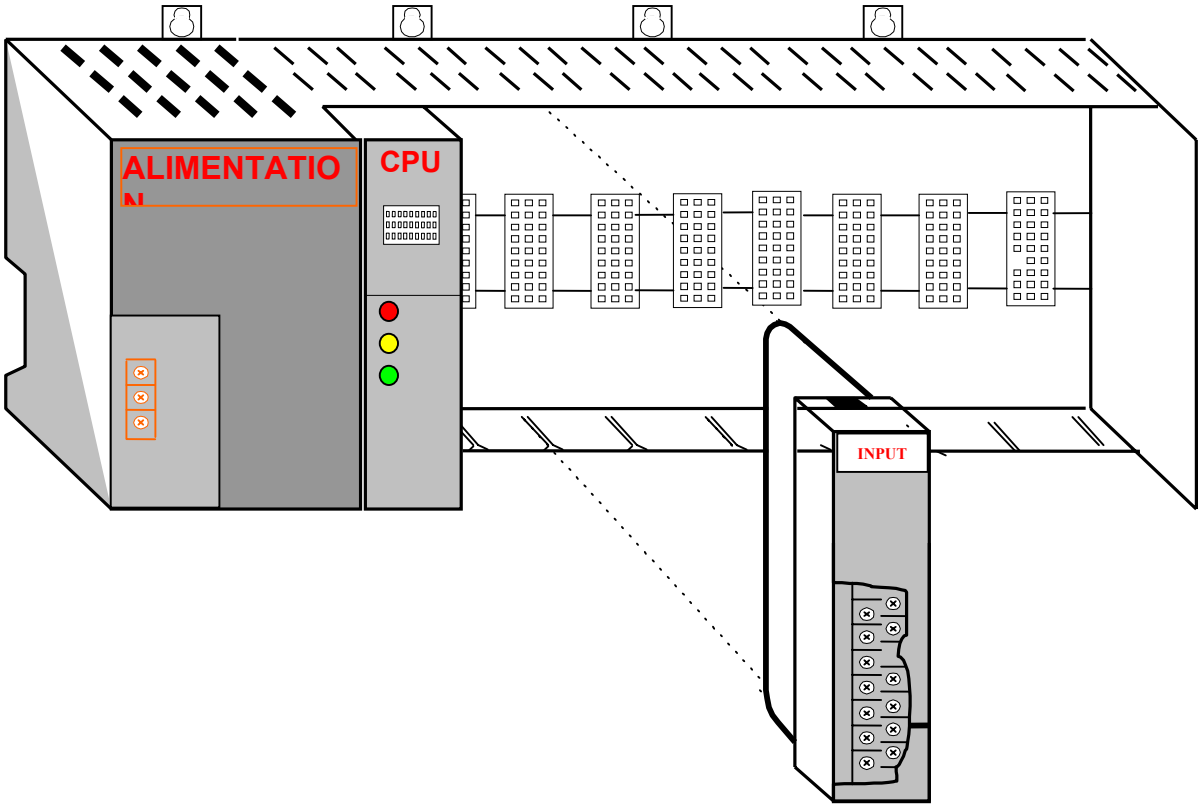


FIGURE 3:API MODULAIRE

Exemple 2 :automate modulaire :

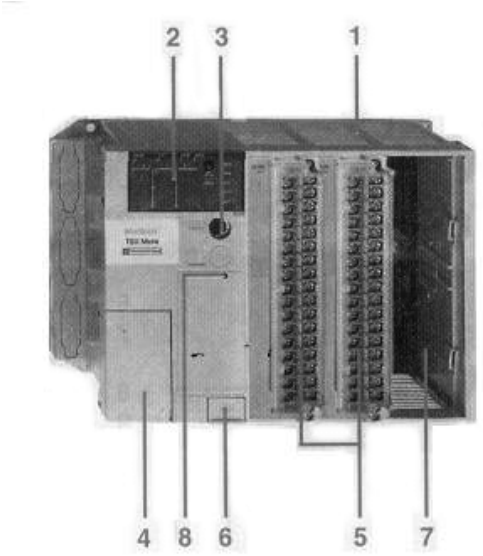


Fig. 4 automate modulaire :

L'automate TSX 37-08 comprend :

1- Un bac à 3 emplacement.
2- Un bloc de visualisation centralisé.
3- Une prise terminal repérée TER.
4- Une trappe d'accès aux bornes d'alimentation.
5- Deux modules à 16 entrées et 12 sorties « Tout ou Rien » positionnés dans le premier et le deuxième emplacements (positions 1, 2, 3 et 4).
6- Une trappe d'accès à la pile optionnelle.
7- Un emplacement disponible.
8- Un bouton de réinitialisation

I.4 - Les applications de l'automate :

Les automates trouvent leur application en milieu industriel, domestiques. On cite quelques exemples courants :

Exemple n°1: Feux de carrefour

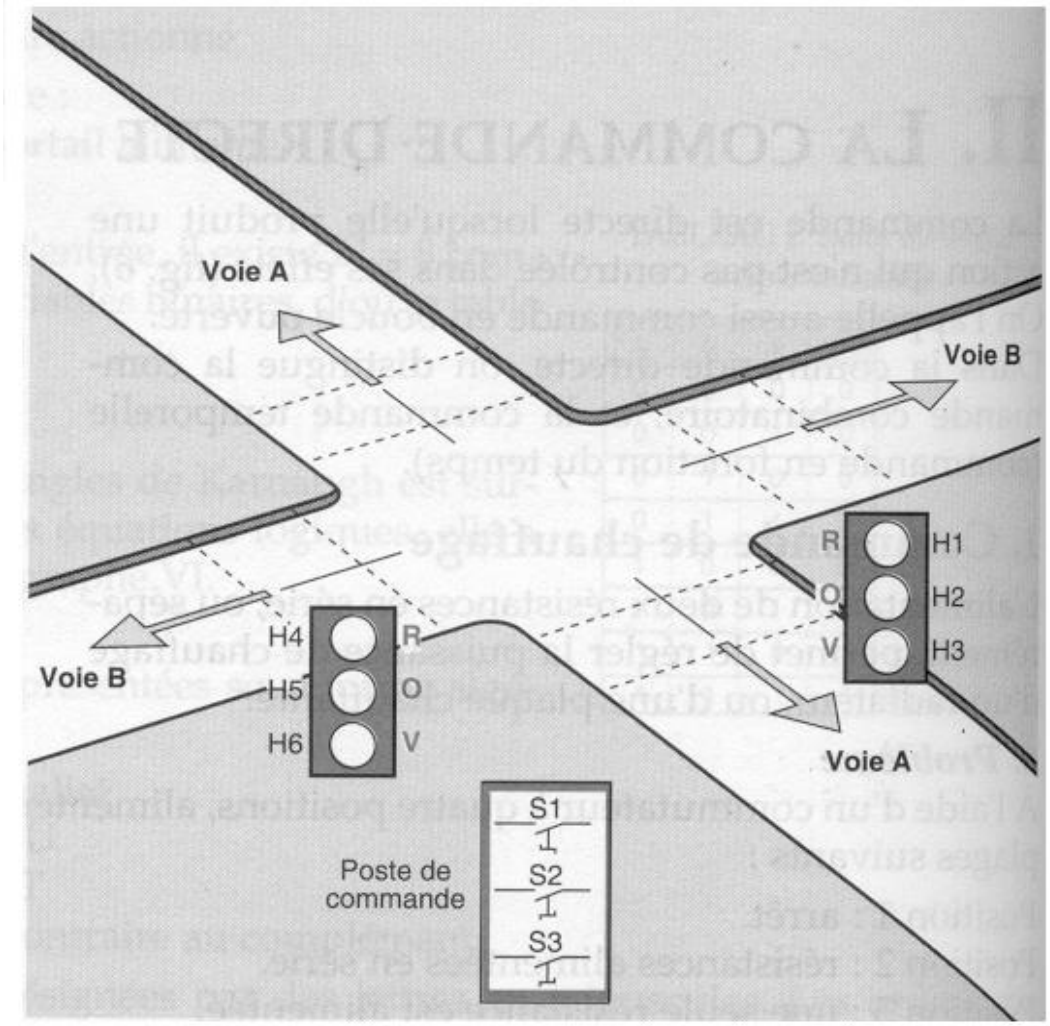


Fig. 5 : Feux de carrefour

Description

On régle la circulation d'un carrefour de deux voies A et B par des feux tricolores (Rouges, orange, vert).

