## I. Historique

Les automates programmables industriels sont apparus à la fin des années soixante, à la demande de l'industrie automobile américaine (GM), qui réclamait plus d'adaptabilité de leurs systèmes de commande.

Les coûts de l'électronique permettant alors de remplacer avantageusement les technologies actuelles.

- ***Avant*** **:** utilisation de relais électromagnétiques et de systèmes pneumatiques pour la réalisation des parties commandes ⇒ ***logique câblée***

  *Inconvénients* : cher, pas de flexibilité, pas de communication possible.

- ***Solution*** **:** utilisation de systèmes à base de microprocesseurs permettant une modification aisée des systèmes automatisés ⇒ ***logique programmée***

Les ordinateurs de l'époque étant chers et non adaptés aux contraintes du monde industriel, les automates devaient permettre de répondre aux attentes de l'industrie.

- **Contraintes du monde industriel :**
  - Influences externes :
    - *poussières,*
    - *température,*
    - *humidité,*
    - *vibrations,*
    - *parasites électromagnétiques, …*
  - Personnel :
    - *mise en œuvre du matériel aisée (pas de langage de programmation complexe)*
    - *dépannage possible par des techniciens de formation électromécanique*
    - *possibilité de modifier le système en cours de fonctionnement*
  - Matériel :
    - *évolutif*
    - *modulaire*
    - *implantation aisée*

- **Domaines *d'emploi des automates :***

On utilise les API dans tous les secteurs industriels pour la *commande des machines* (convoyage, emballage...) ou des *chaînes de production* (automobile, agroalimentaire ...) ou il peut également assurer des fonctions de *régulation de processus* (métallurgie, chimie ...).

Il est de plus en plus utilisé dans le domaine du *bâtiment* (tertiaire et industriel) pour le contrôle du chauffage, de l'éclairage, de la sécurité ou des alarmes.

- **Nature des informations traitées par l'automate**

Les informations peuvent être de type :

- ***Tout ou rien*** (T.O.R.) : l'information ne peut prendre que deux états (vrai/faux, 0 ou 1 …). C'est le type d'information délivrée par un détecteur, un bouton poussoir …
- ***Analogique*** : l'information est continue et peut prendre une valeur comprise dans une plage bien déterminée. C'est le type d'information délivrée par un capteur (pression, température …)
- ***Numérique*** : l'information est contenue dans des mots codés sous forme binaire ou bien hexadécimale. C'est le type d'information délivrée par un ordinateur ou un module intelligent.

