République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Superieure et de la Recherche Scientifique

Université Echahid Hamma Lakhdar d'El-Oued

FACULTE DE TECHNOLOGIE

DEPARTEMENT DE GENIE ELECTRIQUE

Mémoire de fin d'étude

Présenté pour l'obtention du diplôme de

MASTER ACADEMIQUE

Domaine : Sciences et Technologies

Filière :Electrotechnique

Spécialité : Reseau Electrique

Thème

Contribution Des Automate Programmables Au Contrôle Des Systèmes Industriels

Présenté par :

Mr:GUETARI Dhia eddine , Mr:CHERRAHI Khaled , Mr:MAOUCHE Elhachmi

Devant le jury copossède :

Dr.LABBI Yacine Encadreur

Mr.BELILA Khaled Co-Encadreur

2021-2022


بسم الله الرحمن الرحيم

Remerciements

Nous voudrions remercier du fond du cœur, avant tout, le bon Dieu qui Il nous a donnévolonté, courage et continuitédansnosétudes, Et nous avonsgardéjusqu'àceque nous arrivions à ceniveau, Nous exprimonsnotreprofonde gratitude. Nous remercionstousceux qui ontsupervisénotre supervision

Dr. Yacine Labbi et **Mr. Khaled Belila**

Pour l'honneurqu'elle nous a fait en acceptant de nous encadrerDansce travail et ceux qui ontcontribué à son aide et à ses efforts A prendre en charge en complément des diverses documentations Les moyensdisponibles pour atteindrenotre travail. Nous tenonségalement à remercier tout enseignant qui a contribué à Nous noussommesformés de l'écoleprimaire à l'université. Enfin, nous remercionsnosamisexceptionnels.

Dédicace

A mestrèschers parents
Je vousdoisceque je suisaujourd'huigrâce à votre amour, a
votre patience et vosinnombrables sacrifices.
Quecemodeste travail, soit pour vousune petite compensation
et reconnaissance enversce qui vousavez fait d'incroyable pour
moi.
Quedieu, tout puissant, vouspréserveetvous procure santé et
longue vie afinque je puisse à mon tour vouscombler.
A mestrèschèrestantes, mesoncles, mes frères, mes cousins et
mescousines.
Aucunedédicaceneseraitexprimerassezprofondément enceque
je ressenenversvous.
Je vousdirais tout simplement, un grand merci,jevousaime.
A mestrèschers

Ressemé

Dans ce mémoire, nous avons expliqué les automates programmables industriels et leur importance dans le domaine industriel, nous avons également expliqué comment convertir des graphes en langage de programmation Lader, puis nous avons programmé l'automate programmable PLC Schneider. Ces exemples sont : (Monte-charge industriel, élévateur à quatre étages, machine de mélange et de remplissage de liquide, pulvérisateur agricole ondulé automatique.)

Mots clés : Grafcet, API ,Zelio Soft 2

Abstract

In this memory, weexplain the industrial programmable controllers and their importance in the industrialfield. Wealsoexplained how wecanconvert graphs to the Laderprogramminglanguage, and thenwe program the PLC Schneider programmable logiccontroller. Theseexamples are: (Industrialgoods lift, four-storeyelevator, liquidmixing and filling machine, automaticwavy agricultural sprayer.)

Keywords: Grafcet API Zelio Soft 2

ملخص

في هذه المذكرة
،شرحنا المتحاكمات الصناعية القابلة للبرمجة و أهميتها في المجال الصناعي كما أننا أيضا شرحنا كيف نستطيع تحو يل الغرافسات إلى لغة برمجة Lader، ثم قمنا ببرمجة وحدة تحكم منطقية قابلة للبرمجة PLC Schneider.

هذه الأمثلة هي:

(رفع البضائع الصناعية، مصعد من أربعة طوابق، آلة خلط و تعبئة السوائل، البخاخ الزراعي المحوري الأوتوماتيكي.)

Liste Des Symboles

API:Automates Programmable Industrials

CPU: Central Processeur unit

AC:courant alternatif

DC: direct courant

ROM:mémoire mort

RAM:mémoire vive

PC: processeurconsole

TOR:Tout Ou Rien

LD:LadderDiagram

FBD:Function Block Diagram

LIST: langage de programmation textuelle

GRAFCET: Graphe **F**onctionnel de **C**ommande par **E**tapes et **T**ransitions

SFC: Séquentiel **F**onction **C**hart

PLC:PROGRAMMABLE LOGIC CONTROLLER

SFC: Séquentiel Fonction Chart

CIE: commission internationale d'électrotechnique

En: Input module

CAN: conversion d'une valeur analogique en une valeur numérique

BCD: Décimal code binare

ON:OUVRIR

OFF:FERMER

AI: **Artificiel intelligence**

CAN: conbersion analogique numérique

Liste Des Symboles

TTL: transistor-transistor logique

PID: proportionnel intégral dérivé

CIE: commission internationale d'électrotechnique

IL: liste d'instructions

SFC: Sequential Function Chart

PC: processeur ou console

PIC: Programmable Interrupt Controller

I:intree.

Q : sortie

Listes Des Figure

Listes Des Figure

Listes Des Figure

Listes Des Tableau

Sommaire

CHAPITRE III: Programation Des APIs

Introduction Générale

Introduction générale

L'objectif du développement de la recherche scientifique dans tous les domaines est d'aboutir àun progrès technologique voué principalement au bien être de l'humanité. Par sa nature dechercher à avoir la quantité, la qualité, le confort avec le minimum d'effort et du coût, l'hommea commencé à réfléchir, à concevoir et à réaliser des systèmes autonomes qui peuvent luiassurer tout dont il a besoin. Par conséquent, les outils et les appareils à énergies musculairesactionnés par des opérateurs sont remplacés par leurs équivalents à énergies électriques,mécaniques ou hydrauliques. La substitution d'opérateurs par ces systèmes définitl'automatisation.

Les progrès technologiques de ces dernières années ont abouti au développement des automates programmables industriels (En anglais : Programmable Logic Controller (PLC)) et à une révolution conséquente de l'ingénierie de contrôle/commande.

Actuellement, l'industrie des automates programmables connait un développement technologique remarquable, et suivant le bôme technologique des systèmes à microprocesseur. Ces avancées affectent non seulement la conception des automates programmables, mais aussi l'approche philosophique de l'architecture du système de contrôle. Ces améliorations, en point de vue matérielles, ont touchées :

- Le temps de traitement et d'exécution plus rapide.
- La structure et coût optimisées avec un nombre d'Entrées/Sorties assez important.
- Les interfaces intelligentes d'E/S basées sur microprocesseur ont étendu le traitement distribué.
- Les modules d'E/S distants (remote input/output modules). Comme les avancées matérielles, les avancées logicielles.
- Les petits automates ont reçu des instructions puissantes, qui élargissent le domaine d'application de ces petits automates.
- Les diagnostics et la détection des pannes ont été étendus à partir des diagnostics simples du système, qui diagnostiquent les dysfonctionnements de l'automate et celle du processus.
- Les calculs en virgule flottante ont permis d'effectuer des calculs complexes dans des applications de contrôle nécessitant un calcul statistique, un équilibrage...etc.

❖ Les instructions de traitement et de manipulation des données ont été améliorées et simplifiées afin de prendre en charge des applications complexes de contrôle et d'acquisition de données impliquant le stockage, le suivi et la récupération de grandes quantités de données.

Ce mémoire est une introduction à la programmation des automates, vise à faciliter la tâche des ingénieurs qui entrent en contact avec les automates pour la première fois.

A cet effet, le présent mémoire est divisé en quatre chapitre tels :

La première chapitre est une introduction sur les automates programmables industriels, leurs architectures, leurs fonctionnement, et les différentes caractéristiques des APIs...etc.

La deuxième chapitre sera consacrée à l'étude des différents types d'Entrées/Sorties, et leurs connexions...etc

La troisième chapitre sera consacrée à la programmation des API. Dans cette partie on a présenté les fonctions les plus utilisées tels que les fonctions logiques, temporisation, comptage, régulation...etc. Chaque fonction sera montrée en quatre langages Ladder, FBD , LIST et Grafcet.

Dans le dernier chapitre du mémoire, nous avons expliqué les grafcet et nous y avons appliqué des exemples de la réalité et l'avons converti en langage de programmation Lader, puis nous avons programmé un Automate Programmable Industriel API Schneider.

Ces exemples sont :

1- Monte-charge industriel.
2- Ascenseur quatre étages.
3- Machine de mélange et de remplissage de liquide.
4- Pulvérisateur agricole péristaltique automatique.

Chapitre I

Généralité sur Automates Programmables Industriels (API)

Chapitre I: Généralité sur Automates Programmables Industriels (API)

1- Introduction :

Les **Automates Programmables Industriels (API)** sont apparus aux Etats-Unis vers 1969 où ils répondaient aux désirs des industries de l'automobile de développer des chaînes de fabrication automatisées qui pourraient suivre l'évolution des techniques et des modèles fabriqués.

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 industriels. 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.

Il est en général manipulé par un personnel électromécanicien. Le développement de l'industrie à entraîner une augmentation constante des fonctions électroniques présentes dans un automatisme c'est pour ça que l'API s'est substitué aux armoires à relais en raison de sa souplesse dans la mise en œuvre, mais aussi parce que dans les coûts de câblage et de maintenance devenaient trop élevés. **[1]**

2-Définition:[16]

Les automates programmables ont de nombreuses définitions. Toutefois, ils peuvent être considérés en termes simples comme des ordinateurs industriels dotés des interfaces d'Entrées/Sorties et des fonctions spécialement conçues. La figure I-1 illustrée le diagramme conceptuel d'un système à base d'API

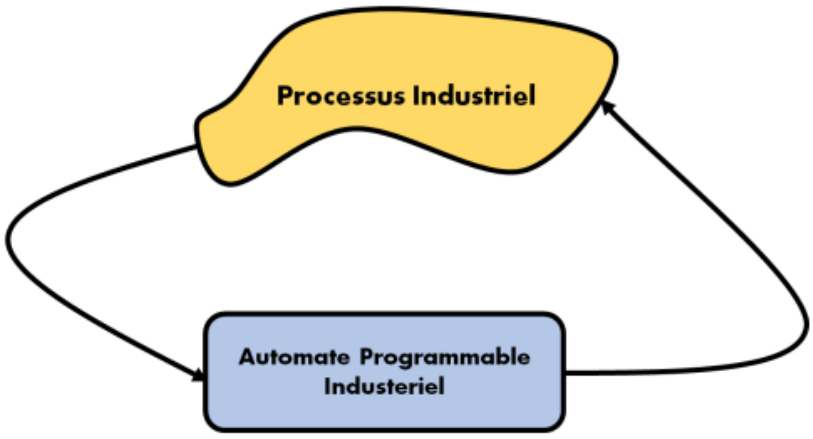


Figure I-1: System à base d'un API [16]

Aussi, il peut être défini comme une forme spéciale de microprocesseur qui utilise une mémoire programmable pour stocker des instructions et mettre en œuvre des fonctions telles que la **logique**, le **séquencement**, la **temporisation**, le **comptage** et résoudre les opérations**arithmétiques**aussi que la fonction de **communication** afin de contrôler des machines et des processus industrielles.
Les concepteurs des APIs ont préprogrammé les automates pour que le programme de contrôle puisse être entré d'une façon simple, plutôt d'une forme intuitive du langage. Les entrées (capteurs tels que les commutateurs,etc.) et les sorties (moteurs, vannes, etc.) du système sont connectées à l'automate chacune identifier par son propre adresse, figureI-2.

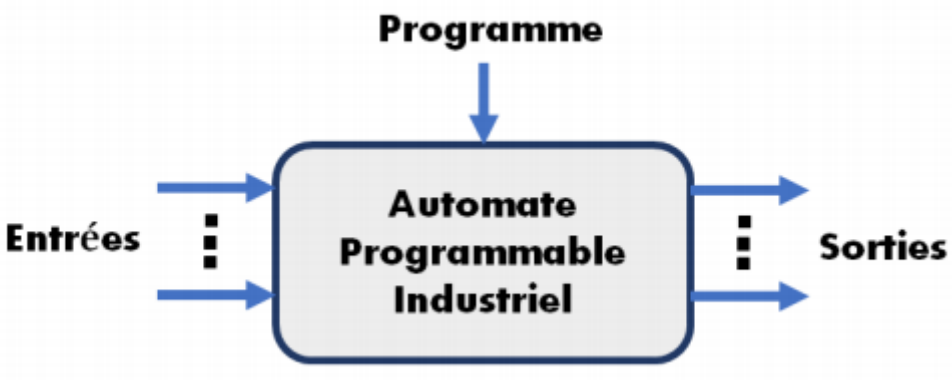


Figure I-2: Automate programmable industriel16]

Par la suite, l'opérateur introduit des séquences d'instructions, un programme, dans la mémoire de l'automate. Ce dernier surveillé ensuite les entrées et les sorties en fonction de ce programme et exécute les règles de contrôle pour lesquelles il a été programmé.

3- Structure interne d'un automate programmable industriel (API):[3]

Les API comportent quatre principales parties (Figure 4.4) :

- Une unité de traitement (un processeur CPU);
- Unemémoire ;
- Des modules d'entrées-sorties ;
- Des interfaces d'entrées-sorties ;

- Une alimentation 230 V, 50/60 Hz (AC) - 24 V (DC).

La structure interne d'un **automate programmable industriel** (API) est assez voisine de celle d'un système informatique simple, L'unité centrale est le regroupement du processeur et de la mémoire centrale. Elle commande l'interprétation et l'exécution des instructions programme. Les instructions sont effectuées les unes après les autres, séquencées par une horloge. *Deux types de mémoire cohabitent :*

- **La mémoire Programme** où est stocké le langage de programmation. Elle est en général figée, c'est à dire en lecture seulement. (ROM : mémoire morte)
- **La mémoire de données** utilisable en lecture-écriture pendant le fonctionnement c'est la RAM (mémoire vive). Elle fait partie du système entrées-sorties. Elle fige les valeurs (0 ou 1) présentes sur les lignes d'entrées, à chaque prise en compte cyclique de celle-ci, elle mémorise les valeurs calculées à placer sur les sorties[14].

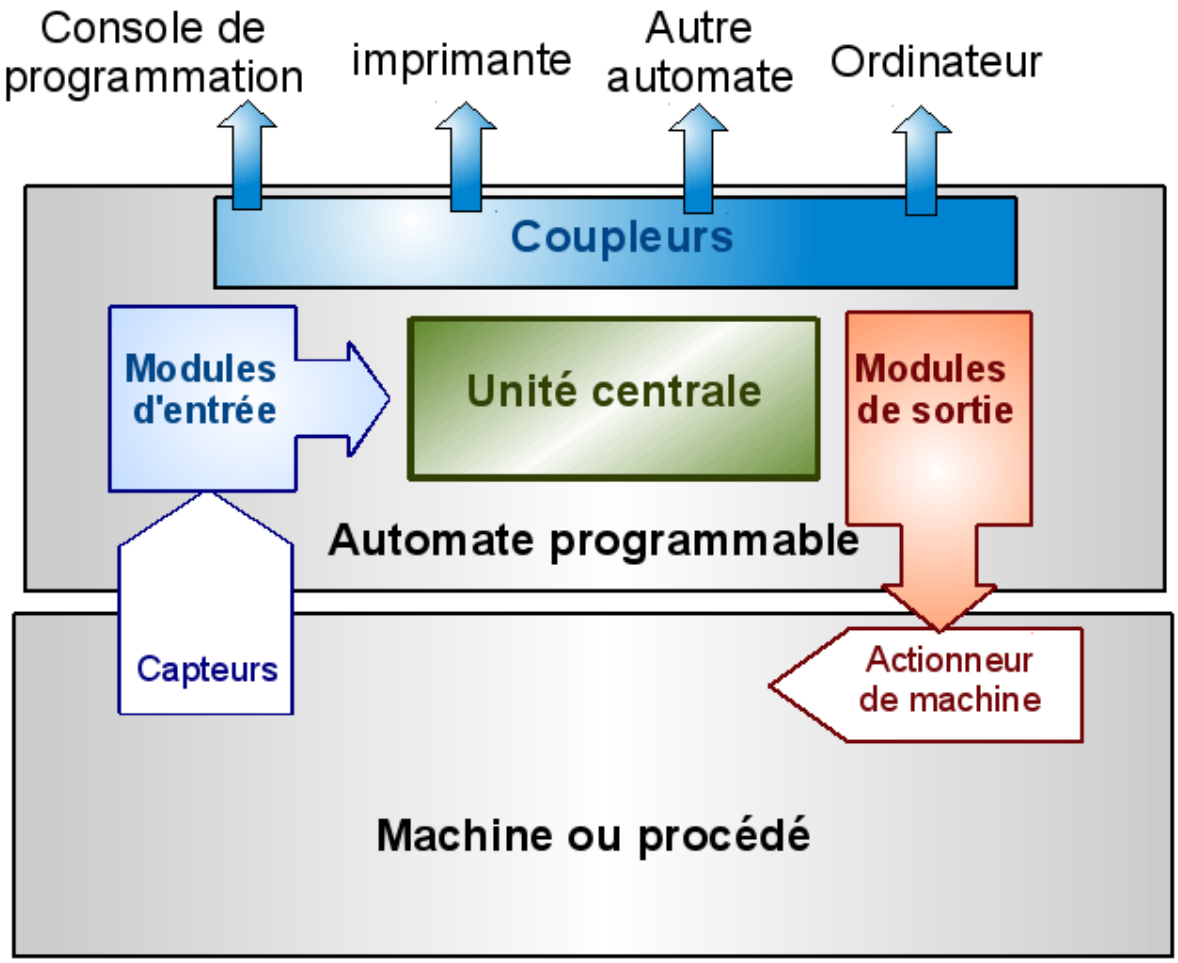


Figure I-3 : Structure interne d'un automates programmables industriels (API[5])

4- Architecture des automates:

a. Aspect extérieur :[3]

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 micro automates. 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. 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.

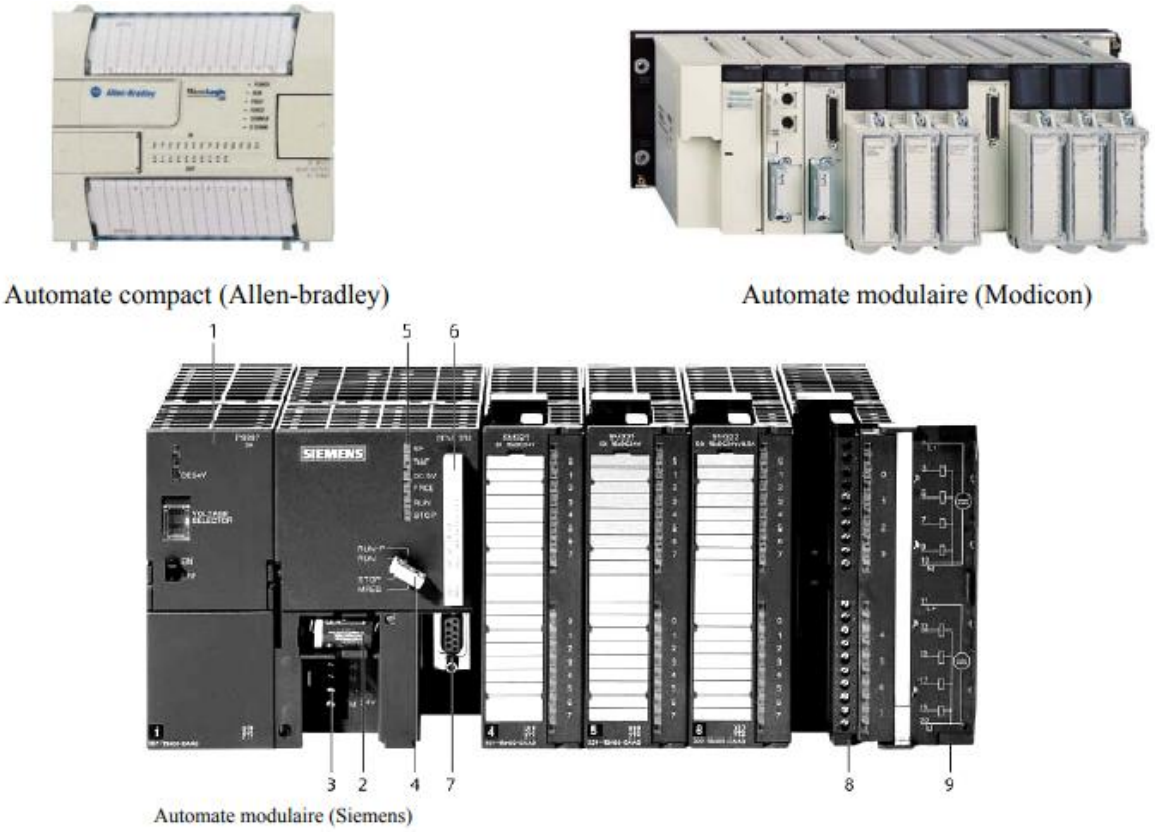


Figure I-4 : Les automates compact et modulaire.

1 Module d'alimentation 6 Carte mémoire

2 Pile de sauvegarde 7 Interface multipoint (MPI)

3 Connexion au 24V cc 8 Connecteur frontal

4 Commutateur de mode (à clé) 9 Volet en face avant

5 LED de signalisation d'état et de défauts

b. Structure interne :[4]

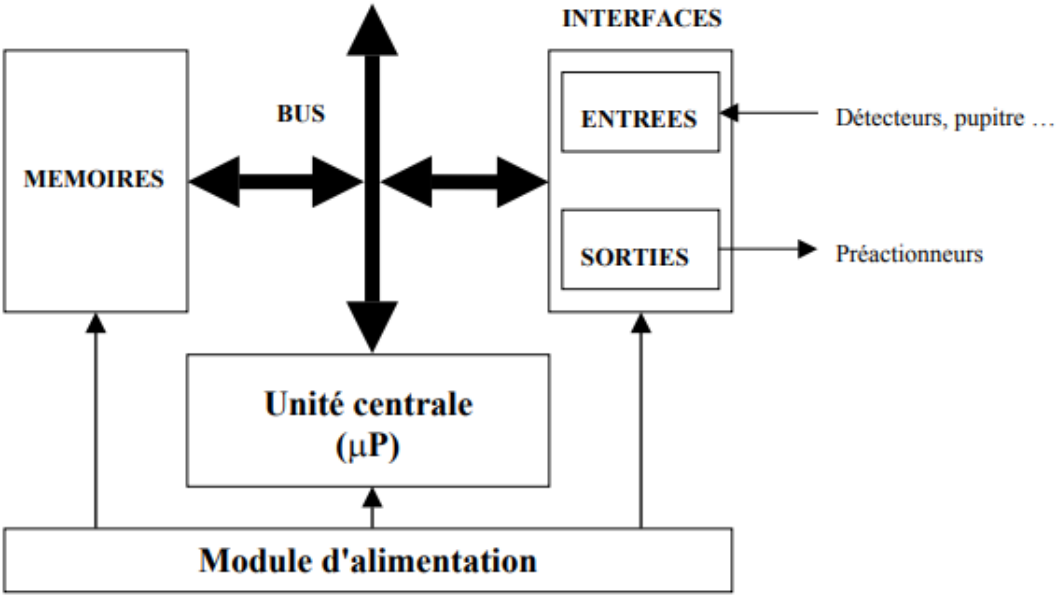


Figure I-5: Structure interne[7]

Module d'alimentation : il assure la distribution d'énergie aux différents modules.

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

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

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.

Interfaces d'entrées / sorties : *ƒ* 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). *ƒ* Interface de sortie : elle permet de commander les divers préactionneurs et éléments de signalisation du S.A.P. tout en assurant l'isolement électrique[5]

c. Fonctions réalisées :

Les automates compacts permettent de commander des sorties en T.O.R et gèrent parfois des fonctions de comptage et de traitement analogique. Les automates modulaires permettent de réaliser de nombreuses autres fonctions grâce à des modules intelligents que l'on dispose sur un ou plusieurs racks. Ces modules ont l'avantage de ne pas surcharger le travail de la CPU car ils disposent bien souvent de leur propre processeur.

Principales fonctions [2]:

Cartes d'entrées / sorties : Au nombre de 4, 8, 16 ou 32, elles peuvent aussi bien réaliser des fonctions d'entrées, de sorties ou les deux. Ce sont les plus utilisées et les tensions disponibles sont normalisées (24, 48, 110 ou 230V continu ou alternatif ...). Les voies peuvent être indépendantes ou posséder des "communs". Les cartes d'entrées permettent de recueillir l'information des capteurs, boutons ... qui lui sont raccordés et de la matérialiser par un bit image de l'état du capteur. Les cartes de sorties offrent deux types de technologies : les sorties à relais électromagnétiques (bobine plus contact) et les sorties statiques (à base de transistors ou de triacs).

Cartes de comptage rapide : elles permettent d'acquérir des informations de fréquences élevées incompatibles avec le temps de traitement de l'automate. Exemple : signal issu d'un codeur de position. Cartes de commande d'axe : Elles permettent d'assurer le positionnement avec précision d'élément mécanique selon un ou plusieurs axes. La carte permet par exemple de piloter un servomoteur et de recevoir les informations de positionnement par un codeur. L'asservissement de position pouvant être réalisé en boucle fermée. *ƒ* Cartes d'entrées / sorties analogiques : Elles permettent de réaliser l'acquisition d'un signal analogique et sa conversion numérique (CAN) indispensable pour assurer un traitement par le microprocesseur. La fonction inverse (sortie analogique) est également réalisée. Les grandeurs analogique sont normalisées : 0-10V ou 4-20mA.

Autres cartes :

- Cartes de pesage
- Cartes de communication (Ethernet ...)
- Cartes d'entrées / sorties déportées

5-Description des éléments d'un API:[6]

5-1- La mémoire:

Elle est conçue pour recevoir, gérer, stocker des informations issues des différents secteurs du système que sont le terminal de programmation (PC ou console) et le processeur, qui lui gère et exécute le programme. Elle reçoit également des informations en provenance des capteurs.

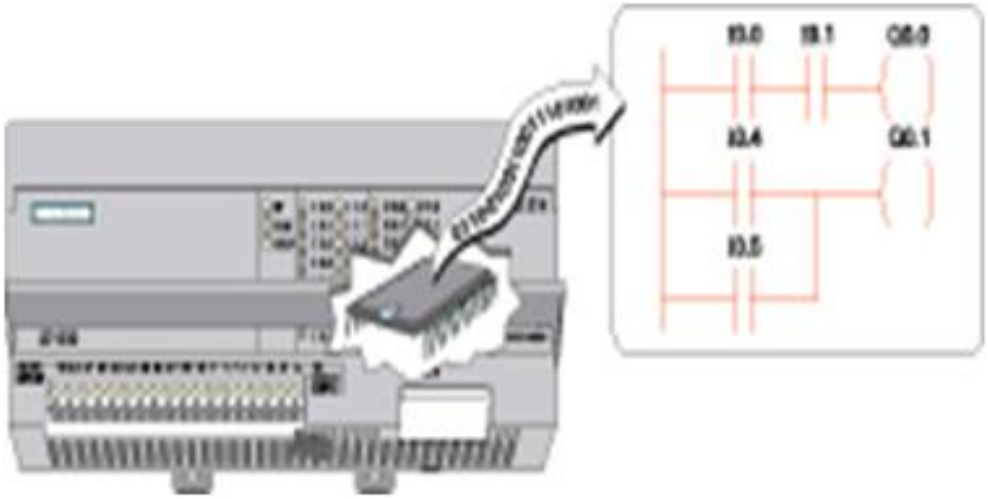


Figure I-6: La mémoire[5]

Il existe dans les automates deux types de mémoires qui remplissent des fonctions différentes :
- La mémoire Langage où est stocké le langage de programmation. Elle est en général figée, 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). Elle s'efface automatiquement à l'arrêt de l'automate (nécessite une batterie de sauvegarde).

