

République Algérienne Démocratique et Populaire  
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE  
SCIENTIFIQUE



Ecole Nationale Polytechnique  
Département Automatique



**PROJET DE FIN D'ETUDES EN VUE DE L'OBTENTION  
DU DIPLOME D'INGENIEUR D'ETAT  
EN AUTOMATIQUE**

**Thème**

**REALISATION D'UNE MAQUETTE DE STATION DE  
POMPAGE A BASE D'AUTOMATE PROGRAMMABLE  
SIEMENS**

*Réalisé par :*

**FOUKA Majda Amina Aida  
FERHAOUI Bouchra**

**Présenté devant le jury composé de :**

Président:	L.ABDELOUL	Chargé de Cours à l'ENP
Rapporteur:	E.M.BERKOUK	Professeur à l'ENP
Examineur:	R.ILOUL	Maître de Conférences à l'ENP

**ENP 2015**

ENP 10, Avenue Hassen Badi, BP. 182, 16200 El Harrach, Alger, Algérie

## **Remerciements**

*Nous tenons à remercier dieu de nous avoir donné la force morale, physique et l'aide pour accomplir ce modeste travail.*

*Nous tenons à remercier notre promoteur **Pr. BERKOUK** pour nous avoir acceptées encadrées et dirigées durant l'élaboration de ce travail ainsi que pour leur assistance et tous leur conseils.*

*Nous remercions également Monsieur **O. STAIHI** pour ses conseils et son aide.*

*Et nous remercions monsieur **M. BOUAZA** qui nous a aidées, conseillées et éclairées sur notre travail tout le long de notre projet.*

*Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptent d'évaluer notre projet.*

*Nous souhaitons aussi remercier tous les enseignants de l'Ecole Nationale Polytechnique d'Alger, et en particulier, Nos professeurs d'Automatique qui nous ont encadrées auparavant et tous nos enseignants pour les connaissances qu'ils nous ont transmis, leur disponibilité et leurs efforts.*

*Que tous ceux qui ont contribué de près ou de loin à la réalisation de ce modeste travail trouvent ici l'expression de notre sincère gratitude.*

# *Dédicaces*

*J'ai le grand honneur de dédier ce modeste travail*

*À Celle à qui mon cœur depuis sa naissance n'a pas pu éprouver  
qu'amour et reconnaissance, à celle qui a donné un sens à mon  
existence en m'offrant une éducation digne de confiance*

*À ma chère Mère.*

*À mon père, pour son amour et son soutien moral depuis mon enfance.*

*Père merci.*

*Aux deux femmes les plus merveilleuses sur cette terre, qui étaient et sont  
toujours là pour moi avec leurs conseils et soutien, ma chère grande maman (que  
dieu te garde parmi nous Ma Alia) et ma chère tante Saadia.*

*À toute ma famille du plus petit au plus grand, mes sœurs, frères,*

*À mon binôme Bouchra et ces cinq années de travail et de complicité,*

*À tous mes amis et camarades de l'École Nationale Polytechnique, et toute la  
promotion Automatique de l'année 2015.*

*À toute personne qui m'a aidée de loin ou de près afin de réaliser ce  
travail.*

*Majda-Amina-Aida*



## Dédicace

*Je remercie Dieu de m'avoir donnée courage et volonté tout au long de mon cursus.*

*Je dédie ce modeste travail à :*

*• mes chers parents que je ne peux remercier assez. Vous vous êtes dépensés pour moi sans compter. En reconnaissance de tous vos sacrifices pour me permettre d'atteindre cette étape de ma vie.*

- Mes sœurs : Samira, Amina et Naila*
- Mon cher fiancé Noureddine*
- Mes beaux-frères : Mourad et Nacereddine*
- Mes adorables nièces : Maram, Manar et Anissa*
- A mon binôme Majda pour tout le travail qu'on a accompli*
- Mes Amies surtout : Abla, Khadidja et à tous mes collègues à l'école nationale polytechnique*  
*A Toute la famille FERHAOUI*

*Bouchra*



## تلخيص

العمل المنجز في المذكرة يتمحور على استعمال مسير صناعي مبرمج "سيمنس" حيث اقترحت دائرة الاتوماتيك للمدرسة الوطنية العليا المتعددة التقنيات صناعة نموذج محطة ضخ المياه وهذا لهدف تعليمي ببرمجة تطبيقات على النموذج. تسيير التحصيل ومعالجة المعلومات محقق من طرف هذا المسير الصناعي. لضبط المحطة استعملنا "برنامج" ستابسات" و لمراقبة المحطة استخدمنا برنامج "وينسيبي فلكسبيل".

## الكلمات المفتاحية

مسير صناعي مبرمج "سيمنس", برنامج "ستابسات", برنامج "وينسيبي فلكسبيل" نموذج محطة ضخ المياه.

## RESUME :

Le travail présenté dans ce mémoire est basé essentiellement sur l'utilisation des automates programmables SIEMENS. Le Département d'Automatique de l'Ecole Nationale Polytechnique a proposé la réalisation d'une maquette de station de pompage afin de l'exploiter dans un objectif didactique pour programmer des manipulations dans la station. Nous avons entrepris la réalisation d'un programme de gestion par le logiciel STEP7, et de supervision par le progiciel WINCC flexible.

### Mots clés:

Automate programmable Siemens S7-314 IFM, logiciel STEP7, logiciel WinCC, Prototype de station de pompage.

## ABSTRACT:

The work presented in this memory is based mainly on the use of programmable SIEMENS. Since The Department of Automation of the national polytechnic school has proposed a realization of a prototype of hydraulic station of pumping in a didactic aim to program different manipulation, we undertook a program of control with the STEP7 software and supervision with the WinCC flexible software.

### Keywords:

Siemens PLC S7-314 IFM, STEP7 software, WINCC software, Prototype of hydraulic station of pumping.

# SOMMAIRE

Introduction générale .....	1
<b>CHAPITRE I: Les automates programmables industriels SIEMENS</b>	
I.1 Introduction.....	4
I.2 Historique.....	4
I.3 Définition.....	4
I.4 Architecture des automates.....	5
I.5 Traitement du programme automate.....	6
I.6 Présentation de la gamme SIMATIC.....	8
I.7 Le logiciel step7.....	13
I.7.1 Description.....	13
I.7.2 Fonctions du logiciel step7.....	13
I.7.3 Applications disponibles .....	13
I.8 Stratégie pour la conception d'une structure programme complète.....	17
I.8.1 Création du projet SIMATIC Step7.....	17
I.8.2 Configuration du matériel.....	18
I.8.3 Définition des mnémoniques.....	20
I.8.4 Edition des programmes.....	21
I.8.5 Simulation de modules Avec PLCSIM.....	23
I.8.6 Chargement du programme dans la CPU.....	24
I.8.7 Surveillance du fonctionnement et diagnostic du matériel.....	24
I.9.Exemples de Création et d'édition d'un projet S7 .....	24
I.9.1. Exemple sur la programmation séquentielle avec le langage Évolué S7-GRAPH.....	24
I.9.2. Exemple sur la Programmation avec entrées/sorties Logiques et Analogique.....	31
I.10.Conclusion.....	37
 <b>CHAPITRE II : Le progiciel de conception des interfaces Homme/Machine Win CC flexible</b>	
II.1 Introduction.....	41
II.2. SIMATIC HMI.....	41
II.3. Utilisation de SIMATIC WinCC flexible .....	42

II.4. Interface logicielle de WinCC flexible .....	42
II.5. WinCC flexible Engineering System.....	42
II.6. Eléments de l'interface utilisateur de WinCC flexible.....	43
II.6.1.Menus et barres d'outils .....	44
II.6.2.Zone de travail.....	45
II.6.3.Fenêtre de projet.....	46
II.6.4.Fenêtre des propriétés.....	46
II.6.5.Bibliothèque.....	47
II.6.6.Fenêtre des erreurs et avertissements .....	48
II.6.7.Fenêtre des objets.....	48
II.7. Créer des vues.....	49
II.8. Générer un fichier de projet.....	50
II.9.Introduction à WinCC flexible Runtime.....	50
II.10. Tester le projet.....	51
II.11. Transférer un projet.....	52
II.12. Exemple d'application sur la Presse Hydraulique .....	52
II.13.Conclusion .....	54
<b>CHAPITRE III : Les réseaux locaux Industriels: Profibus</b>	
III.1 Introduction.....	56
III.2 Echange de données .....	56
III.3 Quelques définitions.....	57
III.4 Réseau MPI et protocole Profibus.....	58
III.4.1 Définition.....	58
III.4.2 Architecture.....	58
III.4.3 Variantes.....	58
III.4.4 Fonctionnement général.....	59
III.4.5 Principe de l'anneau à jeton.....	60
III.4.6 Utilisation des données globales.....	60
III.5 Conclusion.....	62
<b>CHAPITRE IV : Réalisation d'une maquette de station de pompage à base d'automates SIEMENS</b>	
IV.1. Introduction.....	64

IV.2. Structure de la Tâche d'automatisation.....	64
IV.3. Description du Prototype de la station de pompage.....	64
IV.4. Eléments de commande et de supervision du Prototype de la station de pompage.....	66
IV.5. Développement du projet pour gestion du prototype de la station de pompage.....	67
IV.6. Définition des Cahiers de charge.....	68
IV.7. Développement du projet S7 pour la maquette .....	69
IV.7.1. Insertion des stations dans SIMATIC MANAGER.....	69
IV.7.2. Création du programme de gestion dans STEP7 .....	70
IV.7.3. L'interface homme machine HMI configurée par le progiciel « Win CC flexible ».....	71
IV.8. Configuration de la connexion entre API et WinCC Advanced RT.....	72
IV.9. Simulation du programme STEP 7 dans PLCSIM.....	73
IV.10. Visualisation du processus dans WinCC Advanced Runtime.....	73
IV.11. Chargement dans le système cible à partir de la PG/PC.....	74
IV.12 Conclusion.....	74
Conclusion Générale et Perspectives .....	76
Bibliographie.....	77
<b>Annexe A</b> : Description des éléments de l'application « maquette de station de pompage à base de l'API SIEMENS».....	79
<b>Annexe B</b> : Les programmes de l'application «maquette de station de pompages à base d'automates programmable SIEMENS ».....	85



# La liste des Figures

**Figure I.1:** automate modulaire (siemens)

**Figure 1.2:** structure interne d'un API

**Figure I.3:** Les étapes de traitement du programme automate

**Figure I.4:** Présentation de la gamme SIMATIC

**Figure I.5:** S7-200

**Figure I.6:** S7-300

**Figure I.7:**S7-400

**Figure I.8:**S7-1200.

**Figure I.9:**S7-1500.

**Figure I.10:**C7-623

**Figure I.11 :**C7-624

**Figure I.12 :** Ecran tactile (KZ-M7300).

**Figure I.13:**Constituants de la gamme S7-300.

**Figure I.14:** différentes applications du logiciel STEP 7.

**Figure I.15:** Le gestionnaire de projet STEP 7.

**Figure I.16:** Choix de la station SIMATIC 300

**Figure I.17:** Configuration matériels pour la station S300

**Figure I.18:** Sélection des modules

**Figure I.19:** Edition des Mnémoniques.

**Figure I.20:** Edition des programmes

**Figure I.21 :** logiciel de simulation PLC-SIM

**Figure I.22 :** Une perceuse automatique

**Figure I.23 :** Création d'un nouveau projet

**Figure I.24 :** Configuration du matériel

**Figure I.25:**Insertion d'un bloc fonctionnel

**Figure I.26:** Création de la première étape

**Figure I.27:** Insertion des actions

**Figure I.28:** Saisie des actions

**Figure I.29:** Grafset final

**Figure I.30 :** Configuration des paramètres du bloc et enregistrement du bloc fini

**Figure I.31:** Insertion du bloc fonctionnel(FB1) dans le bloc organisationnel(OB1)

**Figure I.32:** Simulation du programme S7 GRAPH

**Figure I.33 :** Schéma de la presse hydraulique

**Figure I.34 :** Graphcset du fonctionnement automatique de la presse hydraulique

**Figure I.35 :** Programme du fonctionnement automatique de la presse hydraulique

**Figure I.36 :** Fonction « SCALE »

**Figure I.37 :** Fonction du mode manuel

**Figure I.38 :** Fonction du mode automatique

**Figure II.1 :** présentation de l'interface WinCC flexible.

**Figure II.2 :** Eléments de l'interface WinCC.

**Figure II.3 :** La forme graphique d'une Vue.

**Figure II.4 :** fenêtre de projet.

**Figure II.5 :** fenêtre des propriétés.

**Figure II.6 :** Elément de la bibliothèque.

**Figure II.7 :** Fenêtre des erreurs et avertissements.

**Figure II.8 :** Fenêtre des objets.

**Figure II.9 :** Vue de commutation entre le mode manuel et automatique.

**Figure II.10:** Vue de mode Manuel

**Figure II.11 :** Vue de mode Automatique

**Figure II.12 :** Vue analogique

**Figure III.1 :** les mots mémoires appartenant à chaque automate

**Figure III.2 :** L'échange de ces mots mémoires « partagés » entre les stations

**Figure III.3 :** Classification des réseaux Profibus.

**Figure IV.1 :** Prototype de station de pompage.

**Figure IV.2 :** Synoptique du prototype.

**Figure IV.3 :** Schéma Electrique de la station

**Figure IV.4 :** diagramme de passage entre les 2 manipulations.

**Figure IV.5 :** Grafsets des deux manipulations.

**Figure IV.6:** Grafsets de la manipulation supplémentaire.

**Figure IV.7 :** Création du projet « Maquette de station de pompage »

**Figure IV.8 :** La table des mnémoniques.

**Figure IV.9 :** Le projet S7 « maquette de station de pompage ».

**Figure IV.10 :** Vue de WinCC flexible.

**Figure IV.11 :** Configuration de La liaison MPI.

**Figure IV.12 :** La vue « démarrage »

**Figure IV.13 :** La vue « maquette »

**Figure IV.14 :** La vue « caractéristique »

**Figure A.1 :** démarreur progressif ATS 01N103FT

**Figure A.2 :** relais de niveau

**Figure A.3 :** Schéma de principe de l'alimentation stabilisée

**Figure A.4:** Schéma d'un relais électromécanique

## **La liste des Tableaux :**

**Tableau I.1:** Constituants de la gamme S7-300.

**Tableau II.1:**Description des éléments de Menu Sous WinCC flexible.

**Tableau IV.1 :** les étapes du développement du projet.

## Symboles et abréviation

<b>AA</b>	sortie analogique
<b>AC</b>	Alternative Curent
<b>AE</b>	entrée analogique
<b>AI</b>	Analog Input
<b>AO</b>	Analog Output
<b>API</b>	Automate Programmable Industriel
<b>ASCII</b>	American Standard Code for Information Interchange
<b>CHAR</b>	caractère
<b>COUNT</b>	Schéma à Contact
<b>CP</b>	Communication Processor
<b>CPU</b>	Central Processing Unit
<b>DA</b>	Sortie Digitale
<b>DB</b>	Data Bloc
<b>DC</b>	Direct Current (courant continu)
<b>DE</b>	entrée digitale
<b>DI</b>	Digital Input
<b>DO</b>	Digital Output
<b>DP</b>	Decentralized Periphery
<b>EEPROM</b>	Electrically Erasable Programmable Read Only Memory
<b>E/S</b>	Entrée/Sortie
<b>FB</b>	Fonctionnel Bloc
<b>FC</b>	Fonction
<b>FM</b>	Function Module
<b>HW Config</b>	HardWare Configuration
<b>IHM</b>	Interface Homme-Machine
<b>IM</b>	Coupleurs d'extension
<b>INT</b>	INTeger
<b>LED</b>	Light Emitting Diode
<b>LIST</b>	Le langage de liste d'instructions
<b>LOG</b>	LOGigramme
<b>MPI</b>	Multi-Point Interface
<b>OB</b>	Organisationnel Bloc

<b>OP</b>	Operator Pupiter
<b>PC</b>	Personnel Computer
<b>PCMCIA</b>	Personal Computer Memory Card International Association
<b>PG</b>	la console de programmation sur le terrain
<b>PLC</b>	Programmable Logic Controller
<b>PROFIBUS</b>	Process Field Bus
<b>PROM</b>	Programmable Read Only Memory
<b>PS</b>	Gamme des alimentations stabilisees de Siemens
<b>PtP</b>	Point to Point
<b>RAM</b>	Random Access Memory
<b>ROM</b>	Read Only Memory
<b>SAP</b>	Système Automatisé de Production
<b>SFB</b>	System Fonctionnel Bloc
<b>SFC</b>	System Fonction
<b>SIMATIC</b>	Siemens Automatic
<b>SM</b>	Signal Module
<b>S7</b>	Step 7
<b>TIA</b>	Totally Integrated Automation
<b>TOR</b>	Tout Ou Rien
<b>T.R.T</b>	temps de réponse total
<b>µP</b>	Micro-Processeur

## ***Introduction générale***

Par la place sans cesse croissante qu'occupe l'automatisme dans les processus industriels, cette technologie revêt désormais une importance primordiale. Tant dans la production quotidienne que dans l'enseignement.

Et, de nos jours, la technologie d'automation est pratiquement toujours liée à la technique d'entraînement, de réglage ou à l'informatique. Par le développement vertigineux de la technique des Automates Programmables Industriels et de la technologie informatique, la technologie d'automation est devenue l'un des domaines les plus innovants. A cela s'ajoute que les nouvelles solutions industrielles, telles la décentralisation et la visualisation, exigent de nouveaux systèmes didacticiels. En outre, les automates programmables intégrés de cette génération ne sont plus programmés que selon des règles uniformes. Compte-tenu de ces exigences auxquelles sont confrontés les spécialistes de l'automation, il est indispensable aujourd'hui de proposer des systèmes d'entraînement orientés à la pratique, permettant de transmettre à l'apprenti l'état actuel de la technique et la compétence requise dans son maniement.

Les stations de pompes hydrauliques cherchent à se procurer cette solution d'automatisme dans les réseaux d'alimentation en eau potable.

La firme Siemens est l'une des plus grosses entreprises européennes. Qui propose une gamme complète de produits pour l'automatisation industrielle et qui procurant ce type de service industriel. En effet, par le biais de sa gamme SIMATIC, elle intègre tout l'environnement d'automatisation.

Dans le but de former des élèves ingénieurs automaticiens à la maîtrise de cet outil, le Département Automatique de l'ENP a proposé de réaliser une maquette de station de pompage à base d'automate Programmable Siemens afin de l'utiliser par la suite dans le cadre des Travaux pratiques au sein du laboratoire comme un système d'apprentissage didactique pour l'automatisation, La maquette est conçue dans le but d'appliquer les connaissances acquises dans l'informatique industrielle et pour répondre à un certain nombre d'exigences de formation professionnelle.

Nous exposons dans le présent rapport quatre grands chapitres décrivant les volets principaux de notre projet:

Le premier chapitre sera une description du système de contrôle et de commande; nous y présenterons son architecture et ses composants et nous traiterons les blocs et diagrammes fonctionnels utiles lors de la phase de manipulation du logiciel d'automatisme.

Dans le deuxième chapitre, nous décrirons le logiciel WinCC flexible de la firme Siemens qui nous permet de programmer et configurer l'interface Homme/Machine pour Visualiser et maîtriser le processus à tout instant et de maintenir le prototype de la station de pompage en état de marche.

Le troisième chapitre sera une description des réseaux industriels nécessaires pour la communication dans la conception d'une architecture de contrôle-commande d'un système de production automatisé .dans ce chapitre on va présenter l'architecture et les différentes variantes du réseau local industriel.

Le dernier chapitre englobera la présentation de l'application et les différents organes constituant la maquette. Nous décrirons par la suite le cahier des charges des manipulations à effectuer sur la maquette, puis on va traiter les étapes de programmation de ce projet dans le logiciel STEP7 et la conception de l'interface homme-machine via le progiciel WinCC flexible, la communication entre la station et le pupitre de visualisation se fait via un réseau MPI.

On terminera par une conclusion générale qui montre les difficultés qu'un étudiant à ses débuts peut rencontrer, les enseignements qu'il en tire, sans oublier les perspectives.

Les étudiants par la suite vont programmer des manipulations à l'aide du logiciels step7, puis transférer leur programme et visualiser l'enchaînement de leur programme.

# CHAPITRE I

## Automates Programmables Industriels SIEMENS



## I.1. Introduction :

Dans ce chapitre, nous allons présenter brièvement les automates programmables ainsi que la gamme SIMATIC de Siemens et plus précisément l'automate S7 300 que nous avons choisi pour la commande de notre maquette. Par la suite, on abordera la description de l'environnement de développement STEP7 ainsi que les différents langages de programmation, et on va traiter des exemples afin d'illustrer les différentes étapes de gestion d'un projet.

## I.2. Historique [1]

L'automate programmable, souvent appelé automate programmable industriel (API, en anglais PLC pour Programmable logic controller) pour rappeler son domaine privilégié d'utilisation, l'industrie, est apparu voici 40 ans et s'est rapidement répandu dans la production, la logistique, le conditionnement, la gestion technique de bâtiments, etc. Son développement a accompagné celui de l'automatisation de la production, la faisant passer du stade de la machine automatisée à celui du système automatisé de production (SAP), et il en est devenu, avec le robot, un composant majeur, le « fantassin de l'automatisation industrielle », suivant l'expression de C. Laugeau. La question de le ramener à cet outil de communication universel qu'est le PC s'est alors rapidement posée.

## I.3. Définition [1]

Un automate programmable industriel, ou API, est défini suivant la norme française NF-61131 comme un système électronique fonctionnant de manière numérique, destiné à être utilisé dans un environnement industriel, qui utilise une mémoire programmable pour le stockage interne des instructions orientées utilisateur aux fins de mise en œuvre de fonctions spécifiques, telles que des fonctions de logique, de mise en séquence, de temporisation, de comptage et de calcul arithmétique, pour commander au moyen d'entrées et de sorties tout-ou-rien ou analogiques divers types de machines ou de processus. L'automate programmable et ses périphériques associés sont conçus pour pouvoir facilement s'intégrer à un système d'automatisme industriel et être facilement utilisés dans toutes leurs fonctions prévues ».