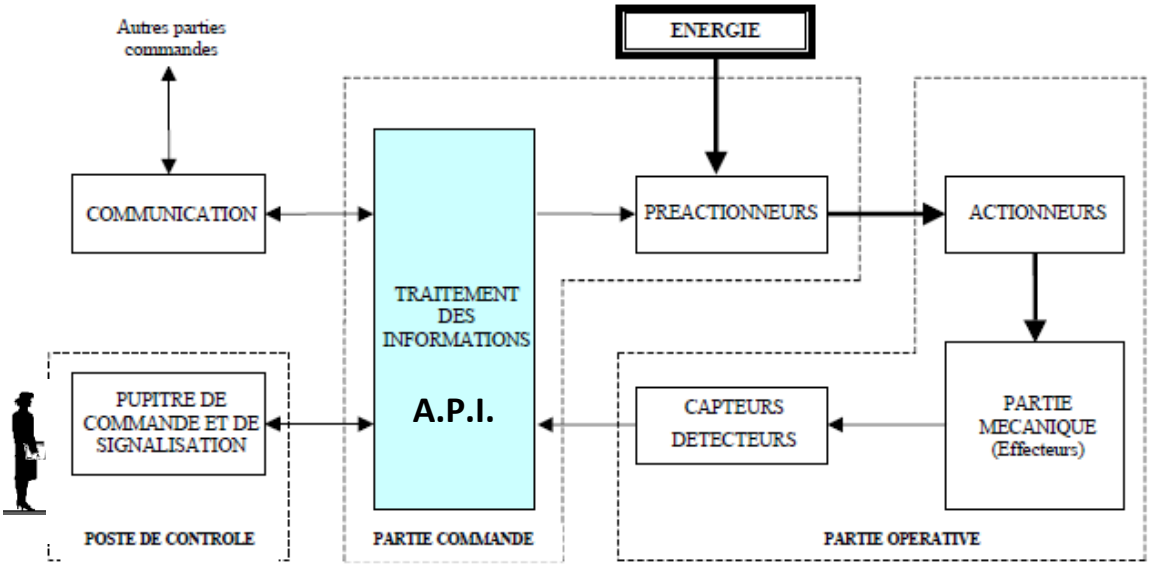
## II. Définition

L'Automate Programmable Industriel (API) est un appareil électronique programmable, adapté à l'environnement industriel, qui réalise des fonctions d'automatisme pour assurer la commande de préactionneurs et d'actionneurs à partir d'informations logique, analogique ou numérique.

## III. Place de l'API dans le système automatisé de production (S.A.P.)

Le système automatisé est soumis à des *contraintes* : énergétiques, de configuration, de réglage et d'exploitation qui interviennent dans tous les modes de marche et d'arrêt du système.

Les automates occupent la partie commande d'où émergent les ordres de fonctionnement vers la partie opérative.




- Le poste de contrôle, composé des pupitres de commande et de signalisation, permet à l'opérateur de ***commander le système*** (consignes de marche, arrêt, départ cycle …).

Il permet également de ***visualiser*** les différents états du système à l'aide de voyants, de terminal de dialogue ou d'interface homme-machine (IHM).

- Le bloc de traitement des informations ***reçoit les consignes*** du pupitre de commande (opérateur) et les ***informations*** de la partie opérative transmises par les capteurs/détecteurs.

En fonction de ces *consignes* ***et*** du *programme* de gestion des tâches implanté dans l'automate programmable, il va ***commander les préactionneurs*** et ***renvoyer des informations au pupitre*** de signalisation ou à d'autres systèmes de commande et/ou de supervision en utilisant un réseau et un protocole de communication.

- Les *préactionneurs*, permettent de ***commander*** **les** ***actionneurs*** et ***assurer le transfert d'énergie*** entre la source de puissance (réseau électrique, pneumatique …) et les actionneurs. Exemple : contacteur, distributeur …

## IV. Architecture des automates

### IV.1. Aspect extérieur

Les automates peuvent être de type *compact* ou *modulaire*.

- De type ***compact***, on distinguera les *modules de programmation* (LOGO de Siemens, ZELIO de Schneider, MILLENIUM de Crouzet ...) des ***microautomates***. Il intègre le processeur, l'alimentation, les entrées et les sorties. Selon les modèles et les fabricants, il pourra réaliser certaines fonctions supplémentaires (comptage rapide, E/S analogiques ...) et recevoir des extensions en nombre limité. Ces automates, de fonctionnement simple, sont généralement destinés à la commande de petits automatismes.


Automate monobloc
TSX Nano

Les automates ***monobloc*** comprennent en général :

1 - Une prise pour raccordement du terminal permettant la programmation à partir d'une console ou d'un ordinateur.

2 et 3 - Des prises pour des fonctions de communication spécialisées avec d'autre équipement.

4- Ecran sur lequel on peut visualiser: - l'état des entrées et sorties relais "TOR" - l'état automate.

5 - Un raccordement de l'alimentation secteur.

6 - Une alimentation capteurs (en général 24 VDC/150mA).

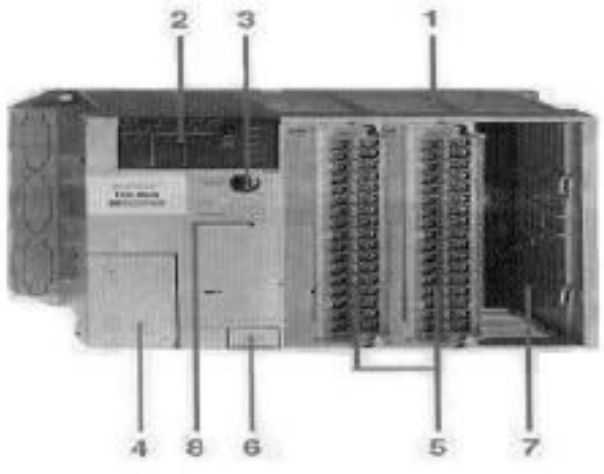
7 - Un raccordement des capteurs d'entrées.