Repartition des zones memoires :

- Table image des entrées

- Table image des sorties
- Mémoire des bits internes
- Mémoire programme d'application

5-2- Le processeur:[5]

Son rôle consiste d'une part à organiser les différentes relations entre la zone mémoire et les interfaces d'entrées et de sorties et d'autre part à exécuter les instructions du programme.

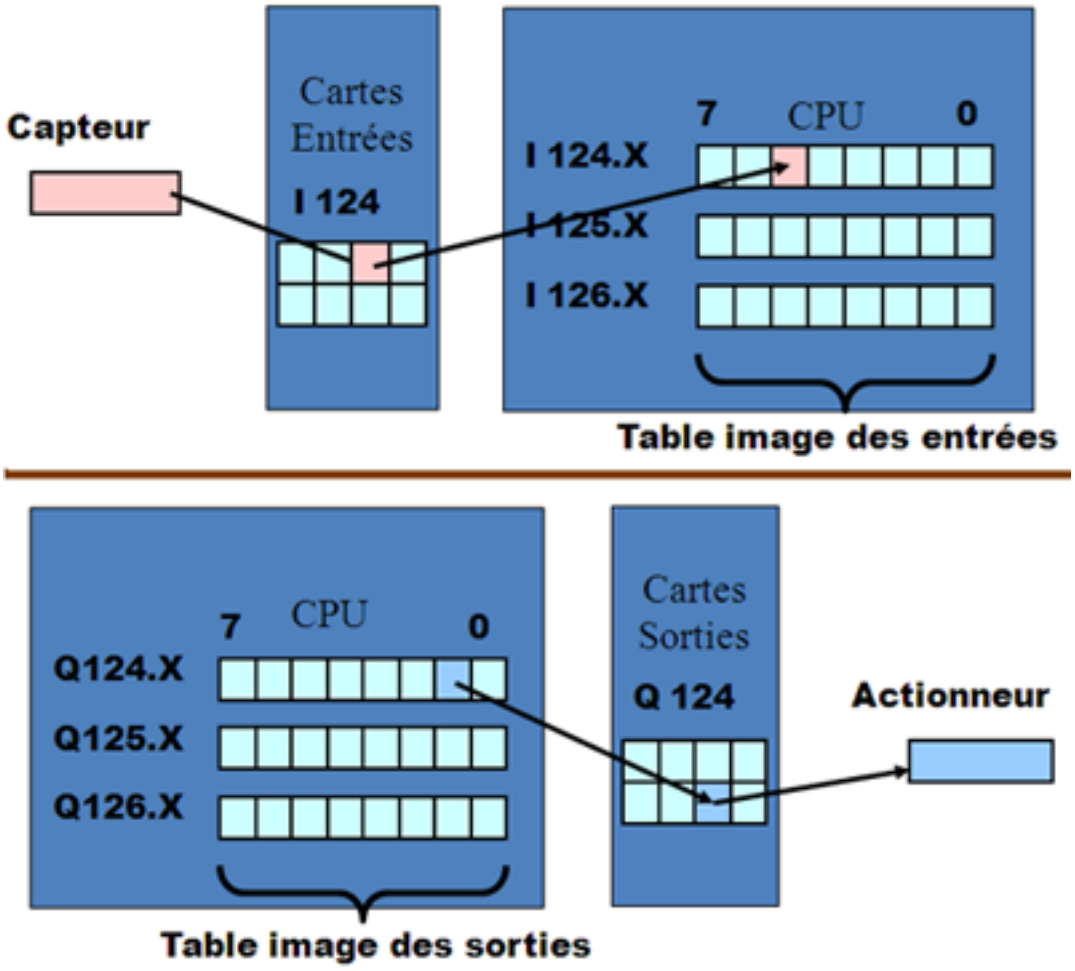
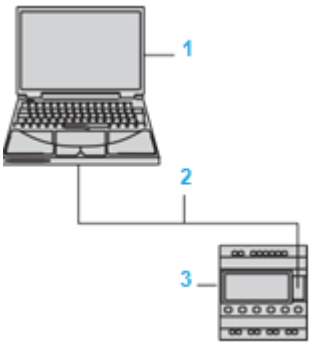
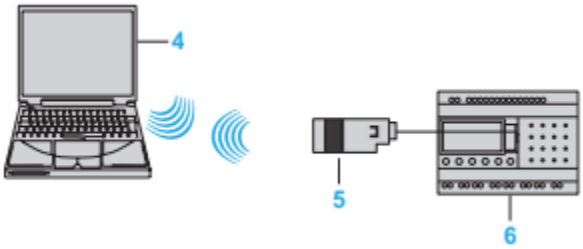
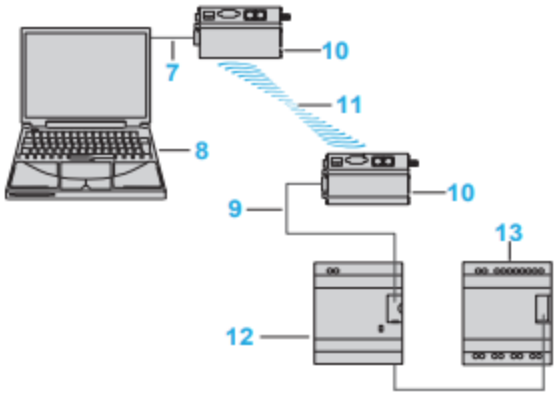


Figure I-7: Les interfaces d'entrées/sortie[7]

5-3- L'alimentation électrique:

Tous les automates actuels sont équipés d'une alimentation 240 V 50/60 Hz, 24 V DC. Les entrées sont en 24 V DC et une mise à la terre doit également être prévue.[5]

6-communication:[12]

Liaisonparcâble 	1 PC de programmation. 2 Câble USB (SR2USB01) ou câble liaison série (SR2CBL01) (1). 3 Module ZelioLogic compact ou modulaire.
Liaison par câble 	4 PC de programmation avec technologie Bluetooth intégrée (1). 5 Interface Bluetooth (SR2BTC01) pour module Zelio Logic (1). 6 Module ZelioLogic compact ou modulaire.
Liaison par Modem 	7 Câble de liaison PC-modem SR1CBL03 8 PC de programmation. 9 Câble de liaison Interface Modem fourni avec SR2COM01(1). 10 Modem d'émission/réception de données SR2MOD02 (1). 11 Liaison téléphonique ou radiophonique. 12 Interface de communication SR2COM01. 13 Module ZelioLogic compact ou modulaire.

7-Critères de choix d'un automate:

Le choix d'un automate programmable est en premier lieu le choix d'une société ou d'un groupe et les contacts commerciaux et expériences vécues sont déjà un point de départ. Les grandes sociétés privilégieront deux fabricants pour faire jouer la concurrence et pouvoir "se retourner" en cas de "perte de vitesse" de l'une d'entre elles. Le personnel de maintenance doit toutefois être formé sur ces matériels et une trop grande diversité des matériels peut avoir de graves répercussions. Un automate utilisant des langages de programmation de type GRAFCET est également préférable pour assurer les mises au point et dépannages dans les meilleures conditions. La possession d'un logiciel de programmation est aussi source d'économies (achat du logiciel et formation du personnel). Des outils permettant une simulation des programmes sont également souhaitables. Il faut ensuite quantifier les besoins :

-Nombre d'entrées / sorties : le nombre de cartes peut avoir une incidence sur le nombre de racks dès que le nombre d'entrées / sorties nécessaires devient élevé.

- Type de processeur : la taille mémoire, la vitesse de traitement et les fonctions spéciales offertes par le processeur permettront le choix dans la gamme souvent très étendue.

- Fonctions ou modules spéciaux : certaines cartes (commande d'axe, pesage ...) permettront de "soulager" le processeur et devront offrir les caractéristiques souhaitées (résolution, ...).

- Fonctions de communication : l'automate doit pouvoir communiquer avec les autres systèmes de commande (API, supervision ...) et offrir des possibilités de communication avec des standards normalisés (Profibus ...)[9]

8- Différents types de réseaux d'automates :[5]

8-1- Réseau en étoile:

Un centre de traitement commun échange avec chacune des autres stations. Deux stations ne peuvent pas échanger directement entre elles (Figure I-8). Exemple le réseau de terrain BITBUS de la société INTEL

Avantages :

- Grande vitesse d'échange.
- Différent types de supports de transmission.

- Pas de gestion d'accès au support.

Inconvénients :

- Coût global élevé.
- Evolutions limitées.
- Tout repose sur la station centrale.

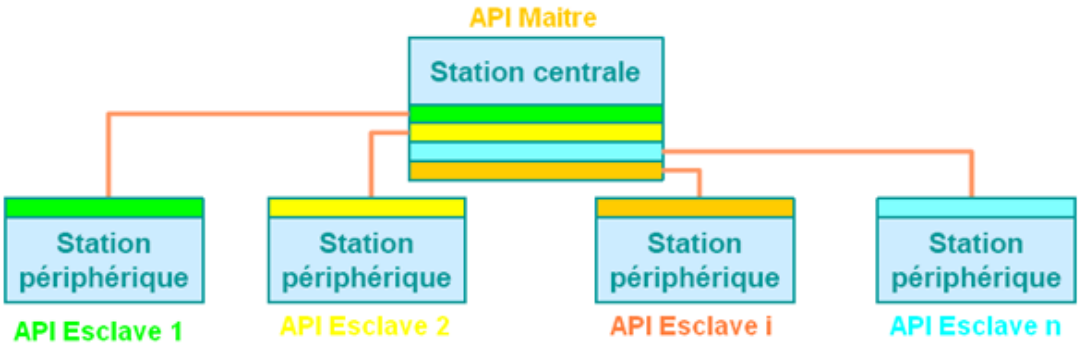


Figure I-8: Interconnexion par entrées/sorties déportées[5]

8-2- Réseau en anneau:

Chaque station peut communiquer avec sa voisine. Cette solution est intéressante lorsqu'une station doit recevoir des informations de la station précédente ou en transmettre vers la suivante (Figure I-9).

Figure I-9: Topologie Anneau[5]

Avantages :

- Signal régénérédoncfiable.
- Contrôle facile des échanges (le message revient à l'émetteur).

Inconvenient :

- Chaque station estbloquante.
- Une extension interrompe momentanément le réseau.

8-3- Réseau hiérarchisé:

C'est la forme de réseaux la plus performante. Elle offre une grande souplesse d'utilisation, les informations pouvant circuler entre-stations d'un même niveau ou circuler de la station la plus évoluée (en général un calculateur) vers la plus simple, et réciproquement (Figure I-10).

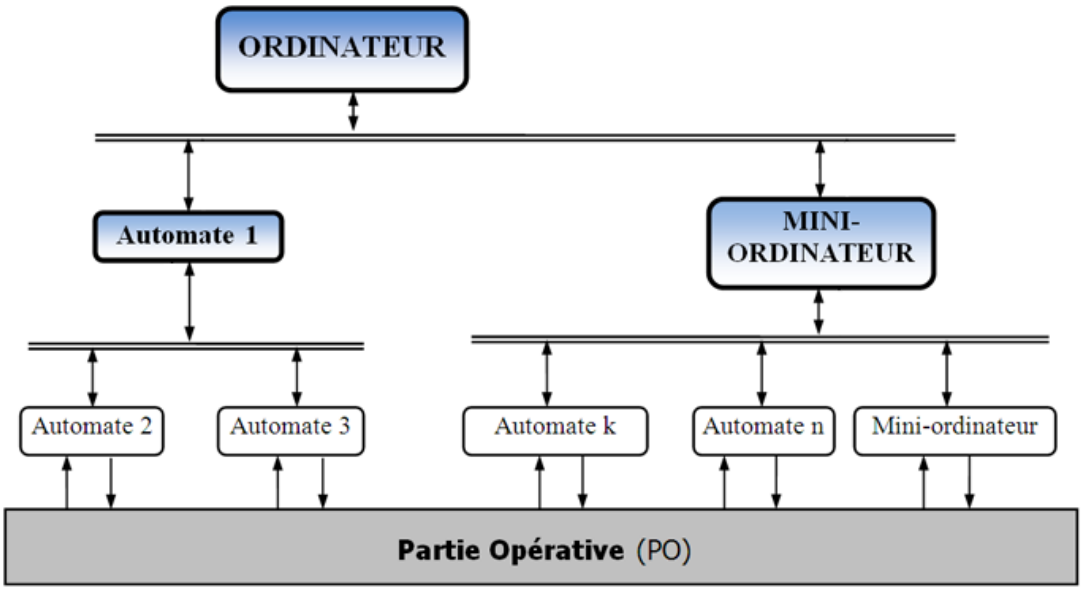


Figure I-10: Réseau hiérarchisé

9-CONCLUSION [6]

L'automate est un bon produit, facile à programmer, à connecter, adapté aux conditions industrielles. L'expansion considérable de ses possibilités, et celle corrélative de son marché, le prouvent. Il ne faut pas vouloir en faire une solution miracle. Dans tous les cas :

une bonne analyse du problème à résoudre ;

le respect des règles d'installation ;

un léger surdimensionnement pour préserver des marges de modification,

sont les conditions d'une implantation réussie, dont la durée de vie dépassera largement celles habituelles dans le monde informatique, dont l'API est pour partie issu.

Chapitre II

Interfaces d'entrées/Sorties

1- Introduction:

Les modules d'entrées/sorties (I/O) dans un API sont des interfaces entre le system (CPU) et le monde extérieur(processus). Leurs buts est pour permettre les connexions des dispositifs d'entrées tels que les capteurs et les dispositifs de sortie tels que des moteurs. Le présent chapitre donne un aperçu sur les différents types d'entrées/sorties, leurs connexions électriques et quelques interfaces spécieux utilisées dans un API.

2-Définition

Un module d'entrée (En :Input module) est un circuit électronique qui permet la conversion d'une valeur analogique en une valeur numérique (CAN) exploitable par le processeur de la carte CPU. De même, un module de sortie (En : Output module) est un circuit électronique qui permet la conversion des données numérique, venus depuis la carte CPU, en données analogiques exploitables par le processus. Cependant, le processeur de la carte CPU doit avoir des entrées et des sorties numériques avec un niveau de tension entre 0 et 5V.

3-Types des signaux d'Entrées/Sorties

Généralement, Il existe trois types des signaux d'entrées/sorties : Tout ou Rien (TOR), numériques et analogiques, et :

- Tout ou Rien : Essentiellement juste un signal marche / arrêt (0/24V...etc.),
- Numérique : Une séquence d'impulsions, figure (II-1).
- Analogique : Un signal dont la valeur est liée à la valeur de la quantité mesurée (0-10V, 4-20mA,...etc.), figure (II-1).

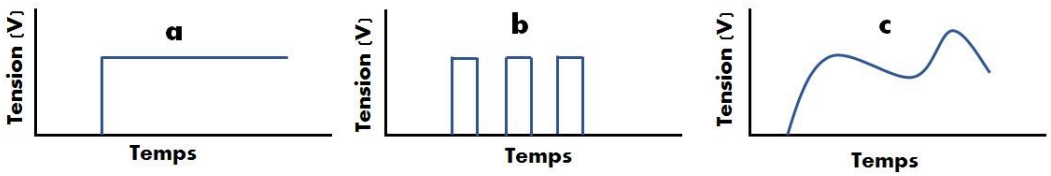


Figure II-1: Type des signaux :(a) Tout ou Rien, (b) Numérique, (c) Analogique

4-Modules d'Entrées

4-1- Modules d'Entrées TOR

Les modules d'entrées TOR connectent des dispositifs d'entrées (capteurs) à l'automate programmable sous forme d'un signal numérique. Ce type d'entrée n'ayant que deux états (ON / OFF, OUVERT / FERMÉ, VRAI / FAUX, etc.).

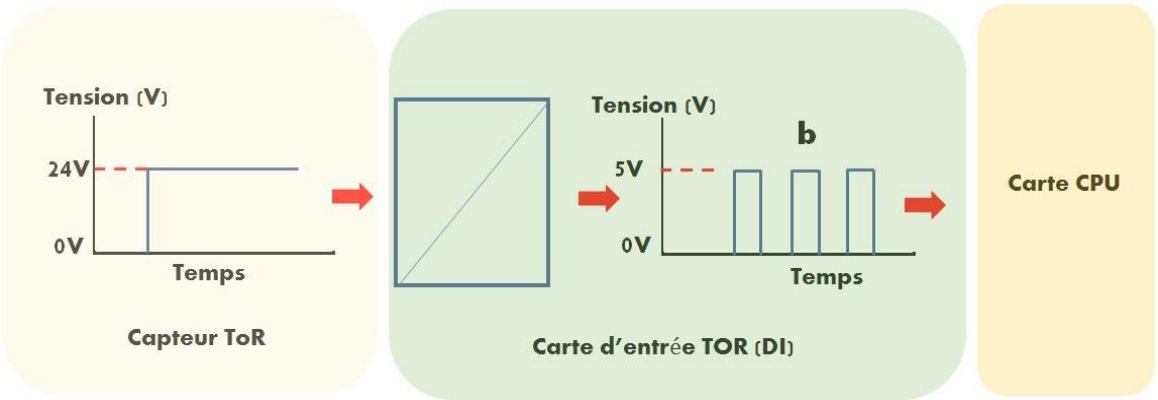


Figure II-2: Signal d'entrée Tout Ou Rien

Table II-1: Examples d'entrées TOR

Entrées TOR
-Fin de course
-Capteur photoélectrique
-Boutons poussoir
-Contacts de relais
-Etc.

En réalité, il y a deux formes d'entrées TOR; soit sous forme des bits simples, qui contrôlent une entrée codée sur un bit, soit sous forme des bits multiples, qui contrôlent de nombreuses

entrées. Ce dernier est appelé modules multibits, recevant plusieurs entrées, tels que les commutateurs à molette utilisés dans les interfaces de registre (BCD),où le transfert de données qui est représenté par le mot (Word).

4-1-1-Signaux des entrées TOR

Le signal reçu par les interfaces TOR peut être différent en matière de forme; alternatif ou continu et/ou de grandeurs (par exemple, 120 VCA, 12 VCC).

Le tableau (3.2) répertorie les valeurs standards utilisées pour les entrées TOR.

Table II-2: Standards des entrées TOR

Standards des entrées TOR
+24 V AC/DC
+48 V AC/DC
+120 V AC/DC
+230 V AC/DC
-Niveau TTL
-Entrée isolée
+5-50 V DC (sink/source)

4-2-Cartes d'entrées analogiques

Les cartes d'entrées analogiques sont utilisées dans les applications où le signal est continu. Contrairement aux signaux TOR, qui ne possèdent que deux états (ON et OFF), les signaux analogiques ont un nombre infini d'états. Par exemple, la température, donne un signal analogique car il varie continuellement par la variation de la température. Le tableau (3.5) présente quelques types des capteurs analogiques.

Table II-3: Exemples des entrées analogiques

Capture san alogiques
-Capteur de débit
-Capteurs d'humidité
-Potentiomètres
-Capteur de pression
-Capteur de vibrations
-Capteur de température
-Capteur de niveau
-Capteur de vitesse - Positionneur de vanne - Etc.

Ce type de carte permet de converti les signaux analogiques, venus depuis les capteurs analogiques, en des signaux numériques exploitables par la carte CPU de l'automate.

La figure (II-3) illustre la séquence numérique survenant lors de la lecture d'un signal d'entrée analogique. Le carte (AI) transforme le signal analogique, par l'intermédiaire d'un convertisseur analogique-numérique(CAN), en un mot de 12 bits qui sera stocké dans la mémoire image d'entrée. La valeur analogique stockée dans la mémoire image d'entrée sera au format BCD ou binaire.

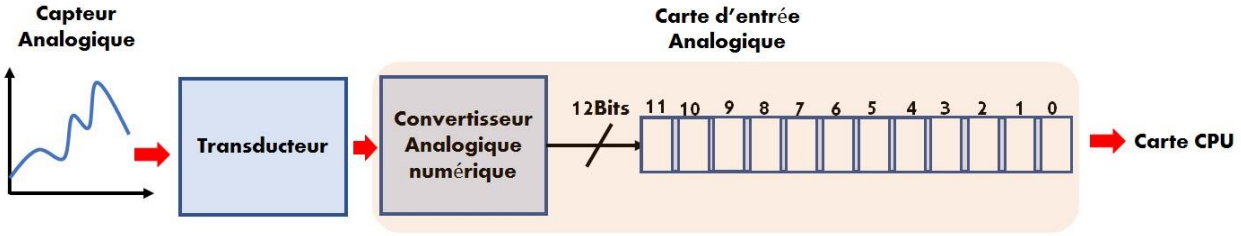


Figure II-3: Principe de fonctionnement d'une carte d'entrée analogique

Le rôle du transducteur est de transformer le signal d'entrée en un signal électrique normalisé que l'entrée analogique peut reconnaître (exemple 0-100 bar en 4-20mA).

Notant que le transducteur génère un signal électrique de très faible niveau (courant ou tension), donc on doit amplifier ce signal par l'utilisation des transmetteurs, qui à leurs tour envoient le signal à la carte d'entrée analogique (figure II-4).

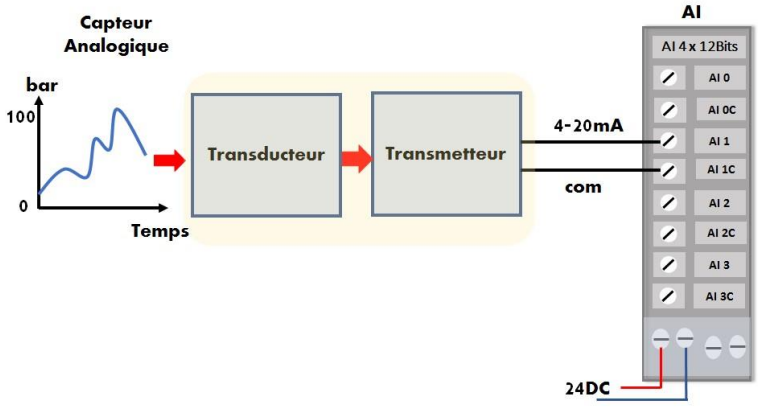


Figure II-4: Connexion d'une carte d'entrée analogique

En raison des nombreux types de transducteurs disponibles au marché, les cartes d'entrées analogiques ont plusieurs caractéristiques électriques standards.
Le tableau (3.6 répertorie les valeurs électriques standard de courant et de tension pour les cartes d'entrées.

Table II-4: Valeurs électriques standard des entrées analogiques

Valeurs denture sélectriques
+ 4-20mA
+ 0 à +1 V(DC)
+ 0 à +5 V(DC)
+ 0 à +10 V(DC)
+ 1 à +5 V(DC)
±5 V(DC)
±10 V(DC)

4-2-1-Connexion électriques des entrées analogiques

Les cartes d'entrées analogiques fournissent généralement une impédance d'entrée élevée (dans la gamme des mégaohms) pour les signaux d'entrées de type tension. Pour les cartes d'entrées de type courant fournissent une impédance d'entrée faible (entre 250 et 500 ohms). ce qui entraine

de prendre en considération le bon choix des transducteurs transmetteur en point de vue courant et tension.

Les cartes d'entrées analogiques peuvent recevoir des entrées asymétriques ou différentielles, dans le premier cas, les communs sont relie ensemble (un seul commun), par contre le dans le cas différentielle chaque entrée est indépendante des autres (chacune son commun). Selon le fabricant, une carte d'entrée peut être configurée en mode asymétrique ou différentiel à l'aide d'un commutateur.

Les figures (II-5) et (II-6) illustrent les connexions électriques des entrées analogiques pour des entrées différentielles et asymétriques respectivement.

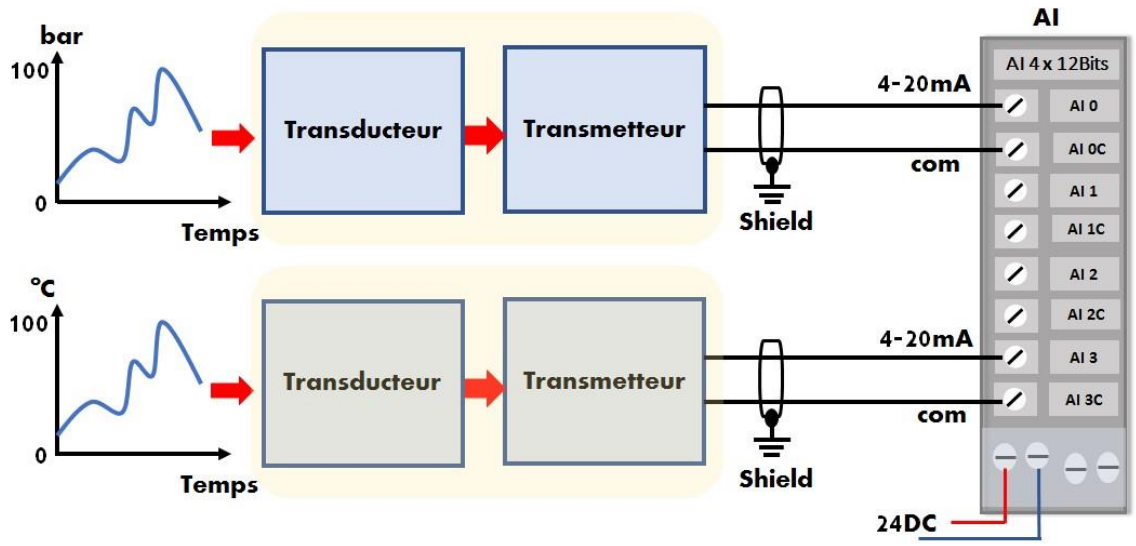


Figure II-5: Connexion différentielle d'une entrée analogique

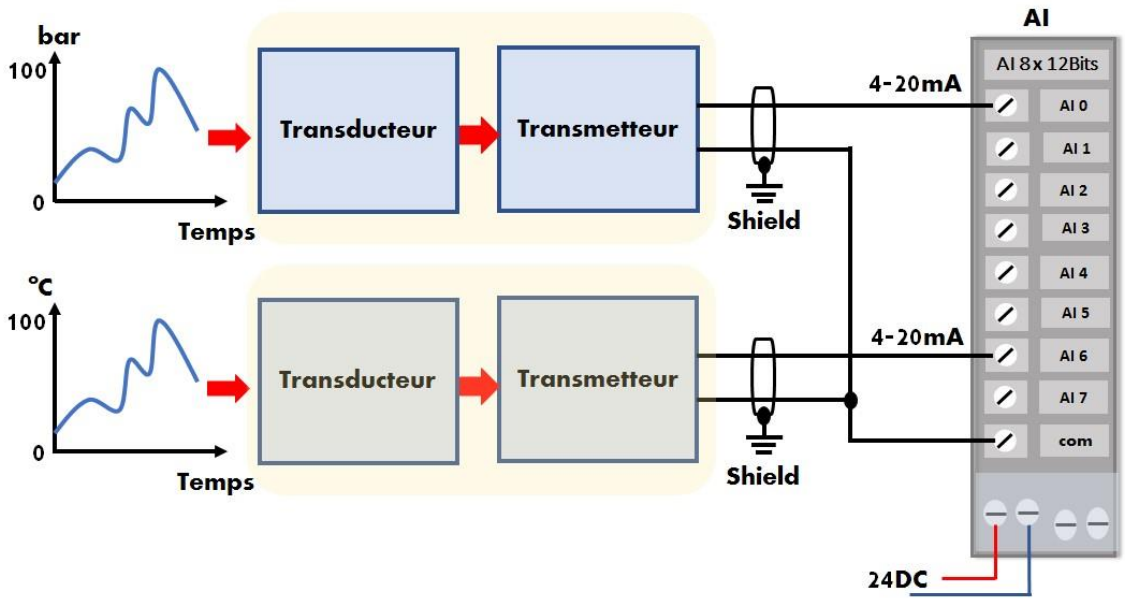


Figure II-6: Connexion asymétrique d'une entrée analogique

NB. Chaque entrée de la carte analogique fournit des circuits de filtrage et l'isolation des signaux pour protéger la carte contre le bruit de champ. D'autre part, l'utilisateur doit également envisager une protection contre les autres parasites électriques lors de l'installation par l'utilisation des câbles blindés (Shield). Ces câbles réduisent les déséquilibres d'impédance de ligne et maintiennent un bon rapport de réjection de mode commun des niveaux de bruit, tels que les fréquences des lignes d'alimentations.