A partir de là, et en respectant les caractéristiques essentielles qui ressortent de cette définition souvent un peu laborieuse, nous considérerons plus simplement comme automate programmable un système :

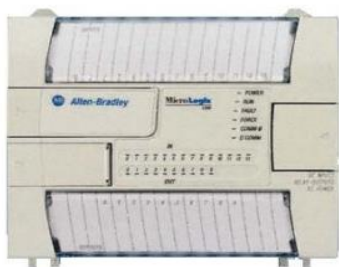
- construit autour d'un processeur numérique, spécifique ou non ;
- pouvant être relié à de nombreux signaux physiques ;
- fonctionnant dans des conditions industrielles grâce à une protection adaptée ;
- doté d'un logiciel de programmation permettant un traitement rapide des fonctions logiques (tout-

ou-rien, TOR dans le monde industriel) ;

– doté de possibilités d’échanges avec d’autres processeurs. Ceci constitue un « noyau » minimal susceptible de nombreuses variantes et extensions propres à un automate donné.

**I.4. Architecture des automates [24]**

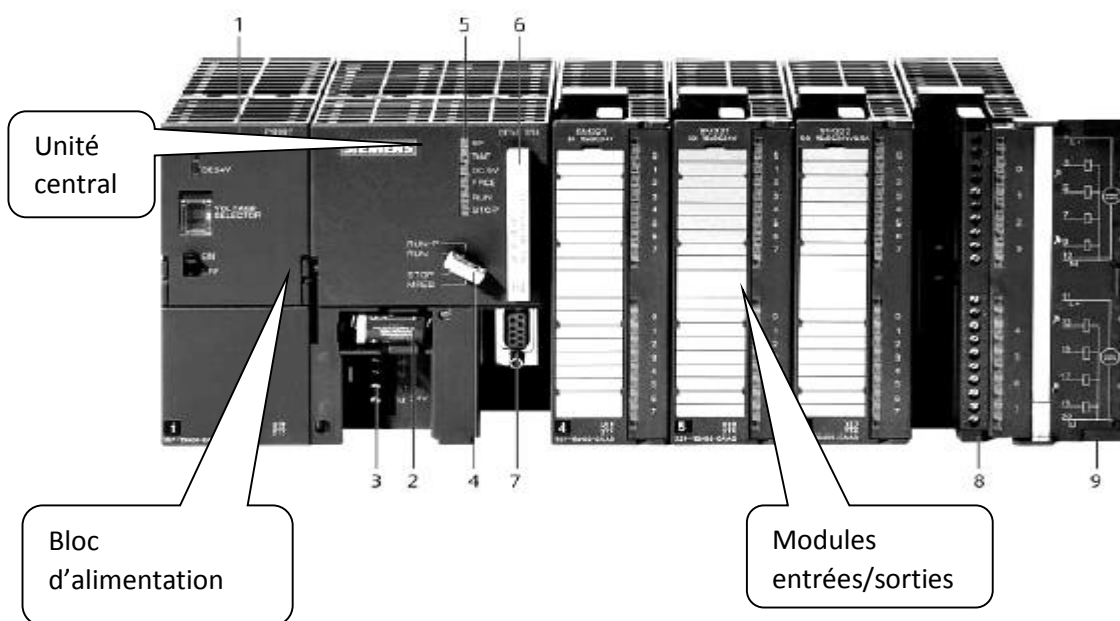
a. Aspect extérieur : Les automates peuvent être de type compact ou modulaire. De type compact, on distinguera les modules de programmation (LOGO de Siemens, ZELIO de Schneider, MILLENIUM de Crouzet ...) des microautomates. Il intègre le processeur, l’alimentation, les entrées et les sorties. Selon les modèles et les fabricants, il pourra réaliser certaines fonctions supplémentaires (comptage rapide, E/S analogiques ...) et recevoir des extensions en nombre limité. Ces automates, de fonctionnement simple, sont généralement destinés à la commande de petits automatismes. 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. Ces automates sont intégrés dans les automatismes complexes où puissance, capacité de traitement et flexibilité sont nécessaires.



Automate compact (Allen-bradley)



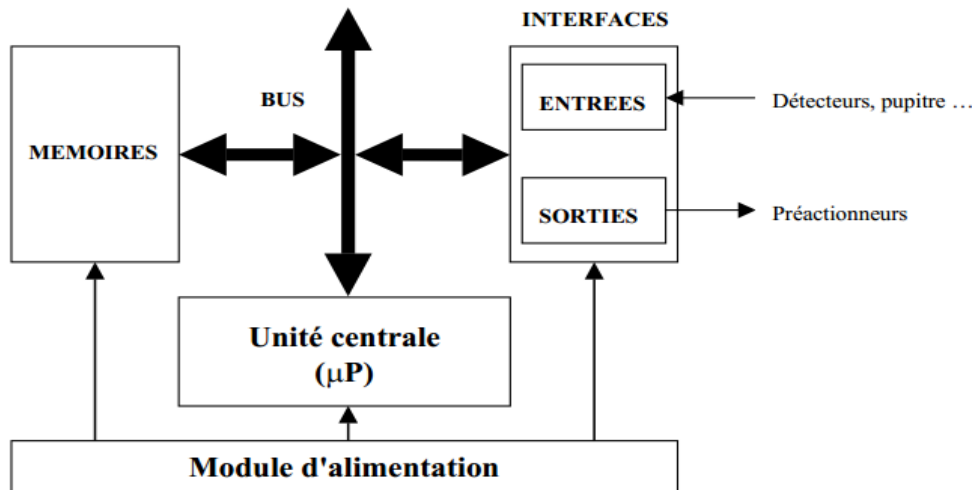
Automate modulaire (Modicon)



**Figure I.1 : automate modulaire (siemens) [24]**

- |   |                              |
|---|------------------------------|
| 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 :**

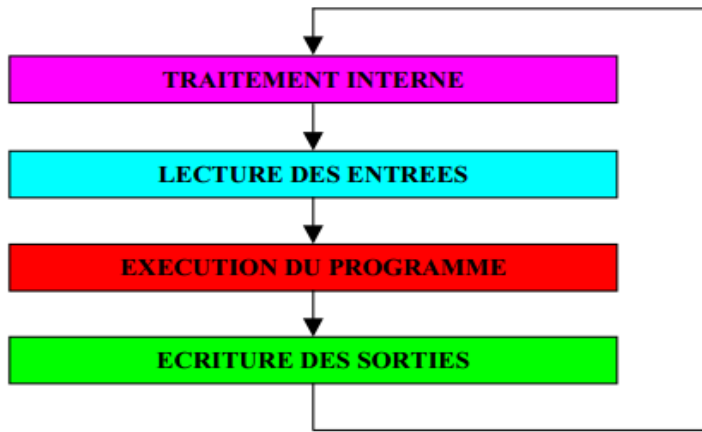


**Figure 1.2 :** *structure interne d'un API* [3]

- 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 :
  - 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).
  - Interface de sortie : elle permet de commander les divers préactionneurs et éléments de signalisation du S.A.P. tout en assurant l'isolement électrique.

## **I.5. Traitement du programme automate** [24]

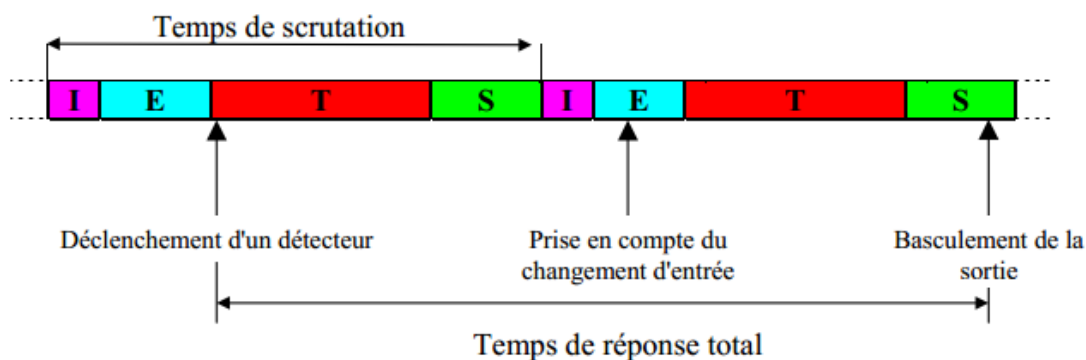
Tous les automates fonctionnent selon le même mode opératoire :



**Figure I.3:** Les étapes de traitement du programme automate [24]

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

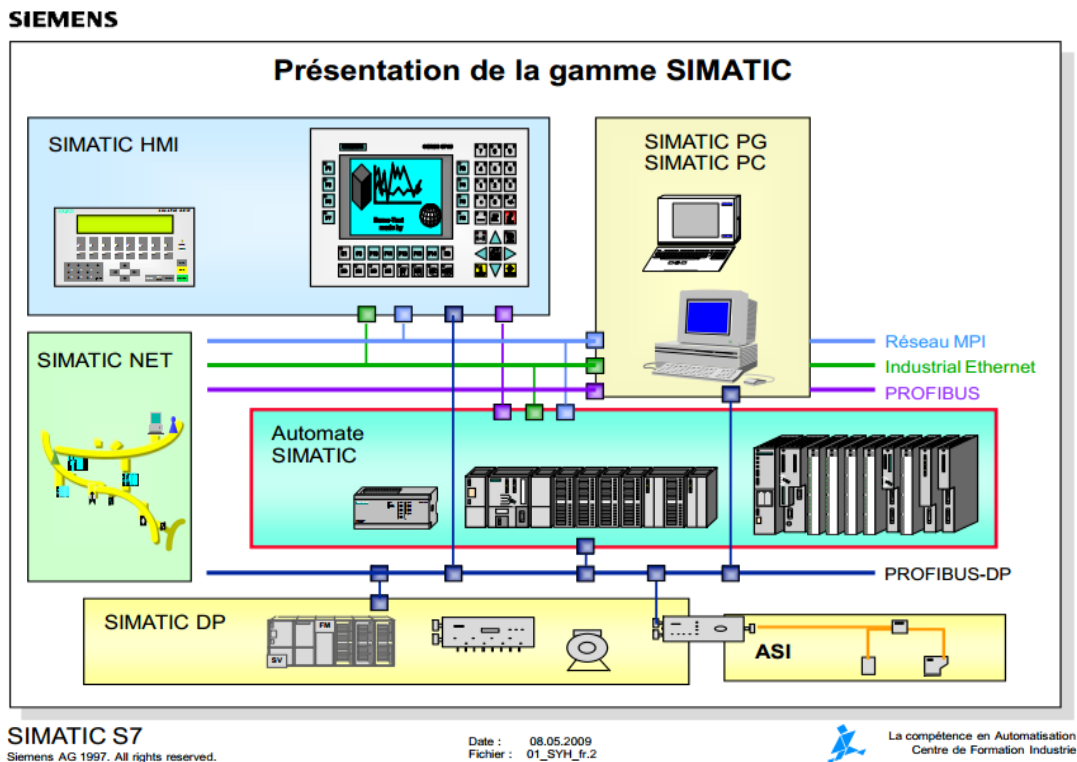
Ces quatre opérations sont effectuées continuellement par l'automate (fonctionnement cyclique). On appelle scrutation l'ensemble des quatre opérations réalisées par l'automate et le temps de scrutation est le temps mis par l'automate pour traiter la même partie de programme. Le temps de réponse total (TRT) est le temps qui s'écoule entre le changement d'état d'une entrée et le changement d'état de la sortie correspondante :



## I.6. Présentation de la gamme SIMATIC : [7]

Le nom de SIMATIC était hier encore associé aux automates et plus précisément aux automates SIMATIC S5. Aujourd'hui SIMATIC est devenu synonyme de l'intégration totale.

**L'intégration totale** est un concept révolutionnaire visant à réunir l'univers de la fabrication manufacturière et l'univers des procédés. Toutes les briques matérielles et logicielles nécessaires à la réalisation d'un projet portent désormais un seul nom : SIMATIC.



**Figure I.4 :** *Présentation de la gamme SIMATIC* [7]

### I.9.1. Les variantes de la gamme SIMATIC

Cette gamme est composée de trois familles :

#### **A. SIMATIC S7:**

SIMATIC S7 désigne un produit de la société SIEMENS et est synonyme de la nouvelle gamme des automates programmables. La famille des systèmes d'automatisation SIMATIC S7 est une brique dans l'architecture de la Totally Integrated Automation - le concept de l'automatisation totale pour la fabrication et la conduite des processus.

La gamme des nouveaux SIMATIC S7 comprend les systèmes d'automatisation suivants :

**A.1. SIMATIC S7-200 :** est un micro-automate compact d'entrée de gamme. Il possède un progiciel qui lui est propre, les propriétés du système S7-200 ne supportant pas une conversion logicielle des programmes S5 en programmes S7.

La famille S7--200 est constituée de micro--automates programmables pouvant commander une large gamme d'appareils .Le S7-200 surveille les entrées et modifie les sorties conformément au programme utilisateur, qui peut contenir des opérations booléennes, des opérations de comptage, des opérations de temporisation, des opérations arithmétiques complexes et des opérations de communication avec d'autres unités intelligentes.



**Figure I.5 : S7-200 [8]**

**A.2.SIMATIC S7-300 :** Le mini-automate modulaire SIMATIC S7-300 fait encore partie de l'entrée de gamme.



**Figure I. 6 :S7-300 [22]**

**A.3.SIMATIC S7-400 :** couvre le milieu, le haut et le très haut de gamme.

Pour faciliter l'orientation, les modules du S7-300 commencent par un 3 et les modules du S7-400 commencent par un 4.



**Figure I.7 :S7-400 [22]**

**A.4.SIMATIC S7-1200 :** Avec son alimentation intégrée et une grande diversité de circuits d'entrée et de sortie intégrés, la CPU S7-1200 constitue un automate puissant. Une fois le programme chargé, la CPU surveille les entrées et modifie les sorties conformément à la logique du programme utilisateur qui peut contenir des instructions booléennes, des compteurs, des temporisations et des instructions mathématiques complexes.



**Figure I.8 :**S7-1200. [9]

**A.5.SIMATIC S7-1500:** Le nouveau contrôleur SIMATIC S7-1500 est conçu pour offrir une facilité d'utilisation maximale. Grâce à de multiples innovations même dans les moindres détails, le contrôleur peut être monté, raccordé et mis en service simplement et rapidement. L'intégration harmonieuse au TIA Portal rend l'ingénierie, la conception de projets et l'utilisation de nouvelles fonctions plus simples que jamais. Elle offre un concept d'utilisation homogène, permettant ainsi une gestion des données commune et cohérente.

La famille d'automates SIMATIC S7-1500 avec la CPU haut de gamme 1518 et les nouveaux automates compacts, la CPU 1511C la CPU 1512C constituent la nouvelle génération d'automates du TIA Portal .



**Figure I.9:**S7-1500. [10]

### **B.SIMATIC C7:**

Le SIMATIC C7 comprend deux sections travaillant indépendamment l'une de l'autre et pouvant communiquer entre elles à travers l'interface MPI interne : la CPU et le pupitre opérateur OP. Les systèmes intégrés compacts C7 existent en deux variantes :

**B.1.SIMATIC C7-623:** Avec un affichage à 4 lignes, 20 caractères par ligne et une hauteur des caractères de 5 mm



**Figure I.10 :** C7-623 [11]

**B.2.SIMATIC C7-624:** La hauteur des caractères est définie dans le logiciel. On a le choix entre 4 x 20 caractères de 8 mm de haut ou 8 x 40 caractères de 4,5 mm de haut.



**Figure I.11:** C7-624 [11]

### C.SIMATIC M7: [12]

SIMATIC M7 est un calculateur industriel compatible PC. Il s'agit d'un système modulaire sous boîtier, construit dans la technique des automates SIMATIC S7. Il peut être intégré dans un automate S7-300/400 ou être utilisé comme système autonome avec une périphérie choisie dans la gamme S7.

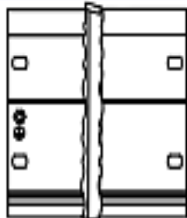






Le M7-300 est capable d'effectuer simultanément avec une seule CPU des opérations en temps réel, par exemple des algorithmes complexes de commande et de régulation ainsi que des tâches de visualisation et de traitement informatique. Les logiciels sous RMOS, DOS ou Windows sont exploitables sur le M7-300. Par ailleurs, avec son architecture normalisée PC, il permet une extension programmable et ouverte de la plate-forme d'automatisation S7.



**Figure I.12 :** Ecran tactile (KZ-M7300) [22]



**I.9.2. Description de la gamme SIMATIC S7 – 300 : [13]**

Constituants	Fonction	Illustration
Profilé support	Il constitue le châssis du S7-300.	
Module d'alimentation (PS)	Il convertit la tension de secteur (120/230 V CA) en une tension continue de 24 V pour l'alimentation du S7-300.	
CPU (module unité centrale) Pile de sauvegarde (option)	La CPU exécute le programme utilisateur. Elle alimente le bus interne du S7-300 en 5 V et communique avec d'autres CPU et avec la PG ou le PC par l'entremise de l'interface MPI.	
Module de simulation (6ES7 374...) avec 8 entrées TOR et 8 sorties TOR	Il vous donne la possibilité de tester le programme utilisateur à la mise en service de l'automate et en cours de fonctionnement : des interrupteurs simulent les signaux des capteurs ; des voyants (LED) visualisent l'état logique des signaux des sorties.	
Câble MPI	Il relie la PG ou le PC à la CPU.	
Console de programmation (PG) avec carte MPI et logiciel STEP 7 installé <b>ou bien</b>	Elle sert à configurer, à paramétrer, à programmer et à tester l'automate S7-300.	
Ordinateur personnel (PC) avec carte MPI ou câble PC/MPI et logiciel STEP 7 installé	Il sert à configurer, à paramétrer, à programmer et à tester l'automate S7-300.	

**Tableau I.1:** Constituants de la gamme S7-300. [13]

## **I.7.Le logiciel step7 :**

### **I.7.1.description :** [17]

STEP 7 est le progiciel de base pour la configuration et la programmation de systèmes d'automatisation SIMATIC. Il fait partie de l'industrie logicielle SIMATIC. Le progiciel de base STEP 7 existe en plusieurs versions :

- STEP 7-Micro/DOS et STEP 7-Micro/Win pour des applications autonomes simples sur SIMATIC S7 - 200.
- STEP 7 pour des applications sur SIMATIC S7-300/400, SIMATIC M7-300/400 et SIMATIC C7 présentant des fonctionnalités supplémentaires :
  1. Possibilité d'extension grâce aux applications proposées par l'industrie logicielle SIMATIC (voir aussi Possibilités d'extension du logiciel de base STEP 7) ;
  2. Possibilité de paramétrage de modules fonctionnels et de modules de communication ;
  3. Forçage et fonctionnement multiprocesseur ;
  4. Communication par données globales ;
  5. Transfert de données commandé par événement à l'aide de blocs de communication et de blocs fonctionnels ;
  6. Configuration de liaisons.

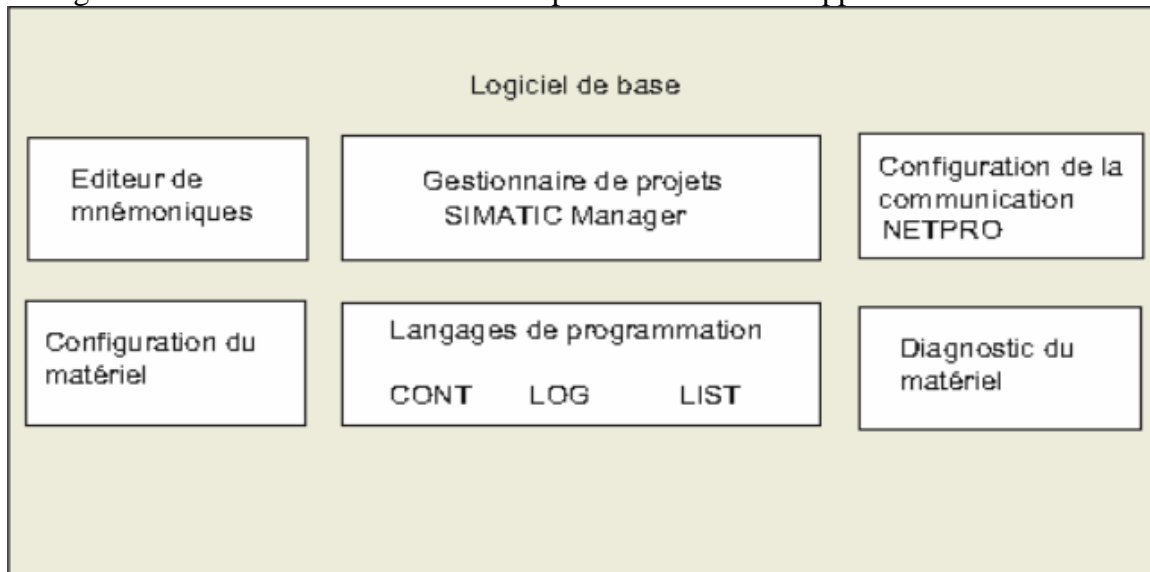
### **I.7.2.Fonctions du logiciel step7** [17]

Le logiciel step7 nous assiste dans toutes les phases du processus de création de nos solutions d'automatisation, comme par exemple :

- la création et la gestion de projets,
- la configuration et le paramétrage du matériel et de la communication,
- la gestion des mnémoniques,
- la création de programmes, par exemple pour les systèmes cibles S7,
- le chargement des programmes dans des systèmes cibles,
- le test de l'installation d'automatisation,
- le diagnostic lors de perturbations de l'installation.
- Introduction sur le produit et installation

### **I.7.3.Applications disponibles**

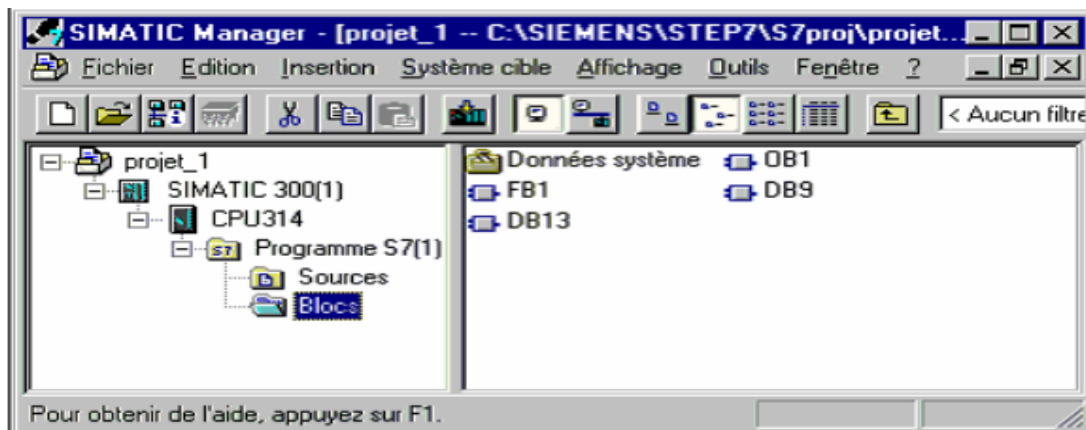
Le logiciel de base STEP 7 met à notre disposition différentes applications :



**Figure I.13:** différentes applications du logiciel STEP 7 [17]

### A. Gestionnaire de projets SIMATIC

Le gestionnaire de projets SIMATIC gère toutes les données relatives à un projet d'automatisation quel que soit le système cible (S7/M7/C7) sur lequel elles ont été créées. Le gestionnaire des projets SIMATIC démarre automatiquement les applications requises pour le traitement des données sélectionnées.