Exemple 2 : Portail coulissant.

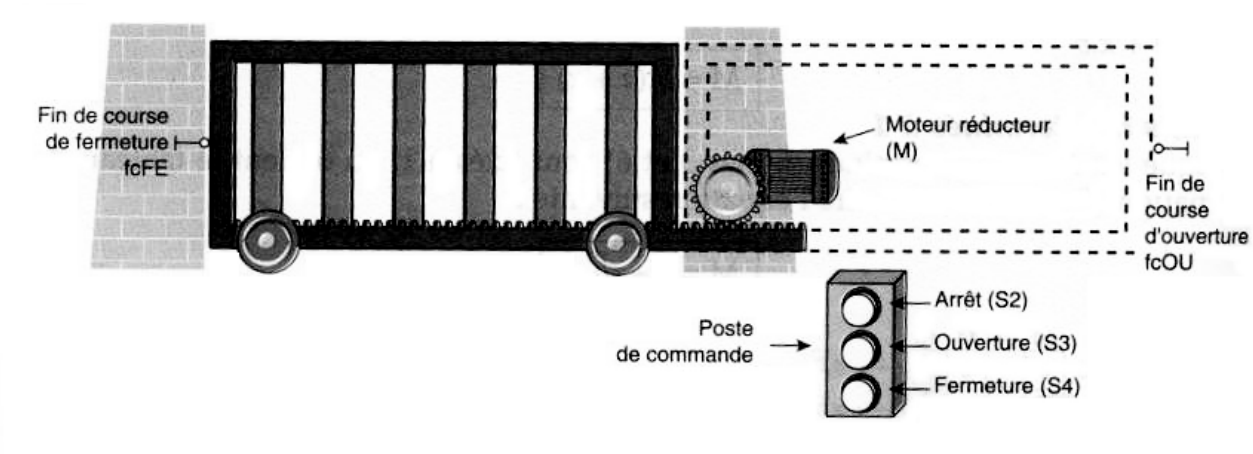


Fig. 6 : Portail coulissant.

Soit un portail coulissant à commander :

- Le portail étant fermé, le contact fin de course fcFE est actionné ;
- On appuie sur le bouton-poussoir d'ouverture S3, le moteur actionne le portail et provoque son ouverture ;
- En fin d'ouverture, le contact fin de course fcOU est actionné, il signale l'ouverture du portail, et il coupe l'alimentation du moteur.

L'action sur le bouton-poussoir de fermeture provoque l'inversion de sens de marche du moteur, et la fermeture du portail.

Le portail libère le contact fcOU, et se déplace jusqu'à actionner le contact fcFE qui provoque l'arrêt du moteur.

Exemple 3 système de perçage

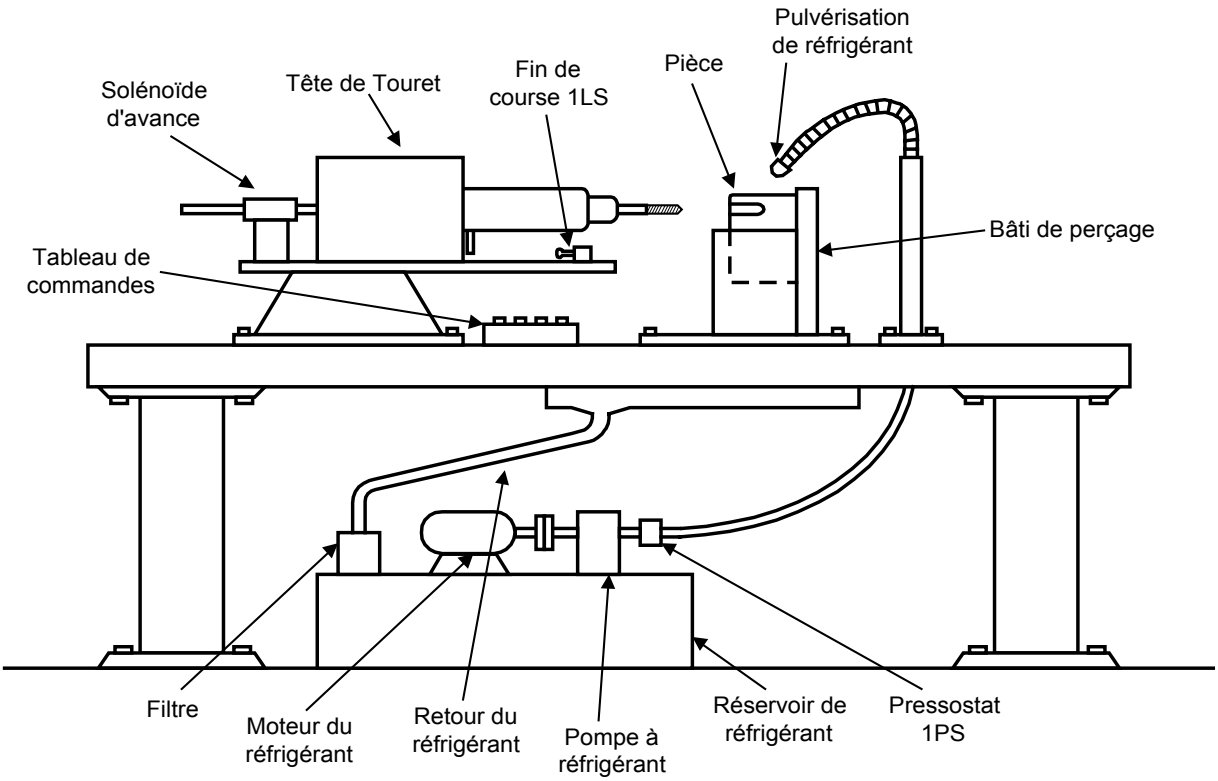


Fig. 7 système de perçage

Exemple 4° système de pompage

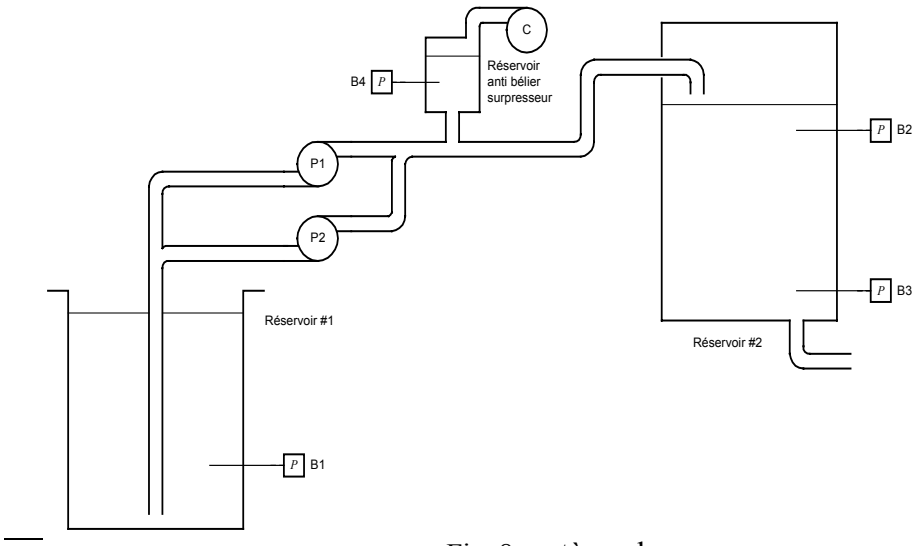


Fig. 8 système de pompage

I.5 - Les différents modules d'entrée/ sortie :

Les modules d'entrée / sortie sont les interfaces qui permettent de communiquer avec le micro presseur .

