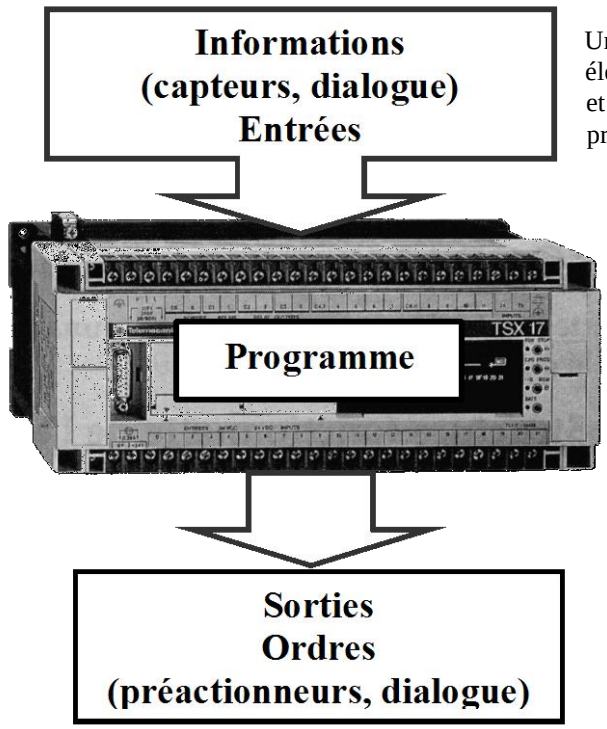
L'Automate Programmable Industriel

1. Définition

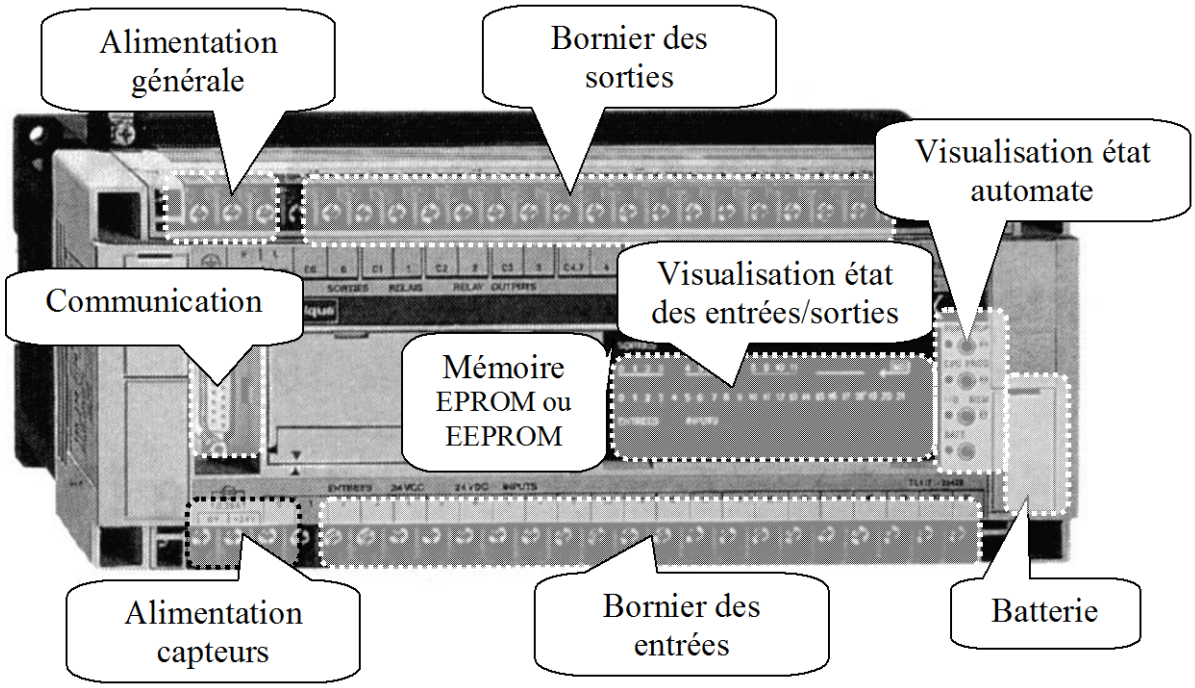


Un Automate Programmable Industriel (API) est une machine électronique programmable par un personnel non informaticien et destiné à piloter en ambiance industrielle et en temps réel des procédés ou parties opératives.