**Figure I.14:** Le gestionnaire de projet STEP 7 [17]

### B. Editeur des mnémoniques

L'éditeur des mnémoniques nous permet de gérer toutes les variables globales. Nous disposons des fonctions suivantes :

1. définition de désignations symboliques et de commentaires pour les signaux du processus (entrées/sorties), mémentos et blocs
2. fonctions de tri,
3. importation/exportation avec d'autres programmes Windows.

La table des mnémoniques qui en résulte est mise à disposition de toutes les applications. La modification de l'un des paramètres d'une mnémonique est de ce fait reconnue automatiquement par toutes les applications.

### C. Diagnostic du matériel

Le diagnostic du matériel fournit un aperçu de l'état du système d'automatisation. Dans une représentation d'ensemble, un symbole permet de préciser pour chaque module, s'il est défaillant ou pas. Un double clic sur le module défaillant permet d'afficher des informations détaillées sur le défaut.

Les informations disponibles dépendent des différents modules :

- affichage d'informations générales sur le module (p.ex. numéro de commande, version, désignation) et son état (p.ex. défaillant),
- affichage d'erreurs sur les modules (p.ex. erreur de voie) de la périphérie centrale et des esclaves DP,
- affichage des messages de la mémoire tampon de diagnostic.

Pour les CPU, des informations supplémentaires s'affichent :

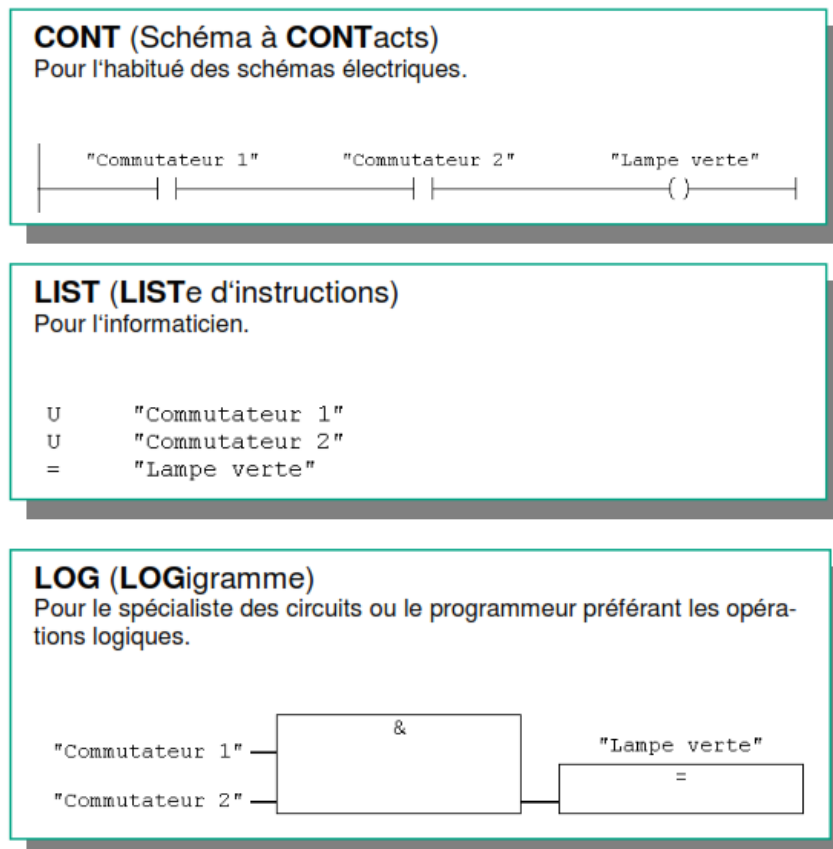
- causes de défaillance dans le déroulement d'un programme utilisateur
- durée de cycle (le plus long, le plus court et dernier),
- possibilités et charge de la communication MPI,
- performances (nombre d'entrées/sorties, de mémentos, de compteurs, de temporisations et de blocs possibles).

### D. Langages de programmation [2]

Les langages de programmation CONT, LIST et LOG pour S7-300/400 font partie intégrante du logiciel de base.

- **Le schéma à contacts (CONT)** est un langage de programmation graphique. La syntaxe des instructions fait penser aux schémas de circuits.  
CONT permet de suivre facilement le trajet du courant entre les barres d'alimentation en passant par les contacts, les éléments complexes et les bobines.
- **La liste d'instructions (LIST)** est un langage de programmation textuel proche de la machine. Dans un programme LIST, les différentes instructions correspondent, dans une large mesure, aux étapes par lesquelles la CPU traite le programme. Pour faciliter la programmation, LIST a été complété par quelques structures de langage évolué (comme, par exemple, des paramètres de blocs et accès structurés aux données).
- **Le logigramme (LOG)** est un langage de programmation graphique qui utilise les boîtes de l'algèbre de Boole pour représenter les opérations logiques. Les fonctions complexes,

comme par exemple les fonctions mathématiques, peuvent être représentées directement combinées avec les boîtes logiques.



**Figure I.15:** mode de représentation des langages basiques de programmation STEP 7 [18]

Nous disposons des logiciels de langage optionnels (langage évolué) suivants pour la programmation des automates programmables SIMATIC S7-300/400.

- GRAPH est un langage de programmation permettant la description aisée de commandes séquentielles (programmation de graphes séquentiels). Le déroulement du processus y est subdivisé en étapes. Celles-ci contiennent en particulier des actions pour la commande des sorties. Le passage d'une étape à la suivante est soumis à des conditions de transition.
- HiGraph est un langage de programmation permettant la description aisée de processus asynchrones non séquentiels sous forme de graphes d'état. A cet effet, l'installation est subdivisée en unités fonctionnelles pouvant prendre différents états. Ces unités fonctionnelles peuvent se synchroniser par l'échange de messages.
- SCL est un langage évolué textuel conforme à la norme DIN EN 61131-3. Il comporte des éléments de langage que l'on trouve également sous une forme similaire dans les langages de programmation Pascal et C. SCL convient donc particulièrement aux utilisateurs déjà habitués à se servir d'un langage de programmation évolué. Nous pouvons, par exemple, faire appel à SCL pour programmer des fonctions très complexes ou se répétant souvent. [2]

Nous disposons aussi des Langages graphiques [2]

- CFC pour S7 et M7 est un langage de programmation permettant l'interconnexion graphique de fonctions existantes. Ces fonctions couvrent un large éventail allant de combinaisons logiques simples à des régulations et commandes complexes. Un grand nombre de ces fonctions est disponible sous la forme de blocs dans une bibliothèque. La programmation se fait en copiant des blocs sur un diagramme et en reliant les connecteurs de blocs par des lignes.

#### **E. Paramétrage de l'interface PG-PC [16]**

Cet outil sert à paramétrer l'adresse locale des PG/PC, la vitesse de transmission dans le réseau MPI ou PROFIBUS en vue d'une communication avec l'automate et le transfert du projet.

### **I.8. Stratégie pour la conception d'une structure programme complète**

La mise en place d'une solution d'automatisation avec STEP 7 nécessite la réalisation des tâches fondamentales suivantes :

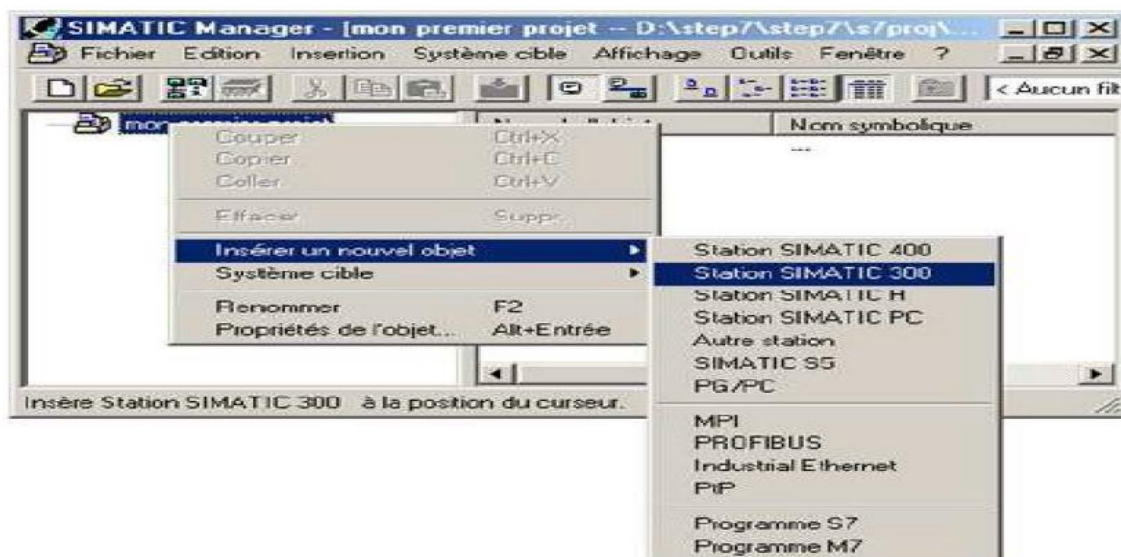
#### **I.8.1. Création du projet SIMATIC Step7**

Un projet comprend deux données essentielles, les programmes et la configuration du matériel, on peut commencer par définir l'une ou l'autre, mais tout d'abord il faut démarrer le programme SIMATIC Manager. Ce programme est l'interface graphique qui permet la manipulation du projet et l'accès aux autres programmes de STEP7.

Pour en créer un nouveau, il suffit de cliquer sur le bouton « Nouveau projet », attribuer un nom et valider. Ensuite il faut choisir une station de travail. Une station SIMATIC représente une configuration matérielle S7 comportant un ou plusieurs modules programmables. Il existe différents types:

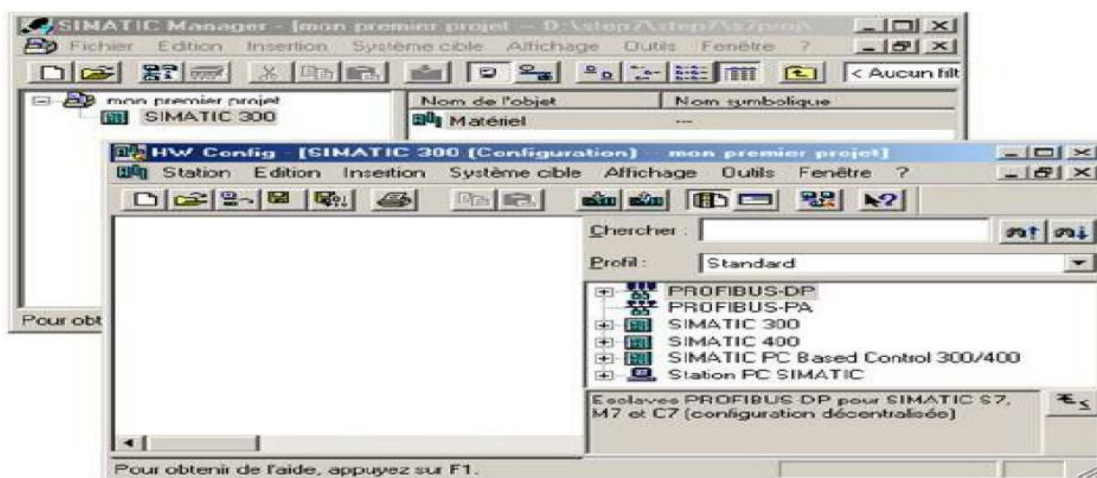
- SIMATIC 400
- SIMATIC 300
- SIMATIC S5
- SIMATIC H : Automate insensible aux défaillances, il se compose de 2 CPU du même type, en cas de problème elle commute de l'une vers l'autre sans perte de données.
- SIMATIC PC : ou Station PC, représente un PC contenant des composants SIMATIC : des applications (WinCC, par ex.), ou une carte CPU enfichée dans le PC.
- PG/PC : Outils de programmation pour contrôleurs SIMATIC, c'est une console de programmation compatible avec le milieu industriel.

- Autres stations : se sont soit des appareils d'autres fabricants ou bien des stations de SIMATIC S7 contenus dans un autre projet.



**Figure I.16:** *Choix de la station SIMATIC 300*

Pour commencer, le plus simple est de configurer le matériel, d'éditer les programmes puis les charger dans la CPU. Avec un double clic sur « Matériel », on démarre l'application HW Config.



**Figure I.17 :** *Configuration matériels pour la station S300*

### I.8.2. Configuration du matériel

Pour configurer le matériel, il suffit de faire glisser des éléments du catalogue dans l'emplacement approprié, on choisit le Rack, l'alimentation, la CPU et les E/S... Dans le catalogue on trouve les modules qu'on peut affecter à chaque type de station, on distingue:

- C7 : Système intégré compact qui regroupe automate programmable et interface homme machine (pupitre opérateur) pour la réalisation de commandes de machines sous encombrement réduit.

- CP : Communication Processor, module de communication (PROFIBUS, Industriel Ethernet,...).
- FM : (Function Module), il regroupe les modules de fonctions (régulation, comptage...).
- IM : Coupleurs d'extension, il permet l'ajout d'autres modules.
- M7 : Modules d'extension et cartouches interface pour SIMATIC M7.
- PS : Module d'alimentation.
- Rack : Support mécanique.
- Routeur : Relie Industriel Ethernet à PROFIBUS.
- SM : Signal Module, c'est le module d'E/S, il contient le AI module d'entrées analogiques, le AO module de sorties analogiques, le DI module d'entrées TOR et le DO module de sorties TOR.
- CPU : L'Unité Centrale, noté CPU xxx a b.
  - xxx est la famille de la CPU
  - a, b sont les propriétés de la CPU (éléments additionnels, port de communication...). Par exemple :
    - C : compact, la CPU intègre des modules E/S ainsi que des fonctions spécialisées.
    - PtP : Peer to Peer, la CPU intègre un port de communication Point to Point.
    - H : Fault-tolerant, des unités de traitements insensibles aux défaillances
    - DP : Decentralized Periphery, la CPU intègre un port de communication PROFIBUS.

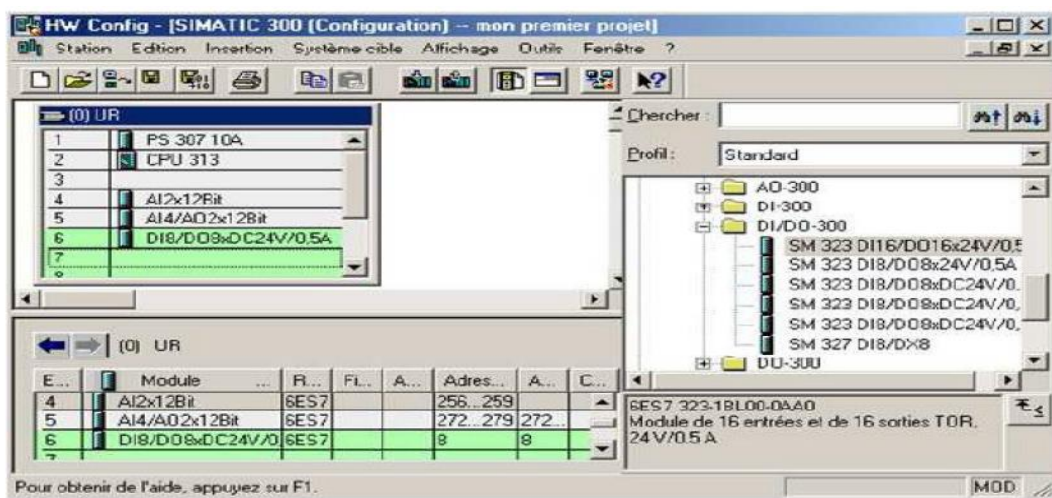


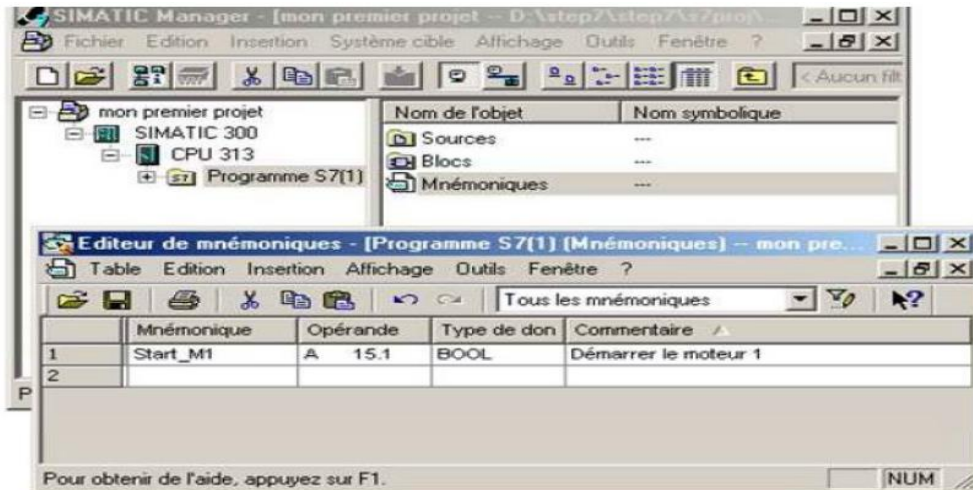
Figure I.18 : Sélection des modules



Si l'insertion de l'élément choisi est possible dans le Rack, la case appropriée devient verte. Une fois le matériel choisi on sauvegarde, on compile et on charge dans la CPU.

### I.8.3. Définition des mnémoniques

De retour dans le SIMATIC Manager, on trouve de nouveaux éléments. On commence par créer les mnémoniques dans la section programmes.



**Figure I.19 :** *Edition des Mnémoniques.*

En affectant des noms symboliques aux adresses absolues, les programmes deviennent plus lisible, faciles à corriger et à mettre à jour.

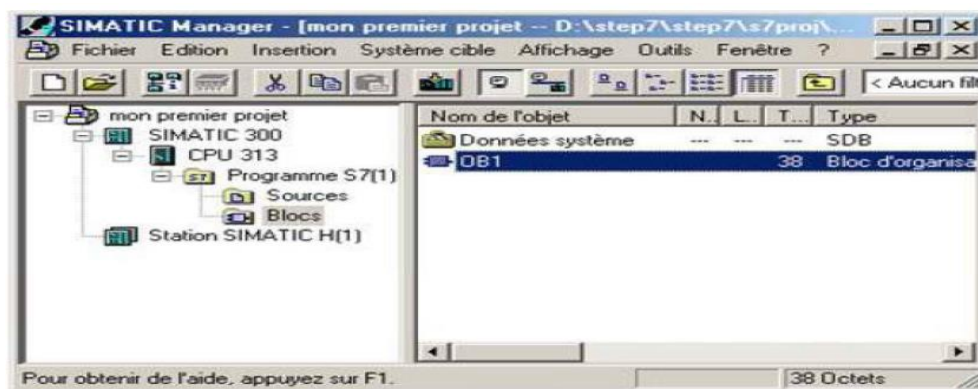
Il y a quatre différents types d'opérande : le bit, l'octet, le mot et le double mot. Ces types définissent l'accès à une zone mémoire. Pour chaque opérande un certain type de données est permis :

- Pour le bit : BOOL : variable booléenne (True ou False, 1 ou 0).
- Pour l'octet : deux types de données sont possibles :
  1. BYTE: nombre hexadécimal de B#16#0 à B#16#FF.
  2. CHAR : Caractère ASCII, 'A', 'B'...
- Pour le mot : quatre types de données sont possible :
  1. WORD : nombre hexadécimal de W#16#0 à W#16#FFFF.
  2. INT : nombre entier de -32768 à 32767.
  3. S5TIME : Durée S7 en pas de 10 ms (valeur par défaut), de S5T#0H\_0M\_0S\_10MS à S5T#2H\_46M\_30S\_0MS.
  4. DATE : Date en incréments de 1 jour, de D#1990-1-1 à D#2168-12-31.
- Pour le double mot : cinq types de données :
  1. DWORD: nombre hexadécimal de DW#16#0000\_0000 à DW#16#FFFF\_FFFF.

2. DINT : nombre entier de L#-2147483648 à L#2147483647.
3. REAL : nombre à virgule flottante, Limite supérieure : 3.402823e+38 Limite inférieure : 1.175 495e-38.
4. TIME : Durée en incréments de 1 ms, de -T#24D\_20H\_31M\_23S\_648MS à T#24D\_20H\_31M\_23S\_647MS.
5. TIME\_OF\_DAY : Heure en pas de 1ms, de TOD#0:0:0.0 à TOD#23:59:59.999.