8 - Un raccordement des pré actionneurs de sorties.

9 - Un raccordement pour des entrées spécifiques.

10 - Un cache amovible pour protection des borniers à vis.

- De type ***modulaire***, le processeur, l'alimentation et les interfaces d'entrées/sorties résident dans des unités séparées (***modules***) et sont fixées sur un ou plusieurs ***racks*** contenant le "fond de panier" (bus plus connecteurs). Ces automates sont intégrés dans les automatismes complexes où puissance, capacité de traitement et flexibilité sont nécessaires.




Automate
modulaire TSX37-08

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

## IV.2. Structure interne

### IV.2.1. Module d'alimentation :

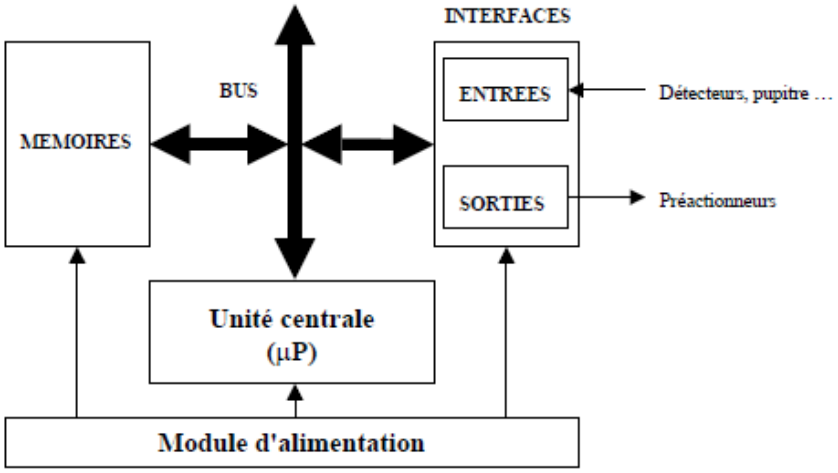
Composé de blocs qui permettent de fournir à l'automate l'énergie nécessaire à son fonctionnement. Cette énergie sera dimensionnée en fonction des consommations des différentes parties. En règle générale, un voyant positionné sur la façade indique la mise sous tension de l'automate. On distingue trois sources d'alimentation : alimentation *alternative*, *continue* ou *auxiliaire*.

- ***L'alimentation alternative***: délivre, à partir du secteur 110Vou 220V, les principales tensions dont l'automate a besoin : 24V, 12V ou 5V en continu. Ces alimentations comportent deux parties : le redressement et la régulation. En cas d défaillance de l'automate, certaines alimentations sont dotées d'un contact à relais et d'un voyant.
- ***L'alimentation continue***: certains automates fonctionnent à partir de tensions continues fournies par des alimentations externes. Elle nécessite l'utilisation de transformateurs à enroulements séparés.
- ***Les alimentations auxiliaires*** : Lorsque l'alimentation principale ne peut pas assurer le fonctionnement de toutes les cartes, il faut prévoir des alimentations auxiliaires. Celles–ci sont implantées soit à l'intérieur du coffret, soit dans des racks d'extension, soit dans les colonnes d'interface, ou encore à l'extérieur de l'automate.

**IV.2.2. Unité centrale** : à base de **microprocesseur**, elle réalise toutes les fonctions logiques, arithmétiques et de traitement numérique (transfert, comptage, temporisation ...).

**IV.2.3. Le bus interne** : il permet la communication de l'ensemble des blocs de l'automate et des éventuelles extensions.

**IV.2.4. Mémoires** : Elles permettent de stocker le système d'exploitation (ROM ou PROM), le programme (EEPROM) et les données système lors du fonctionnement (RAM). Cette dernière est généralement secourue par pile ou batterie. On peut, en règle générale, augmenter la capacité mémoire par adjonction de barrettes mémoires type PCMCIA.