5-Modules de sorties

5-1- Modules de sorties TOR

Comme les modules d'entrées TOR, les modules de sorties TOR connectent des dispositifs de sorties (actionneurs ou prés-actionneurs) de champ à l'automate programmable. Ce type de sortie n'ayant que deux états (ON / OFF, OUVRIR / FERMER, VRAI /
FAUX, etc.).

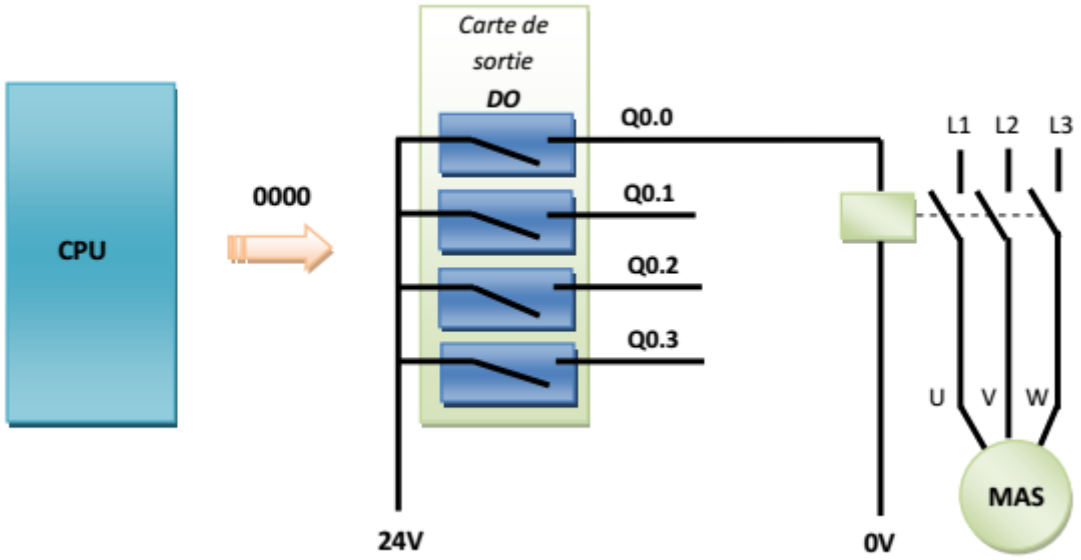


Figure II-7 : Principe de connexion des sorties état au repos

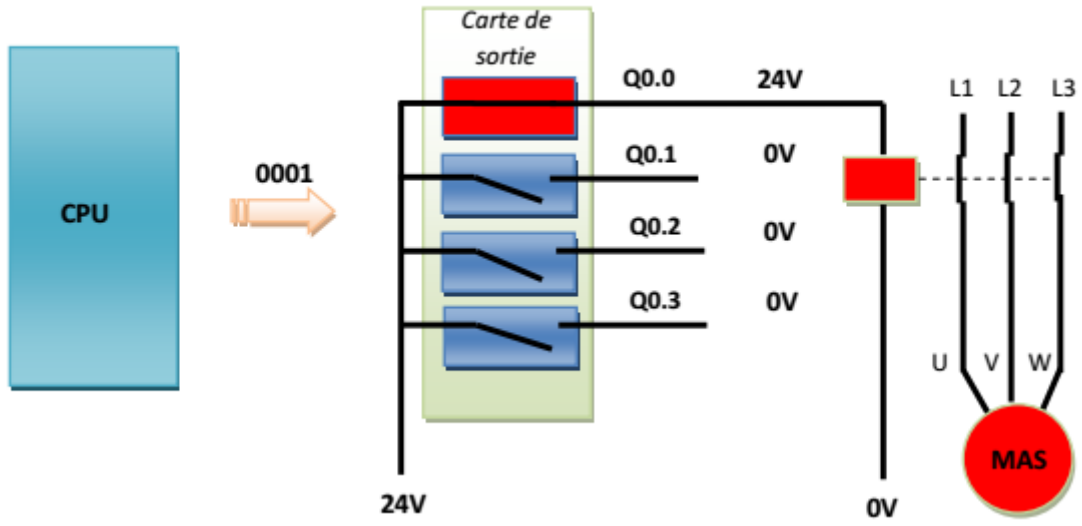


Figure II-8 : Principe de commande des sorties état actionnée

Le tableau (II-5) montre quelques applications utilisées en pratique pour ce type de sortie

Sorties TOR

-Relais électromagnétiques
-Ventilateurs
-LEDs
-Démarreurs
-Electro Vannes TOR
-conducteur
-ETC

Le tableau (II-6) répertori les valeurs standards utilisées pour les sorties de type TOR.

Standards des sorties TOR
+24 V AC/DC
+48 V AC/DC
+120 V AC/DC
+230 V AC/DC
Niveau TTL
Contact (relais)
Sortie isolée
+5-50 V DC (sink/source)

5-2-Cartes de sorties analogiques

Les cartes de sorties analogiques sont utilisées dans les applications nécessitant le contrôle des actionneurs répondant à des niveaux de tension ou de courant d'une façon continu.Un exemple de ce type de carte utilisé pour le débit volumétrique d'une sortie de pompe (voir Figure II-9)

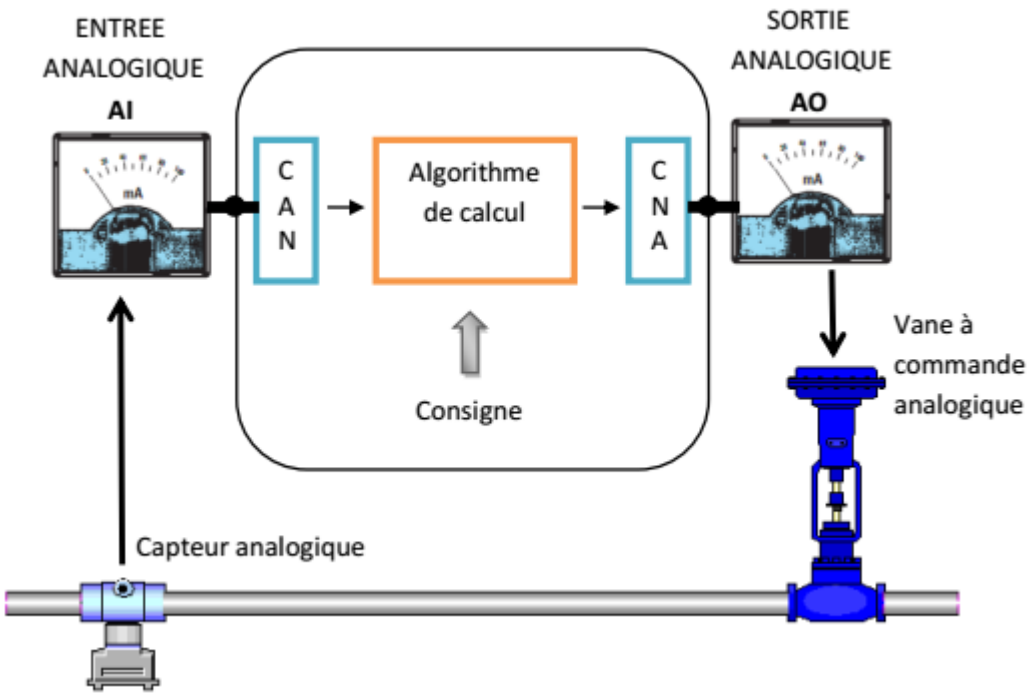


Figure II-9 : Principe de commande des sorties état actionnée

Comme les entrées analogiques, les cartes de sorties analogiques sont généralement connectées à des dispositifs de contrôle via des transducteurs (voir Figure II-10). Ces transducteurs amplifient, réduisent ou modifient le signal de tension en un signal analogique qui, à son tour de contrôler l'actionneur de sortie.

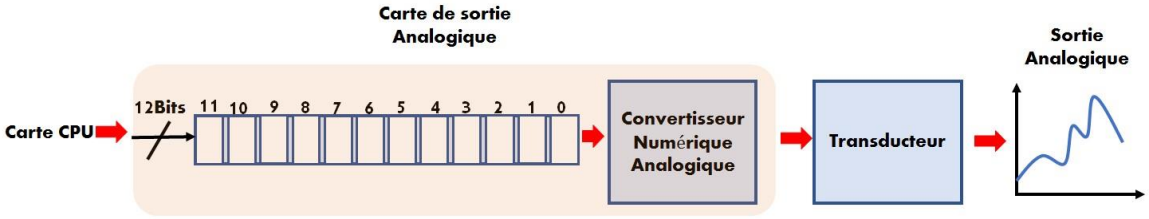


Figure II-10: Sortie analogique

Table II-7: Valeurs électriques standard des sorties analogiques

Valeurs de sortie électriques

+ 4-20mA
+ 10-50mA
+ 0 à +5 V(DC)
+ 0 à +10 V(DC)
±2.5 V(DC)
±5 V(DC)
±10 V(DC)

5-2-1-Connexion électriques des sorties analogiques

Les cartes de sorties analogiques sont disponibles dans des configurations allant de 2 à 8 sorties par carte, mais en moyenne, la plupart des cartes disposent de 4 à 8 canaux de sorties analogiques. Ces canaux peuvent être configurés comme sorties asymétriques ou différentielles.

Les figures (II-11) et (II-12) illustrent les connexions électriques des sorties analogiques pour des sorties différentielles et asymétriques respectivement.

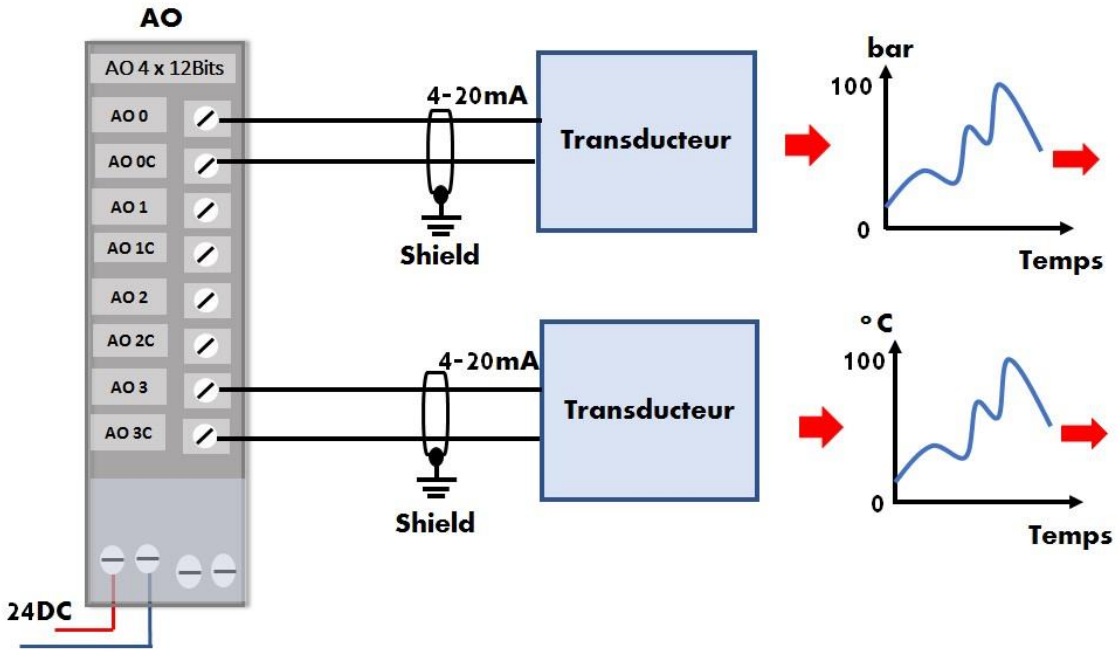


Figure II-11 Connexion différentielle d'une sortie analogique

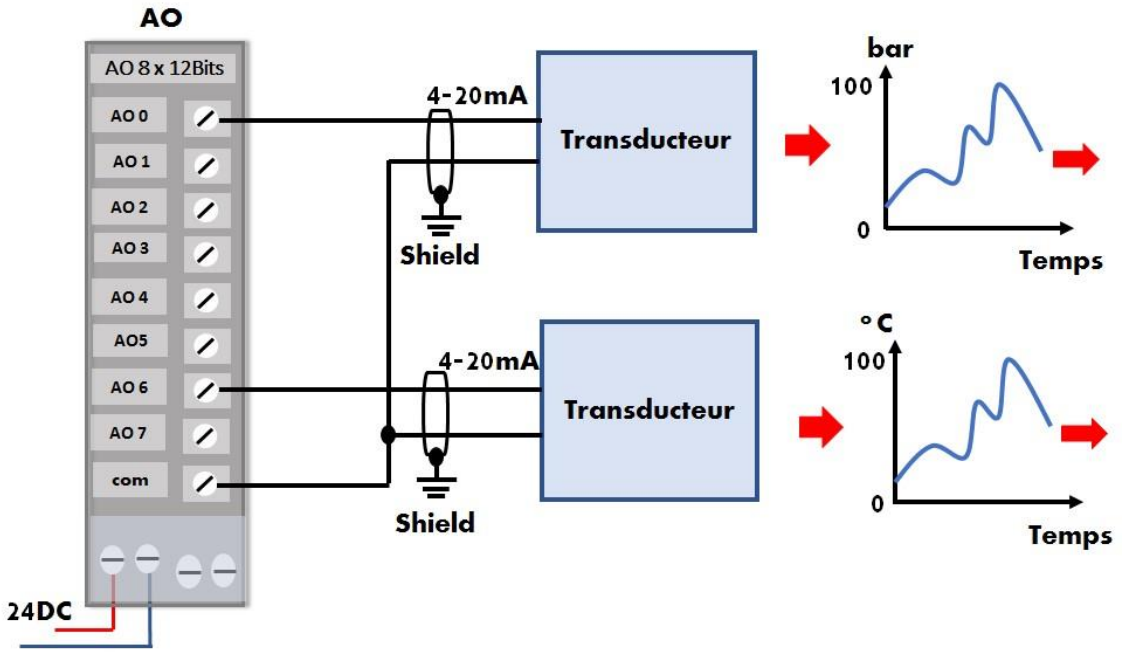


Figure II-12: Connexion asymétrique d'une sortie analogique

6-Cartes d'entrées/sorties spéciaux

Les cartes spéciaux des d'E/S assurent la liaison entre les automates programmables et les appareils nécessitant des types de signaux particuliers. Ces signaux spéciaux, qui diffèrent des signaux analogiques et numériques standards, ne sont pas très courants et ne surviennent que dans 5 à 10% des applications d'automates. Cependant, sans ce type des cartes, les processeurs (CPU) ne seraient pas en mesure d'interpréter ces signaux et de mettre en œuvre des programmes de contrôle, on citant par exemple , les cartes de régulation PID, carte de comptage rapide, les cartes de position, les cartes ASCII, les cartes réseaux pour les E/S distants, les cartes de traitement basé sur la logique floue...etc.. Les cartes spéciaux d'E/S peuvent être divisées en deux catégories :

- Les cartes d'actiondirecte
- Les cartesintelligentes

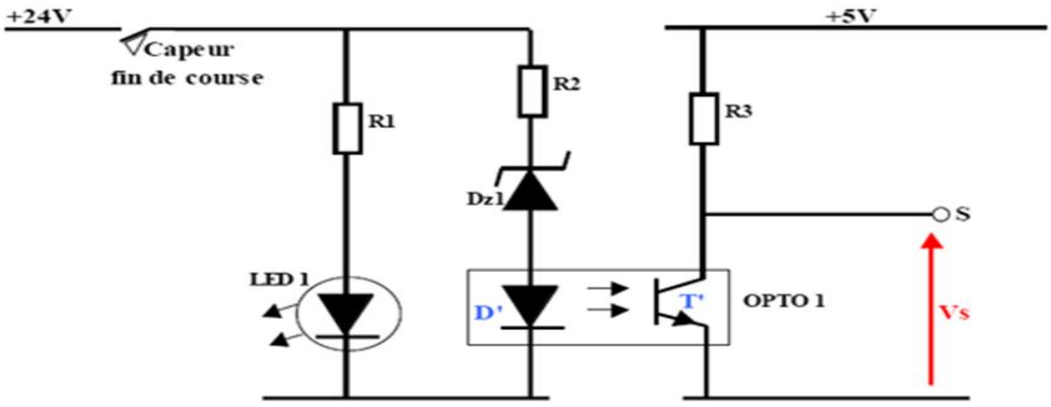


Figure II-13: Exemple d'une carte d'entrées typique d'un API

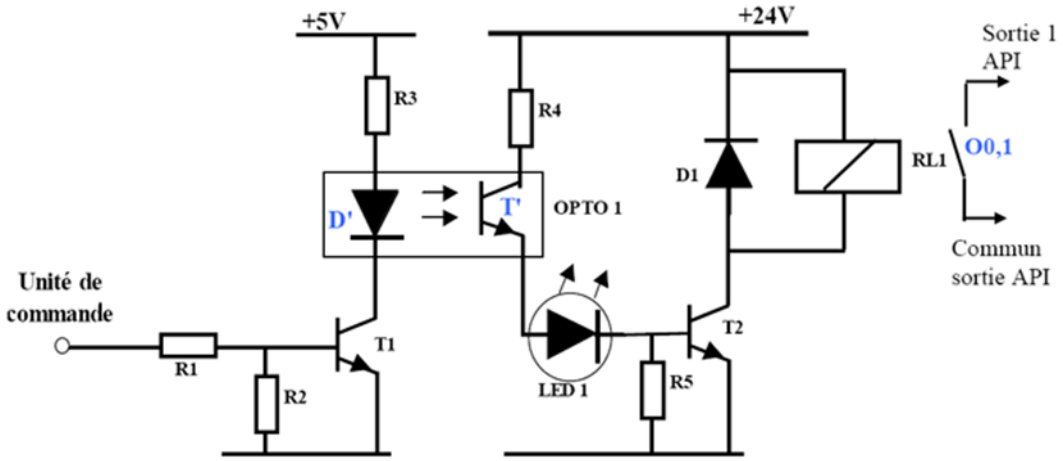


Figure II-14: Exemple d'une carte de sortie typique d'un AP

6-1-Les cartes d'actions directes

Les cartes d'E/S à action directe sont des cartes qui se connectent directement aux capteurs d'entrée ou avec les actionneurs de sortie. Ces cartes pré-traitent les signaux d'entrée ou de sortie et transmettent ces informations pré-traitées directement au processeur de l'API (voir Figure II-15). Ce type des cartes d'E/S comprennent des cartes qui pré-traitent généralement les signaux de faible tension comme dans le cas des thermocouples et rapides comme la grandeur de vitesse, que les autres cartes d'E/S standard ne peuvent pas le lire.

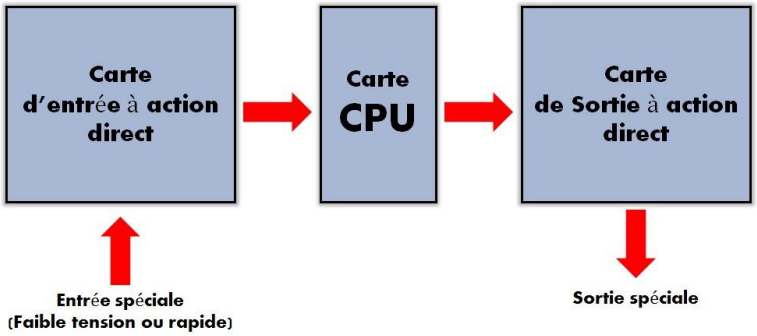


Figure II-15: Cartes d'action directe

6-2-Cartes intelligentes

Des cartes d'E/S intelligentes à fonctions spéciales incorporent des microprocesseurs intégrés pour ajouter le facteur d'intelligence . Ces cartes intelligentes peuvent effectuer des tâches de traitement complètement indépendantes du processeur CPU et la fonction scan l'API (voir Figure II-16). Ils peuvent aussi avoir des commandes numériques, ainsi que analogiques.

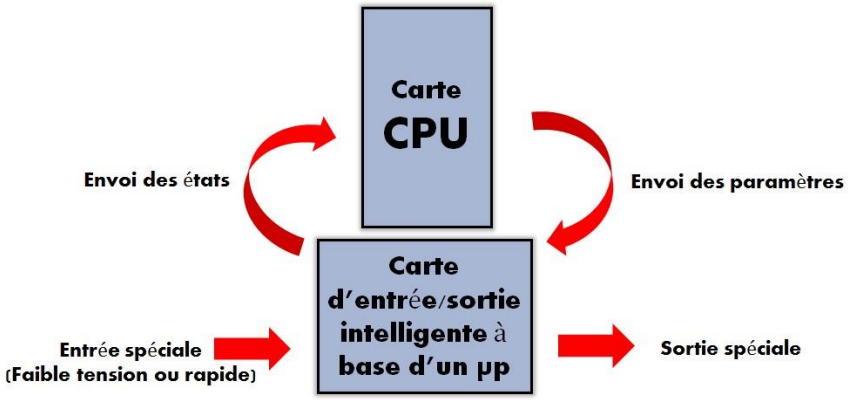


Figure II-16: Cartes d'action directe

7- Câblage des entrées / sorties d'un automate:

7-1Alimentation de l'automate (voir schéma ci-après):

L'automate est alimenté généralement par le réseau monophasé 230V ; 50 Hz mais d'autres alimentations sont possibles (110V etc ...).
La protection sera de type magnéto-thermique (voir les caractéristiques de l'automate et les préconisations du constructeur).
Il est souhaitable d'asservir l'alimentation de l'automate par un circuit de commande spécifique (contacteur KM1).

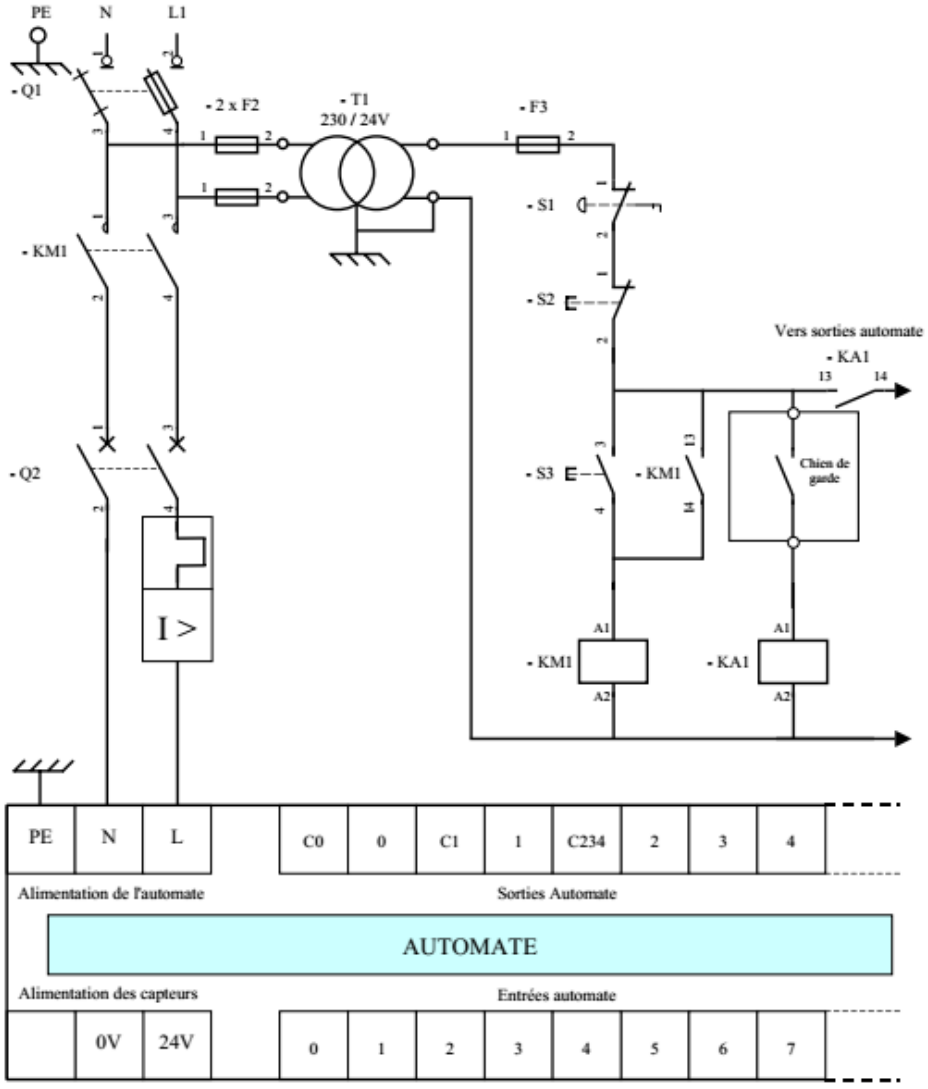


Figure II-17:Alimentation de l'automate

7-2 Alimentation des entrées de l'automate:

L'automate est pourvu généralement d'une alimentation pour les capteurs/détecteurs (attention au type de
logique utilisée : logique positive ou négative).
Les entrées sont connectées au OV (commun) de cette alimentation.
Les informations des capteurs/détecteurs sont traitées par les interfaces d'entrées.

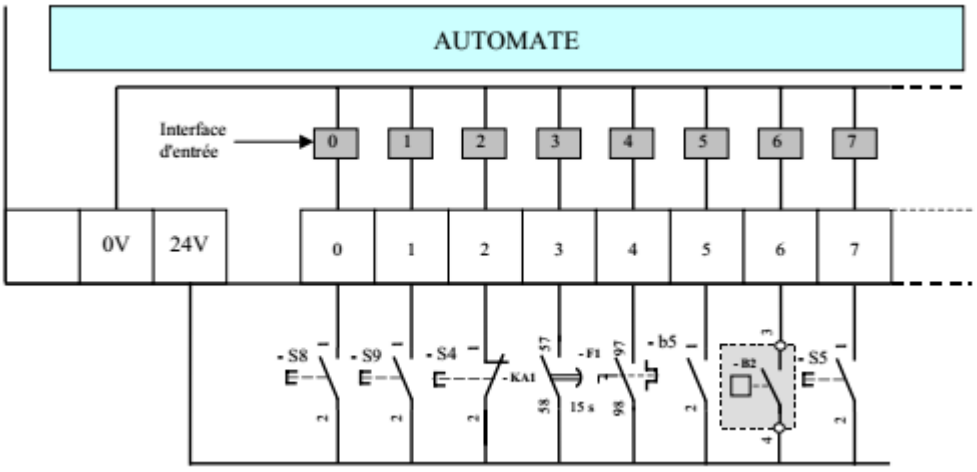


Figure II -18: Alimentation des entrées de l'automate

7-3.Alimentation des sorties de l'automate:

Les interfaces de sorties permettent d'alimenter les divers préactionneurs.
Il est souhaitable d'équiper chaque préactionneur à base de relais de circuits RC (non représentés).