#### I.8.4. Edition des programmes

Dans la section 'bloc' du SIMATIC Manager, on trouve par défaut le bloc d'organisation 1 'OB1' qui représente le programme cyclique. On peut rajouter d'autres blocs à tout moment par un clic droit dans la section Bloc de SIMATIC Manager.



**Figure I.20 :** *Edition des programmes*

Deux programmes différents s'exécutent dans la CPU: le système d'exploitation et le programme utilisateur.

Le système d'exploitation, organise toutes les fonctions et procédures dans la CPU qui ne sont pas liées à une tâche d'automatisation spécifique. Il gère le déroulement du démarrage à chaud et du redémarrage, l'actualisation de la mémoire image des entrées et l'émission de la mémoire image des sorties, l'appel du programme utilisateur, la gestion des zones de mémoire l'enregistrement des alarmes et l'appel des OB d'alarme...

Le programme utilisateur contient toutes les fonctions nécessaires au traitement des tâches d'automatisation spécifique. Ce programme doit être créé et chargé dans la CPU par l'utilisateur. Il détermine les conditions pour le démarrage à chaud et le redémarrage de la CPU (par exemple, initialiser des signaux), il traite les données du processus (par exemple, combiner des signaux binaires, lire et exploiter des valeurs analogiques), il doit réagir aux alarmes et traiter les perturbations dans le déroulement normal du programme.

Le STEP7 permet de structurer le programme utilisateur en le subdivisant en différentes parties autonomes ou dépendantes. Ceci permet d'écrire des programmes importants mais clairs, simples à tester et à modifier.

- **Blocs d'organisation**

Les blocs d'organisation (OB) constituent l'interface entre le système d'exploitation et le programme utilisateur. Ils sont appelés par le système d'exploitation et gèrent le traitement du programme cyclique et des programmes déclenchés par alarmes, ainsi que le comportement à la mise en route de l'automate programmable et le traitement des erreurs.

Les blocs d'organisation définissent l'ordre (événements de déclenchement) dans lequel les différentes parties du programme sont traitées. L'exécution d'un OB peut être interrompue par l'appel d'un autre OB (les OB de priorité plus élevée interrompent les OB de priorité plus faible). Les blocs d'organisations les plus prioritaires sont ceux de la mise en route (OB100, OB101 et OB102) et le moins prioritaire est le cycle en arrière-plan (OB90).

On appelle alarmes les événements qui déclenchent l'appel d'un OB donné. Le type d'alarme définit la classe de priorité de celle-ci.

- **Fonctions et blocs fonctionnels**

On peut programmer chaque bloc d'organisation en tant que programme structuré en créant des fonctions (FC) et des blocs fonctionnels (FB).

- Les blocs fonctionnels (FB) sont des blocs de code associés à des blocs de données d'instance, dans lesquels sont sauvegardés les paramètres effectifs et les données statiques des blocs fonctionnels.
- Les fonctions (FC) sont des blocs de code sans rémanence, c'est-à-dire qu'ils ne sont pas associés à des blocs de données, les paramètres effectifs ne sont pas sauvegardés automatiquement.

De plus il existe les blocs fonctionnels système (SFB) et les fonctions système (SFC), qui sont des fonctions préprogrammées. Ils peuvent être appelés à partir du programme utilisateur. On trouve des fonctions système pour la copie de blocs de données, le contrôle du programme utilisateur, la gestion des alarmes horaires et temporisées...

- **Bloc de données**

Les blocs de données (DB) servent à l'enregistrement de données utilisateur. Les blocs de données globales servent à l'enregistrement de données qui peuvent être utilisées par tous les autres

blocs. Les blocs de données d'instances sont affectés à des blocs fonctionnels. Les différents blocs cités ci-dessus peuvent être édités avec l'application « CONT LIST LOG » .

La programmation des blocs de codes peut se faire à l'aide de trois applications :

1. CONT LIST LOG : elle permet de programmer des blocs d'organisations OB, des blocs fonctionnels FB et des fonctions FC.
2. GRAPH : elle permet de programmer des blocs fonctionnels FB.
3. SCL : elle permet de créer des sources de code. Une source de code est un fichier texte, qui contient une suite d'instructions, une fois compilé il peut être transféré dans la CPU. On peut trouver dans un même fichier source tout le programme utilisateur, c'est-à-dire les blocs d'organisations, les blocs fonctionnels et les fonctions.

### I.8.5. Simulation de modules Avec le logiciel PLCSIM [14]

S7-PLCSIM dispose d'une interface simple permettant de visualiser et de forcer les différents paramètres utilisés par le programme (comme, par exemple, d'activer ou de désactiver des entrées).

En outre, S7-PLCSIM possède les fonctions suivantes :

- On peut créer des « fenêtres » dans lesquelles on a la possibilité d'accéder aux zones de mémoire d'entrée et de sortie, aux accumulateurs ainsi qu'aux registres de la CPU de simulation. On peut également accéder à la mémoire par adressage symbolique (il faut juste charger la table des mnémoniques dans 'options', puis sur 'outils' 'insérer mnémoniques').
- On peut sélectionner l'exécution automatique des temporisations ou encore les définir et les réinitialiser manuellement.
- On a la possibilité de changer l'état de fonctionnement de la CPU (STOP, RUN et RUNP) comme pour une CPU réelle. De plus, on dispose d'une fonction de pause qui permet d'interrompre momentanément la CPU, sans affecter l'état du programme.

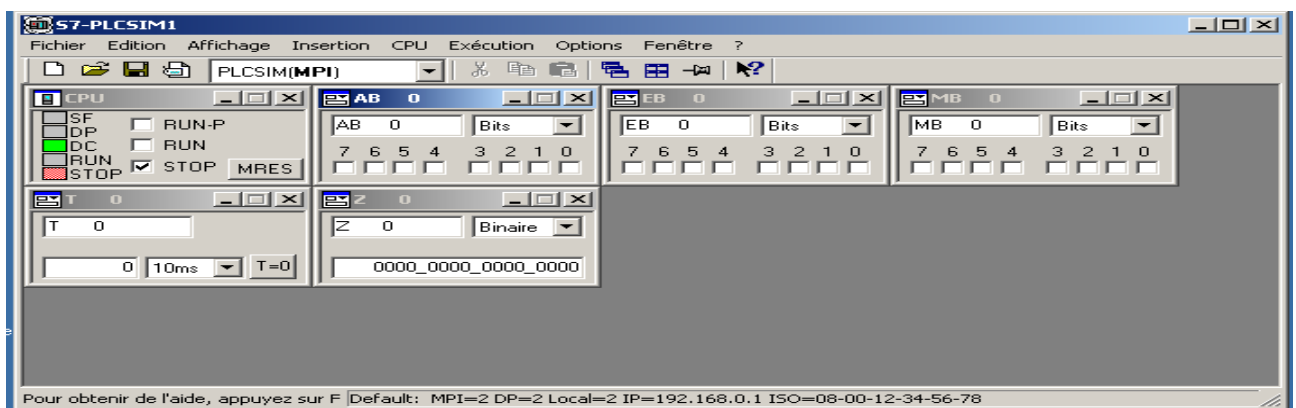


Figure I.21 : logiciel de simulation PLC-SIM [14]

### I.8.6. Chargement du programme dans la CPU

Une fois la configuration, le paramétrage et la création du programme terminés, on peut transférer le programme utilisateur dans le système cible. La CPU contient déjà le système d'exploitation.

### I.8.7. Surveillance du fonctionnement et diagnostic du matériel [6]

La détermination des causes d'un défaut dans le déroulement d'un programme utilisateur se fait à l'aide de la « Mémoire tampon de diagnostic », accessible depuis le SIMATIC Manager.

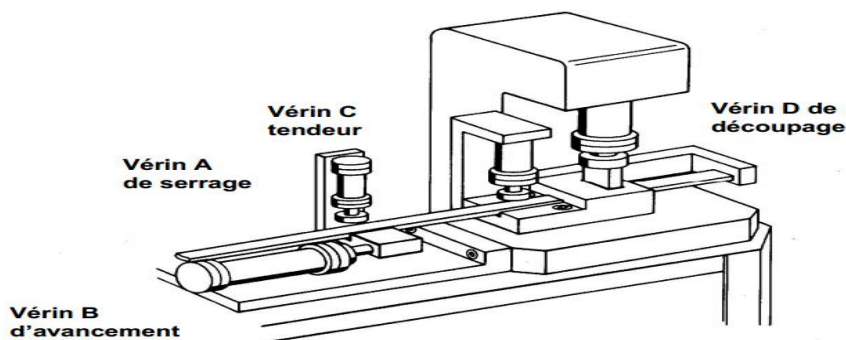
## I.9. Exemples de Création et d'édition d'un projet S7 :

Pour bien illustrer les démarches de conception d'un programme, on va traiter deux exemples dans le premier exemple on va programmer avec le langage évolué S7-GRAPH avec les différentes étapes de création d'un projet S7, et le deuxième exemple décrit la programmation avec des entrées/sorties logiques avec le langage CONT et LIST et la programmation avec des entrées/sorties analogiques.

### I.9.1. Exemple sur la Programmation Séquentielle avec le langage Evolué S7-GRAPH :

On va programmer une commande pas à pas à l'aide de l'outil graphique S7-GRAPH. Les étapes suivantes montrent la procédure au moyen d'un exemple détaillé.

#### A. Descriptions des tâches de commande [20]



**Figure I.22 :** Une perceuse automatique [20]

On souhaite créer une commande pour une perceuse automatique. La barre va être découpée par un fonctionnement commun de l'unité de travail et de la lame.

Le déplacement est produit par le vérin d'avancement (vérin B), qui est entraîné par les mouvements de va et vient de la pince de serrage pneumatique (vérin A). Si la barre se trouve bloquée contre un arrêt, elle est retenue par le tendeur (vérin C). Ensuite le découpage peut commencer (vérin D) et en même temps se produit l'ouverture de la pince de serrage (vérin A). Si

la pince de serrage (vérin A) est ouverte, alors le retour (vérin B) s'effectue jusqu'en position initiale. Quand le découpage est terminé (vérin D) et que l'unité de travail a atteint sa position initiale, le tendeur (vérin C) s'ouvre et un nouveau cycle de travail peut commencer.

Le démarrage est déclenché en appuyant sur la pédale de démarrage, si tous les vérins se trouvent dans leurs positions finales.

### B. Mise en forme chronologique

Le piston du vérin A descend et applique une pression,

le piston du vérin B se déplace et décale la barre jusqu'à la butée,

le piston du vérin C descend et tend la barre dans la direction de la perceuse,

le piston du vérin A remonte (la pince de serrage est alors ouverte) et le vérin D descend (Découpage), le piston du vérin B revient (l'unité de poussée revient à la position initiale) et le piston du vérin D remonte, le piston du vérin C remonte et relâche la tension.

- Afin d'étudier la perceuse automatique, on doit d'abord assigner les variables conditionnelles et les éléments de travail du Graph-7.

### C. Variables conditionnelles :

- **Les entrées :**

S0	E	1.0	BOOL	START
S1	E	0.0	BOOL	capteur de fin de course de la sortie du verin A
S2	E	0.1	BOOL	capteur de fin de course de la rentree du verin A
S3	E	0.2	BOOL	capteur de fin de course de la sortie du verin B
S4	E	0.3	BOOL	capteur de fin de course de la rentree du verin B
S5	E	0.4	BOOL	capteur de fin de course de la sortie du verin C
S6	E	0.5	BOOL	capteur de fin de course de la rentree du verin C
S7	E	0.6	BOOL	capteur de fin de course de la sortie du verin D
S8	E	0.7	BOOL	capteur de fin de course de la rentree du verin D

- **Les sorties :**

VERIN_A	A	0.0	BOOL	Sortie/rentrée du piston du vérin A
VERIN_B	A	0.1	BOOL	Sortie/rentrée du piston du vérin B
VERIN_C	A	0.2	BOOL	Sortie/rentrée du piston du vérin C
VERIN_D	A	0.3	BOOL	Sortie/rentrée du piston du vérin D

### D. CREATION D'UN PROGRAMME S7-GRAPH:

Démarrer SIMATIC- Manager et créer un nouveau projet avec le nom : Perceuse automatique

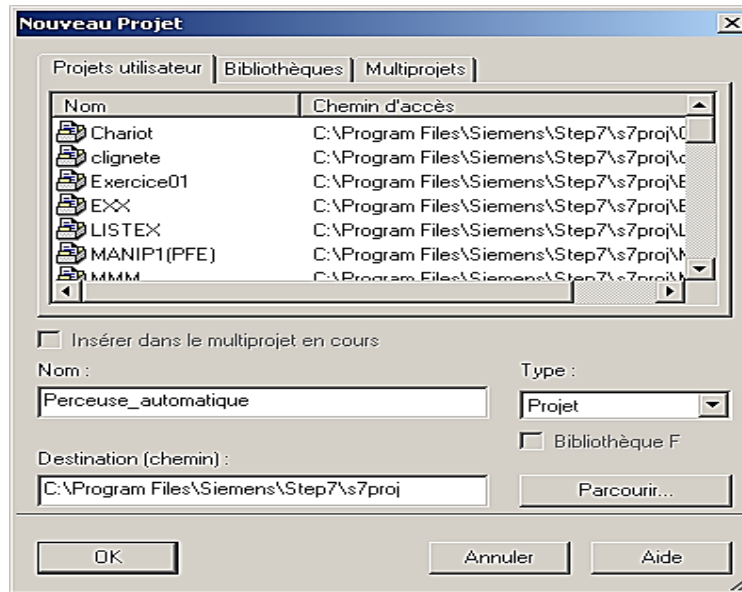


Figure I.23 : Création d'un nouveau projet [20]

Exemple de configuration :

- On choisit une CPU 314 IFM
- Double-clic sur **Matériel** : on insère un module d'entrée/sortie numérique : DI 16/ DO 16xDC24V/ 0,5 A

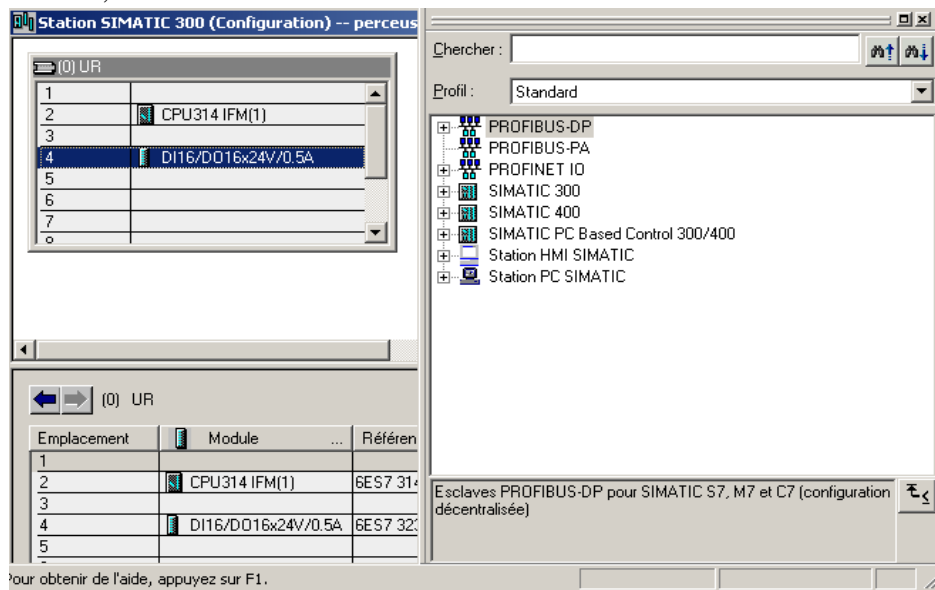


Figure I.24 : Configuration tu matériel

- Ensuite on va Créer la table des mnémoniques : avec double-clic sur mnémoniques

Etat	Mnémonique	Opérande	Type de d	Commentaire
1	Cycle Execution	OB 1	OB 1	
2	G7_STD_3	FC 72	FC 72	
3	S0	E 1.0	BOOL	START
4	S1	E 0.0	BOOL	capteur de fin de course de la sortie du verin A
5	S2	E 0.1	BOOL	capteur de fin de course de la rentrée du verin A
6	S3	E 0.2	BOOL	capteur de fin de course de la sortie du verin B
7	S4	E 0.3	BOOL	capteur de fin de course de la rentrée du verin B
8	S5	E 0.4	BOOL	capteur de fin de course de la sortie du verin C
9	S6	E 0.5	BOOL	capteur de fin de course de la rentrée du verin C
10	S7	E 0.6	BOOL	capteur de fin de course de la sortie du verin D
11	S8	E 0.7	BOOL	capteur de fin de course de la rentrée du verin D
12	TIME_TCK	SFC 64	SFC 64	Read the System Time
13	VERIN_A	A 0.0	BOOL	Sortie/rentrée du piston du vérin A
14	VERIN_B	A 0.1	BOOL	Sortie/rentrée du piston du vérin B
15	VERIN_C	A 0.2	BOOL	Sortie/rentrée du piston du vérin C
16	VERIN_D	A 0.3	BOOL	Sortie/rentrée du piston du vérin D
17				

On insère le bloc fonctionnel FB1 et on choisit le langage de programmation GRAPH.

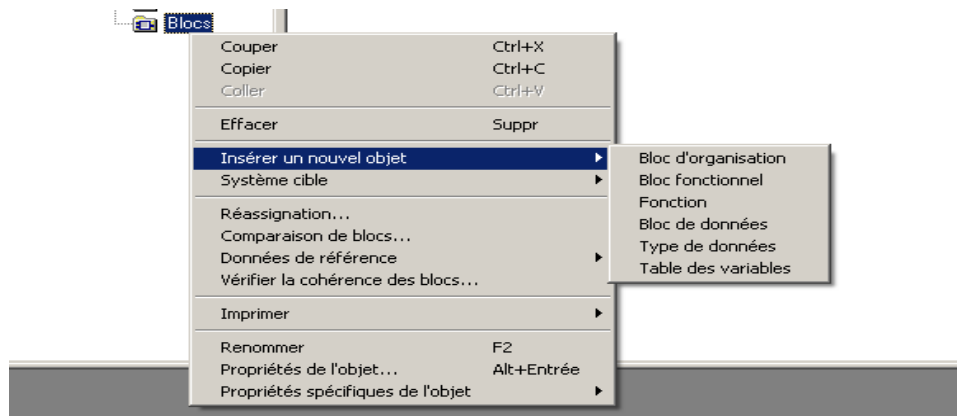


Figure I.25: Insertion d'un bloc fonctionnel

- Le programme S7 GRAPH s'ouvre en cliquant sur FB1 :

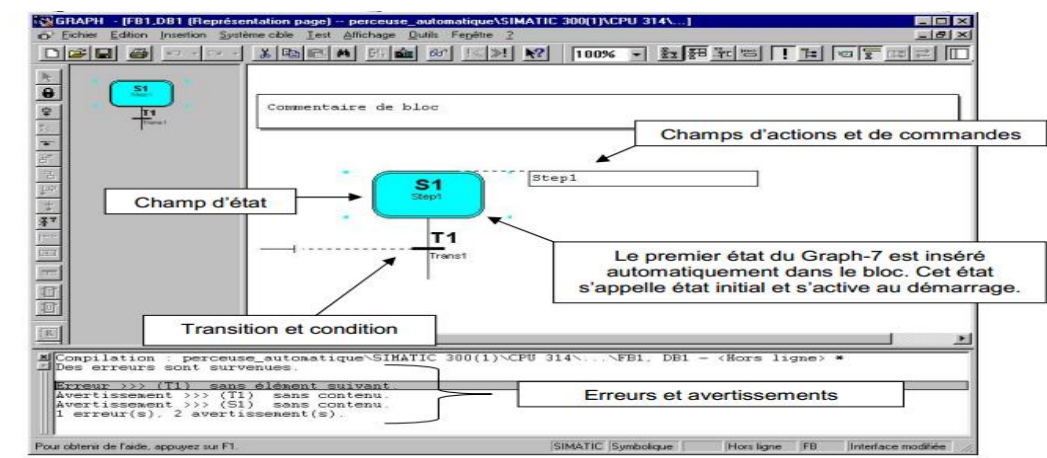
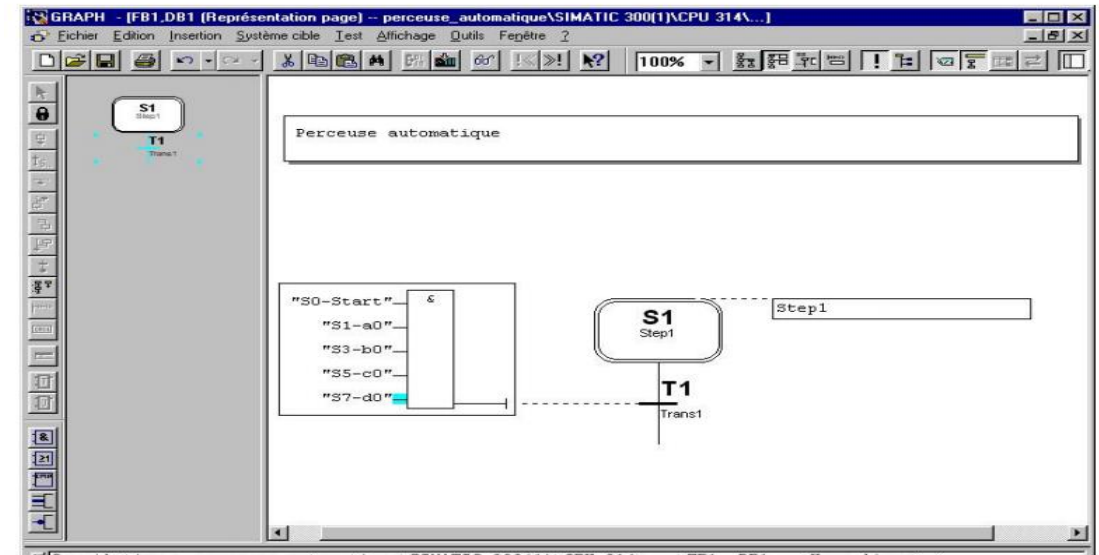


Figure I.25: Composants d'un programme S7 GRAPH



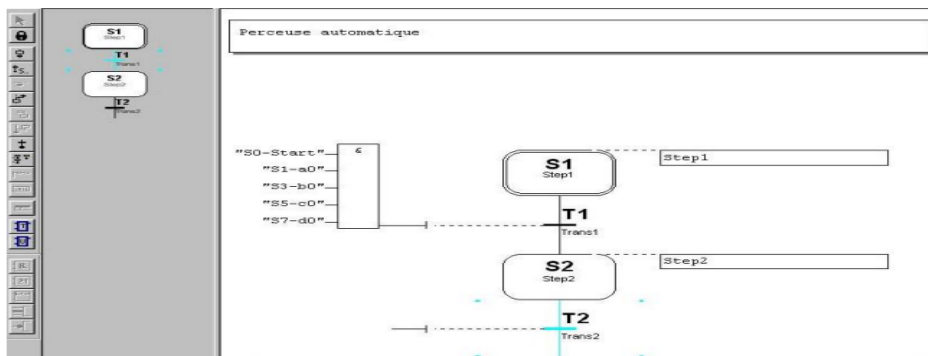
- **les étapes de création du programme :**

1. Double-cliquer et saisir le commentaire de bloc et la désignation de l'étape ;
2. Cliquer sur l'entrée de la transition ;
3. Insérer une boîte ET ;
4. Ajouter des entrées binaires ;
5. Saisir les opérandes de la boîte ET.