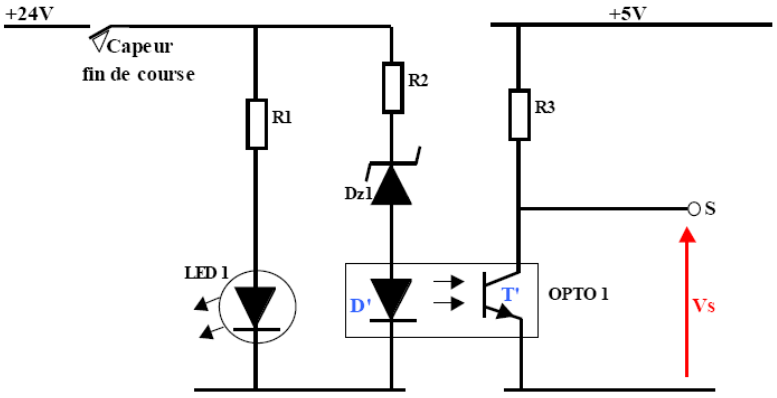

**IV.2.5. Interfaces d'entrées/sorties**

**IV.2.5.1. Interface d'entrée** : Elle permet de recevoir les informations du S.A.P. ou du pupitre et de mettre en forme (filtrage, ...) ce signal tout en l'isolant électriquement (optocouplage).

**a. Interfaces d'entrées TOR**

Elles sont destinées à :

- Recevoir l'information en provenance des capteurs logiques tels que les boutons poussoirs, les pressostats, thermostats, fins de course, capteurs de proximité inductifs ou capacitifs, capteurs photo-électriques, roues codeuses etc.
- Assurer l'adaptation, l'isolement, le filtrage et la mise en forme des signaux électriques. Une diode électroluminescente située sur la carte donne l'état de chaque entrée. Le nombre d'entrées sur une carte est de : 4, 8, 16, 32.



➢ **Fonctionnement**:

Lors de la fermeture du capteur ;

- LED1 signale que l'entrée automate est actionnée
- La led D' de l'optocoupleur s'éclaire
- Le photo transistor T' de l'optocoupleur devient passant
- La tension Vs=Vce = 0V (transistor saturé)

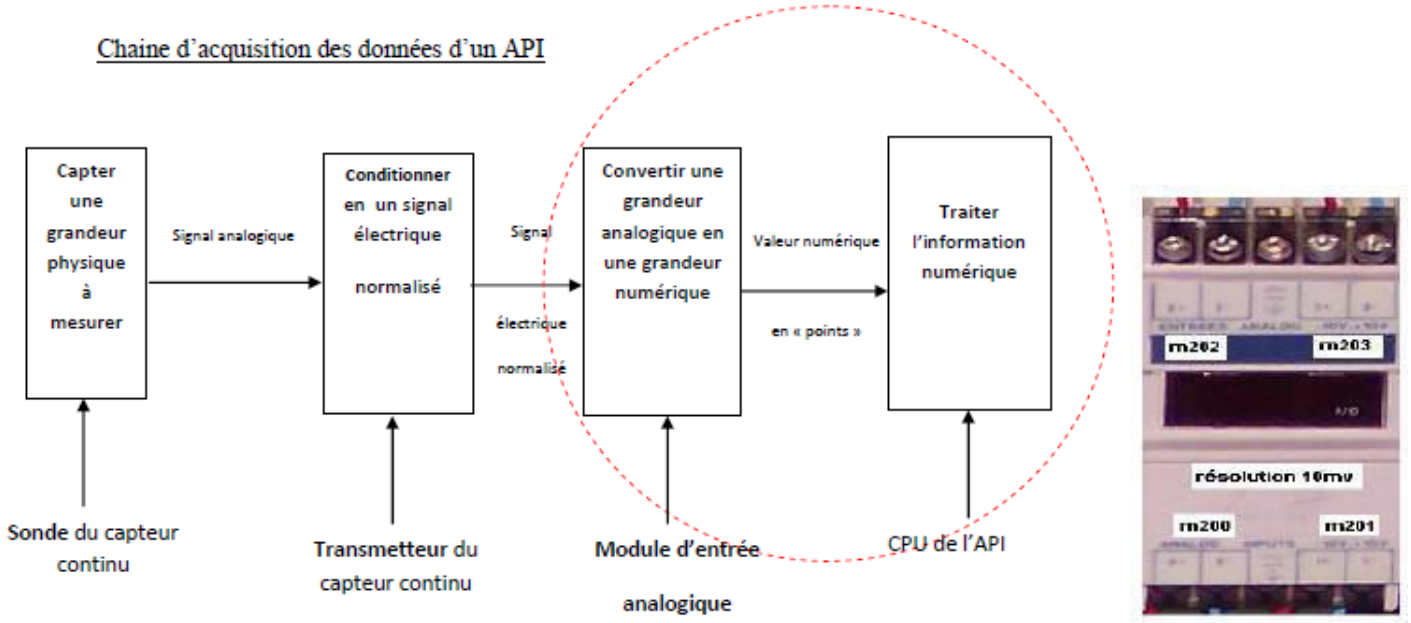
Donc lors de l'activation d'une entrée automate, l'interface d'entrée envoie ***un 0 logique*** à l'unité de traitement et ***un 1 logique*** lors de l'ouverture du contact du capteur (entrée non actionnée).