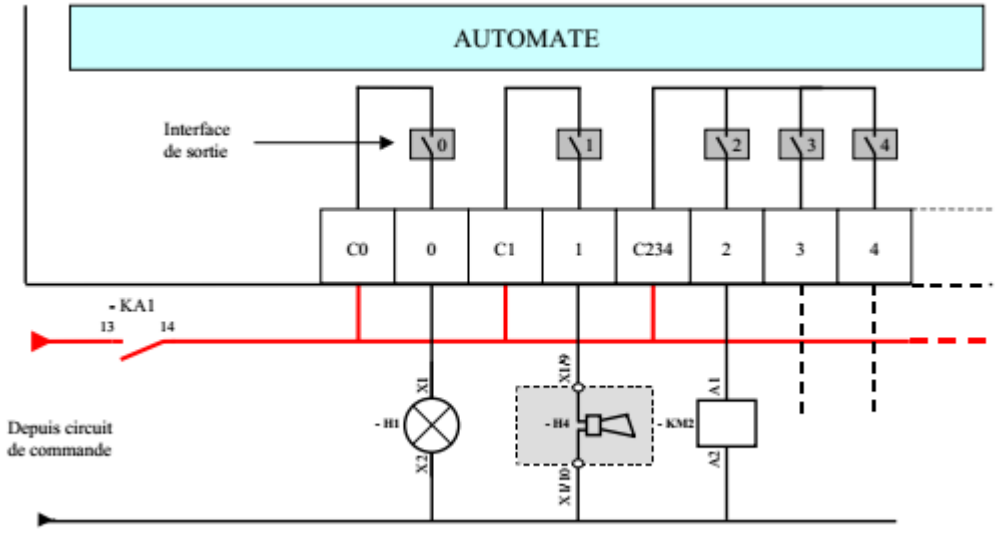


Figure II-19Alimentation des sorties de l'automate

Chapitre III

Programmation des APIs

1-Introduction

Les langages de programmation utilisés pour les automates programmables évoluent depuis la création des APIs à la fin des années 1960. Dans ce contexte, la commission internationale d'électrotechnique (CIE) définit cinq langages de programmation des automates programmables (IEC 61131-3), celles-ci consistent en deux langages textuels, (IL :liste d'instructions) et (ST :texte structuré), et trois langages graphiques, (LD : schéma à contacts), (FBD : schéma fonctionnel)et (SFC :Séquentiels Fonction Chart) sont définis pour structurer l'organisation interne des programmes et des blocs fonctionnels. Dans ce chapitre, nous présenterons que trois types de langages utilisés dans les automates aujourd'hui : IL, FBD et LADDER.

2-Adressages des entrées et des sortie d'un API

Avant d'entamer les trois langages de programmation des API, nous devons savoir l'adressage des entrées et des sorties des API. Pour ce faire, il attribue des adresses à chaque entrée et à chaque sortie. Avec un petit automate, il s'agit probablement d'un nombre, préfixé par une lettre pour indiquer s'il s'agit d'une entrée ou d'une sortie. Par exemple, pour l'automate Mitsubishi, nous pourrions avoir des entrées avec les adresses X400, X401, X402, etc., et des sorties avec adresses Y430, Y431, Y432, etc., le symbole X indiquant une entrée et le symbole Y une sortie.

Pour les automates les plus grands dont composés plusieurs racks, les racks sont numérotés. L'exemple avec le PLC-5 Allen-Bradley, le rack contenant le processeur reçoit le

numéro 0 et les adresses des autres racks sont numérotées 1, 2, 3, etc. Chaque rack peut avoir un certain nombre de cartes et chacune contient un certain nombre d'entrées et / ou de sorties. Ainsi, les adresses peuvent être de la forme illustrée à la figure (III-1). Par exemple, nous pourrions avoir une entrée avec l'adresse *I : 012/03*. Celaindiqueraitune entrée, rack 01, module 2 et l'entrée 03.

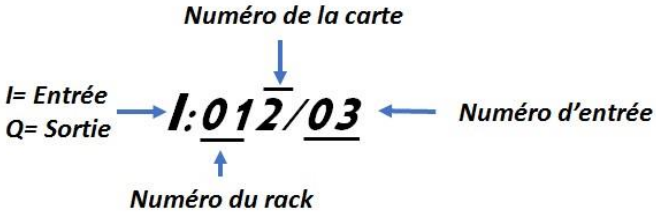


Figure III-1: Adressage PLC-5 Allen-Bradley

Pour le constructeur allemand Siemens SIMATIC S7, les entrées et les sorties sont réparties en groupes de huit. Chacun de ces groupes est appelé un octet et chaque entrée ou sortie d'un groupe de huit est appelée bit. Les entrées et les sorties ont donc leurs adresses en nombre d'octets et de bits, ce qui donne un numéro de module suivi d'un numéro de terminal, et un point (.) Séparant les deux numéros. La figure (III-2) montre l'adressage d'une entrée/sortie dans un automate Siemens. Par exemple, I0.1 est une entrée au bit 1 dans l'octet 0 et Q2.0 est une sortie au bit 0 dans l'octet 2.

Figure III-2: Adressage Siemens SIMATIC S7

3-Langage de programmation API

3-1- langage de programmation ladder LD

Ladder Diagram (LD) ou Langage Ladder ou schéma à contacts est un langage graphique très populaire auprès des automaticiens pour programmer les APIs. Il ressemble un peu aux schémas électriques, et est facilement compréhensible. L'idée initiale du Ladder est la représentation de fonction logique sous la forme de schémas électriques. Par exemple, pour réaliser un ET logique avec des interrupteurs, il suffit de les mettre en série. Pour réaliser un OU logique, il faut les mettre en parallèle.

Ladder a été créé et normalisé dans la norme CEI 61131-3. Il est encore aujourd'hui souvent utilisé dans la programmation des automates programmables industriels, bien qu'ayant tendance à être délaissé e

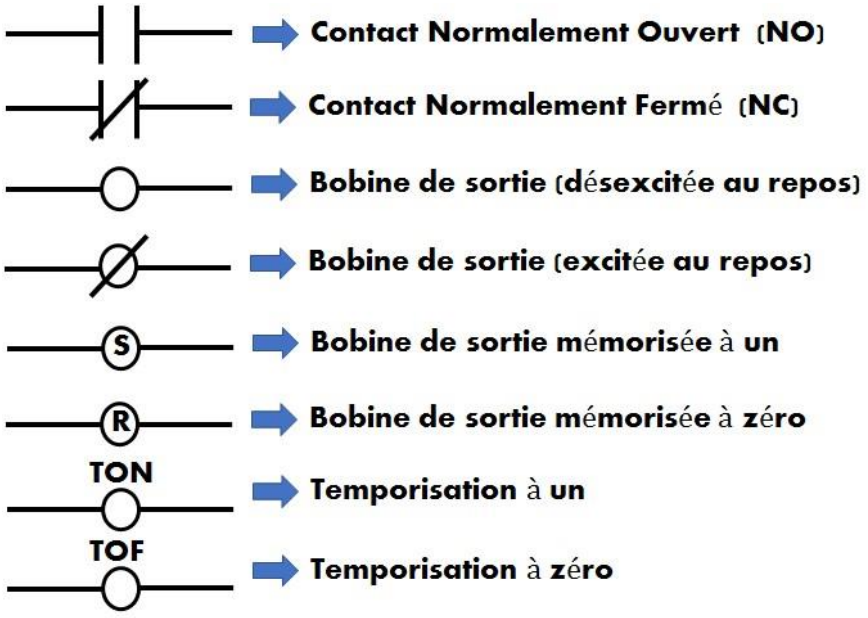


Figure III-3: Quelques symboles de langage Ladder

n faveur de langages plus évolués, et plus adaptés aux techniques modernes de programmation, tels que le ST par exemple, ou encore le Grafcet, plus adapté à la programmation de séquences. Aussi l'avantage de ce langage et dans sa puissance en point de vue diagnostic. La figure (III-4) présente quelques symboles utilisés dans le langage Ladder

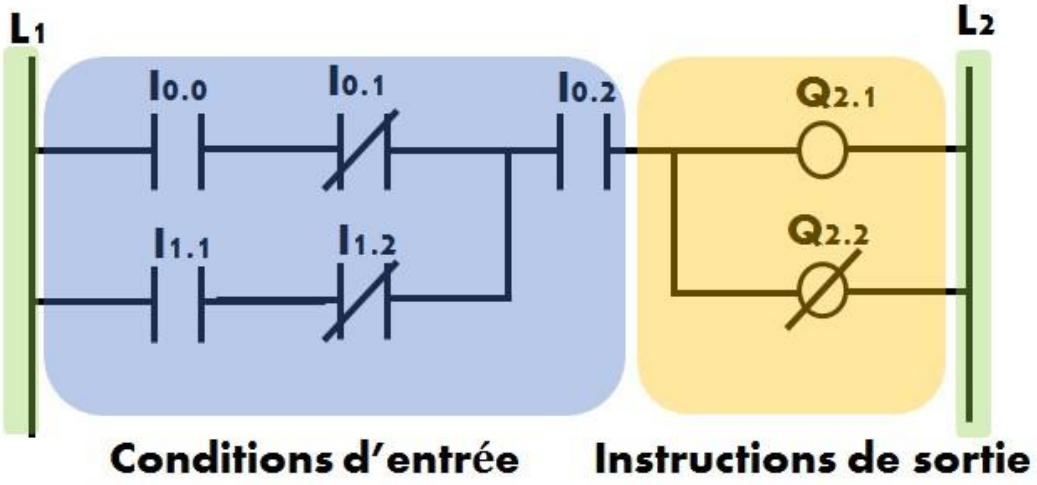


Figure III-4: Présentation du langage Ladder

3-2-langage de programmation FBD*:*

Le diagramme FBD (Function Block Diagram) décrit une fonction entre des variables d'entrée et des variables de sortie. Une fonction est décrite comme un réseau de blocs élémentaires. Les variables d'entrée et de sortie sont connectées aux blocs par des arcs de lien. Une sortie d'un bloc peut aussi être connectée sur une entrée d'un autre bloc.

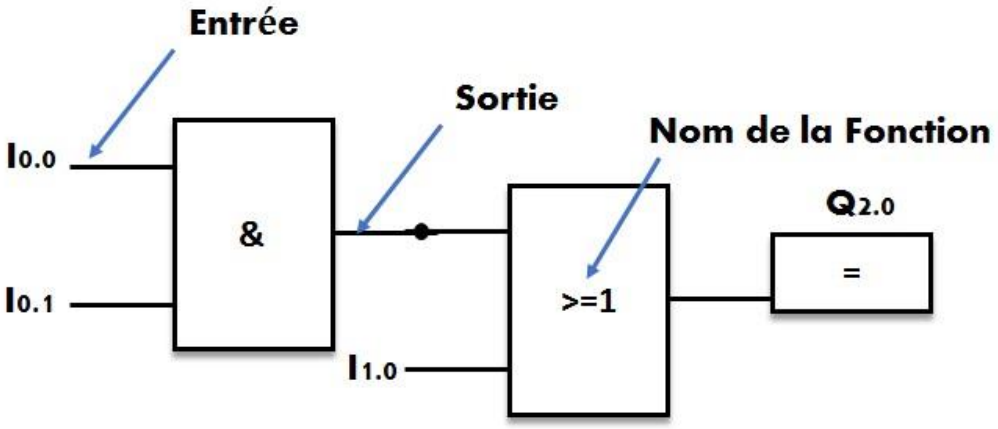


Figure III-5: Présentation du langage FBD

3-3 -langage de programmation LIST*:*

Le langage de programmation textuelle LIST permet de créer des programmes d'application à un niveau proche du matériel et en optimisant le temps d'exécution et la place en mémoire.

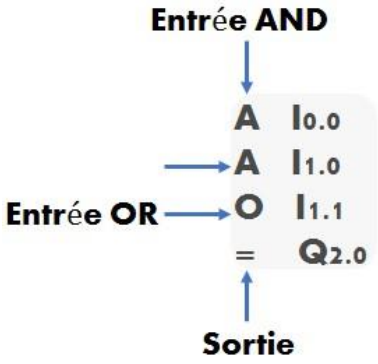


Figure III-6: Présentation du langage LIST

3-4- Programmation à l'aide du GRAFCET (SFC : Séquentiel Fonction Chart

Le GRAFCET, langage de spécification, estutilisé par certainsconstructeursd'automate (Schneider, Siemens) pour la programmation. Parfoisassocié à unlangage de programmation, il permetune programmationaisée des système séquentiels tout en facilitant la mise au point des programmesainsique le dépannage des systèmes.
On peutégalem enttraduireun grafcet en langage en contacts et l'implantersur tout type d'automate. Certainsl ogicie lspermett entune programmation totale en langage GRAFCET etpermettent de s'adapter à la plupart des automates existants (logiciels CADEPA ou AUTOMGEN)

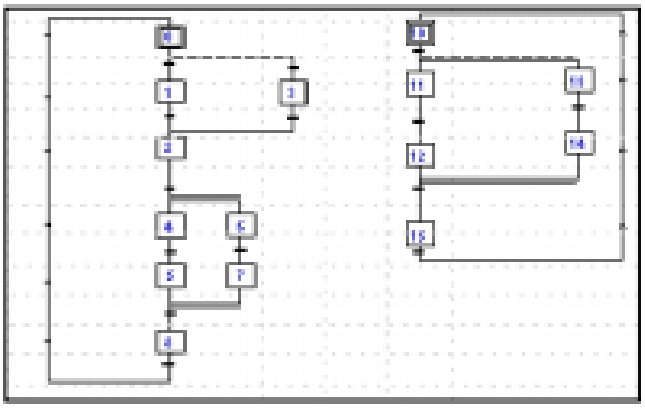


*Figure III-7: grafcet**[7]***

4-Programmation des APIs

Dans cette partie nous allons présenter quelques applications le plus utilisé dans l'industrie et chaque application est accompagner avec des exemples en trois langages cités auparavant.

4-1Fonctions logiques

Dans ce qui suite on va donner quelques fonctions logiques le plus utilisés en trois langages tel que Ladder, FBD et LIST.

Fonction logique AND

Le tableau(III-1)montre, la table de vérité de la fonction AND:

Entrée1	**Entrée2**	**SortieY=A.B**
I0.0	*I0.1*	*Q1.0*
0	0	0
0	1	0
1	0	0
1	1	1

TableIII-1:Table de vérité de la fonction AND

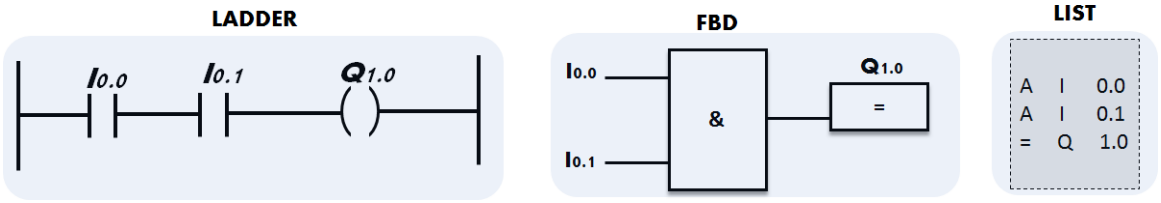


Figure III-8 :Fonction AND sous API

Fonction logique OR

Le tableau(III-3)montrela table devérité de la fonction OR:

Entrée1	**Entrée2**	**SortieY=A+B**
I0.0	*I0.1*	*Q1.0*
0	0	0
0	1	1
1	0	1
1	1	1

Table III-3: Table de vérité de la fonction OR Le

figure(III-9)présente la fonction OR avec les troislangages:

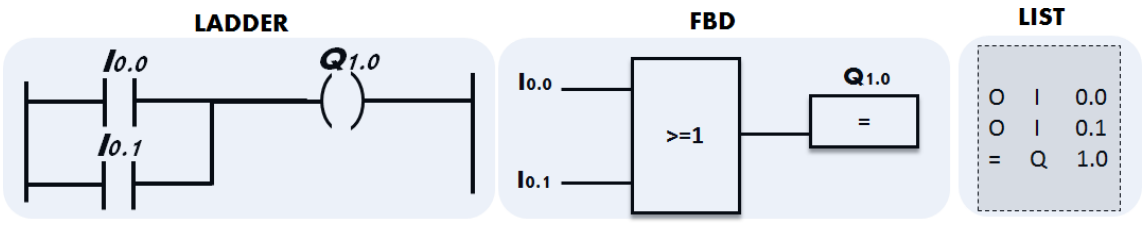


Figure III-9 :Fonction OR sous API

Fonction logique NOT

Le tableau(III-2)montre la tablede véritéde la fonction NOT:

Entrée1	**Sortie**=$\overline{A}$
I0.0	*Q1.0*
0	1
1	0

TableIII-2:Table devérité de la fonction NOT

Le figure(III-10)présente la fonction NOT avec les troislangages:

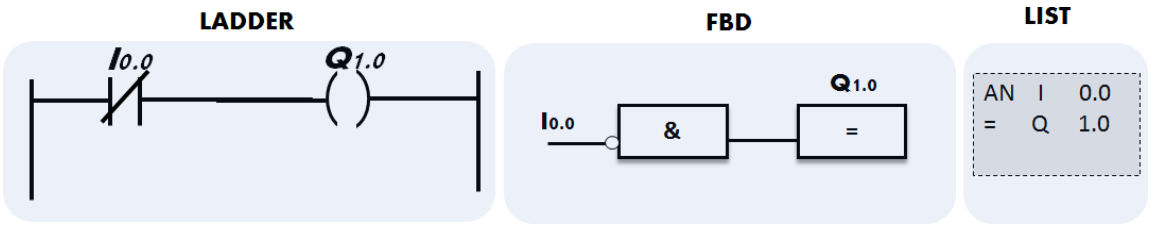


Figure III-10:Fonction NOT sous API

4-2Fonction mémorisation (Latching)

Il existe souvent des situations dans lesquelles il est nécessaire de maintenir une sortie sous tension, quelque soit l'état de l'entrée. Un exemple simple d'une telle situation dans le cas de démarrage du moteur, en appuyant sur un bouton-poussoir marche (start). Bien que le contact du bouton poussoir ne reste pas fermé, le moteur doit continuer à fonctionner jusqu'à l'appui sur le bouton-poussoir d'arrêt (stop).

On appel le contacte Q1.0 l'auto-maintien.

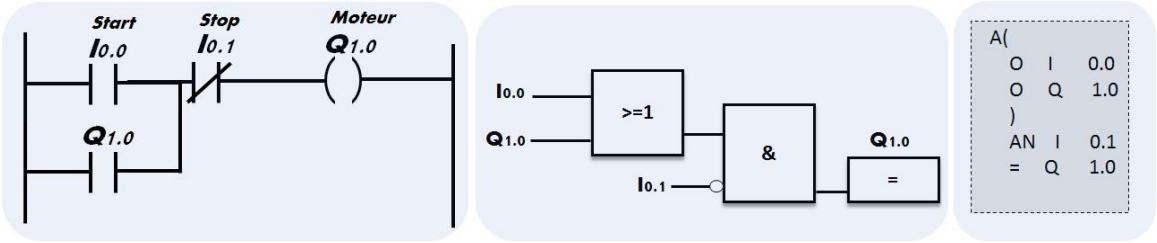


Figure III-11: Fonction mémorisation

4-3Fonction temporisation

Dans de nombreuses applications de contrôle à base d'automates programmables, il est nécessaire de contrôler le temps. Par exemple, un moteur doit fonctionner pendant un intervalle de temps particulier ou peut être allumé après un certain intervalle de temps...etc. Les automates programmables ont donc des minuteries en tant que périphériques intégrés. Les minuteries comptent les secondes ou fractions de seconde en utilisant l'horloge interne du processeur. Dans cette partie on va montré les différents types des minuteries les plus utilisés.

ON-Delay Time: Temporisateur travail

GénéralementtouslesAPIssontdotésdecetypedetemporisateurdontlefonc-tionnement est présenté dans la figure (III-12)

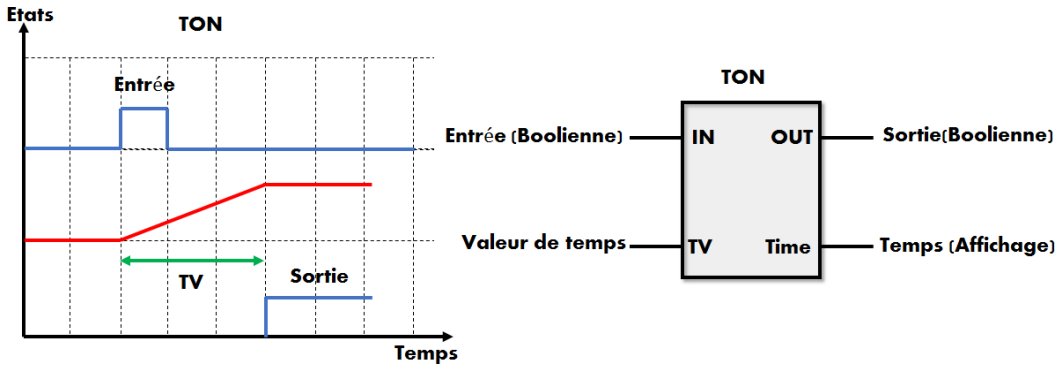


Figure III-12:TemporisateurTON

Sil'entréeINestactivéeletemporisateurTONvadéclencherlatemporisationpendant

Letempsalloué TV , aprèscetemp ,le TON va activer la sortie OUT.

OFF-Delay Time : Temporisateur repos

Un temporisateur de type TOF (minuterie à retard d'excitation) fournit une action retardée.Silalignedecommanden'apasdecontinuité,letemporisateurcommenceà compterlesintervallesdetempsjusqu'àcequelavaleurdetempscumulésoitégaleàla valeur préréglée TV.

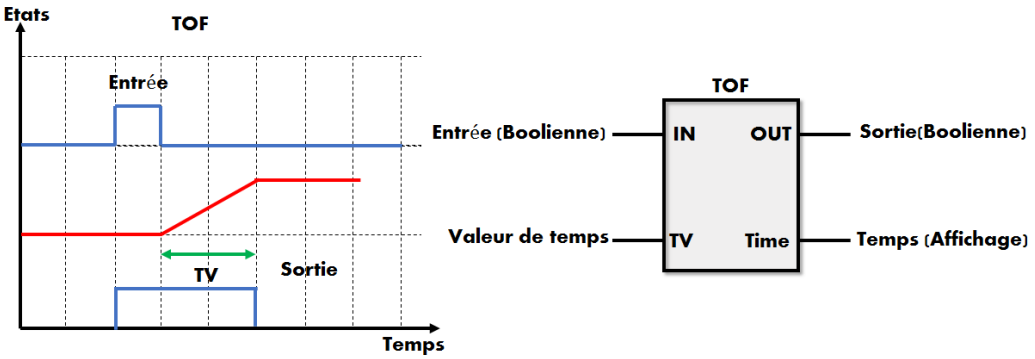


Figure III-13:Temporisateur TOF

UnefoisI'entréeINestdésactivéeletemporisateurTOFvadéclencherlatemporisation pondant le temps alloué TV, après ce temps, le TOF va désactiver la sortie OUT.

Temporisateur d'impulsion(Pulls TIMER)

Ce type de temporisation est utilisé pour produire une sortie de durée fixe à partir d'uneentrée.Lafigure(III-14)montrelediagrammed'échellepourcetypedetemporisateur qui donnera une sortie de out 1 pour une durée fixe prédéterminée de temps quand il y a une entrée a 1, après cette temporisation la sortie devient 1.

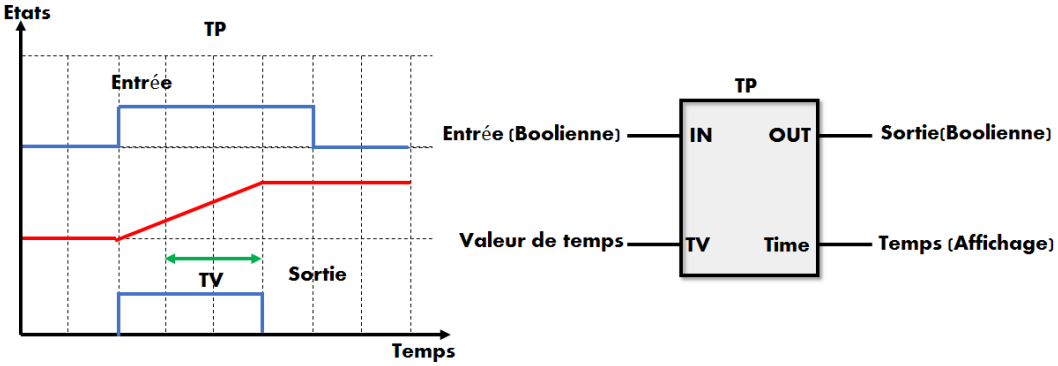


Figure III-14:Temporisateur TP

Temporisateurretentive RTO

LetypedetemporisateurRTOesttrèssimilaireautypeTONetTOF,àl'exception dufaitquelavaleurcumulée(TV)estconservéemêmesilesconditionsdesentréessont à l'état zero.

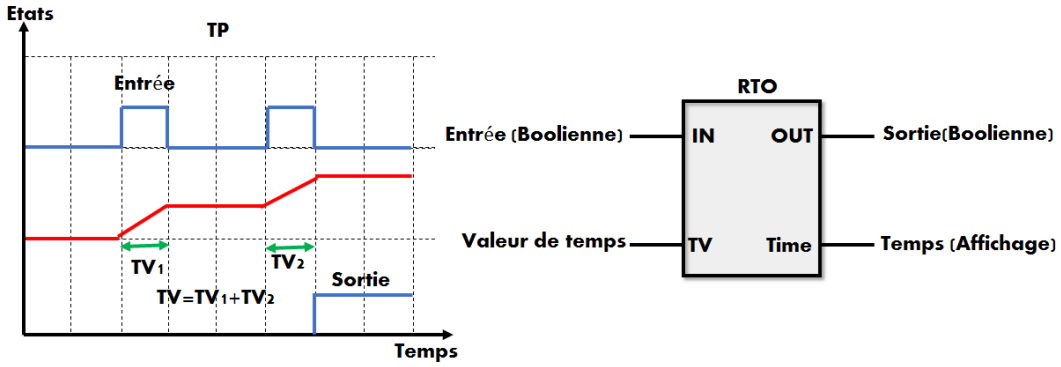


Figure III-15 :Temporisateur RTO

4-4Fonction de comptage

Les compteurs sont fournis en tant qu'éléments intégrés dans les automates et permettent de compter le nombre d'occurrences de signaux d'entrées. Par exemple pour compter le nombre des bouteilles produites ou bien combien de bouteilles sont passées par le convoyeur ou peut-être le nombre de personnes passant par une porte. Cette partie décrit comment ces compteurs peuvent être programmés. Généralement dans les API on trouve deux types des fonctions de comptage, le compteur et le décompteur.