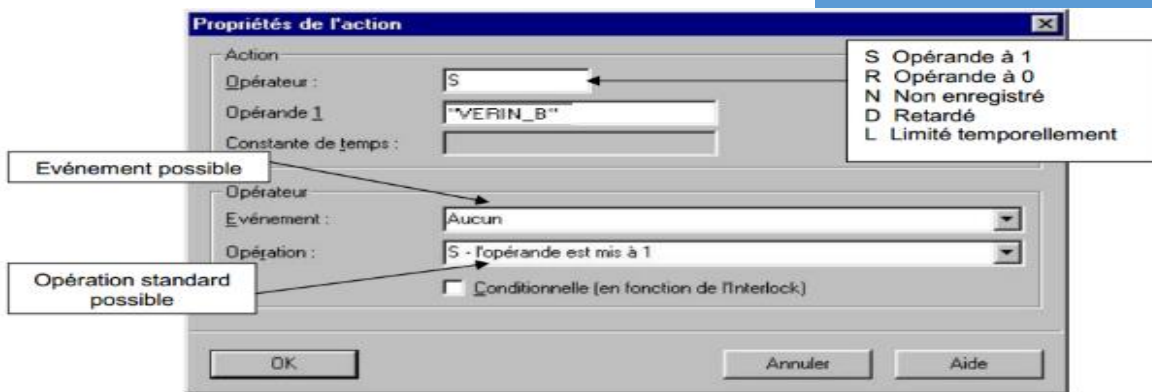
**Figure I.26:** Création de la première étape

6. Cliquer sur la transition T1 et Insérer une étape + transition ;



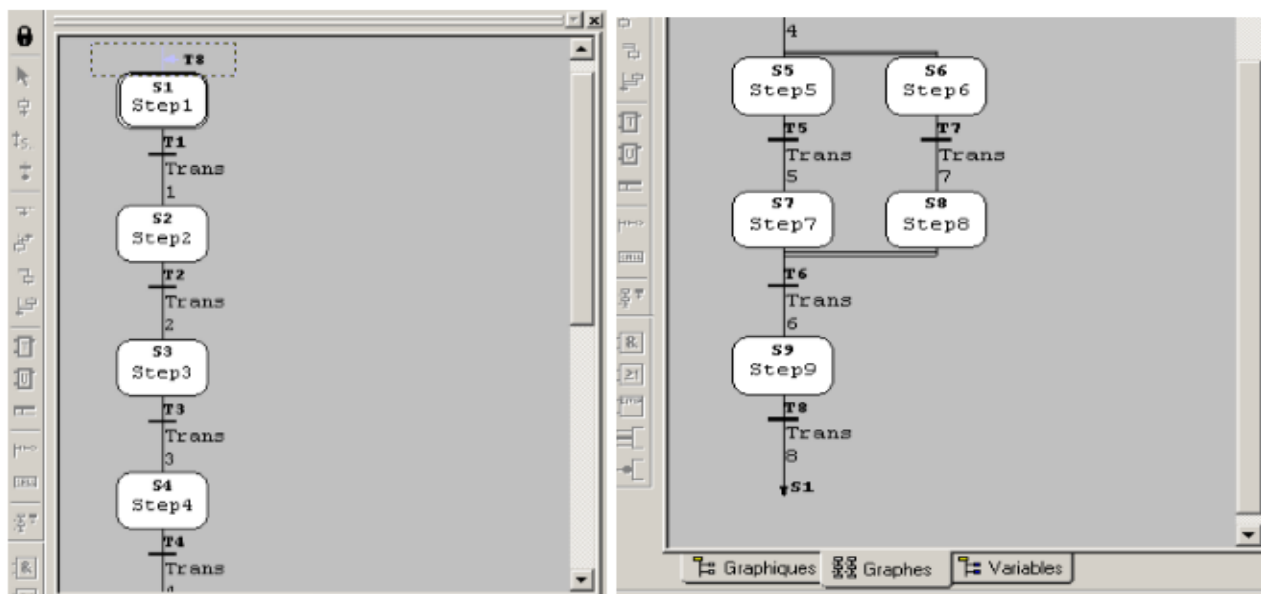
**Figure I.27:** Insertion des actions

7. Cliquer sur le commentaire et saisir le nom de l'étape ;
8. Insérer une action ;
9. Saisir les actions à exécuter en double-cliquant ou avec les boutons droit de la souris en choisissant propriétés de l'objet ;
10. Saisir la transition ;



**Figure I.28:** Saisie des actions

11. On procède de la même façon pour les autres étapes et pour la dernière étape on insère un retour vers l'étape initiale( Insérer un saut).

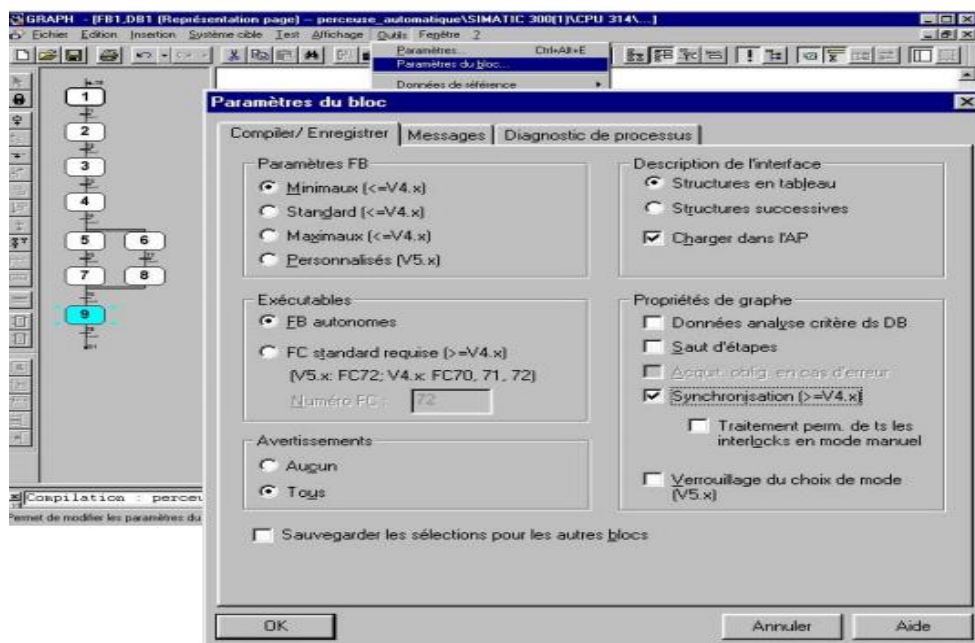


**Figure I.29:** Grafcet final

**E. Configurer les paramètres du bloc et enregistrer le bloc fini**

Avant l'enregistrement du bloc nous devons modifier ses propriétés.

1. Cliquer sur **Outils**.
2. Cliquer sur **paramètres du bloc**.

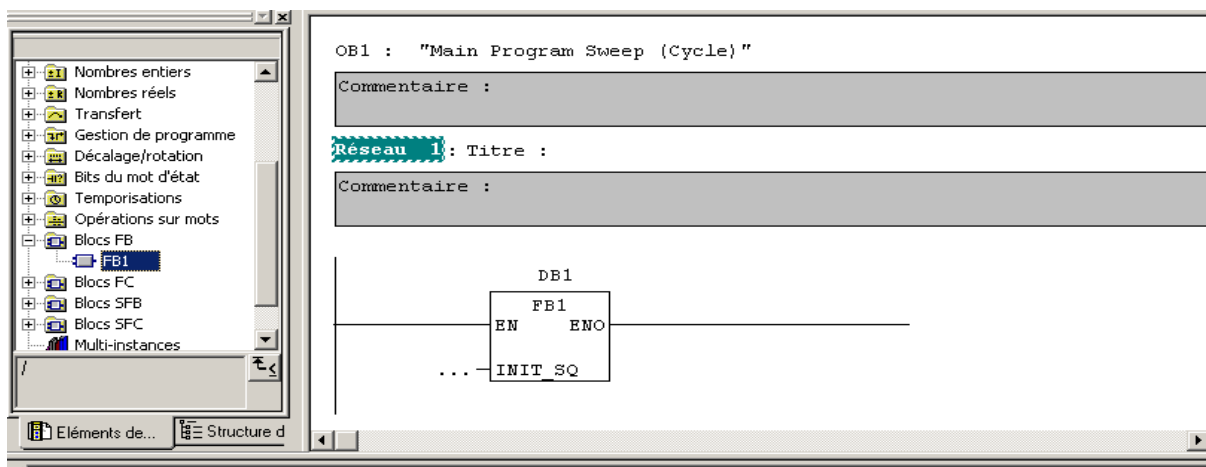


**Figure I.30 :** Configuration des paramètres du bloc et enregistrement du bloc fini

3. Mettre les paramètres FB à Minimaux.
4. Mettre exécutable à FB autonome afin que le FC standard soit relié au bloc fonctionnel.
5. Cocher Synchronisation.
6. Cliquer sur OK.
7. Enregistrer le bloc.
8. Fermer S7-GRAPH.

**F. Traiter le bloc d'organisation OB1 et charger les blocs dans le module**

1. Saisir la rubrique de blocs et la rubrique de réseau.
2. Cliquer dans le champ de saisie.
3. Ouvrir Eléments de programme.
4. Insérer FB1 en double-cliquant.
5. Entrer DB1.
6. Enregistrer le bloc OB1.
7. Fermer CONT/LIST/LOG



**Figure I.31:** Insertion du bloc fonctionnel(FB1) dans le bloc organisationnel(OB1)

8. Cliquer sur blocs et charger dans le module

Après le transfert du bloc dans la CPU, le programme peut être testé avec le logiciel de simulation :

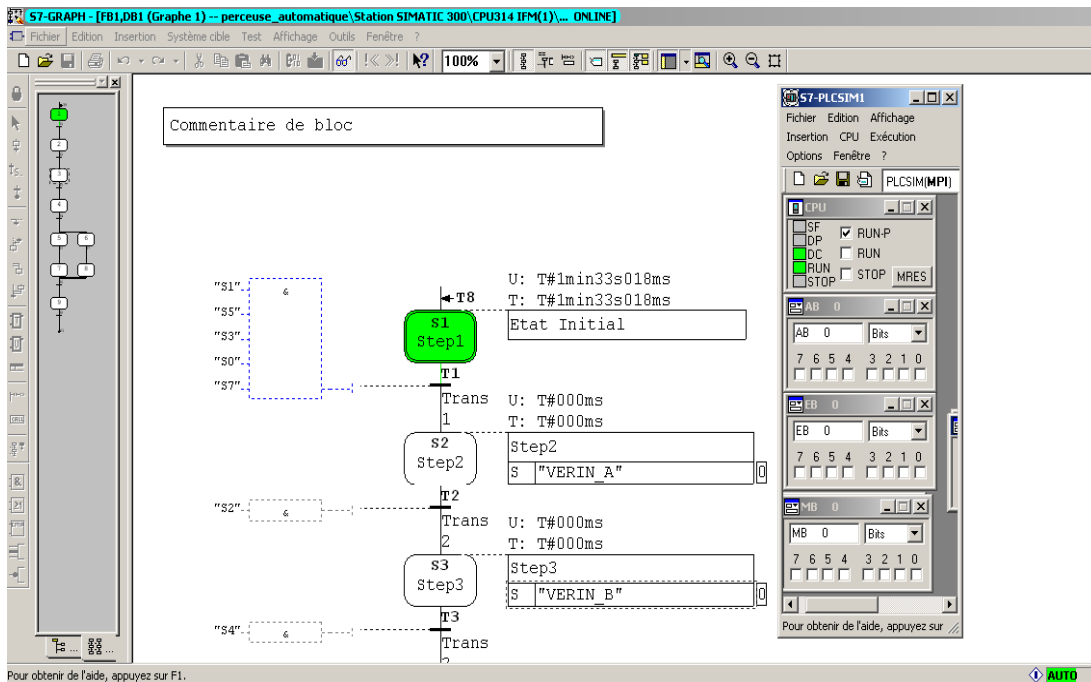


Figure I.32: Simulation du programme S7 GRAPH

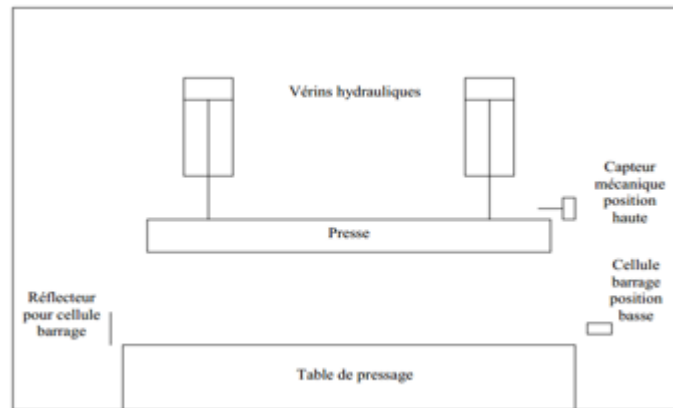
I.9.2.Exemple sur la Programmation avec des entrées/sorties Logique et Analogique:

A. Programmation avec des entrées /sorties Logiques.

Nous allons maintenant commencer la programmation de bloc. Pour cela nous allons nous appuyer sur une première application. Il s’agit d’un premier temps de programmer la commande d’une presse hydraulique en mode manuel. En second, nous programmerons le mode automatique de cette presse.

- Fonctionnement en mode manuel :

Au repos, la presse est en haut avec un retour de capteur mécanique « Presse en position haute », l’appui sur un bouton « Presse » fera sortir les vérins hydrauliques pour descendre la presse sur un capteur barrage « Presse en position basse ». Dès que le bouton « Presse » sera relâché, la presse remontera jusqu’au capteur haut. Remarque : une fois la presse en position basse, l’opérateur devra maintenir le bouton « Presse » pendant le temps de pressage, par contre celle-ci remontera automatiquement au bout de 15 secondes.



**Figure I.33** : Schéma de la presse hydraulique

- Entrées / Sorties :

E0.0 : Capteur position haute (=1 : presse en haut) ----FDC\_H: E 0.0

E0.1 : Capteur position basse (=1 : presse en bas) ----FDC\_B: E 0.1

A0.0 : Commande vérins (=1 : sortir vérins ; =0 : monter vérins) ----CMD\_Presse : A 0.0

M10.0 : Bouton « Presse » (=1 : demande de pressage) et BP\_presse : E 0.2

E 0.4 : MARCHE/ARRET.

----Pression : MW 22

----RESET : M 124.5

----SORTIE\_ANAL:PAW 272

----AUTO/MANU : M 124.4

Maintenant que nous avons toutes les informations nous pouvons lancer le logiciel et ouvrir un nouveau projet

Nous voyons maintenant notre projet complet. Nous allons dans un premier temps ajouter les variables à notre projet.

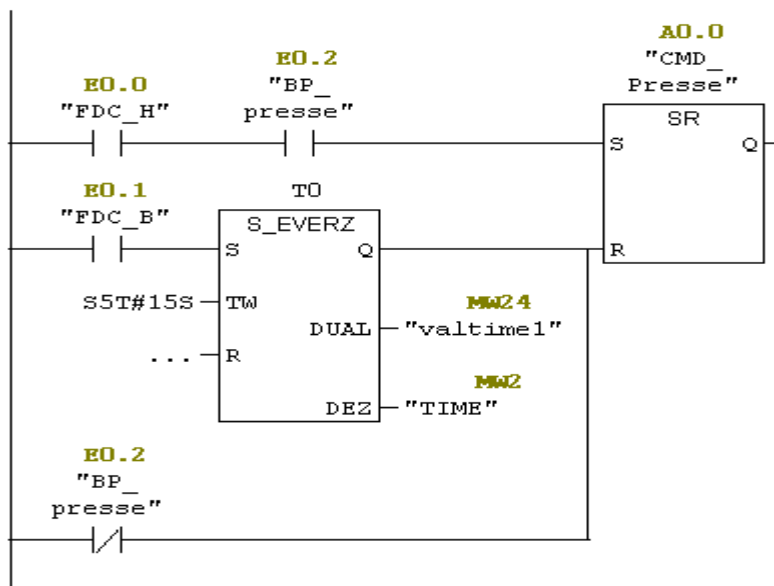
Nous pouvons alors créer notre fonction de gestion du mode manuel de la presse.

FC1 : commande manuel

Commentaire :

**Réseau 1** : commande manuel de presse hydraulique

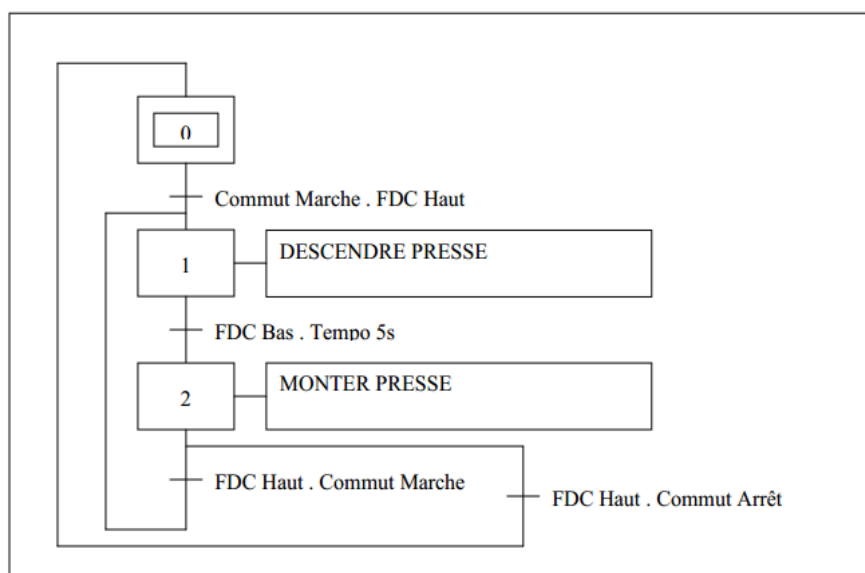
```
CMD_presse: A0.0
Commande vérins (=1 : sortir vérins=>S=1 ;
                =0 : monter vérins=>R=1 )
```



**Figure I.33 :** programme du mode manuel de la presse hydraulique

- Fonctionnement Automatique :

Nous allons maintenant programmer le fonctionnement en automatique de la presse dans une nouvelle fonction. Données de fonctionnement en mode automatique : Au repos, la presse est en haut avec un retour de capteur mécanique « Presse en position haute », l'appui sur un bouton « Marche » fera démarrer le cycle de pressage. L'appui sur un bouton « Arrêt » laissera terminer le cycle puis arrêtera la presse en position haute. Un BP d'arrêt d'urgence fera remonter la presse directement.



**Figure I.34 :** Grafset du fonctionnement automatique de la presse hydraulique

- Entrées / Sorties :

E0.0 : Capteur position haute (=1 : presse en haut)

E0.1 : Capteur position basse (=1 : presse en bas)

E0.2 : AU (=0 : AU enclenché).

A0.0 : Commande vérins (=1 : sortir vérins ; =0 : monter vérins)

M10.1 : Commutateur « Marche/Arrêt » (=0 : cycle à l'arrêt)

Nous pouvons alors créer notre fonction de gestion du mode Automatique de la presse.