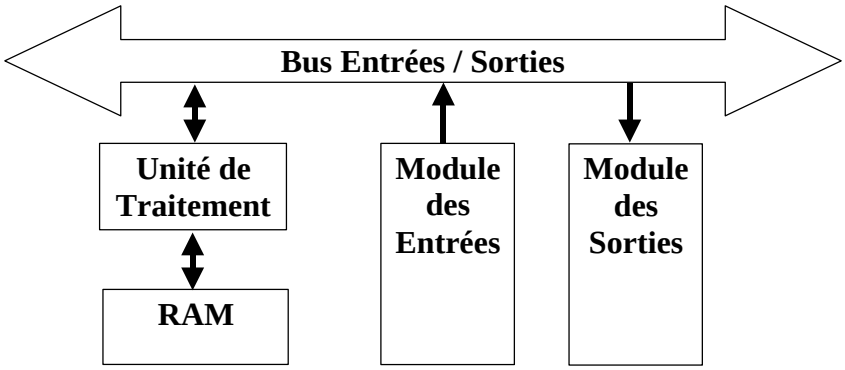
Traiter les informations entrantes pour émettre des ordres de sorties en fonction d'un programme.

Un automate programmable est adaptable à un maximum d'application, d'un point de vue traitement, composants, language.
C'est pour cela qu'il est de construction modulaire.

2. Structure générale



3. Principe de fonctionnement



Le traitement à lieu en quatre phases :

* **Phase 1 : Gestion du système**
* Autocontrôle de l'automate
* **Phase 2** : **Acquisition des entrées**

Prise en compte des informations du module d'entrées et écriture de leur valeur dans RAM (zone DONNEE).

* **Phase 3** : **Traitement des données**

Lecture du programme (située dans la RAM programme) par l'unité de traitement,
lecture des variables (RAM données), traitement et écriture des variables dans la RAM données.

* **Phase 4** : **Emissions des ordres**

Lecture des variables de sorties dans la RAM données et transfert vers le module de sorties.

4. Caractéristiques techniques

Les caractéristiques principales d'un API sont :

•Compact ou modulaire	•Sauvegarde (EPROM, EEPROM, pile, …)
•Tension d'alimentation	•Nombre d'entrées / sorties
•Taille mémoire	•Modules complémentaires (analogique, communication,..)
•Temps de scrutation	•Langage

4.1 Unité Centrale

L'unité centrale est le regroupement du processeur et de la mémoire centrale. Elle commande l'interprétation et l'exécution des instructions programmes. Les instructions sont effectuées les unes après les autres, séquencées par une horloge. Exemple: Si deux actions doivent être simultanées, l'API les traite successivement.

Caractéristiques principales :

- Vitesses de traitement : C'est la vitesse de l'UC pour exécuter 1 K-instructions logiques. (10 à 20 ms/Kmots).
- Temps de réponse : scrutation des entrées, vitesse de traitement et affectation des sorties.

4.2 Mémoire

Deux types de mémoire cohabitent :

* ***La mémoire Langage*** où est stocké le langage de programmation. Elle est en général figé, c'est à dire en lecture seulement. (ROM : mémoire morte)
* ***La mémoire Travail*** utilisable en lecture-écriture pendant le fonctionnement c'est la RAM (mémoire vive).

Attribution des zones mémoire travail en RAM

Nature des Inform.	Désignations	Exploitation	Zones Mémoires
Etats des Capteurs	Variable d'entrée	Evolution de leur valeur en fonction du déroulement du cycle	Zone mémoire des Données
Ordres aux préactionneurs	Variable de sortie		
Résultats de fonctions comptage, tempo...	Variable Interne et / ou Variable mot		
Résultats intermédiaires			
Instructions du cycle dans l'API	Programme	Ecrit 1 fois et lu à chaque scrutation	Zone mémoire PROGRAMME

* ***Sauvegarde :***

Sauvegarde de la RAM (programmes, configuration, données)		Sauvegarde Externe (programme, configuration)
1 heure minimum par pile interne	1an par pile externe	permanente par EPROM (effaçable par ultraviolet), EEPROM (effaçable par courant électrique)....