Compteurs(Up-counter CTU)

Le fonctionnement de ce type de compteur (CTU) est d'ajouter un nombre, chaque foisqu'unévénementseproduit,cecompteurincrémentàun,àlafinilactivelasortie.

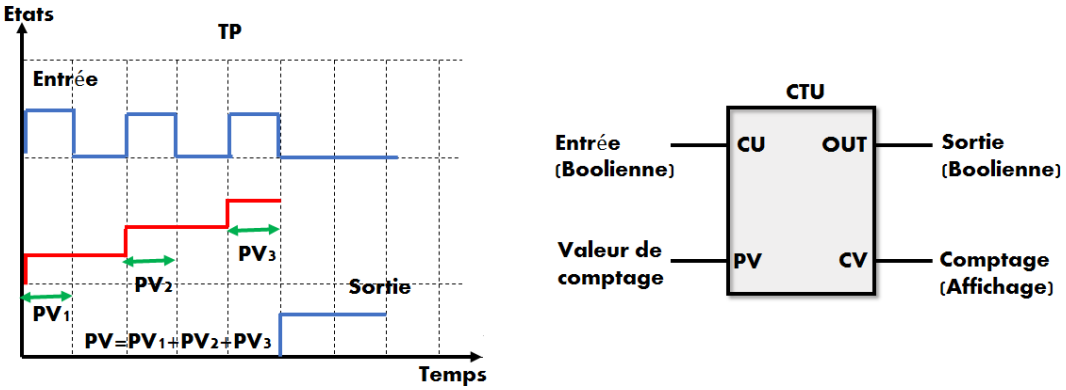


Figure III-16 :Fonction comptage CTU

Compteurs(Up-counter CTD)

Lefonctionnementdudécompteur(CTD)estdediminuerunnombre,chaquefois qu'unévénementseproduit,cedécompteurdécrémenteàun,àlafinilactivelasortie.

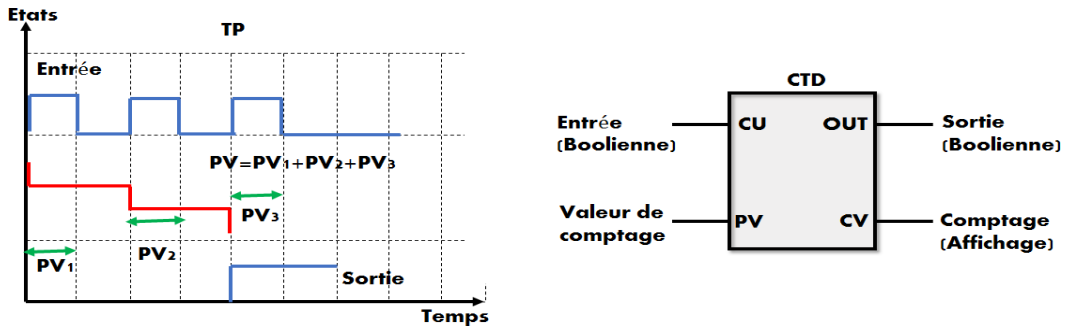


Figure III-17 :Fonction décomptage CTD

4-5Fonction de régulation

La fonction de régulation est une action indispensable dans les automates programmables afin de contrôler les processus industriels, par exemple la régulation de la température d'une chaudière, la régulation de niveau dans un bac de stockage...etc. Généralement dans les APIs, il existe deux types de régulateurs, des régulateurs TOR et des régulateurs PID

Régulation TOR(on-off)

Danscetypederégulation,onpeutdistinguédeuxautrestypes,letypederégulation on-off,figure(4.27.a),etletypeon-offàhystérésisfigure(4.27.b),danslesdeuxcasle contrôleur est essentiellement considéré comme un interrupteur qui fournit un signal de sortie marche/arrêt selon la valeur de l'erreur.

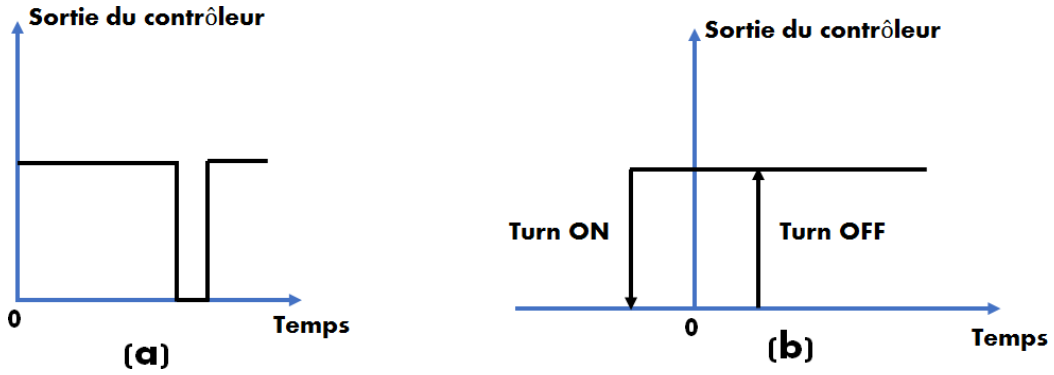


Figure III-18 :Régulation on-off

5-Conclusion

L'exploitation et la programmation des automates programmable industriels ne fait qu'accroitre de jour en jour et devient de plus en plus sollicitée dans plusieurs domaines . Basée principalement sur des connaissances préalables , tels que la logique combina toire , séquentielle et un langage de programmation , elle permet d'automatiser efficacement beaucoup de cas de la productivité . Bien évidemment , ce chapitre traite d'une façon un peut générale la programmation des APIs où on a donné , quelques exemple et des outils les plus utilisés dans l'industrie des API . D'autre part

Chapitre IV

Programmation et simulation d'automates programmables industriels basés sur des GRAFCET

1-INTRODUCTION:

La mise en œuvre d'un robot programmable comprend trois étapes principales :

- Concevoir un système de contrôle qui se fait à l'aide d'outils méthodologiques et les modes de représentation propres à l'automaticien ;

- Programmation de l'automate, qui consiste à transmettre le système de contrôle obtenu Dans le langage de programmation de PLC ;

-Enfin l'exécution du programme pris en charge par un logiciel interne dans l'automate,

Dans ce chapitre, nous appliquerons des exemples concrets et, selon les grafcet , nous programmerons des automates programmables industriels.

2–Définition de GRAFCET

Le **GRAFCET** (**Gra**phe **F**onctionnel de **C**ommande par **E**tapes et **T**ransitions) ou **SFC** (**S**équentiel **F**onction **C**hart) est un outil **graphique** qui décrit les différents comportements de l'évolution d'un **automatisme** et établit une correspondance à caractère séquentiel et combinatoire entre :

- Les **ENTREES**, c'est-à-dire les transferts d'informations de la Partie Opérative vers la Partie Commande,
- Les **SORTIES**, transferts d'informations de la Partie Commande vers la Partie Opérative.

C'est un outil graphique puissant, directement exploitable, car c'est aussi un langage pour la plupart des **API** existants sur le marché. Lorsque le mot **GRAFCET** (*en lettre capitale*) est utilisé, il fait référence à l'outil de **modélisation**. Lorsque le mot **grafcet**est écrit en minuscule, il fait alors référence à **un modèle** obtenu à l'aide des règles du **GRAFCET**.

Le **GRAFCET** comprend :

- des **étapes** associées à des actions ;

- des **transitions** associées à des **réceptivités** ;
- des **liaisons orientées** reliant étapes et transitions.

2-2 - Description du GRAFCET [10]

La description du comportement attendu d'un automatisme peut se représenter par un **GRAFCET** d'un certain « **niveau** ». La caractérisation du «**niveau**» du GRAFCET nécessite de prendre en compte trois dimensions :

- **Le point de vue** , caractérisant le point de vue selon lequel un observateur s'implique dans le fonctionnement du système pour en donner une description. On distingue trios points de vue :
  - Un point de vue système ,
  - Un point de vue Partie Opérative ,
  - Un point de vue Partie Commande .
- **La spécifications**, caractérisant la nature des spécifications techniques auxquelles doit satisfaire la Partie Commande. On distingue troisgroupes de specifications :
  - Specifications fonctionnelles,
  - Specifications technologiques,
  - Specifications opérationnelles.
- **La finesse**, caractérisant le niveau de détail dans la description du fonctionnement, d'un niveau global (ou macro-représentation) jusqu'au niveau de détail complet où toutes les actions et informations élémentaires sont prises en compte.

2-3 - Les concepts de base du GRAFCET [11]

2-3-1 - **étape**

Une **étape** symbolise un état ou une partie de l'état du système automatisé. L'étape possède deux états possibles : **active** représentée par un jeton dans l'étape ou **inactive**. L'étape i, représentée par un carré repéré numériquement, possède ainsi une variable d'état, appelée variable d'étape **X**i. Cette variable est une variable booléenne valant **1** si l'étape est active, **0** sinon.

La situation initiale d'un système automatisé est indiquée par une étape dite **étape initiale** et représentée par un carré double.

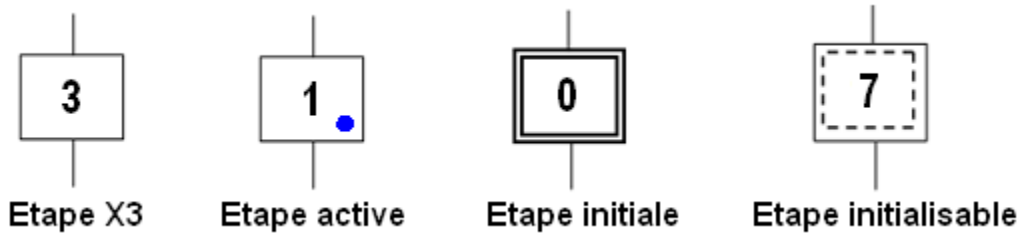


Figure IV-1 : Tape

Remarque : Dans un **grafcet** il doit y avoir au moins une étape initiale.

- Actions associées aux étapes

A chaque étape est associée une **action** ou plusieurs, c'est à dire un ordre vers la partie opérative ou vers d'autres grafcets. Mais on peut rencontrer aussi une même action associée à plusieurs étapes ou une étape **vide** (*sans action*).

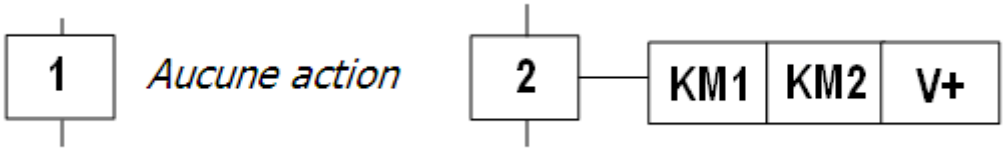


Figure IV-2Actions associées aux étapes

- Transition

Une transition indique la possibilité d'évolution qui existe entre deux étapes et donc la succession de deux activités dans la partie opérative. Lors de son franchissement, elle va permettre l'évolution du système. A chaque transition est associée une condition logique appelée réceptivité qui exprime la condition nécessaire pour passer d'une étape à une autre.

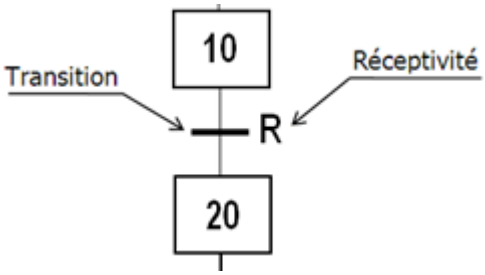


Figure IV-3: TRANSITION

La réceptivité qui est une information d'entrée qui est fournie par :

- l'opérateur : pupitre de commande,
- la partie opérative : états des capteurs,
- du temps, d'un comptage ou tout opération logique, arithmétique...
- du grafcets : d'autres grafcet pour la liaison entre grafcets ou de l'état courant des étapes du grafcet (les Xi),
- d'autres systèmes : dialogue entre systèmes,

Remarque: Si la réceptivité n'est pas précisée, alors cela signifie qu'elle est toujours vraie. (=1)

2-3.4 - Liaisons orientées

Elles sont de simples traits verticaux qui relient les étapes aux transitions et les transitions aux étapes. Elles sont normalement orientées de haut vers le bas. Une flèche est nécessaire dans le cas contraire.

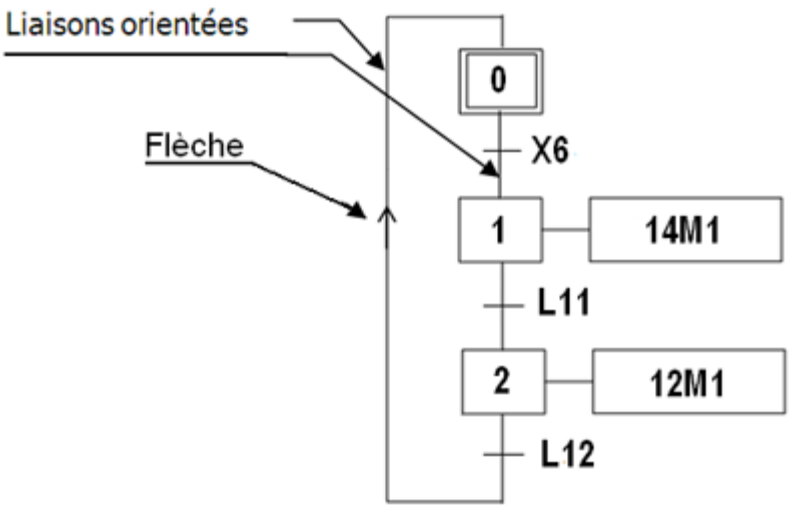


Figure IV-4 : Liaisons orientées

2-4- Les structures de base

2-4-1 - Notion de Séquence :

Une séquence, dans un Grafcet, est une suite d'étapes à exécuter l'une après l'autre. Autrement dit chaque étape ne possède qu'une seule transition AVAL et une seule transition AMONT.

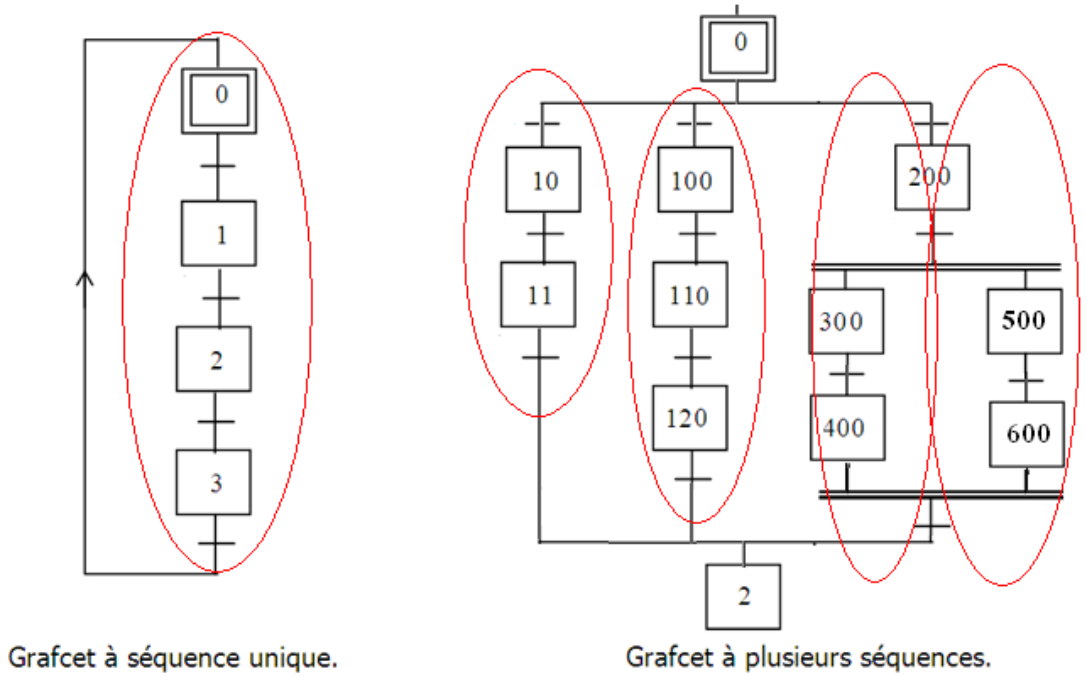


Figure IV-5: Notion de Séquence

2-4-2- Saut détapes et reprise de séquence

Le saut d'étapes permet de sauter une ou plusieurs étapes lorsque les actions associées sont inutiles à réaliser, La reprise de séquence (ou boucle) permet de reprendre, une ou plusieurs fois, une séquence tant qu'une condition n'est pas obtenue.

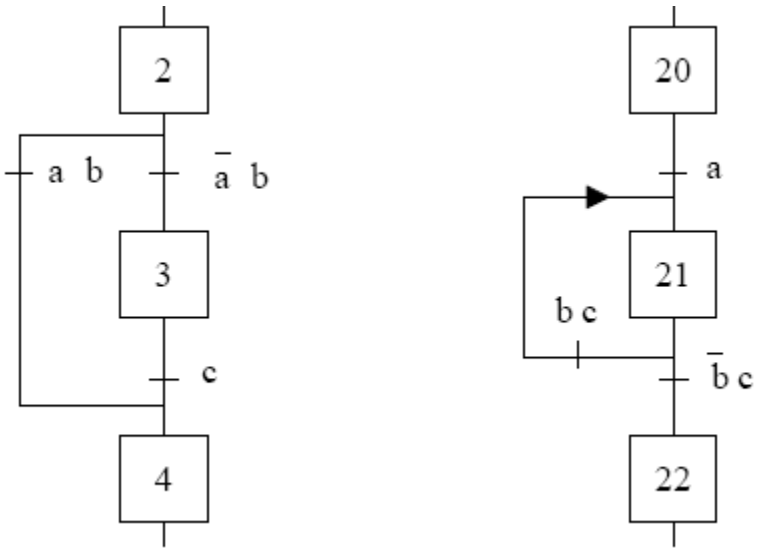


Figure IV-6: Saut détapes et reprise de séquence

2-4-3 - Aiguillage entre deuxouplusieursséquences (Divergence en OU)

On dit qu'il y a **Aiguillage** ou **divergence en OU** lorsque le grafcet se décompose en deux ou plusieurs séquences selon un choix conditionnel. Comme la divergence en OU on rencontre aussi la convergence en OU. On dit qu'il y a convergence en OU, lorsque deux ou plusieurs séquences du grafcet converge vers une seule séquence.

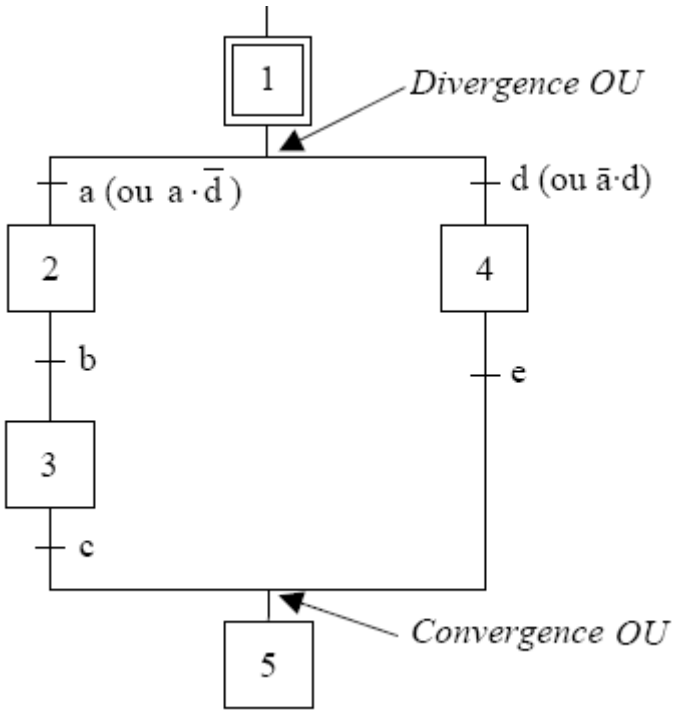


Figure IV-7 :Aiguillage entre deux ou plusieurs séquences

Si les deux conditions a et d sont à 1 simultanément, les étapes 2 et 4 vont devenir actives simultanément, situation non voulue par le concepteur. Donncelle doivent être des conditions **exclusives**

2-4-4 - Parallélisme entre deuxouplusieursséquences (ou séquences simultanées ou divergence convergence en ET) :

Au contraire de l'aiguillage où ne peut se dérouler qu'une seule activité à la fois, On dit qu'on se trouve en présence d'un parallélisme structurel, si plusieurs activités indépendantes pouvant se dérouler en parallèle. Le début d'une divergence en ET et la fin d'une convergence en ET d'un parallélisme structurel sont représentés par deux traits parallèles.

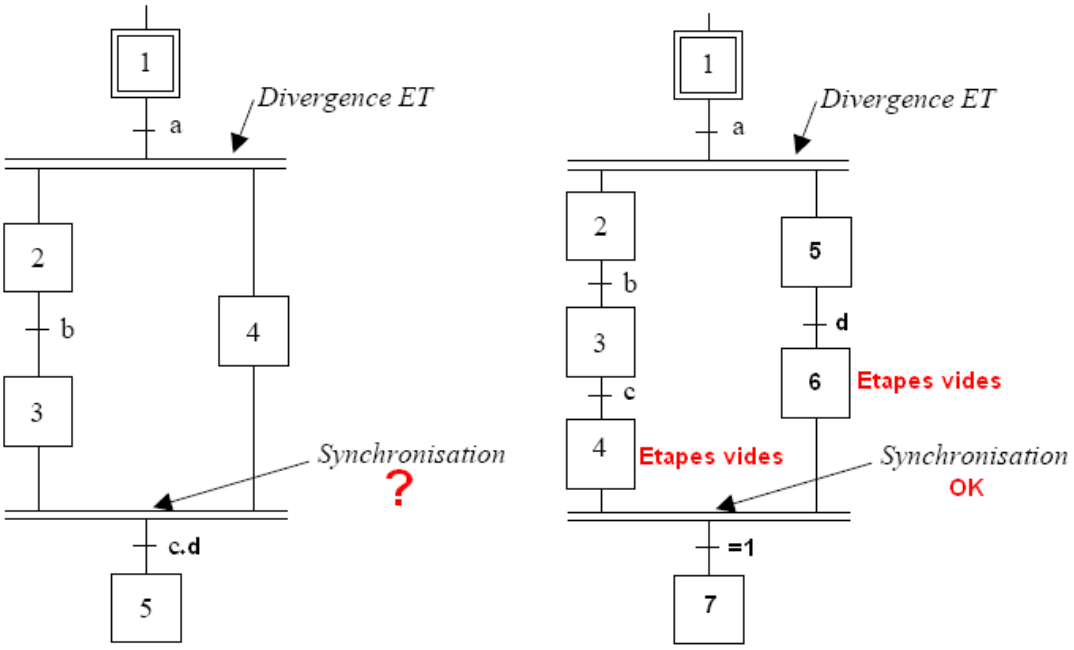


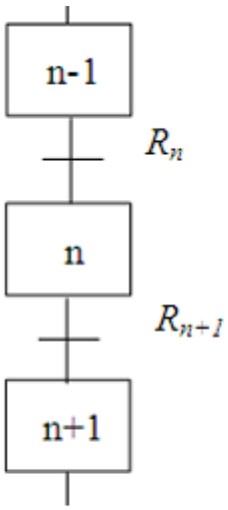
Figure IV-8: Parallélisme entre deux ou plusieurs séquences

La synchronisation permet d'attendre la fin de plusieurs activités se déroulant en parallèle, pour continuer par une seule.

2-5 - Mise en équation d'un grafcet:
2-5-1 - Régale Générale :

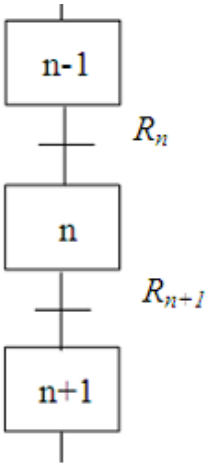
Pour qu'une étape soit activée il faut que :

- L'étape immédiatement précédente soit active ;
- La réceptivité immédiatement précédente soit vraie ;
- L'étape immédiatement suivante soit non active ;
- Après activation l'étape mémorise son état.