- **Solution de programmation du mode Automatique de la presse :**

FC2 : commande automatique

Commentaire :

**Réseau 1**: Timer

Commentaire :

```

U    "X2"                M124.2
L    S5T#5S
SE   T    1
LC   T    1
T    "TIME1"            MW4
T    "valtime2"        MW26

```

**Réseau 2** : S0

```

O    "RESET"            M124.5
O
U    "X3"                M124.3
U    "FDC_H"            E0.0
UN   "MARCHE/ARRET"    E0.4
S    "X0"                M124.0

```

**Réseau 3**: S1

```

U    "FDC_H"            E0.0
U(
U    "X0"                M124.0
U    "MARCHE/ARRET"    E0.4
O
U    "X3"                M124.3
U    "MARCHE/ARRET"    E0.4
)
S    "X1"                M124.1

```

```

Réseau 4 : S2
    U    "X1"                M124.1
    U    "FDC_B"            EO.1
    S    "X2"                M124.2

Réseau 5 : S3
    U    "X2"                M124.2
    U    T      1
    S    "X3"                M124.3

Réseau 6 : R0
    U    "FDC_H"            EO.0
    U    "X0"                M124.0
    U    "MARCHE/ARRET"    EO.4
    R    "X0"                M124.0

Réseau 7 : R1
    U    "X1"                M124.1
    U(
    O    "RESET"            M124.5
    O    "FDC_B"            EO.1
    )
    R    "X1"                M124.1

Réseau 8 : R2
    U    "X2"                M124.2
    U(
    O    T      1
    O    "RESET"            M124.5
    )
    R    "X2"                M124.2

Réseau 9 : R3
    U    "X3"                M124.3
    U(
    O    "RESET"            M124.5
    O
    U    "FDC_H"            EO.0
    U(
    O    "MARCHE/ARRET"    EO.4
    ON   "MARCHE/ARRET"    EO.4
    )
    )
    R    "X3"                M124.3

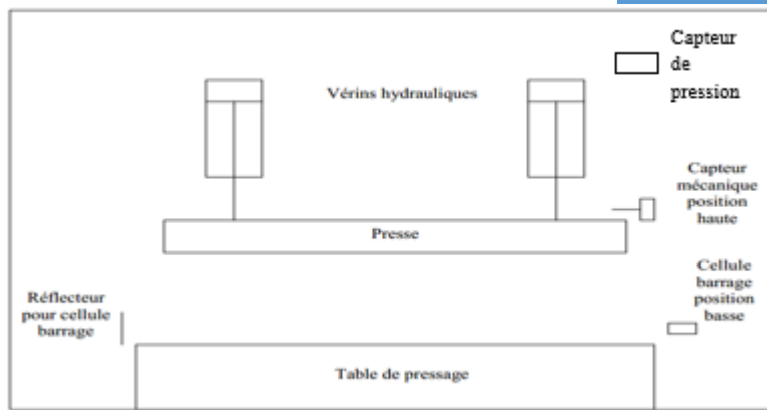
Réseau 10 : CMD_PRESSE
    U    "X1"                M124.1
    S    "CMD_Presse"        A0.0
    U    "X3"                M124.3
    R    "CMD_Presse"        A0.0

```

Figure I.35 : Programme du fonctionnement automatique de la presse hydraulique

## B. Programmation avec des entrées/sorties analogiques :





**Figure I.35 :** Programme du fonctionnement automatique de la presse hydraulique

Sur notre presse, nous avons un pressostat qui mesure la pression qui arrive sur le vérin, ce capteur de pression est relié sur une des entrées analogiques de la CPU en 0/10V, nous retournons l'information de pression d'arrivé de 0 à 15 Bars. Nous allons récupérer la valeur de l'entrée analogique pour permettre de la convertir et l'afficher sur le pupitre en Bar puis de retransmettre cette valeur sur la sortie analogique pour permettre la lecture avec un voltmètre. Les valeurs analogiques sont lues et produites comme des mots d'informations dans l'automate. Chaque valeur analogique « canal » occupe un mot d'entrée et de sortie de périphérie. Le format est un nombre entier (INT). Pour notre application, nous allons lire une valeur de 0 à 10 V avec un module d'entrées analogiques. L'adresse de cette valeur est IW 64. Au début, la valeur est un ENTIER (16 Bits) et doit être normalisé entre 0 et 15 en format REEL puis stockée dans un double mot memento MD 20. Puis pour transférer cette valeur à la sortie ANA, nous avons une valeur entre 0 et 15 stockée sous format REEL sur le double mot memento MD20 devant être normalisée et produite de 0 à 10V par un module de sorties analogiques d'adresse **QW 64**.

$$\text{Equation 1 : } \mathbf{MD20} = (([\mathbf{IW 64}] \text{ convertie en réel} / \mathbf{27648}) * (\mathbf{A} - \mathbf{B})) + \mathbf{B}$$

Avec A : valeur maxi (15)

B : valeur mini (0)

$$\text{Equation 2 : } \mathbf{QW 64} = [((\mathbf{MD20} - \mathbf{B}) / (\mathbf{A} - \mathbf{B})) * \mathbf{27648}] \text{ arrondi à l'entier Avec A : valeur maxi (15) B : valeur mini (0) Pour info : 0 - 10 V correspond à 0 - 27648 points.}$$

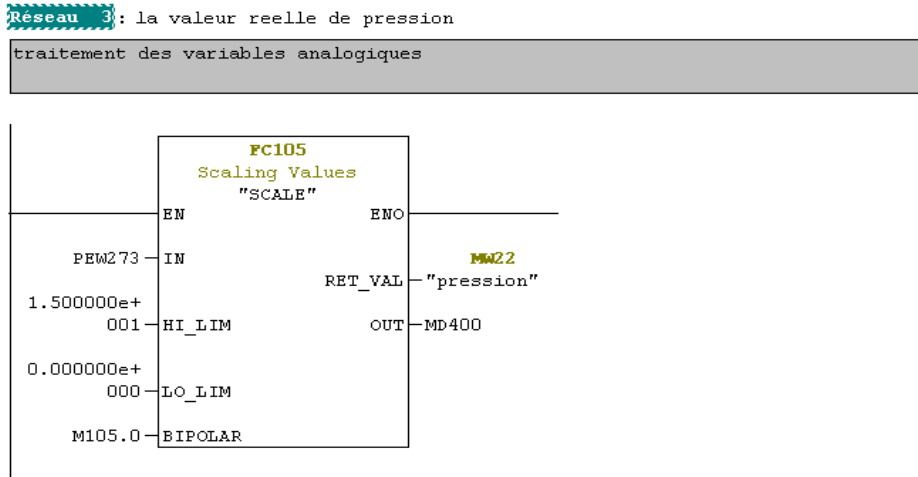


Figure I.36 : Fonction « SCALE »

- Nous allons maintenant programmer l'appel des fonctions dans l'OB cyclique et inclure un OB de démarrage. On ouvre l'OB1 et On ouvre les appels des fonctions. Pour cela, On clique sur la fonction et on la glisse sur le réseau

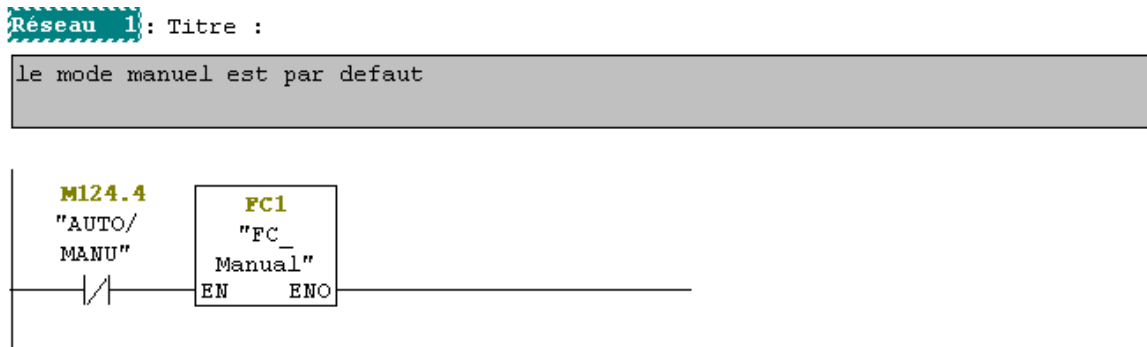


Figure I.37 : Fonction du mode manuel

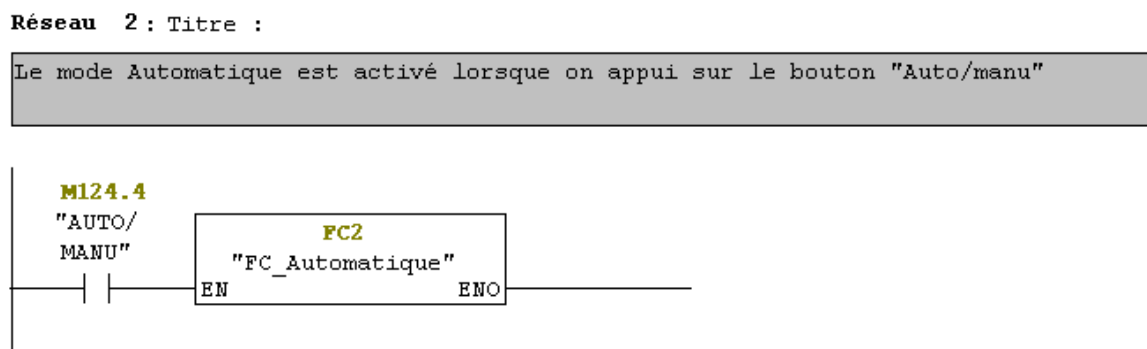


Figure I.38 : Fonction du mode automatique

### I.10.Conclusion

Le choix de l'architecture du système de contrôle, commande et l'élaboration de la documentation technique est une étape cruciale pour l'élaboration du programme d'automatisme pour l'application « Réalisation d'une maquette de station de pompage à base de l'automate

programmable SIEMENS ». Dans ce qui va suivre, nous allons détailler les principaux volets de la programmation de l'automate.

# CHAPITRE II

## **CHAPITRE II**

Le progiciel de conception des interfaces  
Homme/Machine WinCC flexible

## II.1 Introduction

Lorsque la complexité des processus augmente et que les machines et installations doivent répondre à des spécifications de fonctionnalité toujours plus sévères, l'opérateur a besoin d'un maximum de transparence. Cette transparence s'obtient au moyen de l'Interface Homme-Machine (IHM). [4]

Un système IHM constitue l'interface entre l'homme (opérateur) et le processus (machine /installation). Le contrôle proprement dit du processus est assuré par le système d'automatisation. Il existe par conséquent une interface entre l'opérateur et WinCC flexible (sur le pupitre opérateur) et une interface entre WinCC flexible et le système d'automatisation. Un système IHM se charge des tâches suivantes :

- Représentation du processus :  
Le processus est représenté sur le pupitre opérateur. Lorsqu'un état du processus évolue p. ex., l'affichage du pupitre opérateur est mis à jour.
- Commande du processus :  
L'opérateur peut commander le processus via l'interface utilisateur graphique. Il peut p. ex. définir une valeur de consigne pour un automate ou démarrer un moteur.
- Vue des alarmes :  
Lorsque surviennent des états critiques dans le processus, une alarme est immédiatement déclenchée, p. ex. lorsqu'une valeur limite est franchie.
- Archivage de valeurs processus et d'alarmes :  
Les alarmes et valeurs processus peuvent être archivées par le système IHM. On peut ainsi documenter la marche du processus et accéder ultérieurement aux données de la production écoulée.
- Documentation de valeurs processus et d'alarmes :  
Les alarmes et valeurs processus peuvent être éditées par le système IHM sous forme de journal. On peut ainsi consulter les données de production à la fin d'une équipe p. ex.
- Gestion des paramètres de processus et de machine :  
Les paramètres du processus et des machines peuvent être enregistrés au sein du système IHM dans des recettes. Ces paramètres sont alors transférables en une seule opération sur l'automate pour démarrer la production d'une variante du produit p. ex.

## II.2. SIMATIC HMI [21]

SIMATIC HMI offre une gamme complète permettant de couvrir toutes les tâches de contrôle-commande. SIMATIC HMI nous permet de maîtriser le processus à tout instant et de maintenir les machines et installations en état de marche.

### II.3. Utilisation de SIMATIC WinCC flexible [21]

WinCC flexible est le logiciel IHM pour la réalisation, par des moyens d'ingénierie simples et efficaces, de concepts d'automatisation évolutifs, au niveau machine. WinCC flexible réunit les avantages suivants :

- simplicité
- ouverture
- flexibilité

### II.4. Interface logicielle de WinCC flexible[15]

Nous avons accès à toutes les fonctionnalités prises en charge par le pupitre opérateur connecté. Nous démarrons WinCC flexible, soit par l'icône placée sur le bureau de notre ordinateur de configuration, soit par le menu Démarrer de Windows.

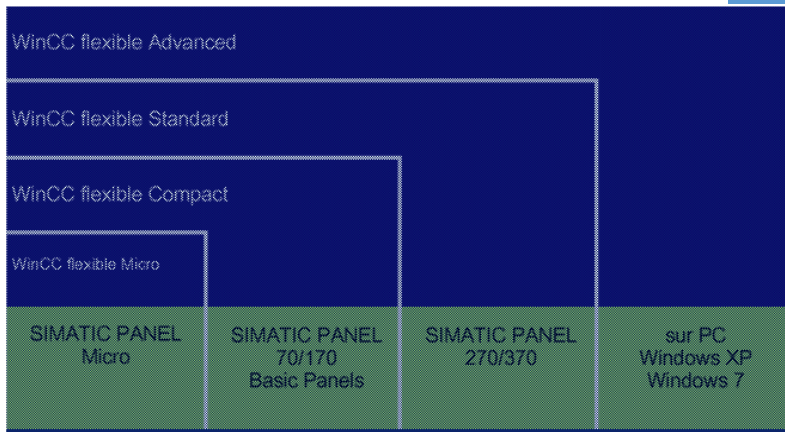


Composants de WinCC flexible : [25]

- Le logiciel de configuration WinCC flexible Advanced nous permet de créer notre configuration sous Windows sur le PC de configuration.
- Le logiciel de visualisation de processus WinCC flexible Runtime nous permet d'exécuter notre configuration sous Windows et de visualiser le processus. Nous utilisons aussi WinCC flexible Runtime sur notre PC de configuration pour le test et la simulation du fichier projet généré.

### II.5. WinCC flexible Engineering System

WinCC flexible est un logiciel modulaire. Chaque incrément d'édition élargit l'éventail des appareils cibles et fonctionnalités pris en charge. Le PowerPack nous permet de passer à tout moment à l'édition supérieure. [25]



- WinCC flexible couvre toute l'étendue de la gamme allant des micro panels à la visualisation simple sur PC.

Lorsque on crée ou on ouvre un projet sous WinCC flexible, l'écran de l'ordinateur de configuration affiche WinCC flexible Workbench. La fenêtre de projet affiche la structure du projet et permet de gérer celui-ci.

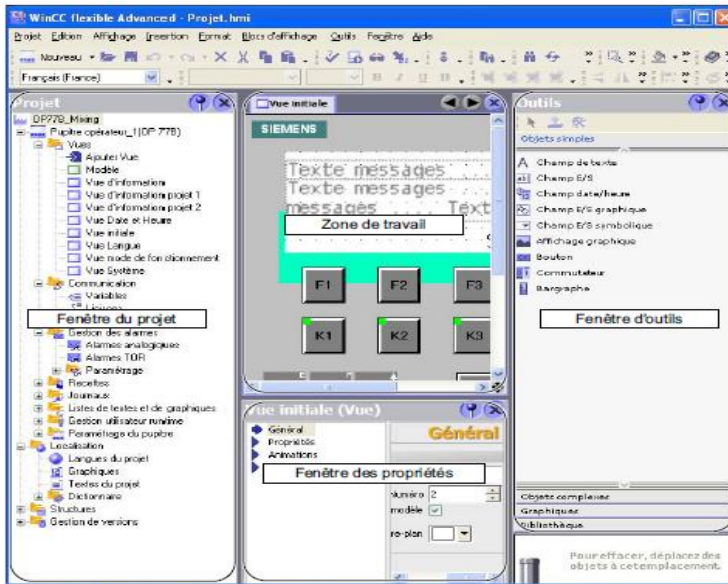
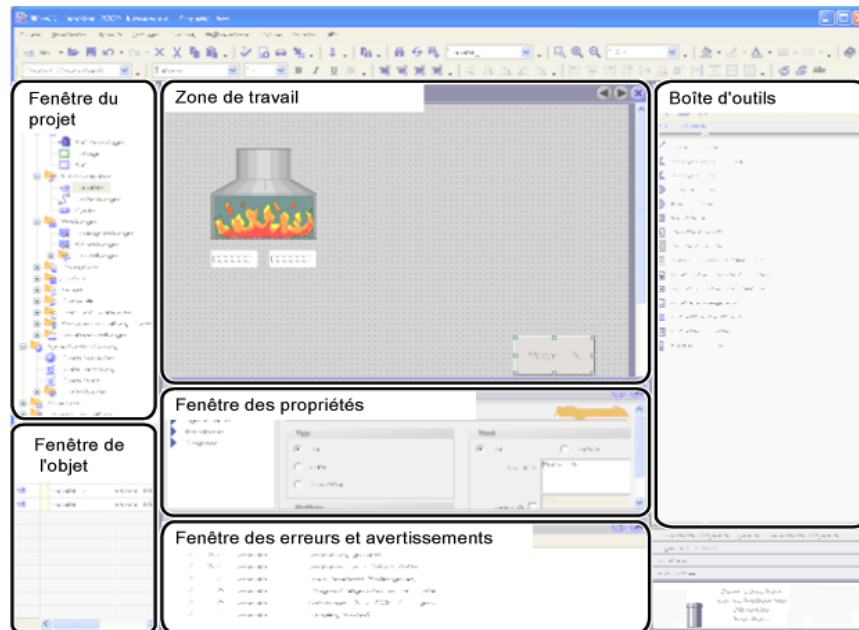


Figure II.1: Présentation de l'interface Wincc flexible. [15]

## II.6. Éléments de l'interface utilisateur de WinCC flexible

L'environnement de travail de WinCC flexible se compose de plusieurs éléments. Certains de ces éléments sont liés à des éditeurs particuliers et uniquement visibles lorsque cet éditeur est activé.





**Figure II.2 : Eléments de l'interface WinCC.**

WinCC flexible se compose des éléments suivants : [25]

**II.6.1.Menus et barres d'outils**

Nous trouverons dans les menus et barres d'outils toutes les fonctions dont nous avons besoin pour la configuration de notre pupitre opérateur. Lorsqu'un éditeur est actif, les commandes de menu ou barres d'outils correspondantes sont visibles.

Lors de la création d'un nouveau projet, les barres d'outils sont positionnées par défaut au bord supérieur de l'écran. La position des barres d'outils est liée à l'utilisateur qui est connecté sous Windows. Lorsque nous démarrons WinCC flexible et que, lors de la session précédente, nous avons déplacé des barres d'outils avec la souris, ces barres d'outils reprennent la position qu'elles avaient à la fermeture de la dernière session.

Les menus suivants sont disponibles sous WinCC flexible :

Menu	Descriptif technique
------	----------------------

"Projet"	Contient des commandes de gestion de projets.
"Edition"	Contient des commandes servant à utiliser le presse-papiers ainsi que des fonctions de recherche.
"Affichage"	Contient des commandes permettant d'ouvrir et de fermer des éléments ainsi que des paramètres des fonctions zoom et plans. Un élément fermé peut être rouvert via le menu "Affichage".
"Insertion"	Contient des commandes pour l'insertion de nouveaux objets.
"Format"	Contient des commandes servant à disposer et à formater des objets de vue.
"Blocs d'affichage"	Contient des commandes servant à créer et éditer des blocs d'affichage.
"Outils"	Contient, entre autres, des commandes servant à changer de langue d'interface et à configurer les paramètres de base de WinCC flexible.
"Script"	Contient des commandes permettant de synchroniser et de vérifier la syntaxe de scripts.
"Fenêtre"	Contient des commandes de gestion de plusieurs vues de la zone de travail, permettant p. ex. de changer de vue.
"Aide"	Contient des commandes d'accès aux fonctions d'aide.

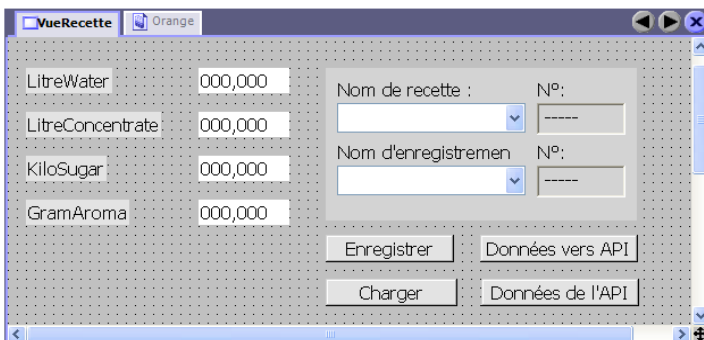
**Tableau II.1:** Description des éléments de Menu Sous WinCC flexible.

Les barres d'outils nous permettent d'accéder rapidement à des fonctions importantes dont nous avons fréquemment besoin. Nous pouvons configurer chaque barre d'outils comme suit :

- Ajouter ou supprimer des boutons
- Modifier la position

### **II.6.2.Zone de travail**

Dans la zone de travail, nous éditons les données de projet soit sous forme de tableau, ce qui est le cas des variables p. ex., soit sous forme graphique, ce qui est le cas des vues de process p. ex.



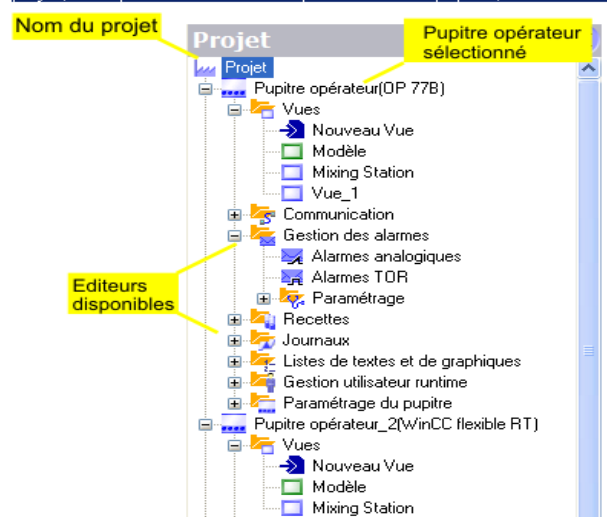
**Figure II.3 :** La forme graphique d'une Vue.

## Chapitre II

Chaque éditeur ouvert est représenté dans la zone de travail dans un onglet particulier. Dans le cas d'éditeurs graphiques, chaque élément est représenté dans un onglet distinct. Si nous avons ouvert plusieurs éditeurs simultanément, un seul onglet est actif. Pour changer d'éditeur, il suffit de cliquer avec la souris sur l'onglet voulu.