On distingue :

- Les interfaces d'entrée.
- Les interface de sortie.

I.5.1 - Interface d'entrée :

a) Interface Tout on rien (TOR)

A partir d'un signal quelconque en entrée, les interfaces fournissent en sortie deux tensions 0V ou 5V Ces interfaces sont de type à contact, ou statique.

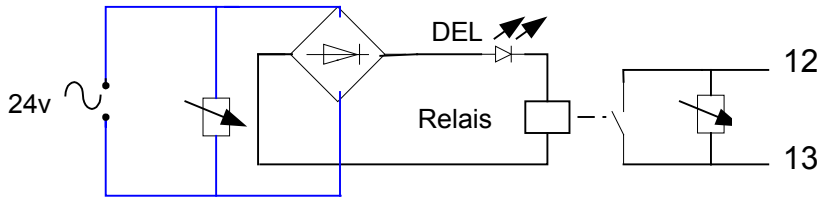


Fig. 9 **:** Interface d'entrée tout ou rien à relais

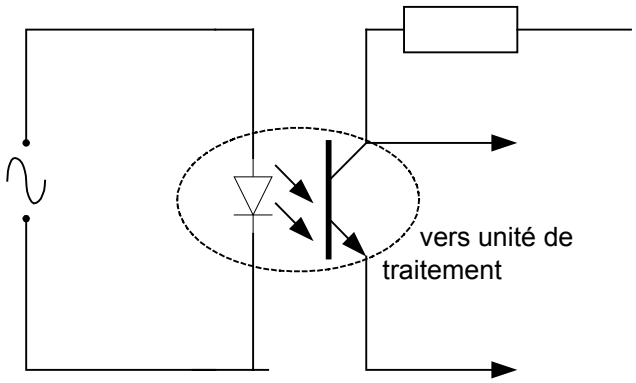


Fig. 10 : Interface d'entrée tout ou rien statique.

b) Transmetteurs analogiques

Les transmetteurs analogiques : Tension / intensité permettent d'adapter les signaux issus des capteurs pour les rendre compatibles avec l'unité de traitement. La variation de la grandeur d'entre est convertie en une variation :

- En tension : de 0V, à10V
- En intensité : de 0 Ma à 20mA, ou de 4 mA à 20mA

Exemple : Transmission de mesure de température effectuée par une sonde PT

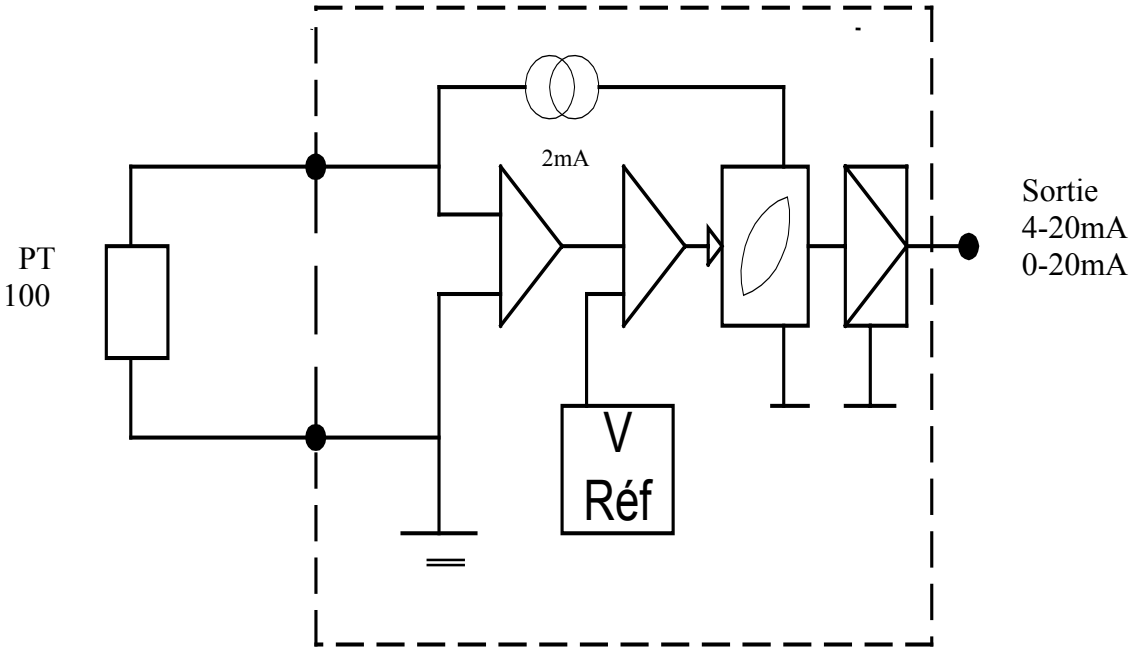


Fig. 11 Interface d'entrée analogique

I.5.2 - Interface de sortie :

a) Interface de sortie tout ou rien

La sortie de l'unité de traitement peut s'effectuer soit sur relais, soit sur transistor (TTL), ou avec un triac.

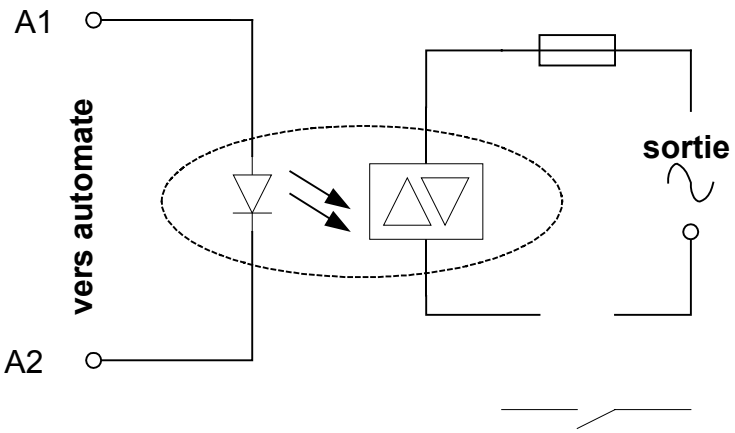


fig.12 Interface de sortie statique avec triac

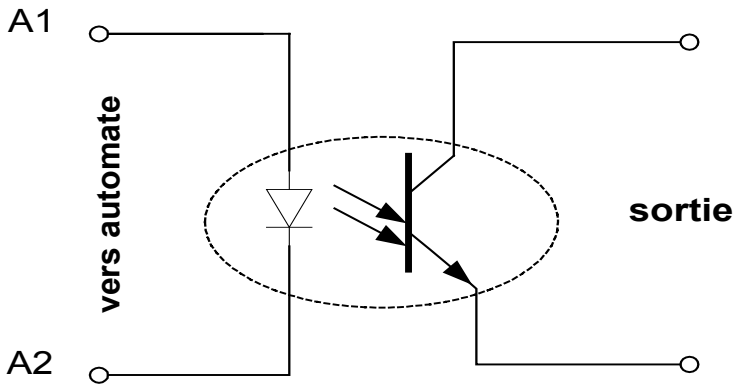


fig.13 Interface de sortie statique à transistor

b) Interfaces de sorties analogiques

Les conventions digitaux /analogiques ont pour fonction de générer un signal analogique normalisé (0-10 V ;0-20mA) à partir d' une information numérique, délivrée par l' unité de traitement et codée en binaire, sur des sorties digitales TOR raccordées aux entrées de l' interface(ou convertisseur).