- SXn-1=Xn+1
- SXn=Xn-1.Rn
- SXn+1=Xn.Rn+1

Equation d'activation de l'étape de rang n



- RXn-1=Xn-1.Rn
- RXn=Xn.Rn+1
- RXn+1=Xn+1

Equation d'activation de l'étape de rang n

Figure IV-9: Régale Générale

3-Modules logiques ZelioLogic

3-1- présentation [12]

Zelio Soft 2 est un logiciel gratuit de configuration des modules logiques ZelioLogic, conçu pour gérer des automatismes simples. Et il peut :

-Programmation en langage ladder ou en langage de programmation de tâches (FBD)

- Simulation, suivi et supervision

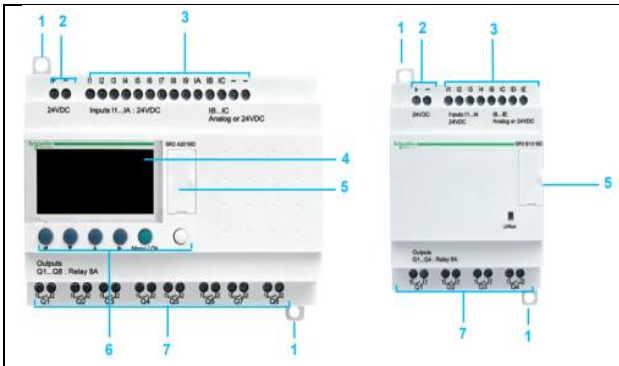
- Importer et télécharger des logiciels

- Imprimez des fichiers personnalisés

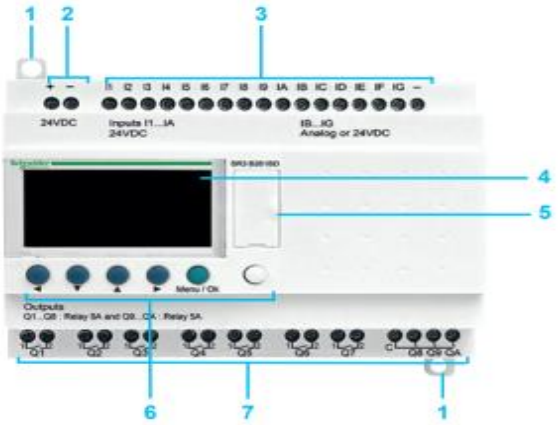
- Regroupement de logiciels automatiquement

3-2- type:

- ❖ Modules logiques compactes

 **Avecafficheur-10,12et20E/SSansafficheur-10,12et20E/S**	Les modules Zelio Logic compacts comprennent en face avant : 1 Deux pattes de fixation rétractables. 2 Deux bornes d'alimentation. 3 Desbornesderaccordementdesentrée. 4 Unafficheur LCDrétroéclairé de 4lignes de18caractères. 5 Un emplacement pour cartouche mémoire ou raccordement au PC ou interfacedecommunicationMode mou auterminauxdedialogueIHM(Magelis Small panel) ou interface Bluetooth. 6 Unclavierde6touchespourla 7programmationet le paramétrage. Desbornesderaccordementdessorties

- ❖ Modules logiques modulaires :

 **Avecafficheur-10et26E/S**	Lesmodules Zelio Logic modulaires comprennent en face avant : 1 Deuxpattesdefixationrétractables. 2 Deuxbornesd'alimentation. 3 Desbornesderaccordementdesentrées. 4 Unafficheur LCDrétroéclairé de 4lignes de18caractères. 5 Un emplacement pour cartouche mémoire ou raccordement au PC ou interfacedecommunicationModemou auterminauxdedialogueIHM(Magelis Small panel) ou interface Bluetooth. 6 Unclavierde6touchespourla 7programmationet le paramétrage. Desbornesderaccordementdessorties

3-3 langage de programmation:
3-3-1 langage contacte (LADDER):

- Definition

Le langage à contacts permet d'écrire un programme LADDER avec des fonctions élémentaires,desblocsfonctionnelsélémentairesetdesblocsfonctionnelsdérivés, ainsi qu'avec des contacts, des bobines et des variables.
Les contactes les bobines peuventêtrecommentés. Du texte peutêtreinséré

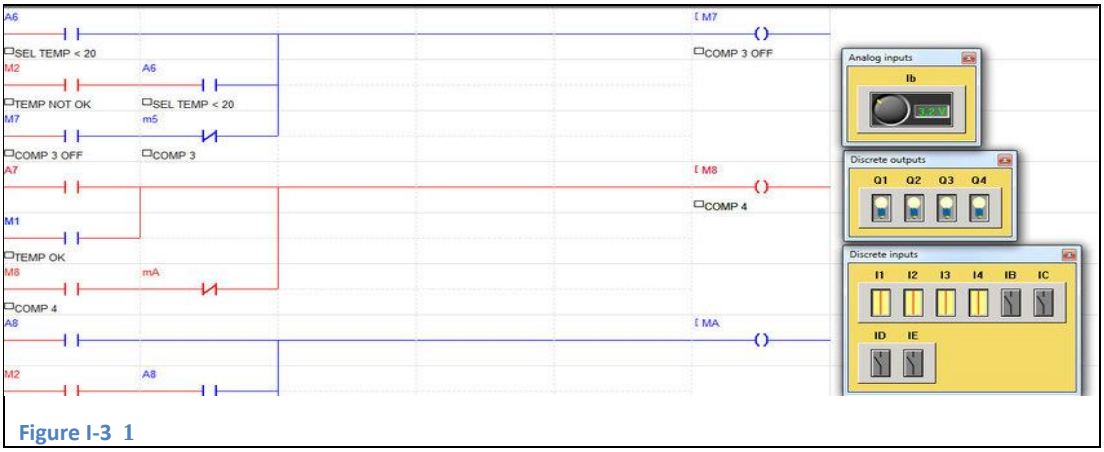


Figure IV-10: langage contacte (LADDER)

3-3-2 langage FBD:

Blocs Fonctionnels (FBD : Fonction Bloc Diagram) : Langage graphique ou 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

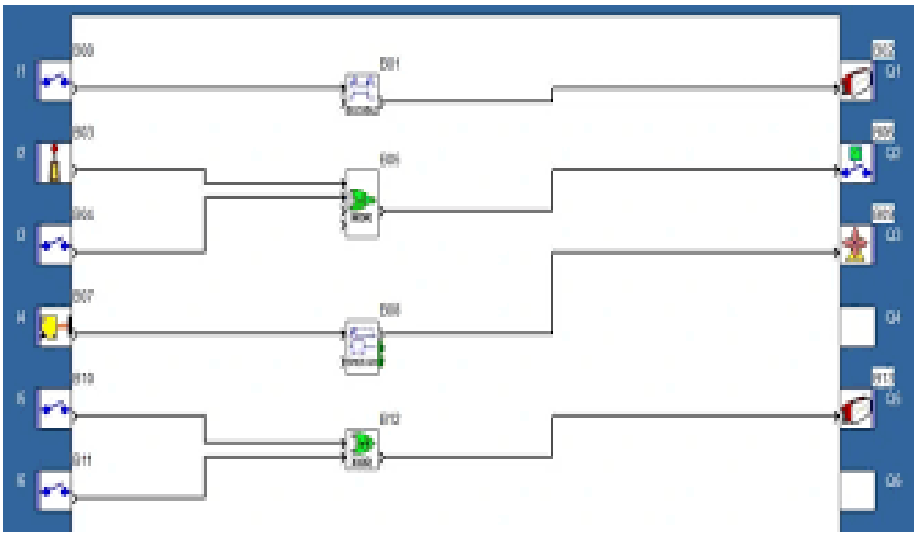


Figure IV-11: langage FBD

4- automgen

4-1 definetion

Automgen permet la création de programmes avec des langages normalisés (norme CEI-1131-3, SysML), la simulation des programmes sur PC, la génération et le téléchargement du code pour des automates programmables ou autres cibles (Arduino, PIC, etc.). Automgen permet également la création d'applications de supervision locale ou sur Internet ainsi que des simulations 3D**.**

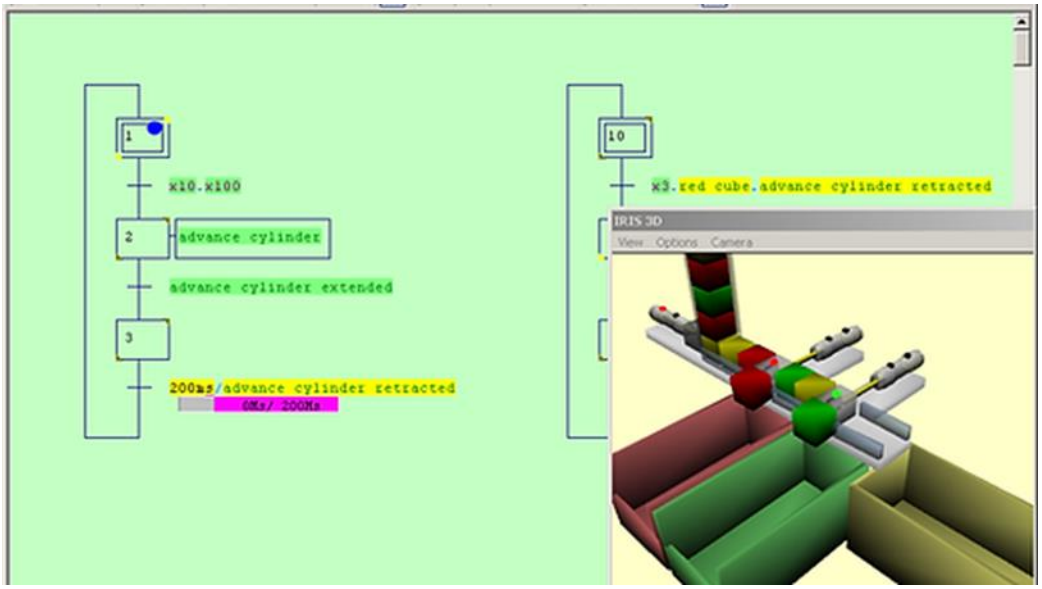


Figure IV-12: utiliser Automgen

4-2-L' application qu'il vous faut:

Le logiciel d'enseignement de l'automatisme le plus utilisé en France

Automgen est la référence des logiciels d'automatismes universels. Créé il y a 30 ans, ce logiciel n'a cessé d'évoluer pour tirer partie des dernières technologies disponibles.

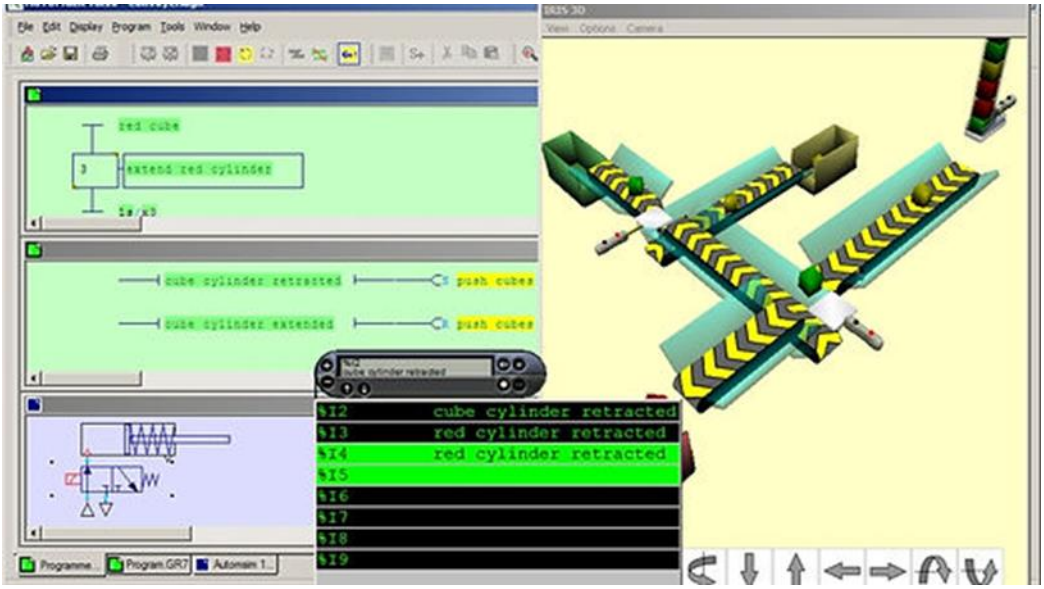


Figure IV-13: un programme automgen

4-3- LA VUE GENERALE

Documentation Ressource- Mise en œuvre du logiciel Automgen V7– page31) LANCER LE LOGICIEL AUTOMGEN V7Double cliquez sur son icône La fenêtre ci-contre apparaît2) LE PROJET Un projet regroupe l'ensemble des éléments composant une application (folios, symboles configuration, objets IRIS, etc ...)Pour créer un nouveau projet Pour ouvrir un projet existant Pour sauvegarder un projet Pensez-y régulièrement ...3) LA VUE GENERAL Elément central de la gestion des applications, le navigateur permet un accès rapide aux différents éléments d'une application : folios, symboles, configuration, impressions, objets IRIS, etc ..

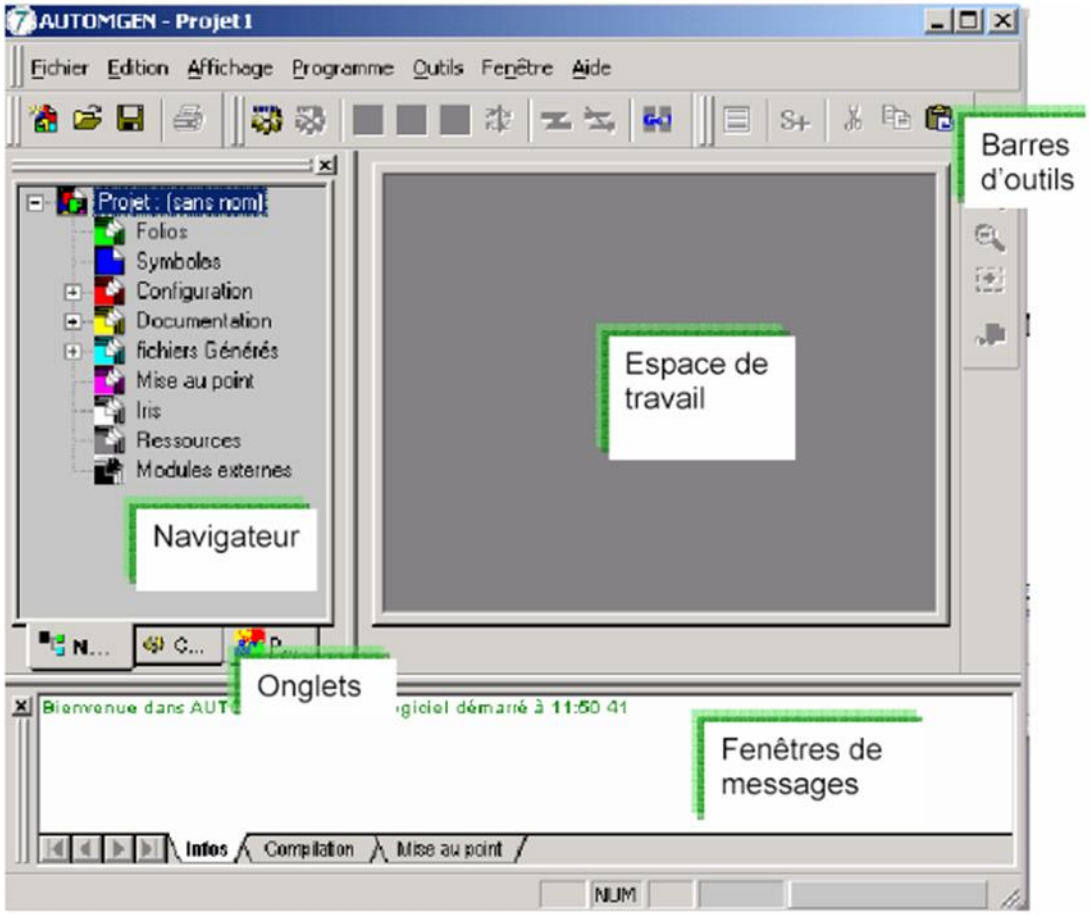
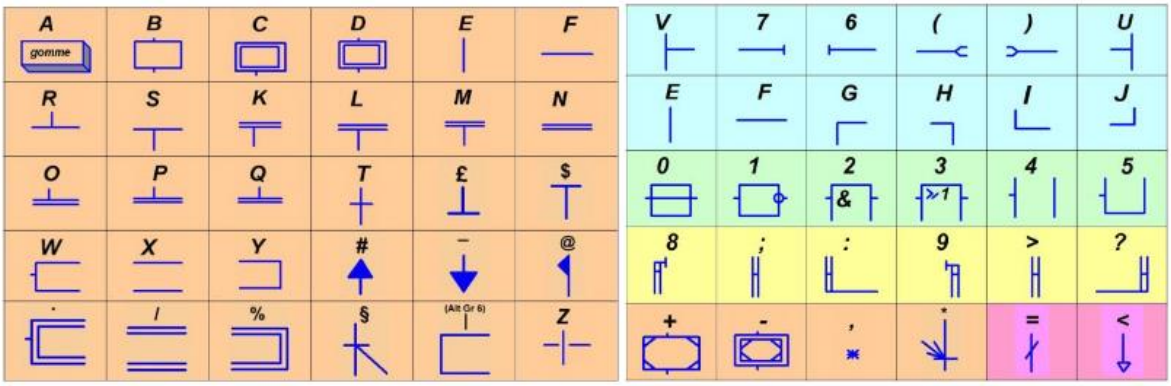


Figure IV-14: LA VUE GENERALE

4-4-Raccourcis clavier



5-programmation et simulation

5-1-Exemple 01 : (monte charge industriel)

❖ Description

_ Le système permet de contrôler un ascenseur à deux étages, en appuyant sur le bouton (M), le moteur tourne dans le sens ascendant (M+), en atteignant le premier étage, il affecte la prise de mouvement (H) et le moteur s'arrête, en appuyant sur le bouton (D) fait tourner le moteur dans le sens de la descente (M-), lorsque l'atteinte du rez-de-chaussée affecte la prise de mouvement (L), le moteur s'arrête, et les deux cas peuvent être arrêtés avec le bouton (S).

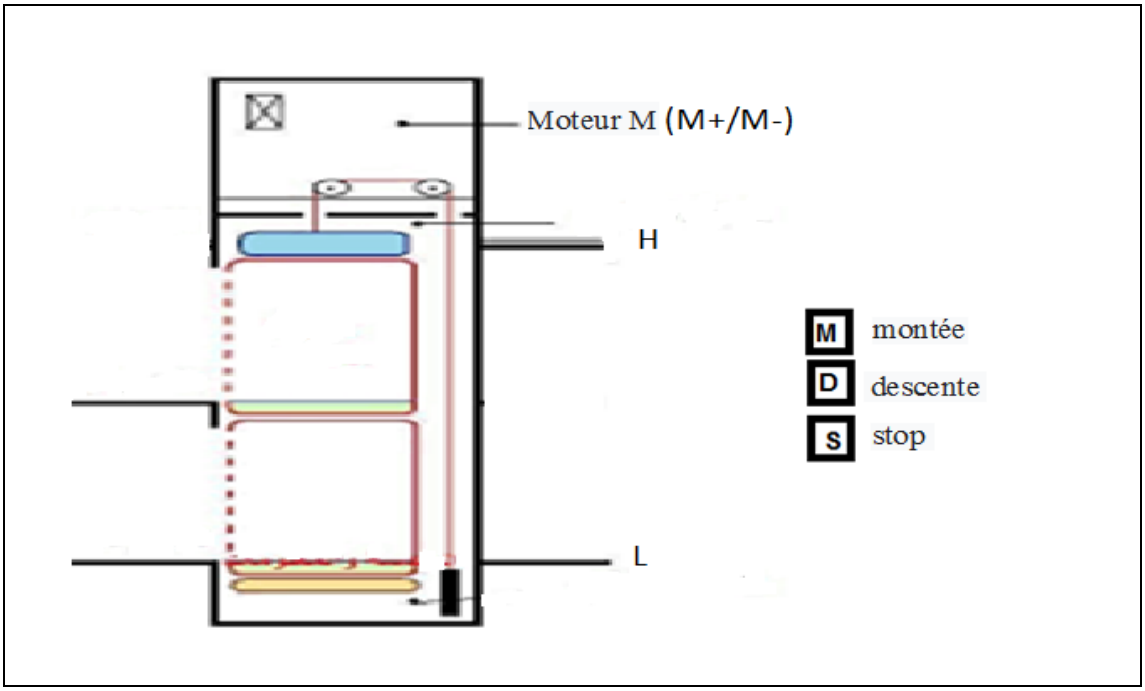


Figure IV-15: Monte charge industrial

M : bouton vers le haut D : bouton vers le bas S : bouton d'arrêt

H : prendre le premier étage L : prendre le rez-de-chaussée

Moteur : M + (haut) M- (haut)

❖ DiagrammeGrafcets par Automgen :

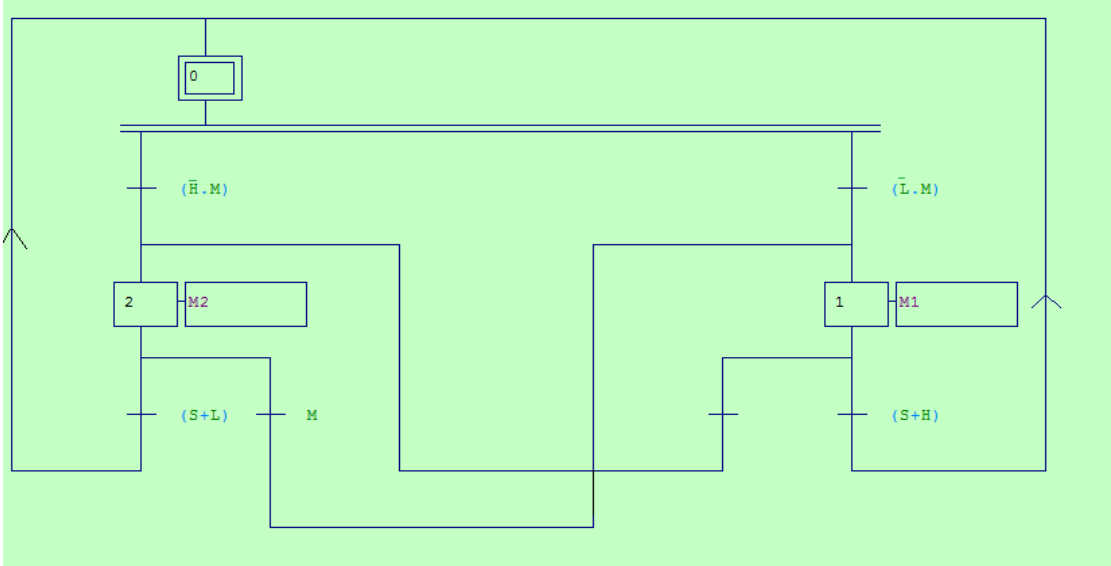


Figure IV-16: GRAFCET -Monte charge industrial

❖ Équations d'activation et d'arrêt :

SET	RSET
$SX_0 = X_1(S+H) + X2(S+L) + \overline{X_0}.\overline{X_1}.\overline{X_2}$	$RX_0 = X_0(\bar{L}.M) + X_0(\overline{H}.D)$
$SX_1 = X_0(\bar{L}+M)X_2.M$	$RX_1 = X_1(S+H) + X_1.D$
$SX_2 = X_0(\bar{H}+M)X_2.M$	$RX_2 = X_2(S+L) + X_2.M$

$/M_1=X_1$ $/M_2=X_2$

❖ programmation et simulation par ZILIO SOFT 2(SYMBOLE LADDER)

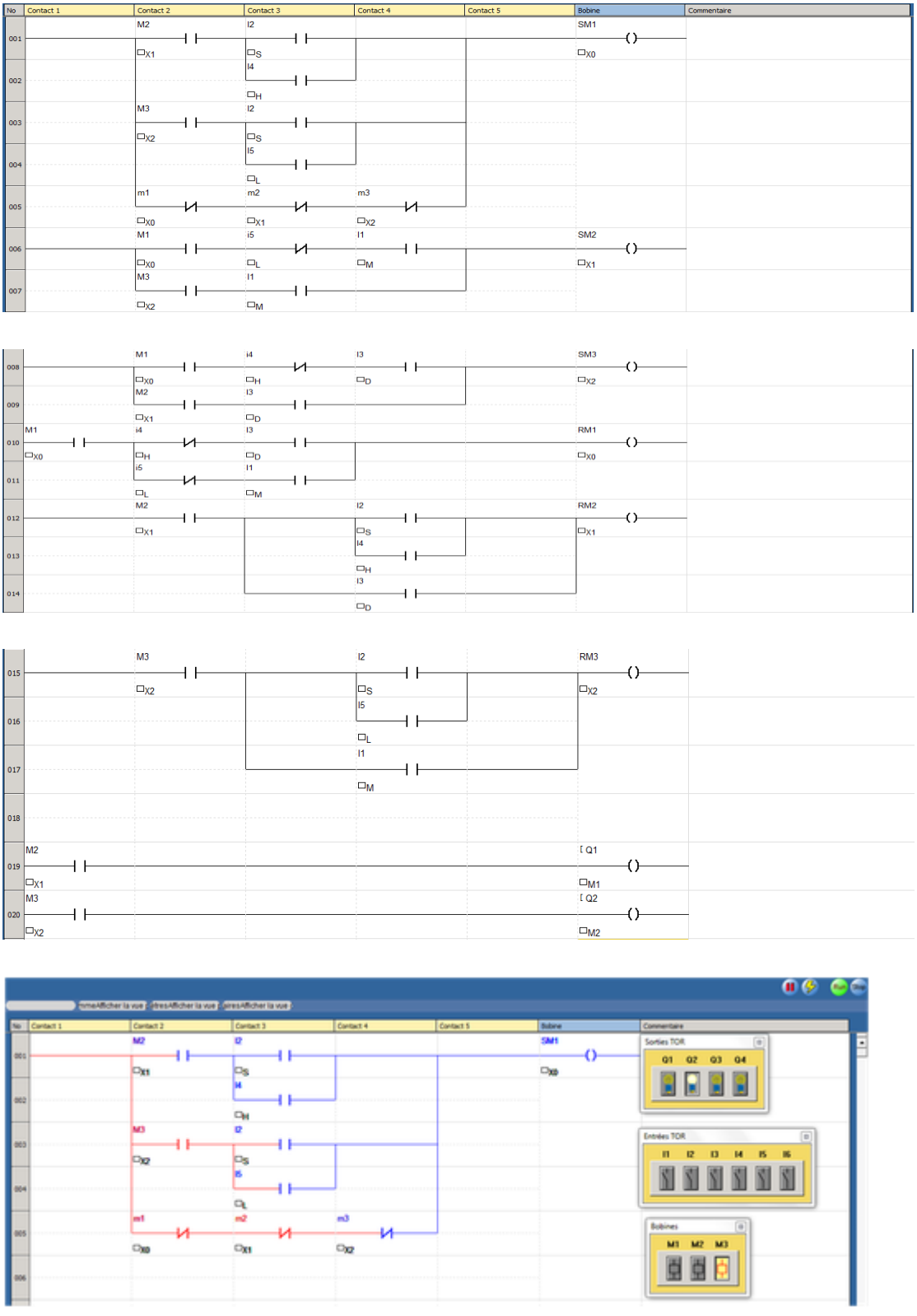


Figure IV-17:SIMULATION- Monte charge industriel

5-2-Exemple 2: (ascenseur quatre étages)

- Description

Pour programmer le système de contrôle d'un ascenseur à quatre étages, nous avons défini les entrées suivantes : entrée P pour contrôler la position de la personne (P1/P2/P3/P4) et E (prise de mouvement) pour contrôler la position de l'ascenseur (E1/E2/E3/E4), et B (prise de mouvement) la porte de l'ascenseur est fermée et l'autre A est la porte de l'ascenseur est ouverte, et les sorties suivantes sont : M (montée), D (descendante), OV (ouvert), FE (fermé)

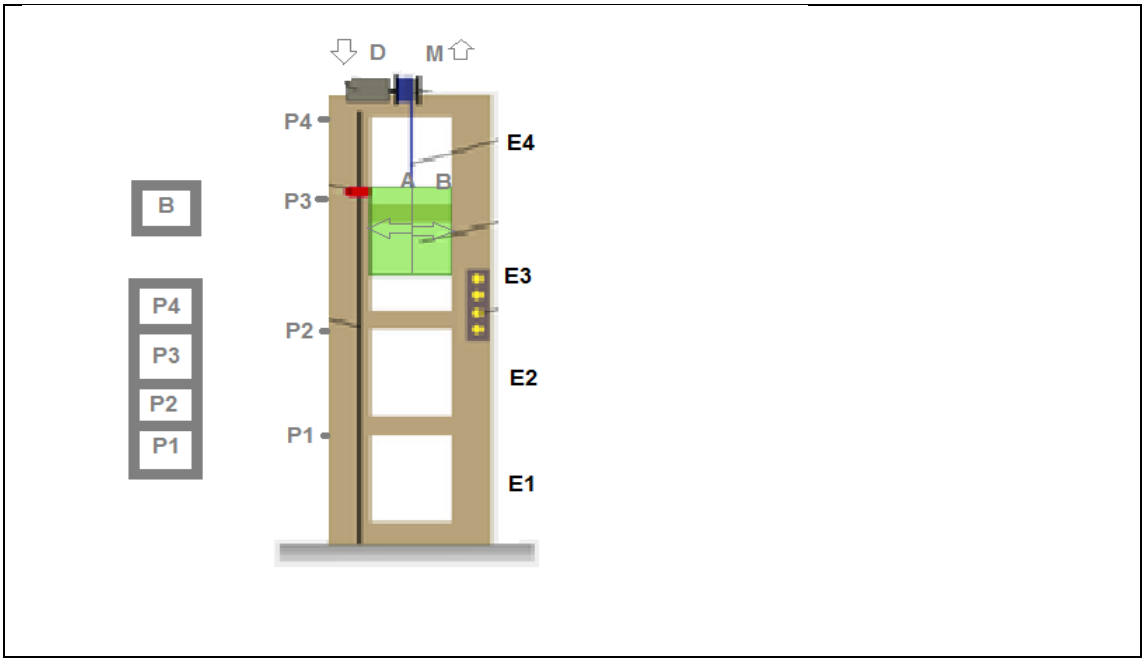


Figure IV-18: ascenseur quatre étages

- Bouton d'appel intérieur et extérieur : E1-E2-E3-E4
- Capteur de position : p1-p2-p3-p4
- Capter Position de porte : A (ouverte) B (fermée)
- Moteur M : (m) montée (d) mouvement descente