### II.6.3.Fenêtre de projet

La fenêtre du projet est le poste central de traitement du projet. Tous les éléments et tous les éditeurs disponibles d'un projet sont affichés sous forme d'arborescence dans la fenêtre du projet et peuvent être ouverts à partir de cette fenêtre. A chaque éditeur correspond une icône qui nous permet d'identifier les objets qui lui sont associés. Seuls les éléments pris en charge par le pupitre opérateur sélectionné apparaissent dans la fenêtre de projet. Dans la fenêtre de projet, nous pouvons accéder aux paramètres du pupitre, à la localisation et à la gestion de versions.



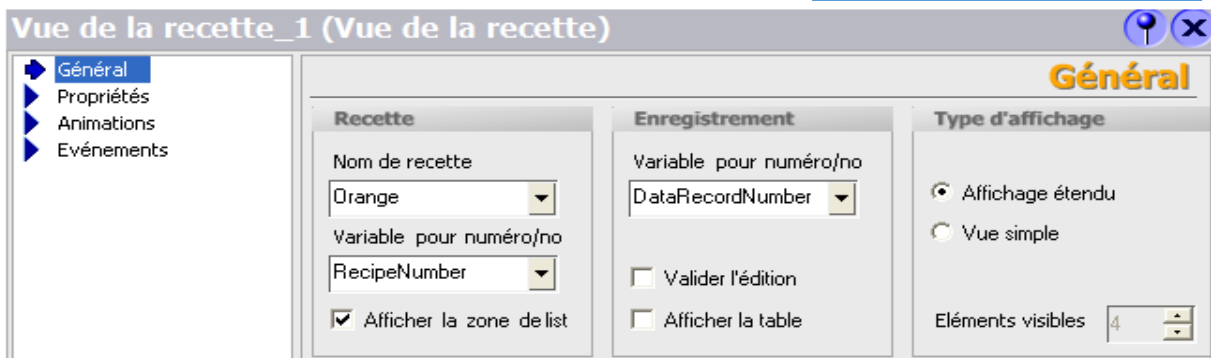
**Figure II.4 :** *fenêtre de projet.*

La fenêtre de projet sert à créer des objets et à les ouvrir pour les éditer. Nous pouvons créer des dossiers pour structurer les objets de notre projet. Nous pouvons ouvrir pour chaque objet un menu contextuel qui regroupe les principales commandes.

Les éléments des éditeurs graphiques sont affichés dans la fenêtre de projet et dans la fenêtre des objets. Les éléments de tableurs sont uniquement affichés dans la fenêtre des objets.

### II.6.4.Fenêtre des propriétés

La fenêtre des propriétés permet de modifier les propriétés d'un objet sélectionné dans la zone de travail. Le contenu de la fenêtre des propriétés dépend de l'objet sélectionné



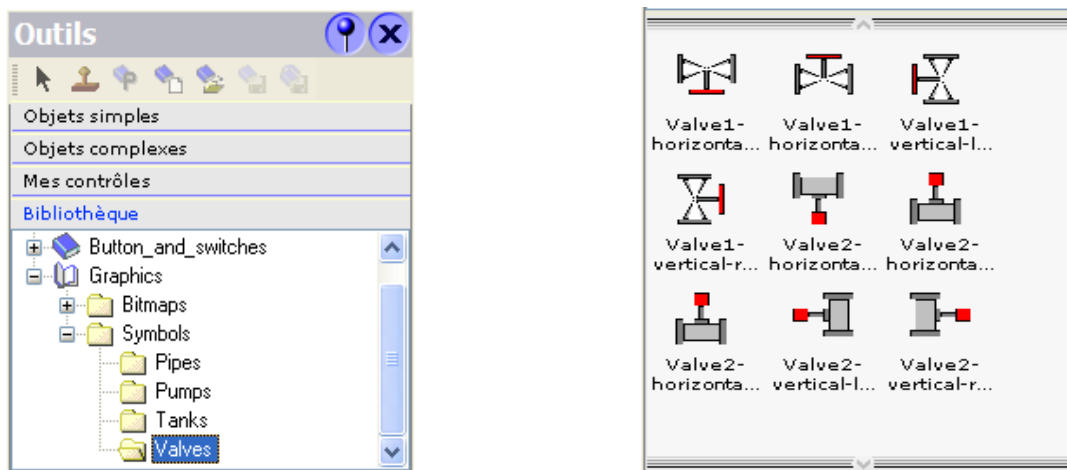
**Figure II.5 :** *fenêtre des propriétés.*

La fenêtre des propriétés affiche les propriétés de l'objet sélectionné classées par catégories. Aussitôt que nous quittons une zone de saisie, les modifications de valeurs effectuées sont actives.

Si nous entrons une valeur inadmissible, celle-ci s'affiche sur fond de couleur. L'info-bulles nous fournit des informations, p. ex. sur la plage des valeurs valides.

### **II.6.5. Bibliothèque**

La bibliothèque fait partie de la fenêtre d'outils. La bibliothèque est le lieu central d'enregistrement des objets fréquemment utilisés. Tout objet enregistré dans la bibliothèque ne doit être configuré qu'une seule fois. Nous pouvons ensuite le réemployer à volonté. Les objets de la bibliothèque permettent d'augmenter la quantité d'objets de vue disponibles et d'améliorer notre productivité lors de la configuration par la réutilisation d'objets préconfigurés.



**Figure II.6 :** *Élément de la bibliothèque.*

Sous WinCC flexible, on distingue la bibliothèque globale de la bibliothèque spécifique au projet :

- Bibliothèque globale

La bibliothèque globale n'est pas enregistrée avec le projet dans la base de données mais sous forme de fichier. Le fichier enregistré est stocké par défaut dans le répertoire d'installation de WinCC flexible. La bibliothèque globale est disponible pour tous les projets.

## Chapitre II

- Bibliothèque spécifique projet

La bibliothèque de projet qui est enregistrée avec les données de projet dans la base de données, est uniquement disponible dans le projet où elle a été créée.

### II.6.6.Fenêtre des erreurs et avertissements

La fenêtre des erreurs et avertissements affiche les événements système générés p. ex. lors du test d'un projet.



**Figure II.7 :** Fenêtre des erreurs et avertissements.

Dans la fenêtre des erreurs et avertissements, les événements système sont affichés par défaut dans leur ordre d'apparition. Les catégories désignent respectivement le module WinCC flexible qui a généré un événement système. Les alarmes système de la catégorie "Compilateur" sont p. ex. générées durant le contrôle de cohérence.

La fenêtre des erreurs et avertissements affiche tous les événements système se rapportant à la dernière action. A chaque nouvelle action, tous les événements système précédents sont écrasés.

### II.6.7.Fenêtre des objets

Lorsque nous sélectionnons des dossiers ou des éditeurs dans la fenêtre de projet, leur contenu s'affiche dans la fenêtre des objets.

La figure suivante montre comment la sélection dans la fenêtre de projet influence l'affichage dans la fenêtre des objets.

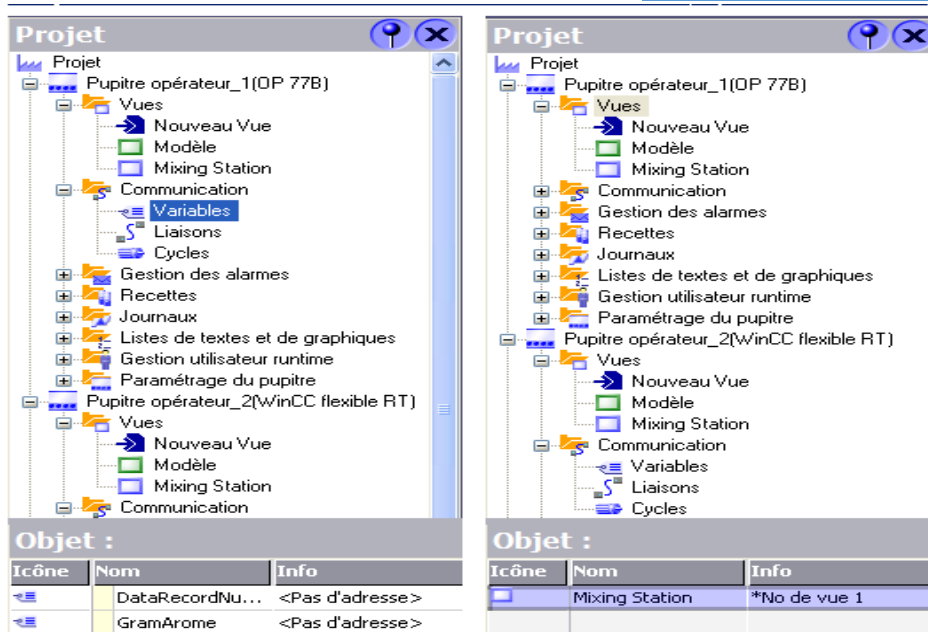


Figure II.8 : Fenêtre des objets.

Nous pouvons effectuer des actions de glisser-déplacer avec tous les objets affichés dans la fenêtre des objets. Les actions de glisser-déplacer suivantes sont p. ex. prises en charge :

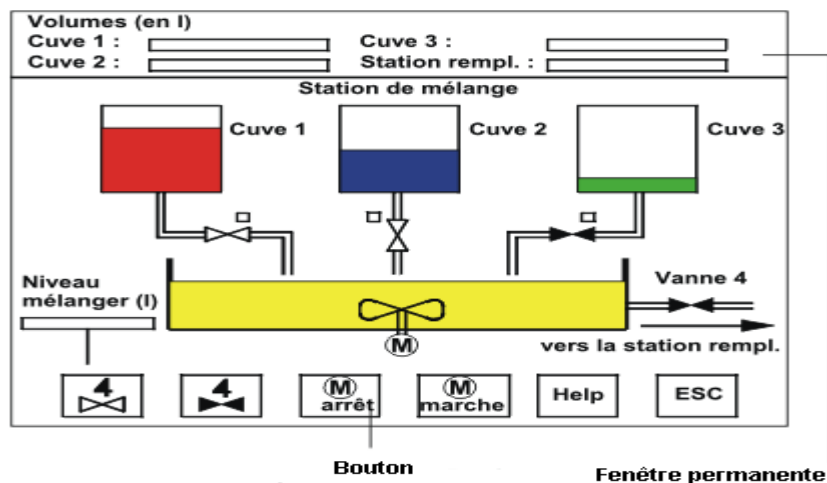
- Placer une variable dans la vue de processus qui se trouve dans la zone de travail.
- Placer une vue de processus dans la vue de processus qui se trouve dans la zone de travail :  
Création d'un bouton vers cette vue de processus.

## II.7. Créer des vues [25]

Dans WinCC flexible, nous créons des vues pour le contrôle-commande des machines et d'installations. Pour créer des vues, vous disposez d'objets prédéfinis permettant de représenter notre installation, d'afficher des procédures et de définir des valeurs de processus.

- **Exemple d'application**

Cette figure présente une vue créée avec WinCC flexible. Les opérateurs de l'installation effectuent avec cette vue le contrôle-commande d'une station de mélange servant à la fabrication de divers jus de fruits. Un mélangeur est rempli de jus de fruits issus de différents réservoirs, en respectant une certaine proportion, puis les différents jus sont mélangés. Le niveau de remplissage des réservoirs et du mélangeur est affiché. De plus, la vue comporte des éléments de commande des vannes de l'installation et du moteur du mélangeur.



Une vue peut être composée d'éléments statiques et d'éléments dynamiques.


- Les éléments statiques, p. ex. le texte et le graphique, ne changent pas au runtime. Dans l'exemple de l'installation de mélange, les inscriptions sur les réservoirs sont p. ex. statiques.
- Les éléments dynamiques varient en fonction de la procédure. Nous visualisons les valeurs de processus actuelles de la manière suivante :
  - A partir de la mémoire de l'automate programmable
  - A partir de la mémoire du pupitre opérateur, sous forme d'affichages alphanumériques, de courbes et de bargraphes.

Les champs de saisie du pupitre opérateur font également partie des objets dynamiques. Dans l'exemple de l'installation de mélange, les niveaux de remplissage des réservoirs sont des objets dynamiques.

## II.8. Générer un fichier de projet

Pour transférer un projet sur notre pupitre opérateur, nous générons un fichier de projet (\*.fwx) dans WinCC flexible.

1. Dans la barre de menus : "Projet > Compilateur > Générer" ou "Projet > Compilateur > Tout régénérer...".

2. Autre solution : Cliquez sur le bouton  dans la barre d'outils.

## II.9. Introduction à WinCC flexible Runtime

WinCC flexible Runtime est un logiciel performant et facile à utiliser pour la visualisation du processus des projets créés avec le logiciel de configuration WinCC flexible Advanced.

WinCC flexible Runtime est conçu pour la visualisation et l'utilisation de machines et de petites installations. Le logiciel Runtime se distingue par son interface utilisateur entièrement graphique, basée sur la technique des fenêtres. Il permet grâce à des temps de réaction rapides une conduite de processus sûre, ainsi qu'une collecte sûre des données. [25]

## II.10. Tester le projet

Le logiciel de configuration WinCC flexible est fourni avec un simulateur qui permet de tester le projet sans automate. Le simulateur est une application en propre. Il nous permet de contrôler le bon fonctionnement des vues, objets de vue et messages configurés.


Le simulateur simule l'automate de la manière suivante :

- Modification définie des valeurs des variables configurées, par exemple, incrémentale, décrémente, sinusoïdale, aléatoire ou par décalage de bit.

Pour simuler le projet terminé, plusieurs possibilités s'offrent :


- Simulation avec raccordement à l'automate

Nous pouvons simuler notre projet en l'exécutant directement dans runtime. Mais dans ce cas, les variables et les pointeurs de zone ne peuvent fonctionner que si notre PC de configuration est raccordé à un automate approprié.

Si notre ordinateur est raccordé à un automate, Runtime nous permet de réaliser une simulation authentique du pupitre opérateur configuré. Pour effectuer la simulation avec WinCC flexible Runtime, on sélectionne dans le menu "Projet" la commande "Générateur > Démarrer runtime". Sinon, on clique dans la barre d'outils "Générateur" sur l'icône .

- Simulation sans raccordement à l'automate

A l'aide du programme de simulation installé en même temps que WinCC flexible Runtime nous pouvons simuler le projet, y compris les variables et les pointeurs de zone sans connexion à un automate. Nous saisissons les paramètres des pointeurs de zone et des variables dans un tableau de simulation qui est lu par WinCC flexible Runtime pendant la simulation.

Pour effectuer la simulation avec le simulateur, on sélectionne dans le menu "Projet" la commande "Générateur > Démarrer runtime avec le simulateur". Sinon, cliquez dans la barre d'outils "Générateur" sur l'icône .

- Simulation en fonctionnement intégré



## Chapitre II


Si nous configurons en mode intégré dans STEP 7, nous pouvons simuler un raccordement à un automate via PLCSIM.

## II.11. Transférer un projet

Le logiciel de configuration et le logiciel WinCC flexible Runtime étant sur un même système, nous procédons comme suit:

1. On crée notre projet (par ex. sous le nom Myproject.hmi), puis on le compile.

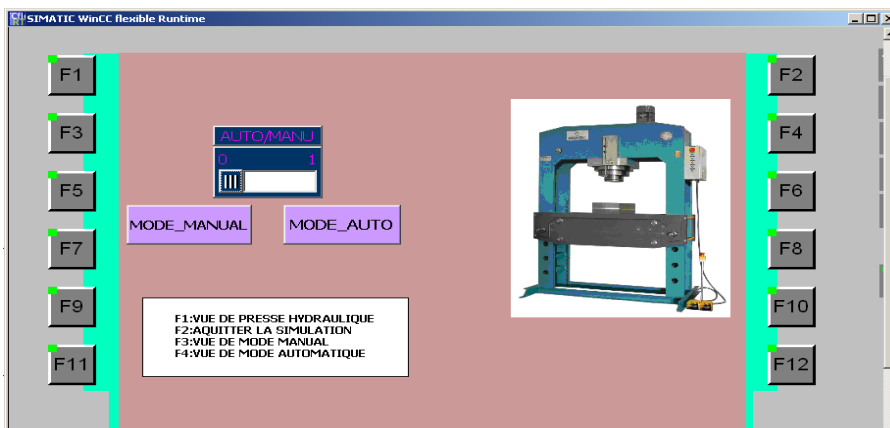
On trouvera ensuite le fichier compilé portant l'extension \*.fwx sous le même répertoire que le fichier projet, par exemple "Myproject.fwx".

2. On démarre le logiciel WinCC flexible Runtime directement à partir du logiciel de configuration en cours. en cliquant dans la barre d'outils "Générateur" sur l'icône .
3. Si une communication est établie avec l'automate, nous pouvons tester ou faire exécuter le projet avec l'automate.

## II.12. Exemple d'application sur la Presse Hydraulique

On établit l'interface de supervision de la presse hydraulique (traitée dans le premier chapitre)

Voici les différents vues sur le logiciel WinCC flexible :



**Figure II.9 :** *Vue de commutation entre le mode manuel et automatique.*

## Chapitre II

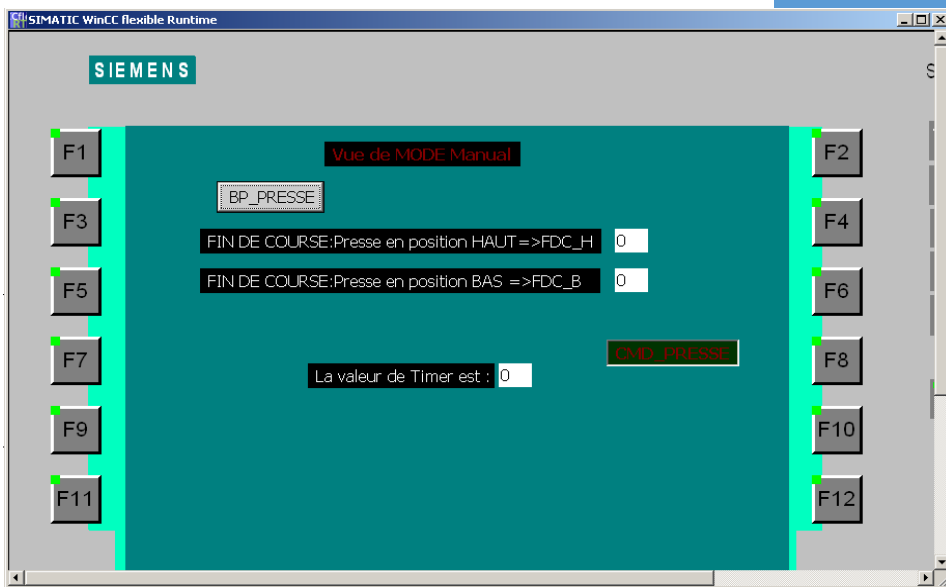


Figure II.10: Vue de mode Manuel



Figure II.11 : Vue de mode Automatique

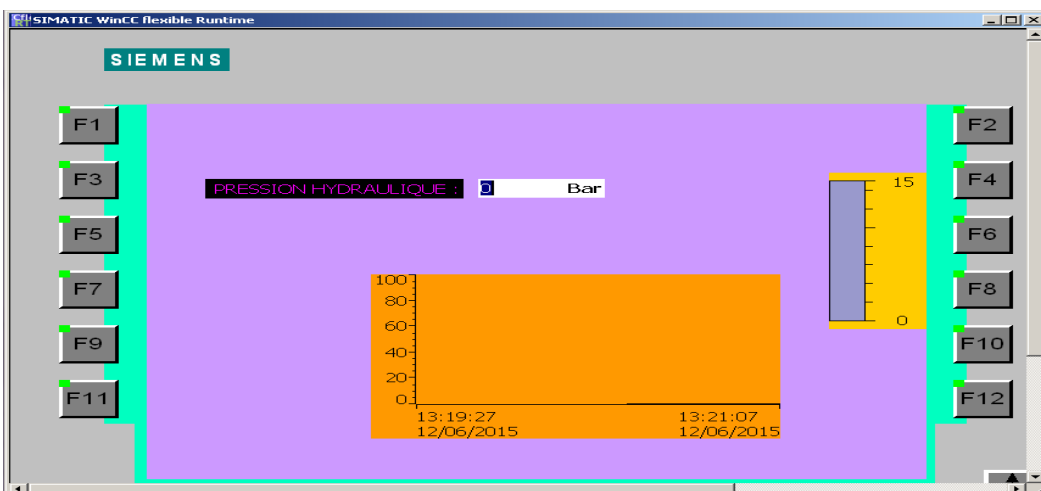


Figure II.12 : Vue analogique

**II.12.Conclusion :**

Siemens propose un grand choix en termes de pupitre de supervision et de contrôle. La description précédente de l'outil WinCC flexible aide l'utilisateur (programmeur de L'IHM ou pupitre) à programmer des interfaces IHM facilitant la communication entre l'homme et sa machine.

# **CHAPITRE III**

## Les réseaux locaux Industriels Profibus

### III .1. Introduction

Dans une entreprise, il peut arriver fréquemment que l'automate, les actionneurs et les capteurs ne soient pas situés au même endroit mais à des distances importantes les uns des autres. L'utilisation d'un réseau industriel permet donc de faire communiquer plusieurs automates, chacun reliés à une partie des capteurs/actionneurs.

La gestion d'une ligne de production est souvent présentée de la manière suivante :

- 1) Capteurs / actionneurs ;
- 2) Contrôle / commande (automates) ;
- 3) Supervision (PC) ;

Le réseau industriel, ou encore réseau de terrain, se place au niveau 2, entre les automates.

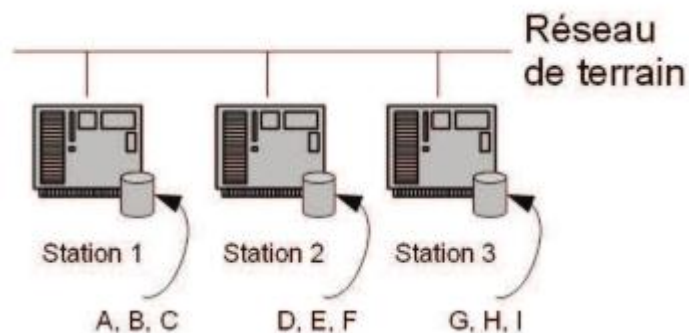
Il existe différents réseaux industriels principalement associés au type d'automate utilisé, les automates Télémécanique utilisent le réseau FipWayn, les automates Siemens utilisent par exemple le réseau Profibus.