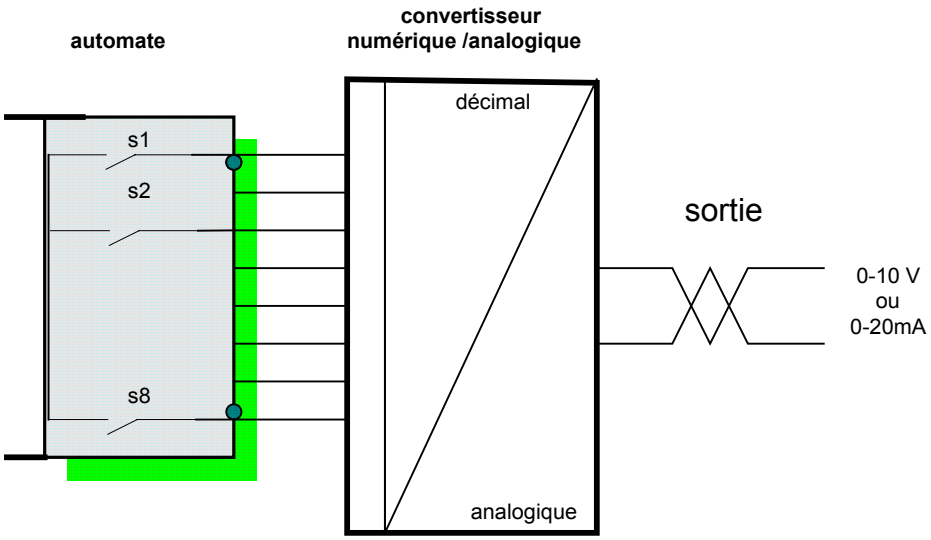


fig14 Interface de sortie numérique/analogique

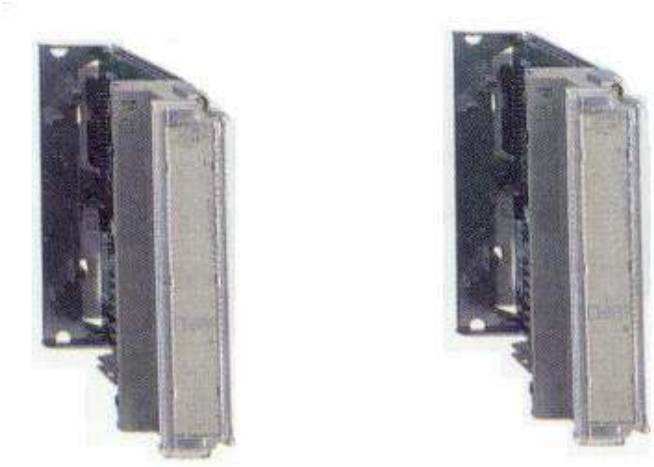
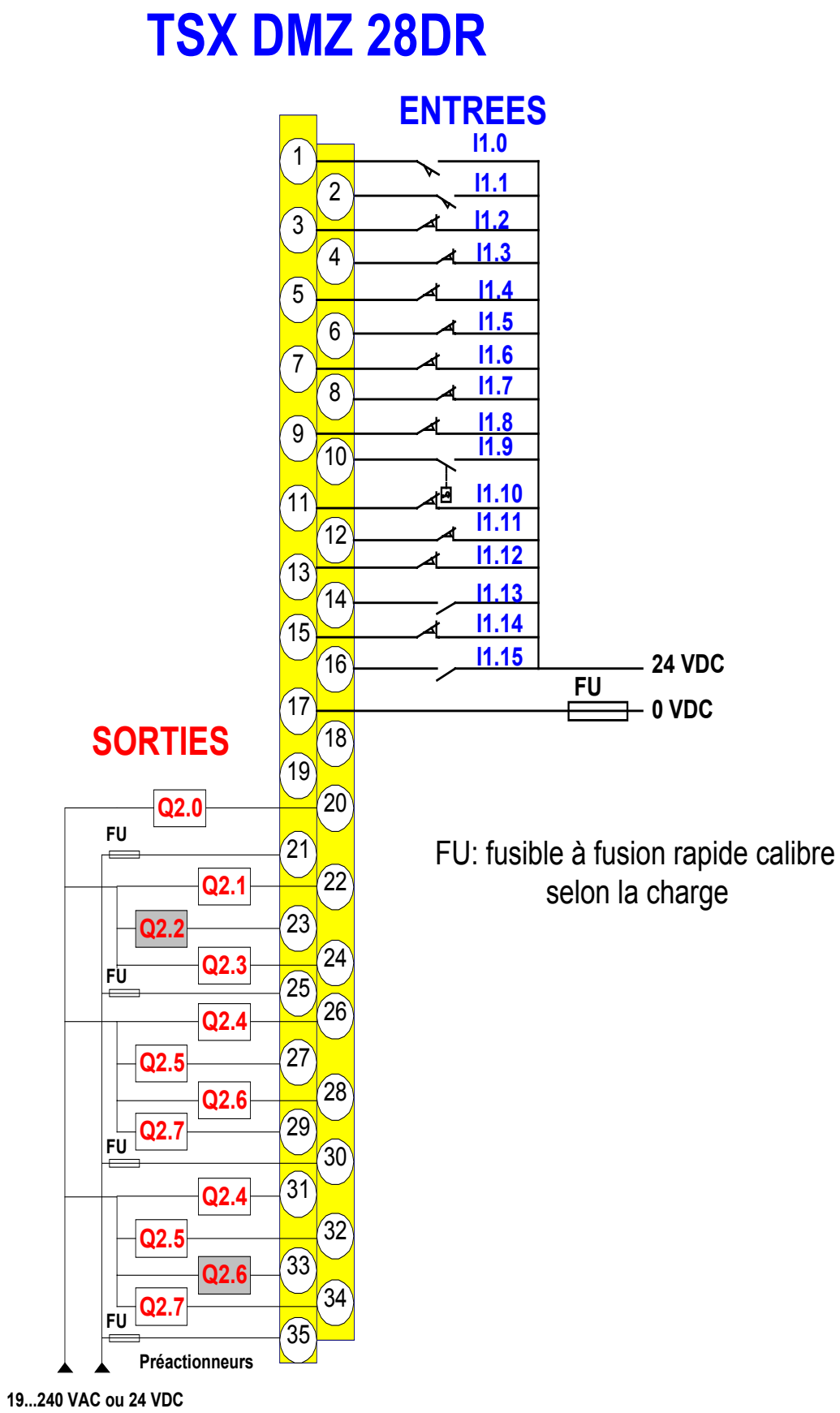


Fig. 15 Modules E/S

Fig 16 Les bornes sont des E/S

Fig 17 : Le n° 1 est un connecteur pour une entrée sortie analogique



F IG 18 Câblage des entrées/sorties TOR :

I.6 - Les logiciels de programmation :

Marque	Automate	Logiciel
Télemecanique	TSX Nano	Pl707
	TSX 3708,TSx22	Pl7-micro
	TSX Premium	Pl7 junior
ALENBRADLEY	SLC 500	APSF
SIEMENS	Serie5:S5 Serie7:S7	Step 5 Step 7

Ce tableau récapitulatif donne le logiciel et le type d'automate conforme à ce dernier .L'opérateur peut communiquer avec l'automate soit à travers un P.C portable ,fixe (Fig.18) ou avec la console (Fig. 19,20). On lie l'automate au PC (ou à la console) par un câble(RS232)

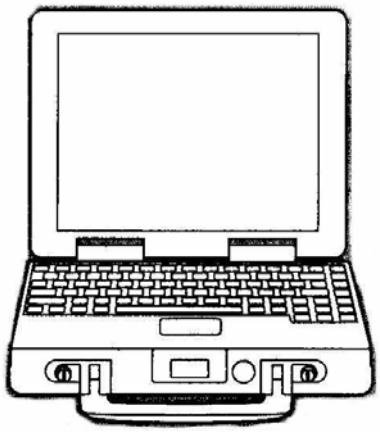


Fig. 19 PC portable

Fig20 console

Fig. 21 console

I.7 - Les étapes à suivre pour raccorder un automate:

Pour raccorder un automate, il est recommandé de suivre :

- Les spécifications du fabricant

- La technique de raccordement

- De vérifier si les modules sont dans leurs embases respectives. Vérifier le type, le numéro du modèle et le diagramme de câblage. Vérifier l'emplacement des embases dans le document pour l'assignation des adresses d'E/S.