❖ DiagrammeGrafcets par Automgen :

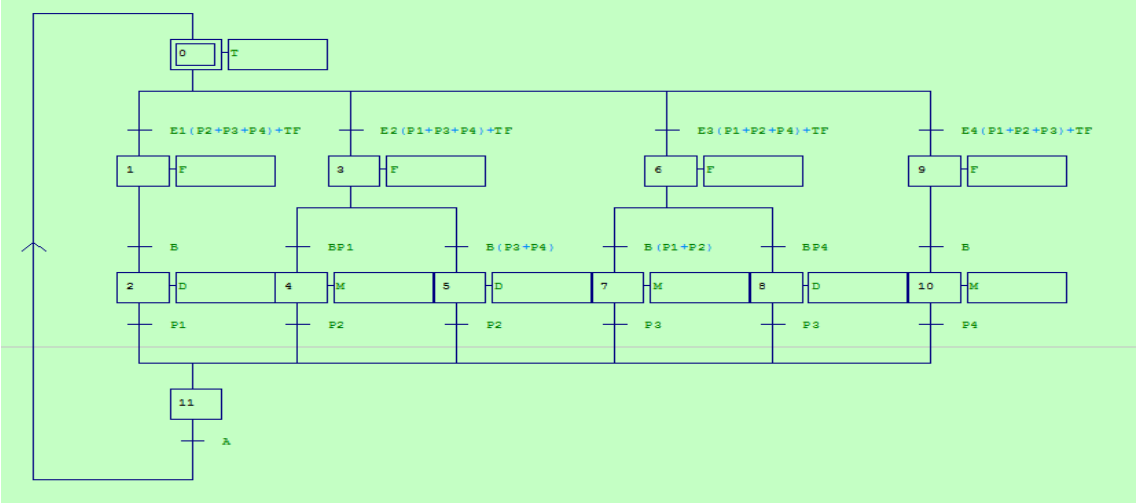


Figure IV-19: GRAFCET-ascenseur quatre étages

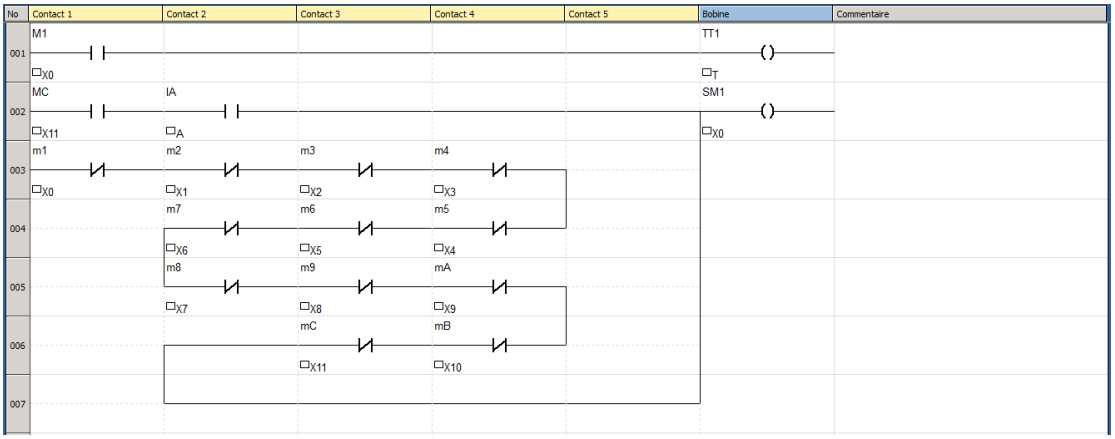
❖ Équations d'activation et d'arrêt :

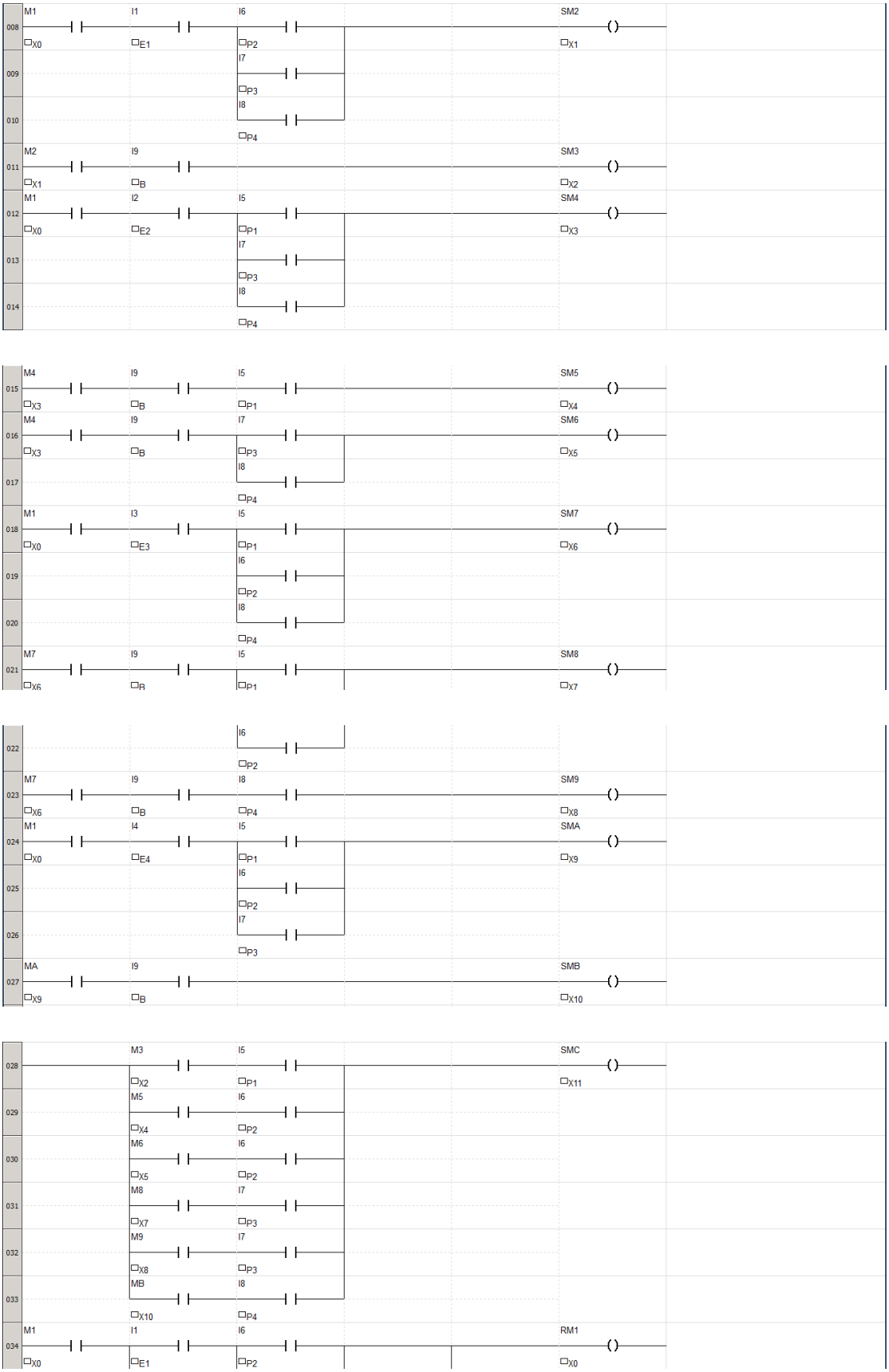
SET	*RSET*
$SX_0 = X_{11}\,.A + (\overline{X_0}.\overline{X_1}.\overline{X_2}.\overline{X_3}.\overline{X_4}.\overline{X_5}.\overline{X_6}.\overline{X_7}.\overline{X_8}.\overline{X_9}.\overline{X_{10}}.\overline{X_{11}})$	$RX_0 = X_0.E_1(P_2+P_3+P_4) + X_0.E_2(P_1+P_3+P_4) + X_0.E_3(P_1+P_2+P_4) + X_0.P_4(P_1+P_2+P_3) + X2.P1 + X4.P2 + X5.P2 + X7.P3 + X8P3 + X10.P4 + B$
$SX_1 = X_0\,.E1(P_2+P_3+P_4)$	$RX_1 = X_1.B$
$SX_2 = X_1\,.B$	$RX_2 = X_2.P_1$
$SX_3 = X_0\,.E2(P_1+P_3+P_4)$	$RX_3 = X_3.B(P_1) + X_3.B(P_3+P_4)$
$SX_4 = X_3\,.B(P_1)$	$RX_4 = X_4.P_2$

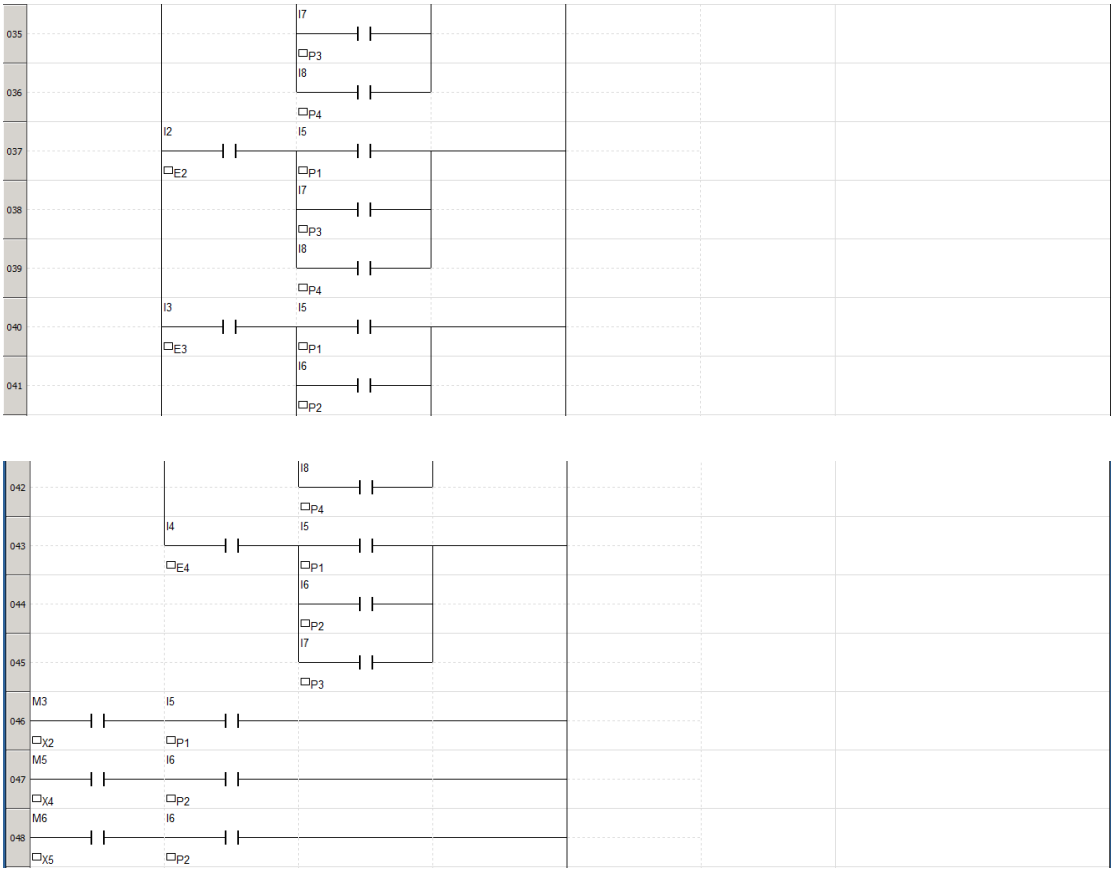
$SX_5 = X_3 \,.\, B(P_3 + P_4)$	$RX_5 = X_5 . P_2$
$SX_6 = X_0 \,.\, E3(P_1 + P_2 + P_4)$	$RX_6 = X_6 . B(P_1 + P_2) + X_6 . B(P_4)$
$SX_7 = X_6 \,.\, B(P_1 + P_2)$	$RX_7 = X_7 . P_3$
$SX_8 = X_6 \,.\, B(P_4)$	$RX_8 = X_8 . P_3$
$SX_9 = X_0 \,.\, E4(P_1 + P_2 + P_3)$	$RX_9 = X_9 . B$
$SX_{10} = X_9 \,.\, B$	$RX_{10} = X_{10} . P_4$
$SX_{11} = X_2 \,.\, P_1 + X_4 . P_2 + X_5 . P_2 + X_7 . P_3 + X_8 . P_3 + X_{10} . P_4$	$RX_{11} = X_{11} . A$

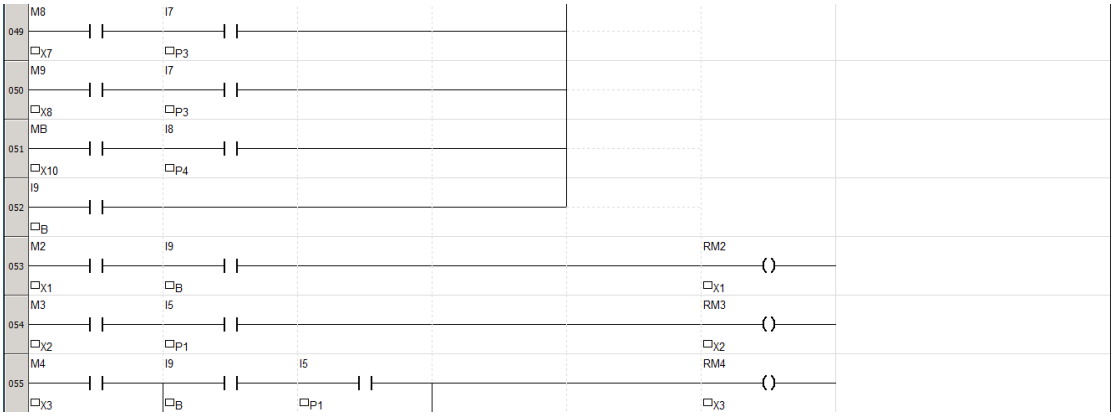
MO=X_4+X_7+X_{10} ;DE=X_2+X_5+X_8 ;FE=X_1+X_3+X_6+X_9+X_0;OV=X_{11} ; T =X_0

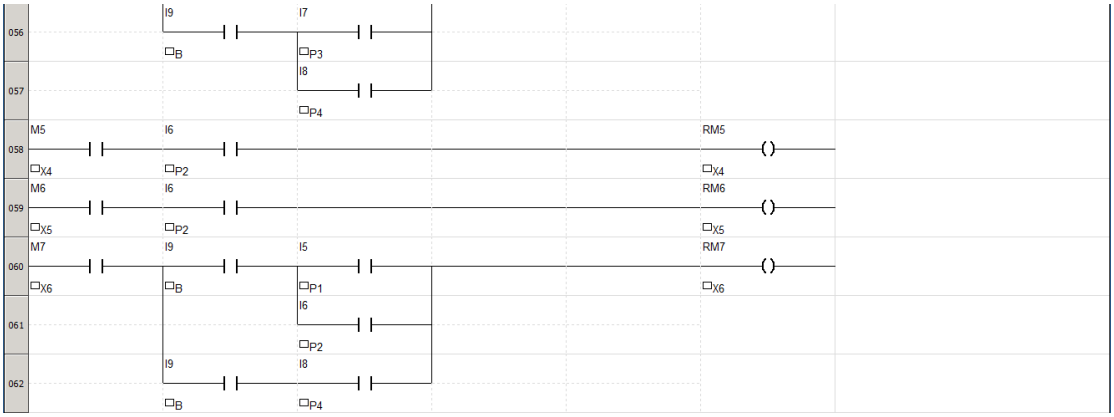
- programmation et simulation par ZILIO SOFT 2(SYMBOLE LADDER)

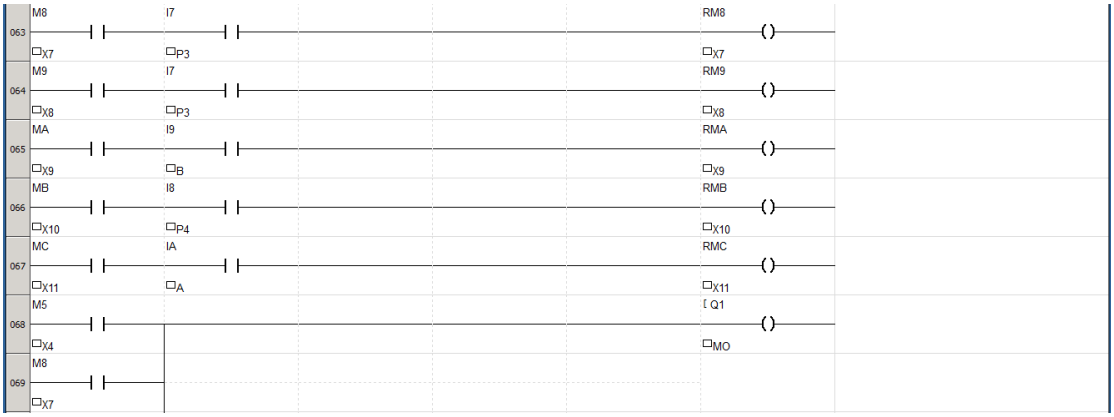


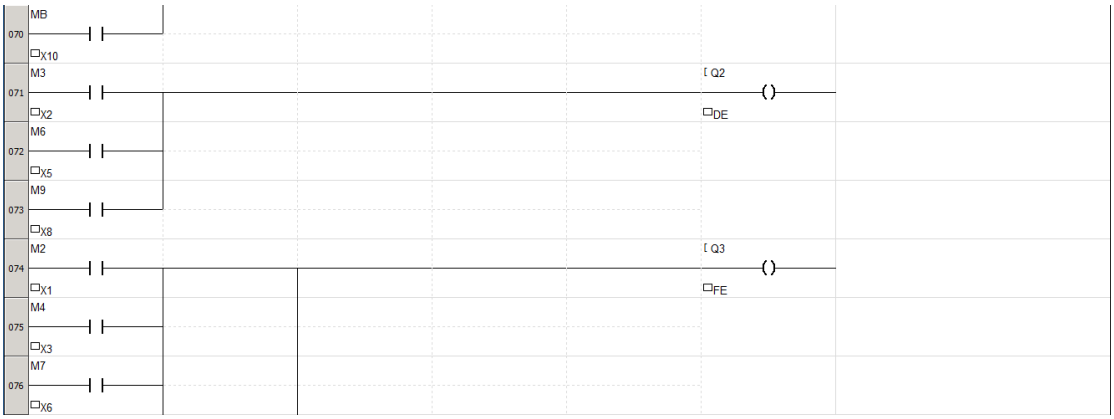

008 M1 X0 I1 E1 I6 P2 SM2 X1
009 I7 P3
010 I8 P4
011 M2 X1 I9 B SM3 X2
012 M1 X0 I2 E2 I5 P1 SM4 X3
013 I7 P3
014 I8 P4
015 M4 X3 I9 B I5 P1 SM5 X4
016 M4 X3 I9 B I7 P3 SM6 X5
017 I8 P4
018 M1 X0 I3 E3 I5 P1 SM7 X6
019 I6 P2
020 I8 P4
021 M7 X6 I9 B I5 P1 SM8 X7
022 I6 P2
023 M7 X6 I9 B I8 P4 SM9 X8
024 M1 X0 I4 E4 I5 P1 SMA X9
025 I6 P2
026 I7 P3
027 MA X9 I9 B SMB X10
028 M3 X2 I5 P1 SMC X11
029 M5 X4 I6 P2
030 M6 X5 I6 P2
031 M8 X7 I7 P3
032 M9 X8 I7 P3
033 MB X10 I8 P4
034 M1 X0 I1 E1 I6 P2 RM1 X0

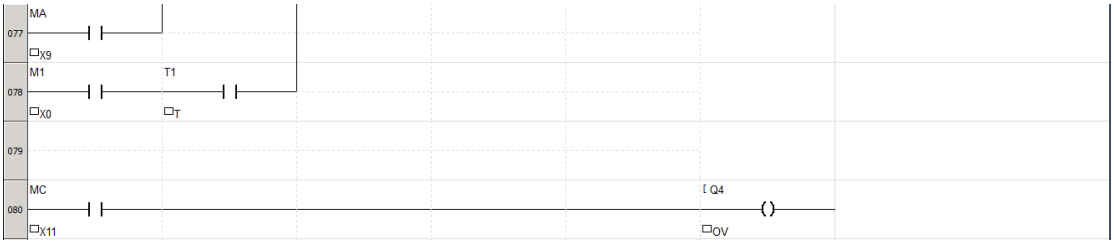

035
I7
P3
I8
036
P4
037
I2
I5
E2
P1
038
I7
P3
039
I8
P4
040
I3
I5
E3
P1
041
I6
P2
042
I8
P4
043
I4
I5
E4
P1
044
I6
P2
045
I7
P3
046
M3
I5
X2
P1
047
M5
I6
X4
P2
048
M6
I6
X5
P2


049
M8
I7
X7
P3
050
M9
I7
X8
P3
051
MB
I8
X10
P4
052
I9
B
053
M2
I9
RM2
X1
B
X1
054
M3
I5
RM3
X2
P1
X2
055
M4
I9
I5
RM4
X3
B
P1
X3


056
I9
I7
B
P3
057
I8
P4
058
M5
I6
RM5
X4
P2
X4
059
M6
I6
RM6
X5
P2
X5
060
M7
I9
I5
RM7
X6
B
P1
X6
061
I6
P2
062
I9
I8
B
P4


063
M8
I7
RM8
X7
P3
X7
064
M9
I7
RM9
X8
P3
X8
065
MA
I9
RMA
X9
B
X9
066
MB
I8
RMB
X10
P4
X10
067
MC
IA
RMC
X11
A
X11
068
M5
Q1
X4
MO
069
M8
X7


070
MB
X10
071
M3
Q2
X2
DE
072
M6
X5
073
M9
X8
074
M2
Q3
X1
FE
075
M4
X3
076
M7
X6


077
MA
X9
078
M1
T1
X0
T
079
080
MC
Q4
X11
OV

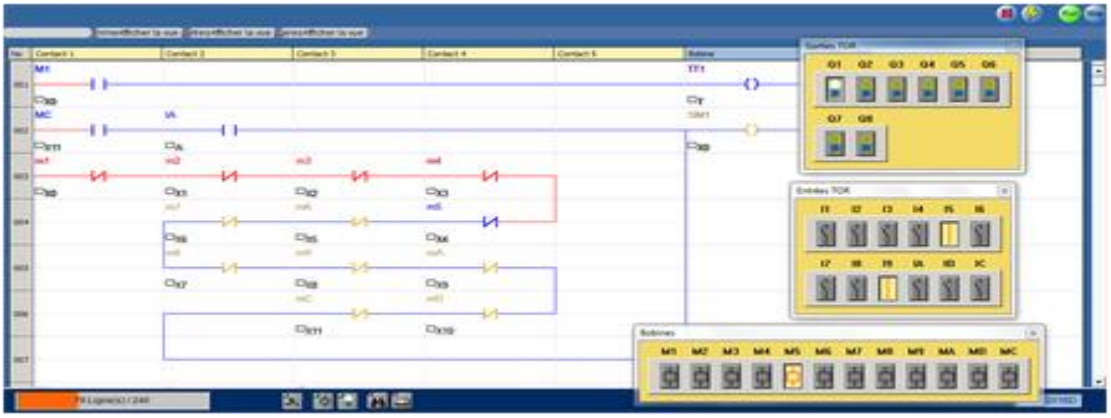


Figure IV-20: SIMULATION-ascenseur quatre étages

5-3-Exemple 3:(machine de melange et remplissage de liquide)

- Description

Le système permet de mélanger deux substances différentes, A et B, en appuyant sur le bouton P, ce qui entraîne l'ouverture des deux électrodes (EV1/EV2) et la vidange de chaque substance dans un réservoir. . Et chacun de (EV3/EV4) s'ouvre pour décharger les deux substances dans le mélangeur, et le moteur de mélange M démarre, et au contact du mélange collecteur Ch, le (EV3/EV4) et le moteur M se ferment, et le moteur du rail fonctionne jusqu'à ce que le ballon atteigne le collecteur (S), puis il s'ouvre (EV5). Et il est rempli jusqu'à ce que le produit atteigne la pince (Ck) pour répéter le processus

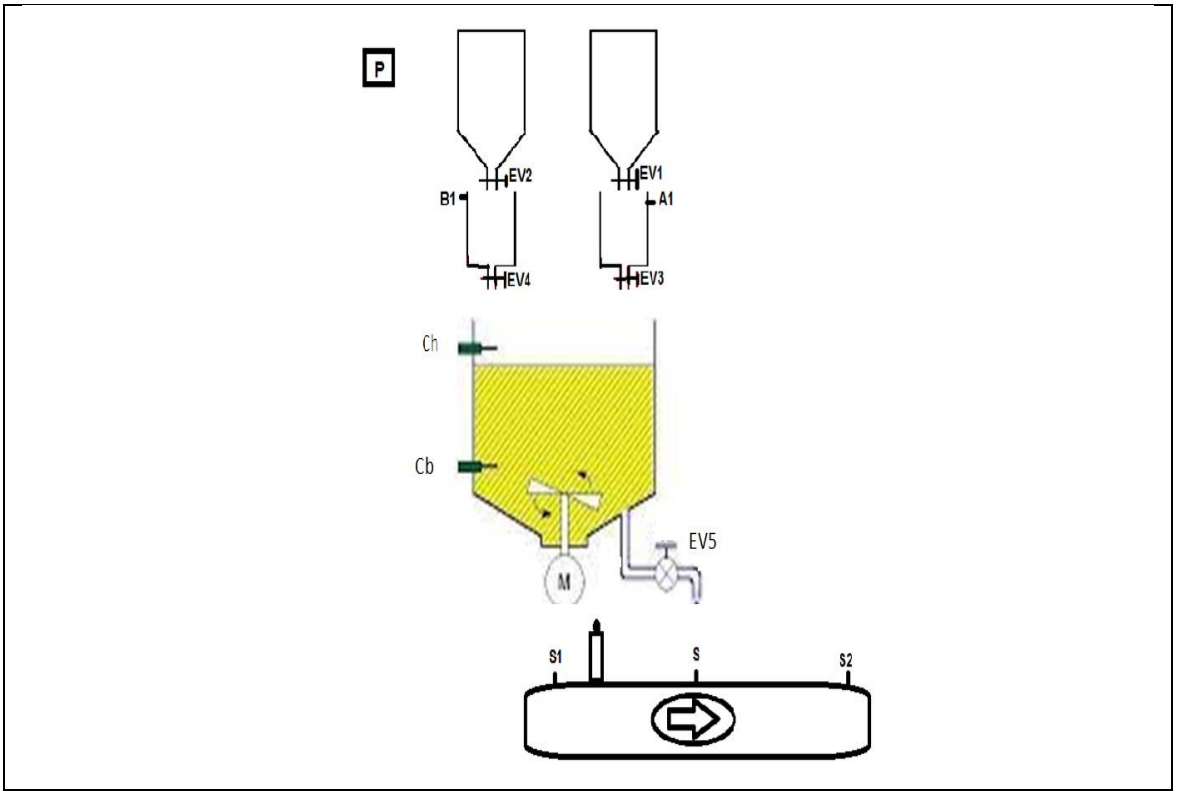


Figure IV-21: machine de melange et remplissage de liquide

- P : Bouton
- EV1-EV2-EV36EV4-EV5 : Electrovanne
- A1-B1-Ch-Cb-S : Capteur
- M : moteur
- M1 : tapis roulant motorisé