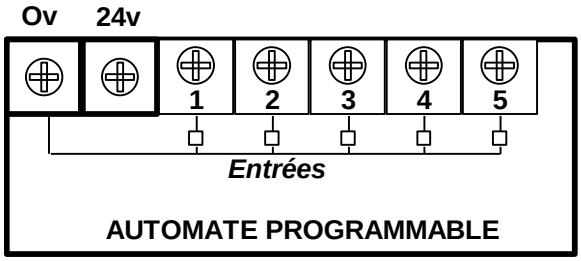
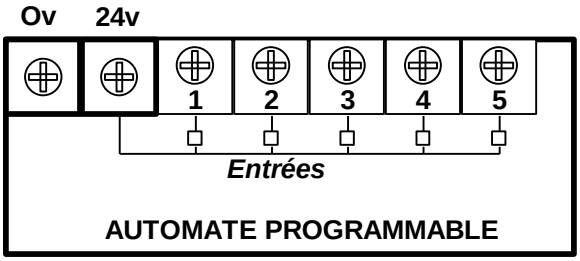
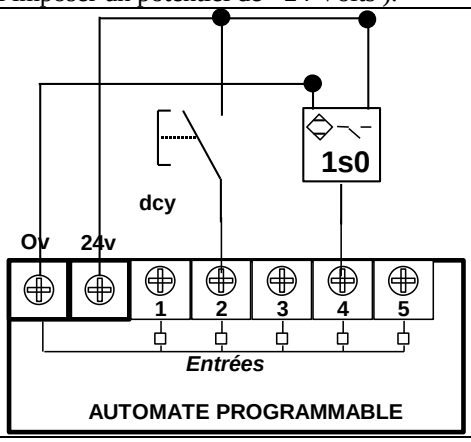
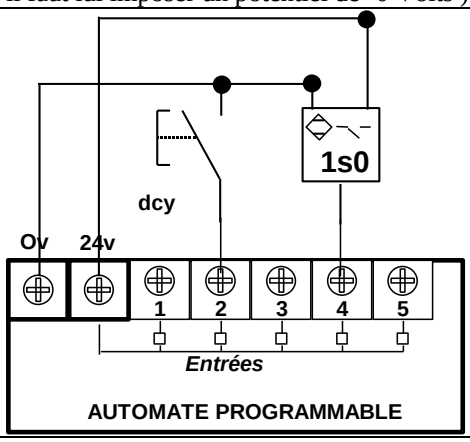
Le transfert de l'EPROM ou EEPROM vers la mémoire RAM de l'automate, s'effectue à chaque reprise secteur et si le contenu de celle-ci est différent.

4.3 *Les Modules Entrées - Sorties*

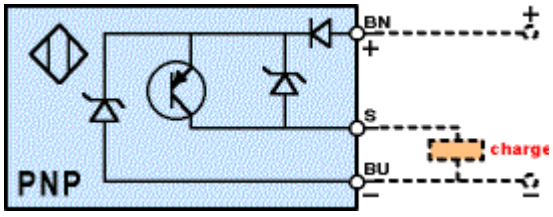
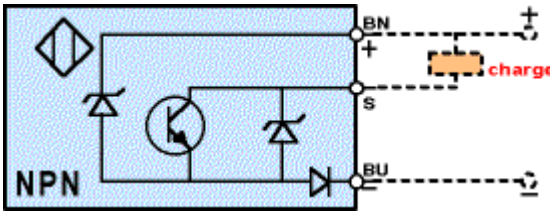
Module d'extension d'Entrées/Sorties TOR Module réseau : communication entre automate	Module d'extension d'Entrées Analogiques 0-10V Module d'extension de Sorties Analogiques 0-10V

4.3.1 Branchement des Entrées TOR

Le principe de raccordement consiste à envoyer un signal électrique vers l'entrée choisie sur l'automate dés que l'information est présente.
L'alimentation électrique peut être fourni par l'automate (en général 24V continu) ou par une source extérieure.
Un automate programmable peut être à **logique positive** ou **négative**.