- De localiser le paquet de fils correspondant à chaque module et le diriger à travers le conduit à l'emplacement du module.

  Identifier chacun des fils dans le paquet et s'assurer qu'ils correspondent à ce module en particulier.

- En commençant avec le premier module, repérer le fil dans le paquée qui se branche à la borne la plus basse. Au point où le fil arrive à la même hauteur que le point de terminaison, plie le fil à angle droit vers la borne.

- De couper le fil pour qu'il dépasse de 6 mm du côté de la vis de la borne. Dégainer l'isolant du fil à approximativement 9 mm

  Insérer le fil sous la plaque de la borne et serrer la vis.

- Si deux modules ou plus utilisent la même source d'alimentation, on peut utiliser du cavalier «jumpers » pour le câblage de la source d'alimentation d'un module à l'autre.

- Si le câble blindé est utiliser, en brancher seulement un bout à la mise à la terre, préférablement au châssis. Ce branchement évitera toutes boucles possibles de retour de masse. L'autre bout doit être coupé et non branché.

- De répéter la procédure de câblage pour chaque fil du paquet jusqu'à ce que le câblage du module soit complété. Après que tous les fils aient été branchés, tirer doucement sur chacun pour s'assurer d'avoir un bon branchement.

De répéter la procédure de câblage jusqu'à ce que tous les modules soient terminés.

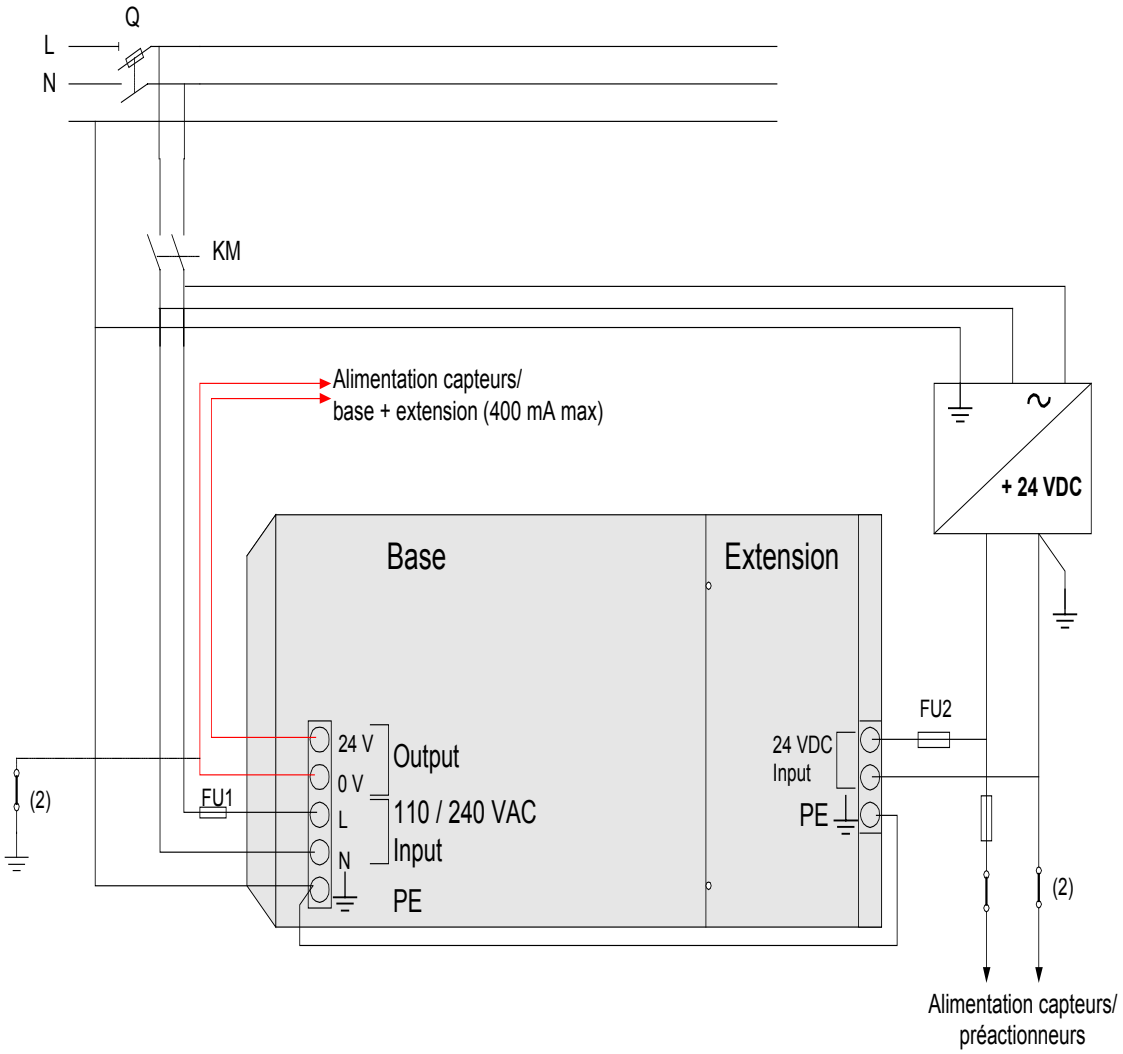


Fig 22 : Raccordement des alimentations :

II - ACCES AUX FONCTIONS D'UN AUTOMATE :

II.1 - Les Langages de programmation :

II.1.1 - Langage à contacts

Structure d'un programme :

Un programme en langage à contacts est composé d'une suite de réseaux de contacts exécutée de façon séquentielle par l'automate :

Dessiné entre deux barres de potentiel, un réseau est un ensemble d'éléments graphiques représentant :
• les entrées/sorties de l'automate (boutons-poussoirs, détecteurs, relais, voyants...),
• des fonctions d'automatismes (temporisateurs, compteurs...),
• des opérations arithmétiques, logiques et spécifiques,
• les variables internes de l'automate.
Ces éléments graphiques sont reliés entre eux par des connexions horizontales et verticales.
(Voir figure 1)