**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'entrée est convertie en une variation :

- En tension : de 0V, à 10V,
- En intensité : de 0 mA à 20 mA, ou de 4 mA à 20 mA.

***Exemple*** : Transmission de mesure de température effectuée par une sonde PT Sur le marché, il existe des modules à 2, 4, 8 voies d'entrées. Une carte standard assure jusqu'à 8 connections.




***Exemple*** : Module d'entrée CAN du TSX 17

Ce module (appelé aussi d'**entrées analogiques**) est muni de 4 entrées (0-15v) traduisant le résultat dans 4 mots respectifs (m200 à m203), la résolution est de **10mv.**

**c. Les cartes d'entrées numériques**

Elles permettent de dialoguer directement avec les roues codeuses. Certaines sont dotées d'un nombre fixe de voies .Chaque voie possède un nombre d'entrées qu'on organise selon ses besoins pour la constitution des mots en respectant les règles prescrites par le constructeur. Généralement ce sont des mots de 16 bits codés en BCD. (Binary Coded Decimal).

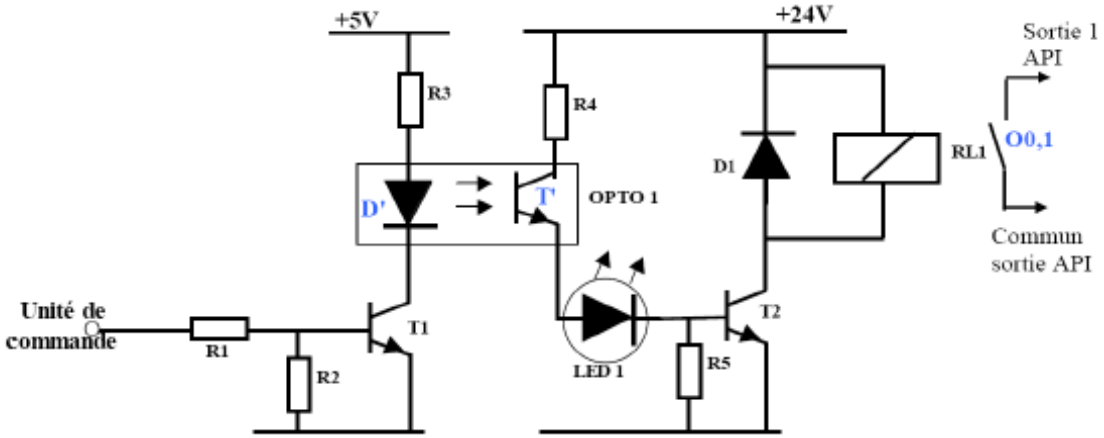
***Exemple*** : pour le nombre 127, il suffit de coder chacun des chiffres 1, 2 et 7 ce qui donne 0001, 0010, 0111.

**IV.2.5.2. Interface de sortie :** Elle permet de commander les divers préactionneurs et éléments de signalisation du S.A.P. tout en assurant l'isolement électrique.

**a. Interfaces de sorties TOR**

Elles sont destinées à :

- Commander les préactionneurs et éléments des signalisations du système
- Adapter les niveaux de tensions de l'unité de commande à celle de la partie opérative du système en garantissant une isolation galvanique entre ces dernières.