Logique positive	Logique négative
Le commun interne des entrées est relié au 0V *Alimentation* Capteurs  EX : l'API TSX 17 fonctionne exclusivement en logique positive (pour mettre une entrée automate au 1 logique, il faut lui imposer un potentiel de +24 Volts).	Le commun interne des entrées est relié au 24V *Alimentation* Capteurs  EX : l'API PB15 fonctionne exclusivement en logique négative (pour mettre une entrée automate au 1 logique, il faut lui imposer un potentiel de -0 Volts).
	

Les détecteurs 3 fils ou électronique sont de deux types **PNP** ou **NPN**.

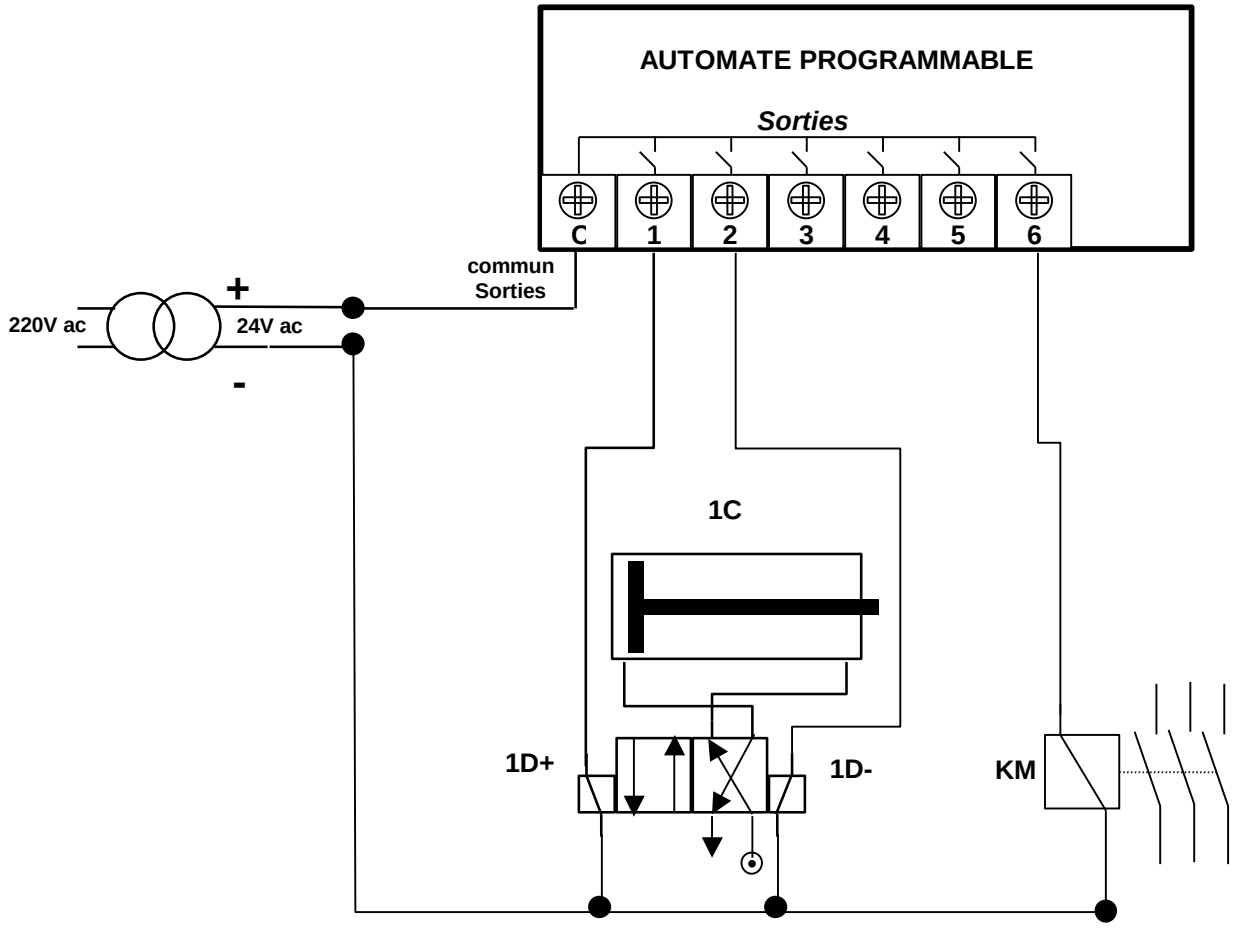
Détecteur PNP pour automate à logique Positive	Détecteur NPN Pour automate à logique Négative
	
Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel + sur la sortie **S** . La charge est branchée entre la sortie **S** et le potentiel - . Ce type de détecteur est adapté aux unités de	Lorsque qu'il y a détection, le transistor est passant (contact fermé). Il va donc imposer le potentiel - sur la sortie S . La charge est branchée entre la sortie S et le potentiel + . Ce type de détecteur est adapté aux unités de traitement qui fonctionnent en logique négative.

traitement qui fonctionnent en logique positive.	
Pour un automate programmable la charge représente l'entrée	

4.3.2 Branchement des sorties

Le principe de raccordement consiste à envoyer un signal électrique vers le préactionneur connecté à la sortie choisie de l'automate dés que l'ordre est émis.

L'alimentation électrique est fournie par une source extérieure à l'automate programmable.



4.4 Terminaux de programmation et de réglage

L'API doit permettre un dialogue avec :

* Le personnel d'étude et de réalisation pour réaliser la première mise en oeuvre (Edition programme, Transfert, Sauvegarde...)
* Le personnel de mise au point et de maintenance de réaliser des opérations sur le système (Forçage, Visualisation de l'état, Modification de paramètres temporisation, compteurs....)

Ce dialogue peut être réalisé par :

* Une Console : Elle sera utilisée sur site. Elle comporte un clavier, un écran de visualisation et le langage de programmation.
* Un Micro-ordinateur avec un logiciel d'assistance à la programmation : Il sera utilisé hors site. Il comprend plusieurs modules pour permettre l'édition, l'archivage, la mise au point des applications.

5. Mise en oeuvre

5.1 Préparation

La Partie Opérative du système, les grafcets de Production Normale, le Dialogue, le GEMMA (Modes de Marches et d'Arrêts), les GRAFCET de Sécurité et de Conduite étant définis, il reste à définir la Partie Commande.
Si le choix se porte sur un automate programmable, celui-ci étant relié aux préactionneurs (affectation Entrées/ Sorties) et ayant son propre langage de programmation, il faut traduire les GRAFCET précédents en un programme.

Tracer les GRAFCET adaptés à l'automate programmable.	⇒ Remplacer les réceptivités et les actions par les affectations des variables d'Entrées/Sorties ⇒ Modifier les structures GRAFCET si nécessaire en fonction des possibilités du langage de programmation. ⇒ Préparer la programmation pour les temporisations, les compteurs, les mémorisations d'action etc.. en respectant la syntaxe du langage de programmation.
Ecrire les équations de sorties	Recherche des conditions d'exécution des actions dans l'ensemble des grafcets et des équations logiques
Noter l'état initial des variables	Etapes actives au démarrage, mots de données pour tempo ou compteur)
Ecrire le programme.	Il existe 2 possibilités d'édition de Programme: ⇒ Ecrire le programme directement dans le langage programmable sur feuille de programmation. (Ex: Langage littéral booléen ou GRAFCET PB15 ou Langage Graphique Schéma à contact ou GRAFCET PL7-2 pour console TSX). Ecriture de l'ossature GRAFCET et des réceptivités, puis des équations de sorties. ⇒ Utiliser un logiciel d'assistance à la Programmation (en général GRAPHIQUE)exemple AUTOMGEN

REMARQUE: Le logiciel AUTOMGEN permet l'édition graphique proche des grafcets, puis l'affectation des entrées/sorties, la génération du programme pour l'automate concerné, la simulation du programme, le transfert et la supervision de son exécution.