❖ DiagrammeGrafcets par Automgen :

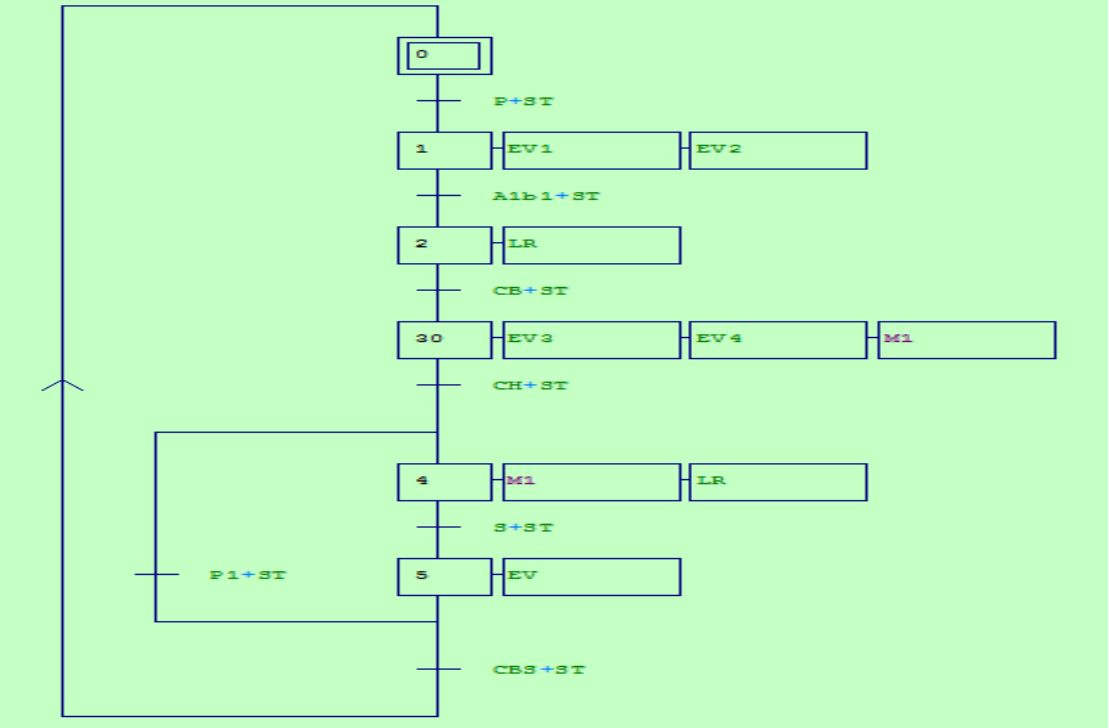


Figure IV-22: GRAFCET-machine de melange et remplissage de liquide

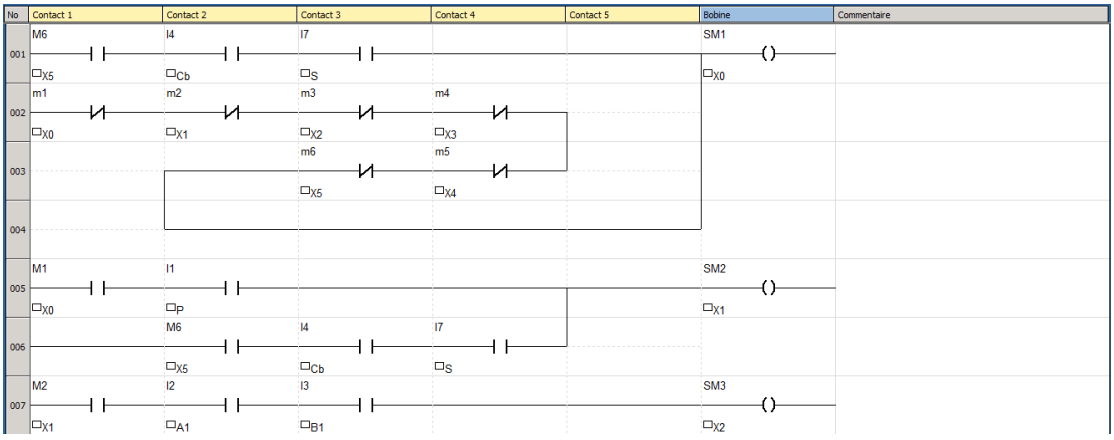
❖ Équations d'activation et d'arrêt (SET/RESET):

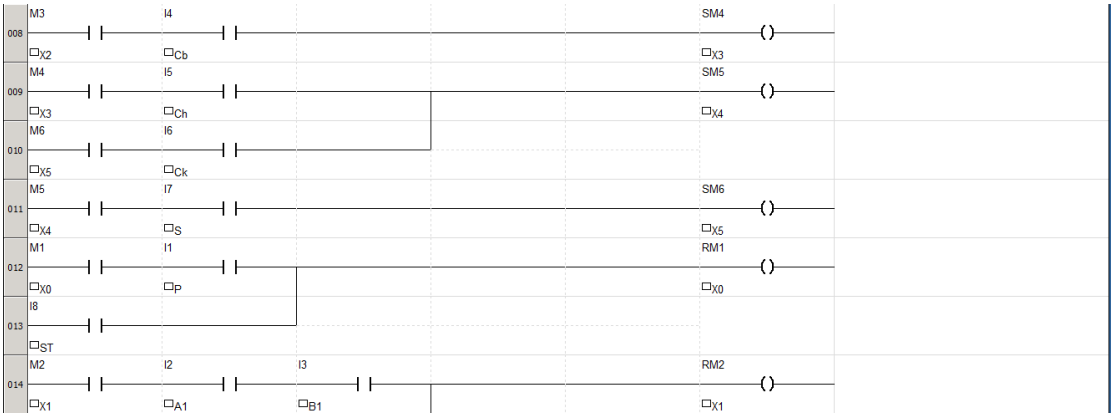
SET	RSET
$SX_0 = X_5(\mathrm{CB}.\mathrm{S}) + \overline{X_0}.\overline{X_1}.\overline{X_2}.\overline{X_3}.\overline{X_4}.\overline{X_5}$	$RX_0 = X_0.P + X_0.ST$
$SX_1 = X_0\,.P$	$RX_1 = X_1(A_1.B_2) + X_0.ST$
$SX_2 = X_1(A_1.B_1)$	$RX_2 = X_2.\mathrm{CB} + X_2.ST$
$SX_3 = X_2\,.CB$	$RX_3 = CH + X_3.ST$
$SX_4 = X_3.CH + X_5.CK$	$RX_4 = X_4.\mathrm{S} + X_4.ST$

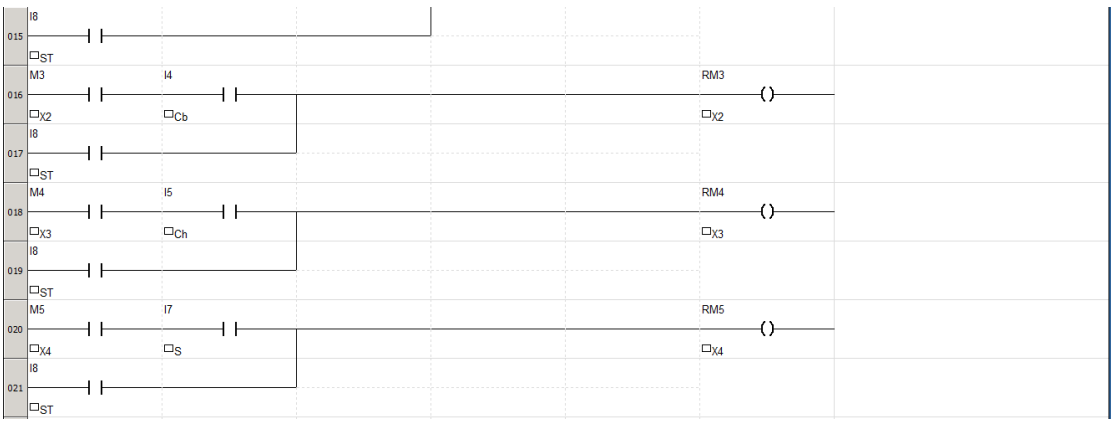
$SX_5 = X_4 \,.S$	$RX_5 = X_5(\mathrm{CB.S}) + X_5.ST + X_5.CK + X_5.ST$

$EV_1 = X_1$;$EV_2 = X_2$;EV_3=X_3 ;EV_4=X_3 ;EV_5=X_5 ;LR=X2+X4 ,M=X3 ;Mt=X4

- programmation et simulation par ZILIO SOFT 2(SYMBOLE LADDER)







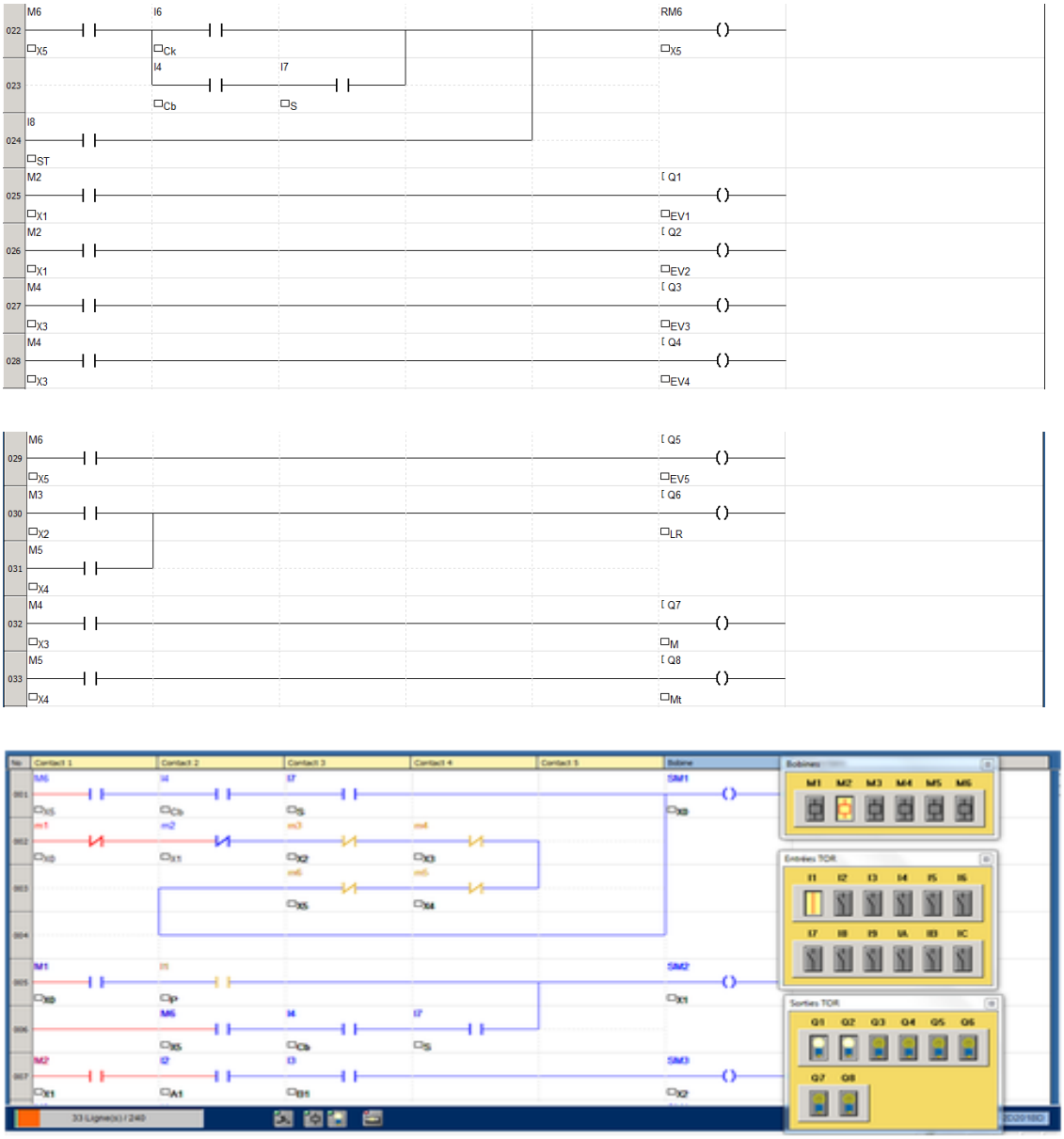


Figure IV-23: SIMULATION-machine de melange et remplissage de liquide

5-4-EXEMPLE 04:(pulvérisateur agricole et péristaltique)

- ❖ Description

Lorsque le bouton P est enfoncé pendant un temps T1=4s, le collecteur EV1 ouvre le robinet et avec lui allume la pompe M et le moteur M1 commence à tourner pendant T2=4H, après le temps T2 le moteur s'arrête, et après un temps T3=10s, le collecteur EV2 ouvre le robinet et démarre le moteur M2 pendant le temps T4=4H M2 après 4 heures

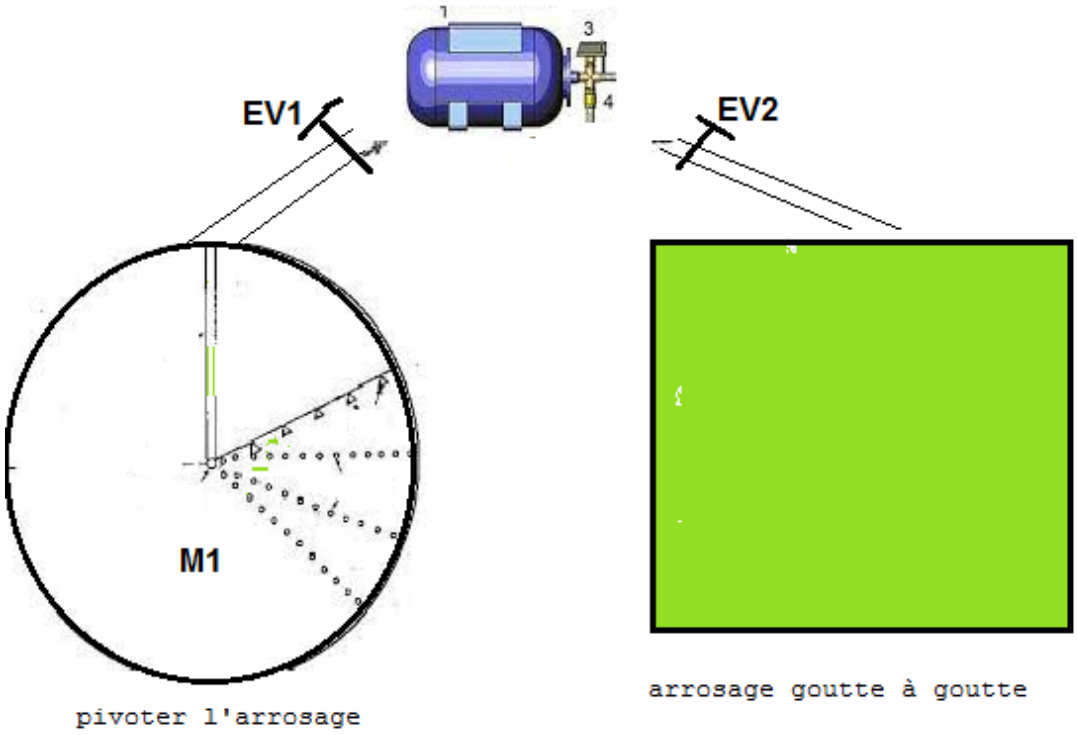


Figure IV-24: pulvérisateur agricole et péristaltique

- P : Bouton Démarrer
- EV1-EV2 : électrovanne
- M : moteur
- M1 : moteur1

❖ Diagramme Grafcets par Automgen :

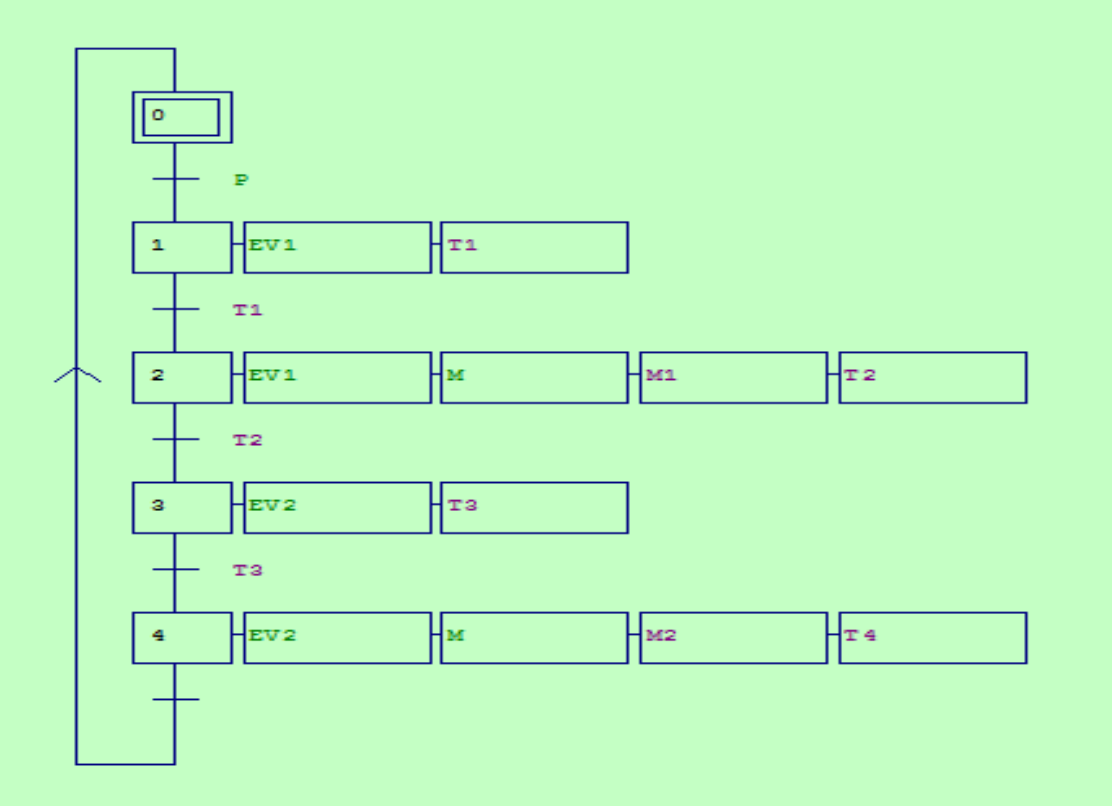


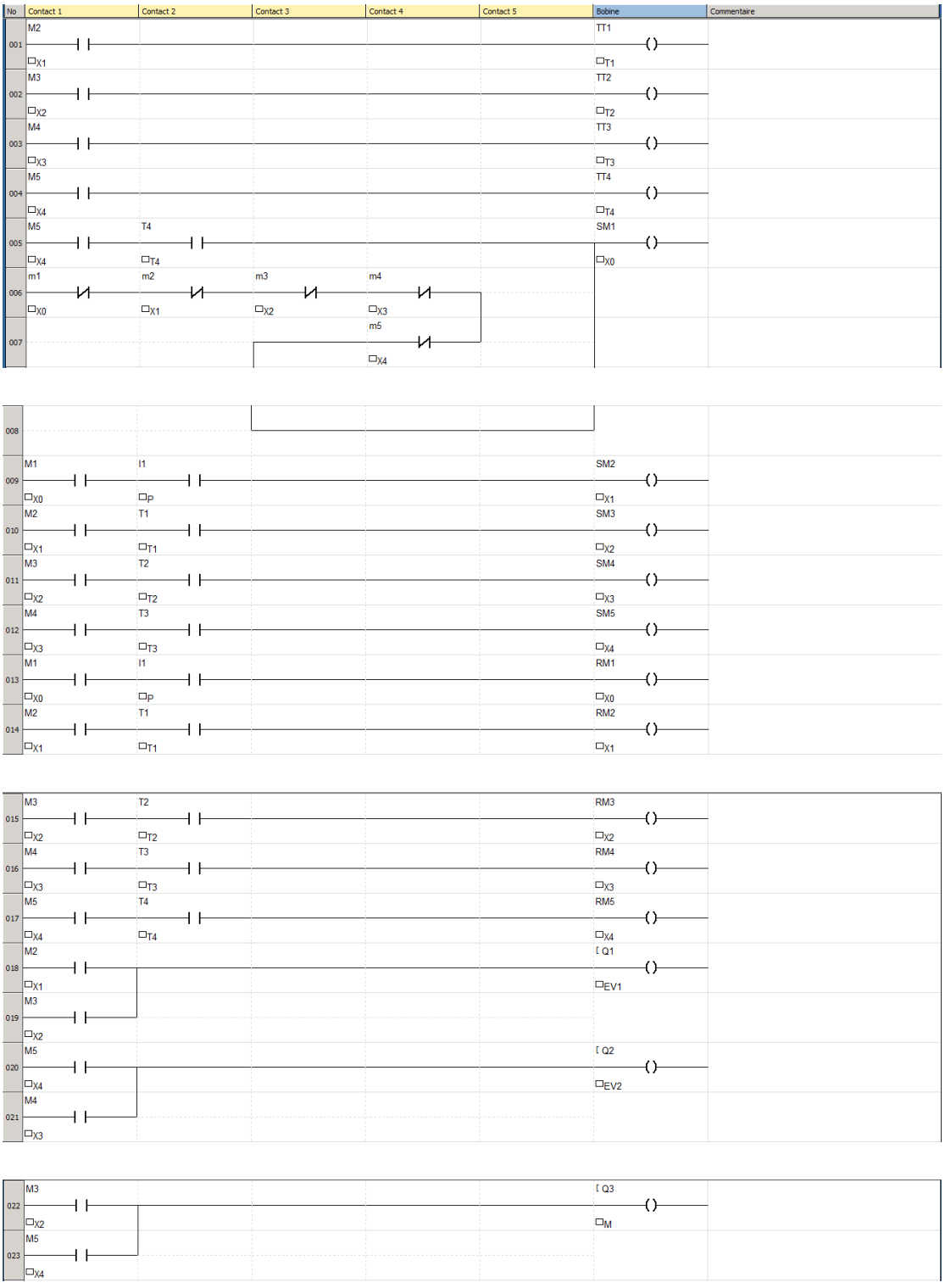
Figure IV-25:GRAFCET- pulvérisateur agricole et péristaltique

❖ Équations d'activation et d'arrêt (SET/RESET):

SET	RSET
$SX_0 = X_4.T_4 + \overline{X_0}.\overline{X_1}.\overline{X_2}.\overline{X_3}.\overline{X_4}$	$RX_0 = X_0 \,.P$
$SX_1 = X_0 \,.P$	$RX_1 = X_1 \,.T_1$
$SX_2 = X_1 \,.T_1$	$RX_2 = X_2 \,.T_2$
$SX_3 = X_2 \,.T_2$	$RX_3 = X_3 \,.T_3$
$SX_4 = X_3 \,.T_3$	$RX_4 = X_4 \,.T_4$

EV1=x1+x2 ;EV2=x3+x4 ;M=X2 +X4 ;M1=X2 ;T1=x1 ;T2=X2 ;T3=X3 T4=X4

- ❖ programmation et simulation par ZILIO SOFT 2(SYMBOLE LADDER)



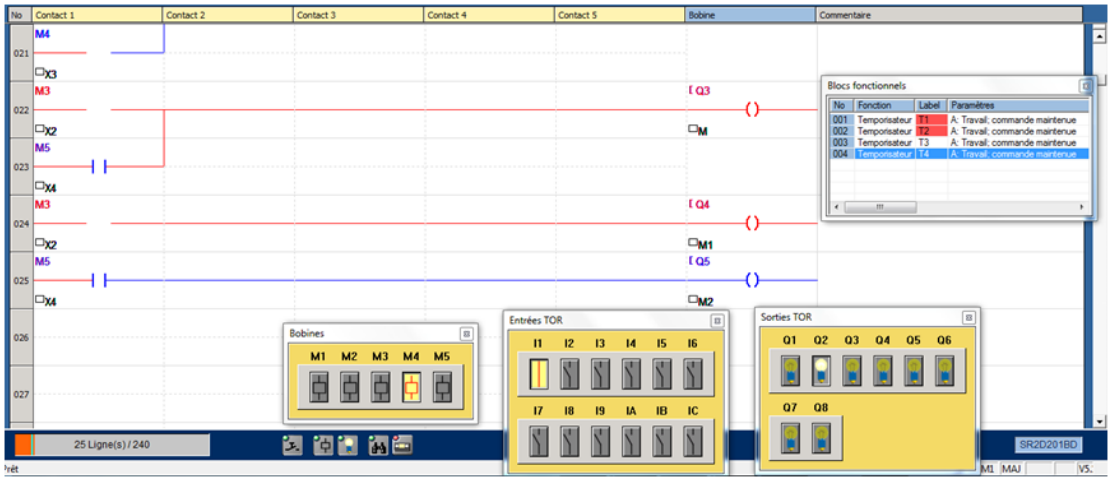


Figure IV-26: SIMULATION-pulvérisateur agricole et péristaltique

Conclusion Générale

Conclusion Générale

Le fonctionnement et la programmation des automates programmables industriels évoluent de jour en jour et la demande est de plus en plus forte dans de nombreux domaines. Principalement basé sur des connaissances antérieures, telles que la logique combinatoire et séquentielle et le langage de programmation, il permet une automatisation efficace De nombreux cas de productivité.

De toute évidence, cette mémoire gère en quelque sorte Les API de programmation peuvent être généralisées car nous avons donné quelques exemples et outils Couramment utilisé dans l'industrie des API.

Nous avons implémenté de nombreux exemples et API de programmation en fonction des Grafcet En revanche, cette version sera accompagnée de travaux pratiques supplémentaires Mis en œuvre dans l'environnement ZlioSoft 2 de Schneider, nous pouvons également appliquer ces exemples à des Automates Programmables Industriels d'autres sociétés.

Listes Des Référence

Listes Des Référence

[1]https://fasoeducation.net/espace_eleves/secondaire/eftp/bac_technologique/automates_programmables_industriels/co/grainIntroduction.html

[2]djarrallah_mohamed "automate_programmable"univ-batna2

(3)Préface I-Aspect extérieur – Nanopdf

[**4]Alain GONZAGA"LES AUTOMATES PROGRAMMABLES INDUSSTRIELS"**

https://www.startimes.com/f.aspx?t=36114937[**5]**

Michel BERTRAND"Automates programmablesindustriels" 10 mars 2001[**6]**

CABLAGE ENTREE SORTIE API DR"DECEMBER 2020]"7[

EC 61131-3 " A standard programmingresource]"8[

2002Mr Hu jean–LOUIS "les progammables"18 DECEMBER[9]

LE GRAFCET"univ-guelma]"10[

LE GRAFCET"univ-biskra]"11[

Zelio Soft – Professionnels | Schneider Electric France"]"12[

Modules logiques ZelioLogic"-14102-FR_Ver2.0.fm]"13[

https://www.iraifrance.com/automgen-industrie]14[

"NOTICE AUTOMGEN"-Lycée PP Riquet St Ore]15[

Dr. Aboubakeur HADJAISSA"api"lagh-univ 13 mars 2019]16[

Dr Ir LECOCQ "Cararactèristiques et mèthodologie de programmation "2005]17[