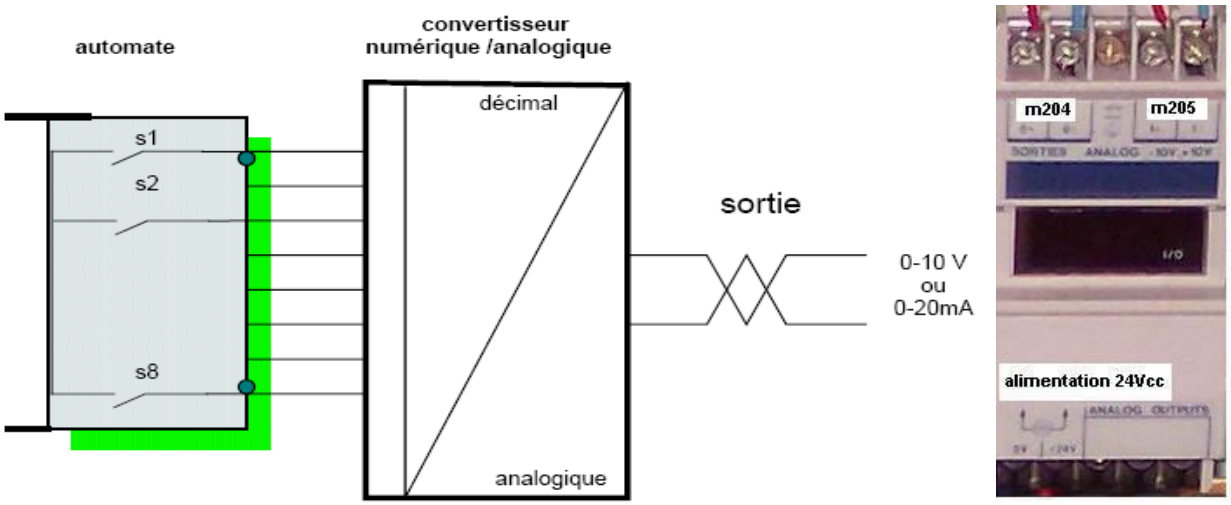

➢ **Fonctionnement :**

Lors de la commande d'une sortie automate ;

- L'unité de commande envoie un 1 logique (5V)
- T1 devient passant, donc D' s'éclaire
- Le photo transistor T' de l'optocoupleur devient passant
- LED 1 s'éclaire et nous informe de la commande de la sortie O0,1
- T2 devient passant
- La bobine RL1 devient sous tension et commande la fermeture du contact de la sortie O 0,1

Donc pour commander une sortie automate l'unité de commande doit envoyer un 1 logique pour actionner une sortie API et un 0 logique pour stopper la commande d'une sortie API

**b. Interfaces de sorties analogiques** Les conversions digitales /analogiques ont pour fonction de générer un signal analogique normalisé (0-10 V ; 0-20 mA) 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).




*Exemple* : Module de sortie analogique du TSX 17

Ce module (appelé aussi "de **sorties analogiques**") est muni de 2 voies (0-10 V) convertissant le résultat de 2 mots respectifs (**m204 à m205**) en une tension de 0 à 10 V, la résolution est de **10mv.**

**c. Les cartes de sorties numériques**

Elles permettent de dialoguer directement avec les afficheurs numériques. Comme pour le cas des entrées numériques, les sorties de ce type sont dotées d'un nombre fixe de voies. Chaque voie possède un nombre d'entrées qu'on organise selon ses besoins pour la constitution des mots, en respectant les règles prescrites par le constructeur. Généralement ce sont des mots de 16 bits codés en BCD.

## V. Fonctionnement

La plus part des automates programmables industriels ont un fonctionnement cyclique .Le processeur est géré en fonction d'un programme qui est une suite d'instructions placées en mémoire. Lorsque le fonctionnement est dit synchrone par rapport aux entrées et aux sorties, le cycle de traitement commence par la prise en compte des entrées qui sont figées en mémoire pour tout le cycle. Le processeur exécute alors le programme instruction par instruction en rangeant à chaque fois les résultats en mémoire. En fin de cycle les sorties sont affectées d'un état binaire, par mise en communication avec les mémoires correspondantes. Dans ce cas, le temps de réponse à une variation d'état d'une entrée peut être compris entre un ou deux temps de cycle (durée moyenne d'un temps de cycle est de 5 à 15 ms). Il existe d'autres modes de fonctionnement, moins courants :

- synchrone par rapport aux entrées seulement ;
- asynchrone.