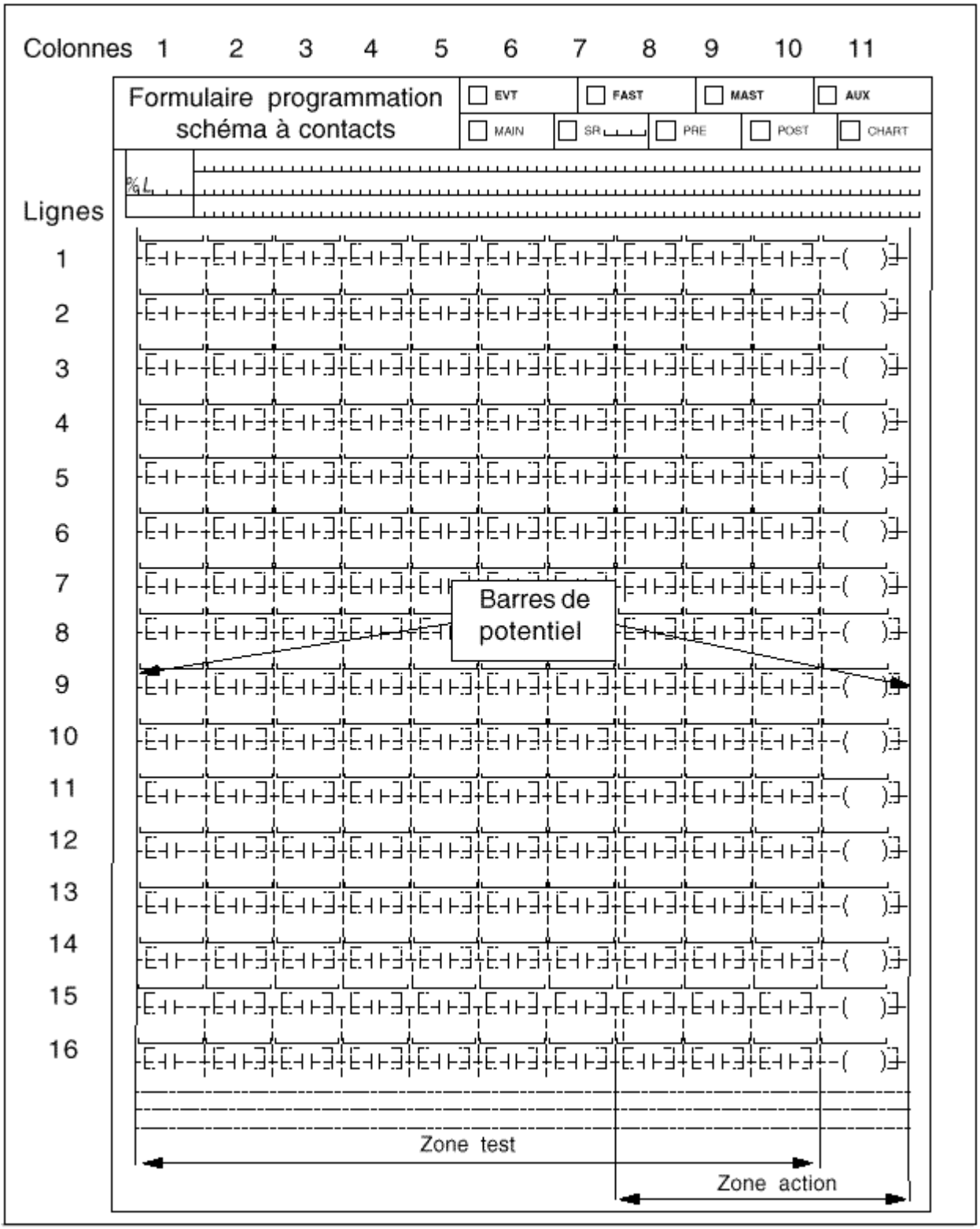


Fig 1 : Structure d'un réseau de contacts

EXEMPLE :

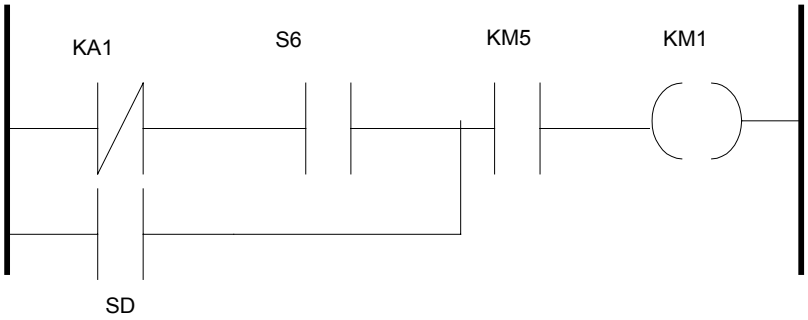


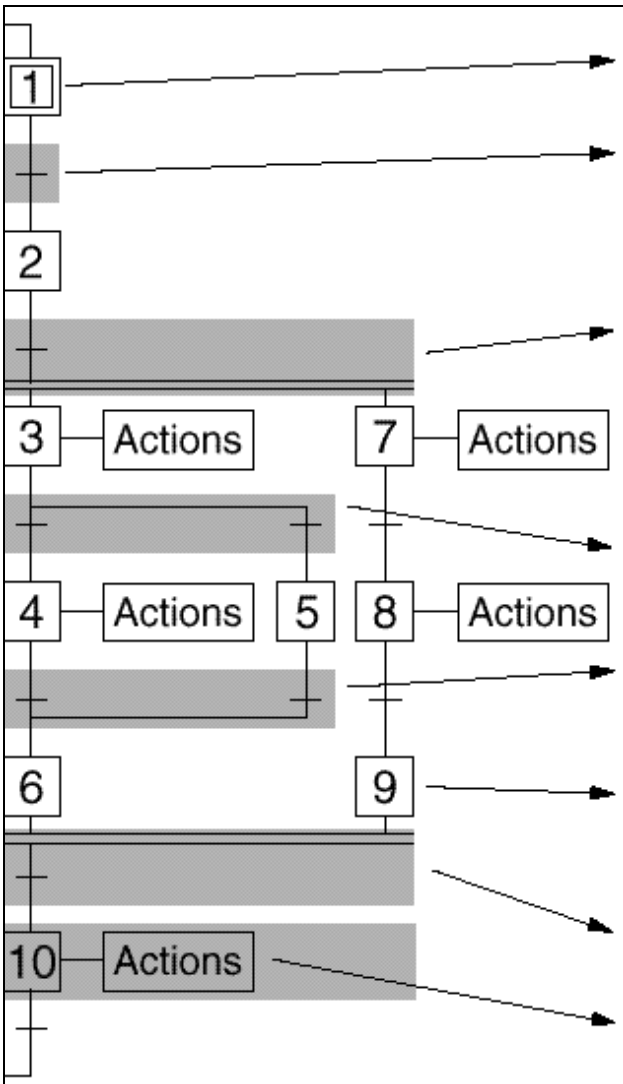
Fig 2 : Exemple écrit avec un langage à contacts

II.1.2 - Langage GRAFCET

Le **GRAFCET** est une représentation graphique qui permet la transcription du fonctionnement d'un système automatique. Il prend en compte les entrées et les sorties, et définit le comportement séquentiel du système.

L'**étape** correspond à une situation élémentaire ayant un comportement stable.

Une **transition** indique la possibilité d ' évolution d 'une étape à l' étape suivante. A chaque transition , on associe une, ou des conditions logiques qui traduisent la notion de **réceptivité**.



Étape initiale : définit la situation initiale de l'automatisme.

Transition : les **réceptivités** associées indiquent les conditions logiques nécessaires au franchissement de cette transition.

Activation simultanée des étapes 3 et 7 (Divergence en ET). Les sous-ensembles formés par les étapes 3, 4, 5, 6 et 7, 8, 9 constituent deux séquences dites simultanées.
Aiguillage (Divergence en OU) à partir de l'étape 3 vers l'étape 4 ou vers l'étape 5.

Fin d'aiguillage (convergence en OU) à partir de l'étape 4 ou de l'étape 5 vers l'étape 6.
Étape de fin de séquence : permet la synchronisation des séquences simultanées.

Désactivation simultanée des étapes 6 et 9 (convergence en ET).