5.2 Transfert du programme dans l'automate programmable

Le transfert du programme peut être fait soit :

* manuellement en entrant le programme et l'état initial à l'aide d'une console de programmation
* automatiquement en transférant le programme à l'aide du logiciel d'assistance, et en réalisant la liaison série entre l'ordinateur et l'automate.

5.3 Vérification du fonctionnement

Lors de sa première mise en oeuvre il faut réaliser la mise au point du système.

⇒ ***Prendre connaissance du système*** (dossier technique, des grafcets et du GEMMA, affectation des entrées / sorties, les schémas de commande et de puissance des entrées et des sorties).

⇒ ***Lancer l'exécution du programme*** (RUN ou MARCHE)

⇒ ***Visualiser l'état des GRAFCET, des variables...***

Il existe deux façons de vérifier le fonctionnement :

- En simulation (sans Partie Opérative).
- En condition réelle (avec Partie Opérative).

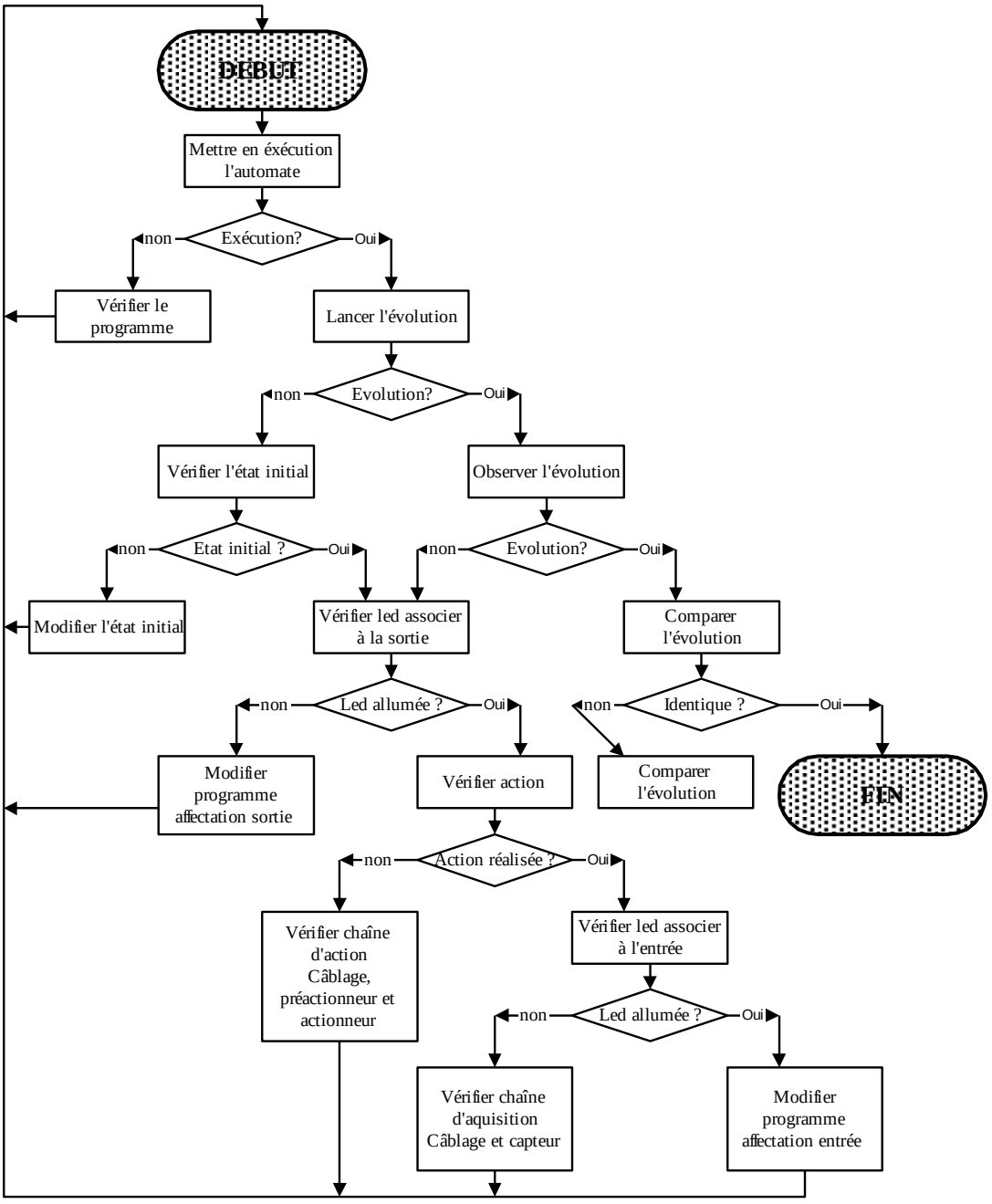
Simulation sans P.O.	**Condition réelle**
Le fonctionnement sera vérifié en simulant le comportement de la Partie Opérative, c'est à dire l'état des capteurs, en validant uniquement des entrées. ⇒ ***Valider les entrées correspondant à l'état initial (position) de la Partie Opérative.*** ⇒ ***Valider les entrées correspondant aux conditions de marche du cycle.*** ⇒ ***Vérifier l'évolution des grafcets (étapes actives).*** ⇒ ***Vérifier les ordres émis (Leds de sorties).*** ⇒ ***Modifier l'état des entrées en fonction des ordres émis (état transitoire de la P.O.).*** ⇒ ***Modifier l'état des entrées en fonction des ordres émis (état final de la P.O.).*** ⇒ **....** Toutes les évolutions du GEMMA et des grafcets doivent être vérifiées.	Le fonctionnement sera vérifié en suivant le comportement de la P.O. ⇒ ***Positionner la P.O. dans sa position initiale.*** ⇒ ***Valider les conditions de marche du cycle.*** ⇒ **Vérifier l'évolution des grafcets et le comportement de la P.O.** ⇒ **…** Toutes les évolutions du GEMMA et des grafcets doivent être vérifiées.