## V.1. Jeu d'instructions

Le processeur peut exécuter un certain nombre d'opérations logiques; l'ensemble des instructions booléennes des instructions complémentaires de gestion de programme (saut, mémorisation, adressage ...) constitue un jeu d'instructions.

Sur une ligne de programme, on trouve systématiquement un *code d'instruction*, suivi éventuellement de *l'adresse de l'opérande* (variable) sur laquelle s'applique l'opération.

### V.2. Opérations logiques de base

Les opérations logiques de base sont :

- Lecture de l'état d'une variable (**Load**, **If**, ... etc);
- Et logique (**AND**);
- OU logique (**OR**, **+** ...);
- Affectation (**=**, **SET**, **OUT** ....);
- Négation (**NOT**, **Non**, **Pas**).

### V.3. Instructions complémentaires

Les instructions complémentaires sont la mémorisation, la temporisation, le comptage, le saut (avant ou arrière ou le deux),…

### V.4. Langages d'automates

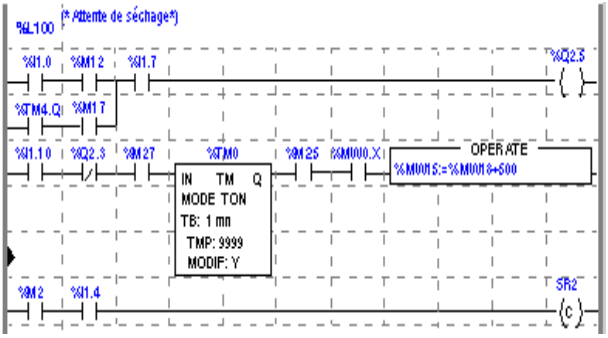
Les automates programmables industriels doivent pouvoir être utilisés facilement par du personnel habitué aux techniques classiques d'automatisation et peu à l'informatique. Ceci a conduit les constructeurs des API à concevoir des langages d'application spécialement adaptés à la réalisation d'automatisme.

On distingue:

- ***Les langages graphiques***
  - **LD** : **L**adder **D**iagram (Diagramme à contacts).

    Langage graphique développé pour les *électriciens*. Il utilise les symboles tels que: contacts, relais et blocs fonctionnels et s'organise en réseaux (labels).

    ***C'est le plus utilisé.***

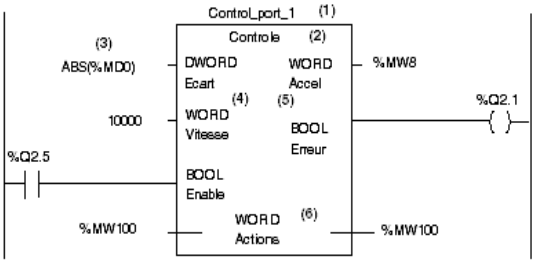



Le langage à relais (Ladder Diagram) est basé sur un symbolisme très proche de celui utilisé pour les schémas de câblage des systèmes à relais. Les symboles les plus utilisés sont donnés au tableau suivant :

| Fonction | Symbole | |
|---|---|---|
| | Européen | Américain |
| Contact normalement ouvert | ---o o--- | |
| Contact normalement fermé | ---o o--- | |
| Début de branchement | | |
| Fin de branchement | | |
| Affectation | ---( )--- | ---( ) |

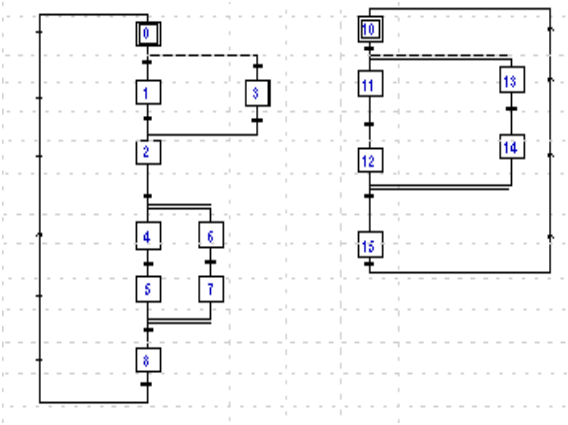
- **FBD** : **F**unction **B**lock **D**iagram (Logigrammes).