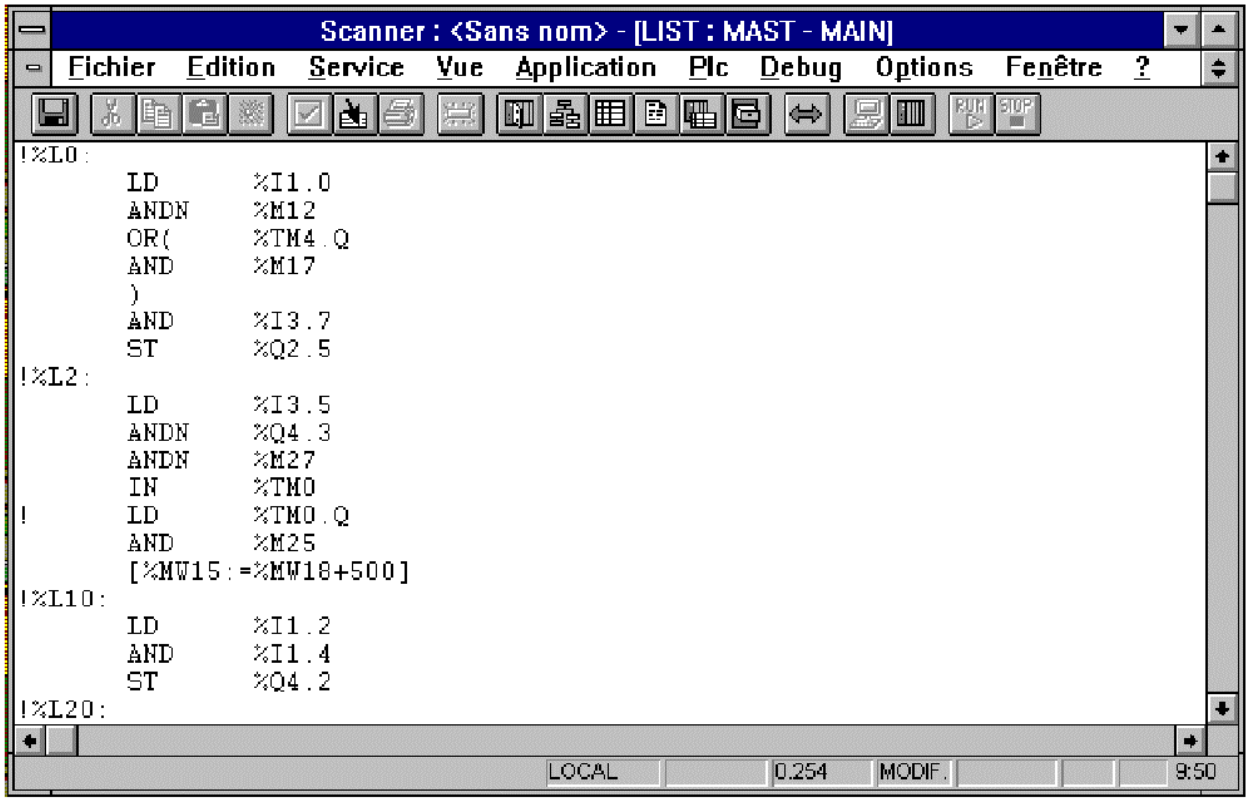
Étape : les **actions** associées ne s'exécutent que lorsque l'étape est active.

Fig 3 : exemple avec langage grafcet

II.1.3 - Présentation du langage liste d'instructions :

a) Principe

Un programme écrit en langage liste d'instructions se compose d'une suite d'instructions exécutées séquentiellement par l'automate.



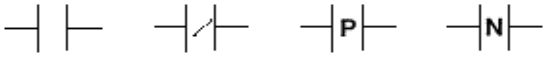
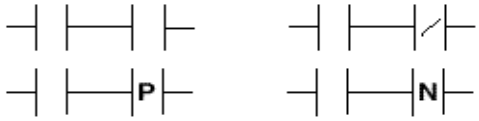
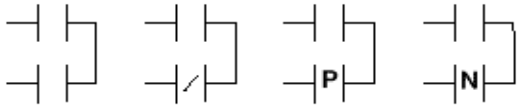
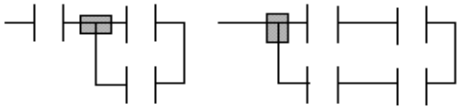
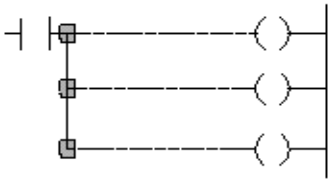
Exemple d'instruction : LD %I1.0

Code instruction — Opérande

Chaque instruction est composée d'un code instruction et d'un opérande.
Ces instructions agissent sur :
• les entrées/sorties de l'automate (boutons-poussoirs, détecteurs, relais, voyants...),
• des fonctions d'automatismes (temporisateurs, compteurs...),
• des opérations arithmétiques et logiques et des opérations de transfert,
• les variables internes de l'automate.
Il existe 2 types d'instructions :
• Instruction de test, dans laquelle figurent les conditions nécessaires à une action, ex : LD, AND, OR...
• Instruction d'action, qui sanctionne le résultat consécutif à un enchaînement de test.
ex : ST, STN, R, ...

b) Les instructions

Instructions de base

Désignation	Instructions	Fonctions équivalentes
Instructions de test	• **LD, LDN, LDR, LDF**	
	• **AND, ANDN, ANDR, ANDF**	
	• **OR, ORN, ORR, ORF**	
	• **AND(, OR(** (8 niveaux de parenthèses)	
	• **XOR, XORN, XORR, XORF** Ou exclusif	
	• **MPS** **MRD** **MPP**	
	• **N**	Négation
Instructions d'action	• **ST, STN, S, R**	
	• **JMP, JMPC, JMPCN**	Permet un branchement (inconditionnel, conditionné à un résultat booléen à 1, conditionné à un résultat booléen à 0) à une instruction étiquetée, amont ou aval.
	• **SRn** **RET, RETC, RETCN**	Branchement en début de sous-programme Retour de sous-programme (inconditionnel, conditionné à un résultat booléen à 1, conditionné à un résultat booléen à 0).
	• **END, ENDC, ENDCN**	Fin de programme, (inconditionnel, conditionné à un résultat booléen à 1, conditionné à un résultat booléen à 0).
	HALT	Arrêt de l'exécution du programme.

c) Structure d'un programme

Généralités

Comme en langage à contacts, les instructions sont organisées en séquence d'instructions (équivalent à un réseau de contacts) appelée phrase. Chaque phrase se compose d'une à plusieurs instructions de test, le résultat de ces instructions étant appliqué à une ou plusieurs instructions d'action.
Une instruction occupe une ligne maximum. Chaque phrase commence par un point d'exclamation (généré automatiquement), elle peut comporter un commentaire et être repérée par une étiquette.

```
! (*Attente de séchage*)
%L2:
LD      %I0.1
AND    %M10
ST      %Q2.5
```

Commentaire

Le commentaire peut être intégré au début d'une phrase et peut occuper 3 lignes maximum (soit 222 caractères alphanumériques), encadrés de part et d'autre par les caractères **(*** et ***)**. Il facilite l'interprétation de la phrase à laquelle elle est affectée, mais n'est pas obligatoire.
Les commentaires s'affichent uniquement à partir de la première ligne de la phrase. En cas de suppression d'une phrase, le commentaire qui lui est associé est également supprimé.
Les commentaires sont mémorisés dans l'automate et sont accessibles à tout moment par l'utilisateur. A ce titre, ils consomment de la mémoire programme

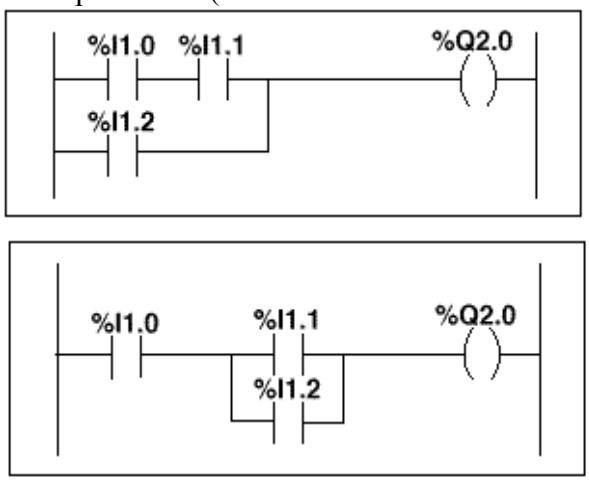
Étiquette

L'étiquette permet de repérer une phrase dans une entité de programme (programme principal, sous-programme, ...) mais n'est pas obligatoire.
Cette étiquette a la syntaxe suivante : %Li avec i compris entre 0 et 999 et se position en début d'une phrase.
Un repère d'étiquette ne peut être affecté qu'à une seule phrase au sein d'une même entité de programme.
Par contre il est nécessaire d'étiqueter une phrase afin de permettre un branchement après un saut de programme.
L'ordre des repères des étiquettes est quelconque, c'est l'ordre de saisie des phrases qui est prise en compte par le système lors de la scrutation.