5.4 Recherche des dysfonctionnements

5.4.1 Causes de dysfonctionnements

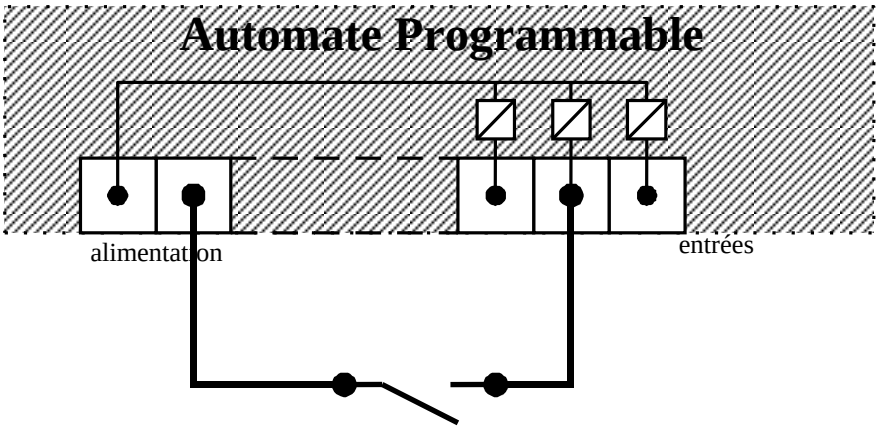
Un dysfonctionnement peut avoir pour origine :
- un composant mécanique défaillant (préactionneur, actionneur, détecteur,...).
- un câblage incorrect ou défaillant (entrées, sorties).
- un composant électrique ou électronique défectueux (interface d'entrée ou de sortie).
- une erreur de programmation (affectation d'entrées-sorties, ou d'écriture).
- un système non initialisé (étape, conditions initiales...).

5.4.2 Méthode de recherche des causes de dysfonctionnement



5.4.3 Vérification du câblage d'une entrée à masse commune

Cette vérification se réalise à l'aide d'un voltmètre.



5.4.4 Vérification du câblage d'une sortie à relais

Cette vérification se réalise à l'aide d'un voltmètre.