Langage graphique où des fonctions sont représentées par des rectangles avec les entrées à gauche et les sorties à droites. Les blocs sont programmés (bibliothèque) ou programmables.
***Utilisé par les automaticiens***.




- **SFC** : **S**equential **F**unction **C**hart (Grafcet).

Le GRAFCET, langage de spécification, est utilisé par certains constructeurs d'automate (Schneider, Siemens) pour la programmation. Parfois associé à un langage de programmation, il permet une programmation aisée des systèmes séquentiels tout en facilitant la mise au point des programmes ainsi que le dépannage des systèmes.
On peut également traduire un Grafcet en langage à contacts et l'implanter sur tout type d'automate.




- ***Les langages textuels***

- **IL** : **I**nstruction **L**ist (Liste d'instructions).

De même nature que l'assembleur (programmation des microcontrôleurs).
***Très peu utilisé par les automaticiens.***

```
! %L0 :
        LD          %I1.0
        ANDN        %M12
        OR (        %TM4.Q
        AND         %M17
        )
        AND         %I1.7
        ST          %Q2.5
```

- **ST** : **S**tructured **T**ext (Texte structuré).

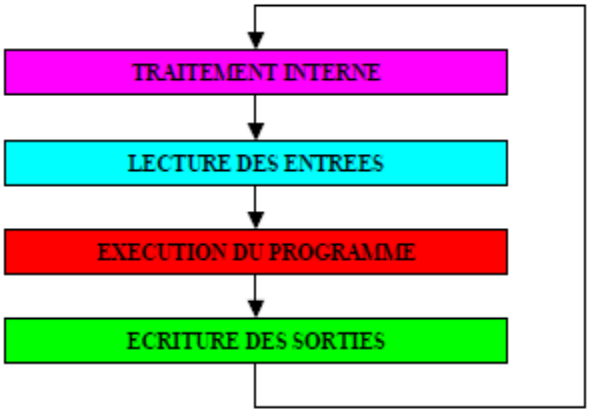
Langage informatique de même nature que le Pascal, il utilise les fonctions comme if ... then ... else ... (si ... alors ... sinon ...)
***Peu utilisé par les automaticiens***.

```
IF %M0 THEN
    FOR %M0099 := 0 TO 31 DO
        IF %M0100 [%M0099]< > 0 THEN
            %M0010 : =%M0100 [%M0099];
            %M0011 : =%M0099;
            %M1 : = TRUE;
            EXIT;                    (*Sortie de la boucle FOR*)
        ELSE
            %M1 : = FALSE;
```

## V.5. Logiciels de programmation

| Marque | Automate | Logiciel |
|---|---|---|
| **Télémécanique** | TSX Nano | PI707 |
| | TSX 3708,TSx22 | PI7- micro |
| | TSX Premium | PI7 junior |
| **ALENBRADLEY** | SLC 500 | APSF |
| **SIEMENS** | Serie 5:S5<br>Serie 7:S7 | Step 5<br>Step 7 |

## V.6. Traitement du programme automate




- ***Traitement interne*** : L'automate effectue des opérations de contrôle et met à jour certains paramètres systèmes (détection des passages en RUN/STOP, mises à jour des valeurs de l'horodateur, ...).
- ***Lecture des entrées*** : L'automate lit les entrées (de façon synchrone) et les recopie dans la mémoire image des entrées.
- ***Exécution du programme*** : L'automate exécute le programme instruction par instruction et écrit les sorties dans la mémoire image des sorties.
- ***Ecriture des sorties*** : L'automate bascule les différentes sorties (de façon synchrone) aux positions définies dans la mémoire image des sorties.

Ces quatre opérations sont effectuées continuellement par l'automate (fonctionnement cyclique).

On appelle ***scrutation*** l'ensemble des quatre opérations réalisées par l'automate et le ***temps de scrutation*** est le temps mis par l'automate pour traiter la même partie de programme. Ce temps est de l'ordre de la dizaine de millisecondes pour les applications standards.