Utilisation des parenthèses

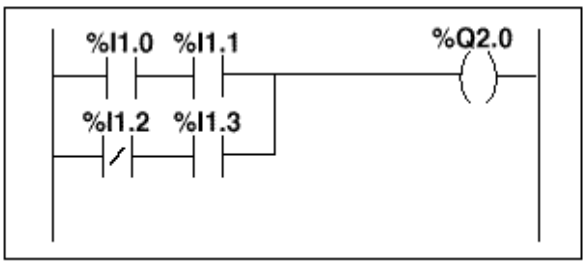
Les instructions AND et OR peuvent utiliser des parenthèses. Ces parenthèses permettent de réaliser des schémas à contacts de façon simple. L'ouverture de parenthèses est associée à l'instruction AND ou OR. La fermeture de parenthèse est une instruction, elle est obligatoire pour chaque parenthèse ouverte.

Exemple : AND(



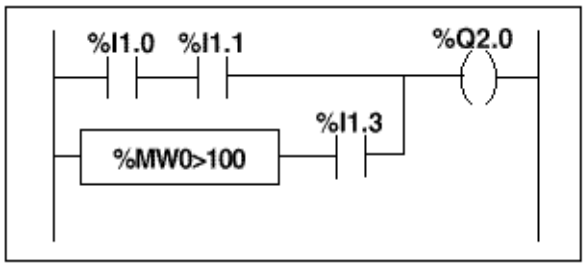

```
LD      %I1.0
AND     %I1.1
OR      %I1.2
ST      %Q2.0

LD      %I1.0
AND(    %I1.1
OR      %I1.2
)
ST      %Q2.0
```

Exemple : OR(




```
LD      %I1.0
AND     %I1.1
OR(N    %I1.2
AND     %I1.3
)
ST      %Q2.0
```

Aux parenthèses peuvent être associées les modificateurs :
• N négation, ex : AND(N ou OR(N,
• F front descendant (**F**alling edge), ex : AND(F ou OR(F,
• R front montant (**R**ising edge), ex : AND(R ou OR(R,
• [comparaison.

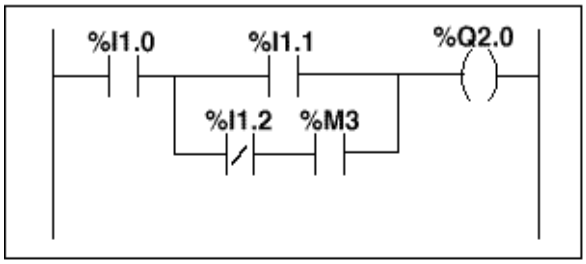



```
LD      %I1.0
AND     %I1.1
OR(     [%MW0>100]
AND     %I1.3
)
ST      %Q2.0
```

Imbrication de parenthèses

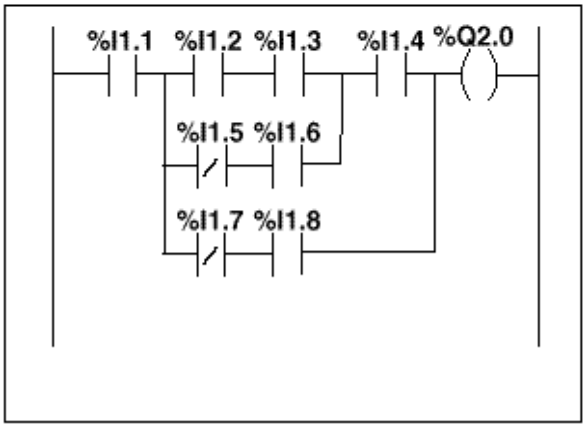
Il est possible d'imbriquer jusqu'à 8 niveaux de parenthèses.

Exemple




```
LD      %I1.0
AND(    %I1.1
OR(N    %I1.2
AND     %M3
)
)
ST      %Q2.0
```

Exemple




```
LD      %I1.1
AND(    %I1.2
AND     %I1.3
OR(N    %I1.5
AND     %I1.6
)
AND     %I1.4
OR(N    %I1.7
AND     %I1.8
)
)
ST      %Q2.0
```

Note :

• chaque parenthèse ouverte doit être impérativement refermée.

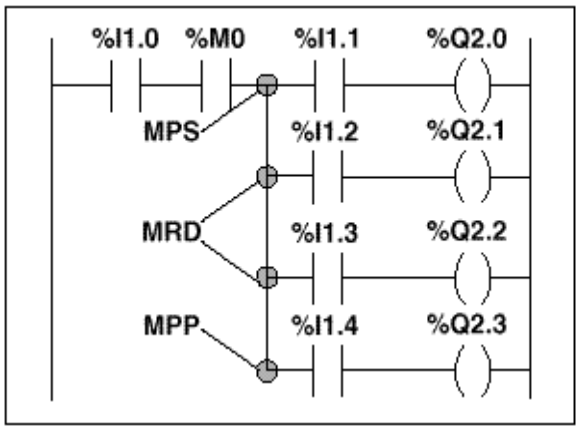
• les étiquettes %Li: ne doivent pas être placées dans des expressions entre parenthèses, ainsi que les instructions de saut JMP et d'appel à sous programme SRi,

• les instructions d'affectation ST, STN, S et R ne doivent pas être programmées entre parenthèses.

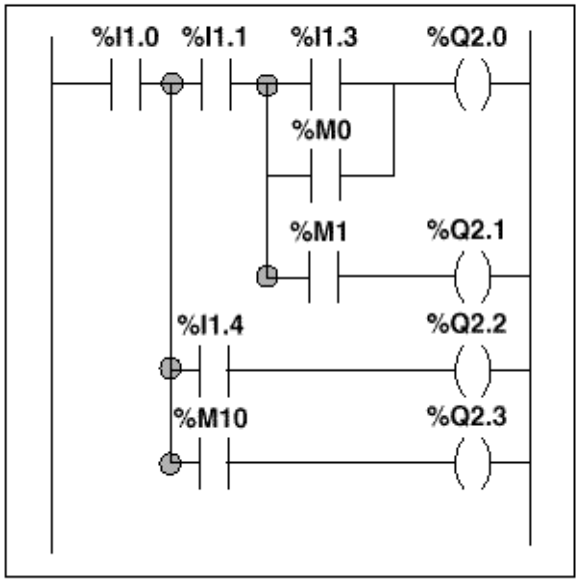
Instructions MPS, MRD, MPP

Les 3 types d'instruction permettent de traiter les aiguillages vers les bobines.
Ces instructions utilisent une mémoire intermédiaire appelée pile pouvant stocker jusqu'à 3 informations booléennes.
L'instruction MPS (Memory PuSh) a pour effet de stocker le résultat de la dernière instruction de test au sommet de la pile et de décaler les autres valeurs vers le fond de la pile.
L'instruction MRD (Memory ReaD) lit le sommet de la pile.
L'instruction MPP (Memory PoP) a pour effet de lire, de déstocker le sommet de la pile et de décaler les autres valeurs vers le sommet de la pile.

Exemples :




```
LD      %I1.0
AND     %M0
MPS
AND     %I1.1
ST      %Q2.0
MRD
AND     %I1.2
ST      %Q2.1
MRD
AND     %I1.3
ST      %Q2.2
MPP
AND     %I1.4
ST      %Q2.3
```




```
LD      %I1.0
MPS
AND     %I1.1
MPS
AND(    %I1.3
OR      %M0
)
ST      %Q2.0
MPP
AND     %M1
ST      %Q2.1
MRD
AND     %I1.4
ST      %Q2.2
MPP
AND     %M10
ST      %Q2.3
```