### III.2. Echange de données

Dans un réseau industriel, les informations relatives aux entrées/sorties de la maquette sont réparties sur chacun des automates selon le câblage réalisé. Or, pour continuer de fonctionner, il est possible qu'une station ait besoin de connaître la valeur d'un capteur relié à une autre station.

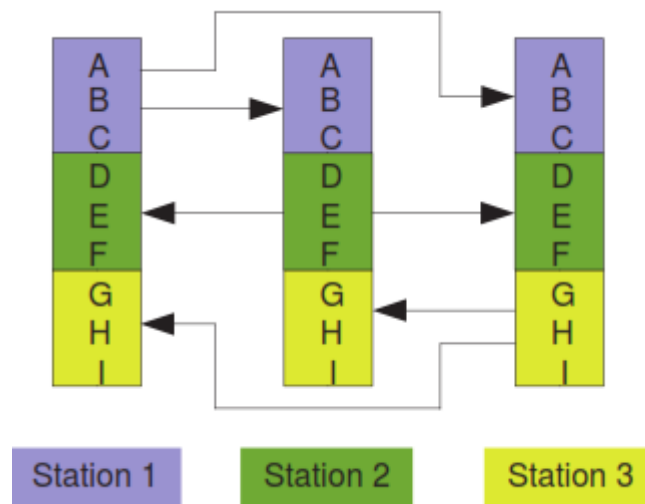
Pour ce faire, une partie des données contenues sur chacun des automates sont périodiquement envoyées sur le réseau par chacun d'entre eux et à l'attention de tous.

Ces données sont en réalité des mots mémoires appartenant à chacun des automates (exemple : mots A, B, C de la station 1).



**Figure III.1 :** *les mots mémoires appartenant à chaque automate*

L'échange de ces mots mémoires « partagés » entre les stations est représenté sur le schéma suivant :



**Figure III.2** L'échange des mots mémoires « partagés » entre les stations

### III .3.Quelques définitions :

\_ **Terrain** : indique un espace délimité géographiquement (usine, atelier, voiture...)

\_ **BUS** : au sens informatique industriel, conducteur ou ensemble de conducteurs communs a plusieurs circuits permettant l'échange de données entre eux avec :

- Liaisons communes ;
- Plusieurs circuits ;
- Reference à la topologie de la configuration. [6]

\_ **Protocoles** : ensemble de règles de communication qui définissent l'émission et la réception de données sur un réseau.

\_ **Réseau** : ensemble d'équipements reliés entre eux pour échanger des informations. Cet ensemble est constitué d'au moins un support de transmission pour l'acheminement des signaux et de protocoles de communication.

Il existe différents protocoles de réseaux industriels principalement associés au type d'automate utilisé (Télémécanique utilise le protocole WorldFip avec un réseau FipWay).

\_ **Réseau local** : Un réseau local couvre une zone géographique limitée sur un domaine privé, par opposition aux réseaux publics ou longue distance.

Parmi les réseaux locaux, on distingue souvent les réseaux locaux d'entreprise et les réseaux locaux industriels. Ils diffèrent essentiellement par les contraintes d'environnement (temps et sûreté de fonctionnement) et par certains services et protocoles mis en œuvre pour tenir compte des différences de besoins des applications qui les utilisent.

\_ **Réseau local industriel** : il est en première approximation un local utilisé dans une usine ou tout système de production pour connecter diverses machines afin d'assurer la commande, la surveillance, la supervision, la conduite, la maintenance, le suivi de produit, la gestion, en un mot, l'exploitation de l'installation de production.

### **III .4. Réseau MPI et protocole Profibus[5]**

#### **III .4. 1. Définition**

Profibus (Process Field Bus) est un réseau de terrain ouvert, permettant de répondre à un large éventail d'applications dans les domaines concernant :

- \_ Les procédés manufacturiers (conduite des procédés séquentiels, procédés discontinus par lots « batch ») ;
- \_ Les procédés continus (conduite, régulation) ;
- \_ La gestion des bâtiments (gestion technique centralisée, gestion technique du bâtiment).

En tant que réseau non propriétaire, Profibus est issu de travaux initiés en 1987 par le ministère fédéral allemand pour la Recherche technologique, comprenant un groupement de sociétés industrielles et d'instituts de recherche allemands, orchestré par Siemens AG. [5]

La norme allemande DIN 19245 a été créée en 1991, normalisée EN 50170 par le Cenelec en 1996, comme norme européenne. Depuis 1999, le réseau Profibus est reconnu dans la norme internationale CEI 61158, avec les autres réseaux ControlNet, P-Net, Fieldbus Foundation High-Speed-Ethernet, SwiftNet, WorldFip, Interbus-S. Avec beaucoup de difficultés, le groupe de travail du projet CEI 61158 a convergé enfin vers un compromis de norme sur les réseaux de terrain en y intégrant sept réseaux très divers et quelquefois rares. Les travaux d'homogénéisation de la norme CEI 61158 se poursuivent, notamment pour affiner les différents types identifiés au niveau des couches 2 et 7.

#### **III .4. 2. Architecture**

Profibus propose une architecture réseau en trois niveaux, autorisant une communication entre matériels hétérogènes et de différents constructeurs, afin de couvrir tous les niveaux d'automatisation d'un système (figure 1) : [5]

- \_ Transmission de données de type action réflexe avec un temps de réaction très court ;
- \_ Raccordement direct de capteurs et d'actionneurs sur le bus ;
- \_ Fonctionnement en sécurité intrinsèque ;
- \_ Dialogue entre automatismes et périphérie décentralisée ;
- \_ Echange de données complexes et volumineuses pour la gestion de cellules.

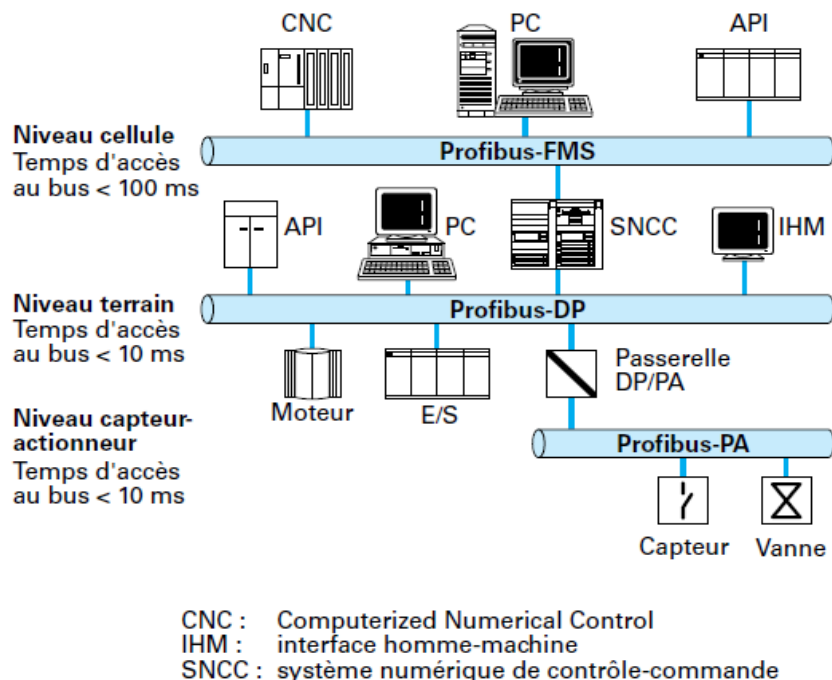
#### **III .4. 3. Variantes**

Profibus se décline en trois variantes de protocoles, répondant chacune à des finalités métiers et applicatives spécifiques :

\_ **Profibus-DP (Decentralized Peripheral)** : destiné aux applications de type maître-esclave en monomaître pour la gestion des équipements d'entrées-sorties déportées avec des temps d'accès Extrêmement courts. Le fonctionnement multimaître est possible.

\_ **Profibus-FMS (Fieldbus Message Specification)** : destiné aux applications nécessitant l'échange entre maîtres pour la synchronisation d'activités de contrôle-commande, basé sur la messagerie MMS (Manufacturing Message Specification).

\_ **Profibus-PA (Process Automation)** : destiné aux applications de contrôle de process nécessitant la communication avec des équipements de terrain (capteurs, actionneurs) permettant une téléalimentation des équipements et un fonctionnement avec sécurité intrinsèque en ambiance explosive.



**Figure III.3** : Classification des réseaux Profibus [5]

### III .4.4. Fonctionnement général

Dans un réseau Profibus, la communication est gérée selon un système d'anneau à jeton : des automates maîtres se passent un jeton alors que des esclaves sont interrogés par leur maître respectif.

Ce procédé est un procédé d'accès au bus hybride (jeton + réseau maître/esclave). La méthode du jeton garantit que toutes stations actives peuvent accéder au bus au moins une fois pendant un



laps de temps défini. D'un autre côté, la méthode maître/esclave permet au maître possédant le jeton de scruter ses esclaves (stations passives). Il peut écrire ou lire sur ses esclaves.

### III .4.5. Principe de l'anneau à jeton [5]

Les stations actives se passent un jeton (télégramme) de manière périodique suivant un ordre préétabli (par ordre d'adresse ascendante). La station possédant le jeton reçoit un droit de parole, d'une durée préétablie, pour communiquer vers les autres stations actives et initier des transactions vers ses esclaves (lecture d'entrées/sorties sur des terminaux déportés). Lorsque la station a fini de communiquer ou que le temps de parole est expiré, elle passe le jeton à la station d'adresse immédiatement supérieure.

Un protocole d'interrogation des successeurs et des prédécesseurs permet de créer automatiquement cet anneau. Ce type de gestion d'accès fournit des bornes au temps de cycle réseau.

Le constructeur Siemens propose une couche supplémentaire reposant sur les protocoles Profibus ou MPI. Cette couche offre principalement 2 types de services :

- \_ La communication par données globales basée sur un principe de diffusion permettant le dialogue entre 15 stations maximum ;
- \_ La communication en mode point à point entre 2 stations (Profibus DP). Elle permet d'initier des transactions entre 2 stations.

### III .4.6. Utilisation des données globales

Les données globales (GD) sont des mémentos, des entrées, des sorties, des temporisations, des compteurs et des zones de blocs de données. La communication par données globales repose sur une table commune à chaque station. Cette table permet de définir l'ensemble des communications.

Chaque transaction par données globales est identifiée de la manière suivante :  $GD_{v, w, z}$  avec :

- v** n° du cercle de connexion. Il regroupe plusieurs stations du réseau ;
- w** n° du paquet, dans le cercle de connexion ;
- z** n° des données dans le paquet. Un paquet peut regrouper plusieurs données ;

#### III .4.6.A. Exemple 1

L'exemple qui suit présente une table de données globales qui paramètre la communication entre 3 stations. Les mnémoniques M1.0, M2.0, etc. correspondent aux bits M1.0, M2.0, etc. de la mémoire des stations.

Identificateur	Station 1	Station 2	Station 3
GD1.1.1	>M1.0		M6.0
GD2.1.1	>M10.2	M2.1	
GD2.1.2	>M10.4	M2.5	
GD2.2.1	M1.2	>M2.6	
GD2.2.2	M1.3	>M2.7	
GD3.1.1	>M10.5	M10.5	M6.1
GD3.2.1	M10.2	M10.2	>M6.3
GD3.2.2	M10.3	M7.2	>M7.2

La donnée globale GD2.1.2 désignent le 2<sup>ème</sup> cercle de communication (il regroupe les stations d'adresse 1 et 2). Il s'agit du 1<sup>er</sup> paquet de ce cercle (communication de la station 1 vers la station 2) et de la 2<sup>ème</sup> zone de données dans le paquet. Le symbole > précise que la station 1 est émettrice de ce paquet et que la zone de données est constituée du bit M10.4 de cette station. La station 2 est réceptrice et stocke le bit reçu dans M2.5.

Lorsque la station 1 a le jeton, les bits M1.0, M10.2, M10.4 et M10.5 de la station 1 sont recopiés respectivement dans les bits M6.0 de la station 3, M2.1 de la station 2, M2.5 de la station 2 et M10.5 de la station 2.

### III .4.6.B Exemple 2

Exemple de table de données globales, vue de l'environnement **Simatic manager**.

	Identificateur GD	Station Magasin (4)\ CPU314 (4)	Station Maître (5)\ CPU314 (5)
1	GD 1.1.1	>MB80.10	MB20.10
2	GD 1.2.1	DB10.DBB0.20	>DB10.DBB0.20
3	GD		
4	GD		

Dans cet exemple, la station nommée Station Magasin\CPU314 envoie une zone de 10

octets commençant à MB80 à la station nommée Station Maître\CPU314 qui les dépose dans les octets MB20 à MB29.

La Station Maître est également définie comme émettrice et envoie 20 octets du DB10 (en commençant à l'adresse 0) à la CPU 314 de la Station Magasin qui les dépose dans les mêmes opérandes.

### **III .5.Conclusion**

Les réseaux de terrain sont devenus quasiment "incontournables" dans la conception d'une architecture de contrôle-commande d'un système de production automatisé (pour des installations de taille conséquente) et leur évolution en terme de services et performances peut remettre en question l'organisation du flux d'informations au sein de cette architecture.

Les réseaux locaux industriels sont variés du point de vue des services qu'ils sont capables de supporter. Le choix d'un RLI doit nécessairement s'appuyer sur les caractéristiques essentielles des échanges qu'il doit assurer, donc sur son profil, plutôt que sur l'utilisation usuelle des différents réseaux. Parmi les principales caractéristiques rappelons : la taille des données ou messages, la sécurité requise (acquiescement, contrôle de flux, etc.), la périodicité (indirectement débit), la règle d'émission (spontanée ou sur requête), la capacité à répondre aux caractéristiques de la coopération entre applications réparties et les contraintes temporelles (délai d'émission, de réception, temps de latence, priorités, etc.).

## **CHAPITRE IV**

Réalisation d'une maquette de station de  
pompage à base d'automates SIEMENS

## IV.1. Introduction :

Le Prototype de La station est conçu dans le but de réaliser plusieurs applications. Tout au long de ce chapitre, nous allons décrire notre prototype tout d'abord d'un point de vue instrumentation, à savoir les différents capteurs ainsi que les actionneurs utilisés. Ensuite, nous décrirons les cahiers des charges des manipulations effectuées sur le prototype de la station de pompage. Nous finirons par décrire les programmes sur STEP7 ainsi que la supervision de ces derniers sur WinCC flexible.

## IV.2. Structure de la Tâche d'automatisation

Dans ce qui suit, un aperçu des différentes étapes de la réalisation du projet et des objets que nous avons créé dans le cadre du projet "Maquette de Station de Pompage" au sein du logiciel STAP 7.

Le projet en gros comprend les étapes de configuration suivantes :

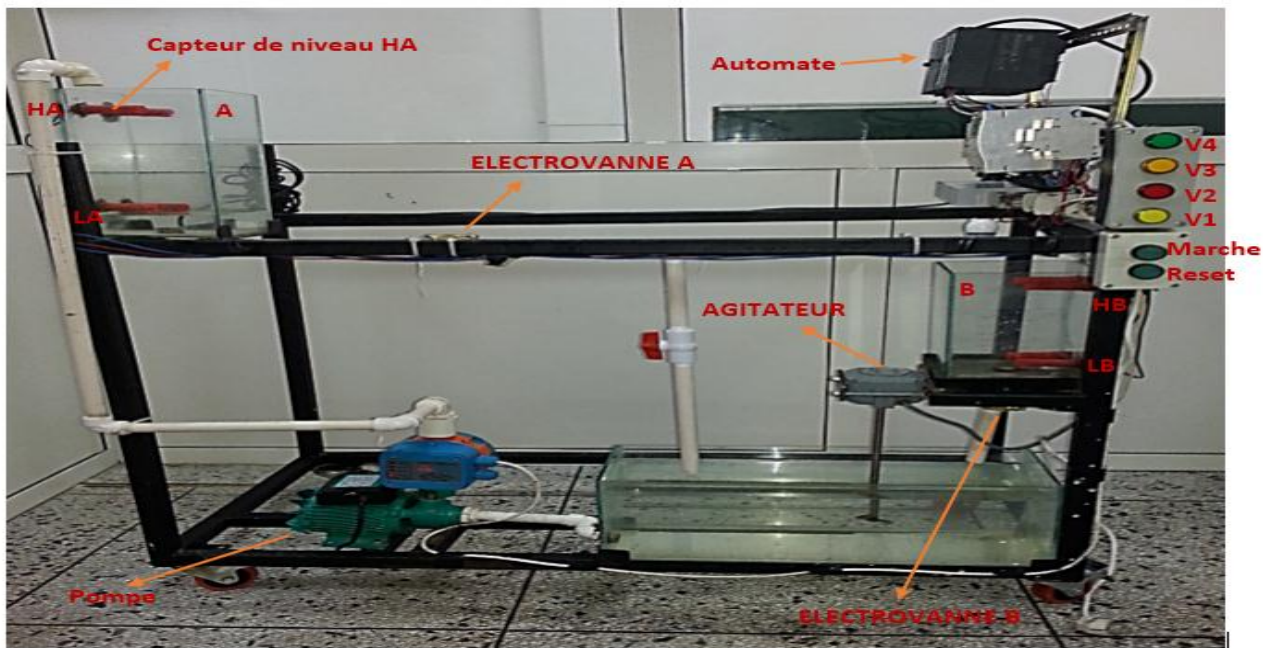
- Détermination des éléments de l'application pour finaliser le cahier des charges ;
- Détermination des éléments auxiliaires de l'installation tels que les détecteurs de niveau, les électrovannes ainsi que la tuyauterie ;
- Création du projet "Maquette de station de pompage" ;
- Insertion et configuration des matériels ;
- Programmer API ;
- Paramétrage de la communication.
- Visualisation du processus ;
- Tester en ligne les manipulations effectuées sur la maquette.

## IV.3. Description du Prototype de la station de pompage

Notre prototype a été élaboré suivant les phases d'un projet, ces phases sont le design, le développement et la construction ;

Les éléments et composants utilisés pour la construction de notre prototype, ont été choisis en fonction des paramètres, qualité, disponibilité et coût.

La figure suivante donne une vue générale de la maquette, qui permet de réaliser de différentes manipulations de même principe que les stations de pompage hydraulique.



**Figure IV.1 :** *Prototype de station de pompage.*

Le principe de fonctionnement sera décrit en détail dans le cahier des charges établi.

L'ensemble des éléments et les caractéristiques du prototype réalisé sont présentés comme suit :

- Le prototype repose sur un support métallique 1.50 m sur 1m et d'une hauteur de 1.50m qui comporte toutes les organes de la station.
- Les trois réservoirs sont en verre transparent dimensionnés comme suit :  
Deux réservoirs (A et B) de 0.15m sur 0.15m sur une hauteur de 0.30m/0.25m.  
Un réservoir (C) de 0.50m sur 0.20m sur une hauteur de 0.20m
- Une pompe centrifuge QB50, couplé à un moteur asynchrone monophasé alimenté à 220 V  
La hauteur manométrique maximale de cette pompe est de 20 m, protégé par un cerveau pour éviter la marche à sec de la pompe.
- Un mélangeur(Agitateur) alimenté en 220 v pour agiter le liquide.
- Un démarreur progressif ATS01N03FT qui améliore les performances de démarrage du mélangeur en permettant un démarrage progressif et contrôlé.
- Les éléments de la tuyauterie sont en PEHD (Polyéthylène Haute Densité) Tigre de diamètre DN40,
- Les deux robinets d'arrêt à opercule DN40, utilisés pour permettre la flexibilité du prototype afin qu'on peut l'utiliser dans deux manipulations différentes (ou plus).
- Quatre relais de niveau JYB-714B alimenté en 220V et muni chacune de 2 bougies de sonde pour détecter les niveaux haut et les niveaux bas des réservoirs A et B.

## Chapitre IV

- Une alimentation stabilisée (Harvest electric HAS-120-24) alimenté en monophasé 220 V et qui délivre du 24V.
- Deux électrovannes DN40 TOR alimentées en 24V DC.
- Une station PC pour la supervision.
- Deux relais électromagnétiques 8 broches servant comme interface de sortie entre l'automate et les actionneurs (pompe et l'agitateur).
- Un disjoncteur pour la sécurité de toute l'installation.
- Deux répartiteurs de tension pour le 24 V et 220 V

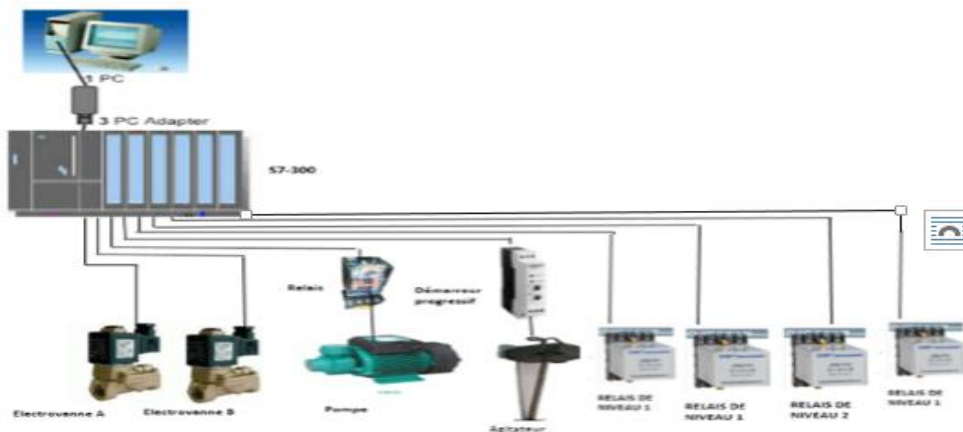


Figure IV.2: Synoptique du prototype.

- **Description du Schéma électrique de la station**

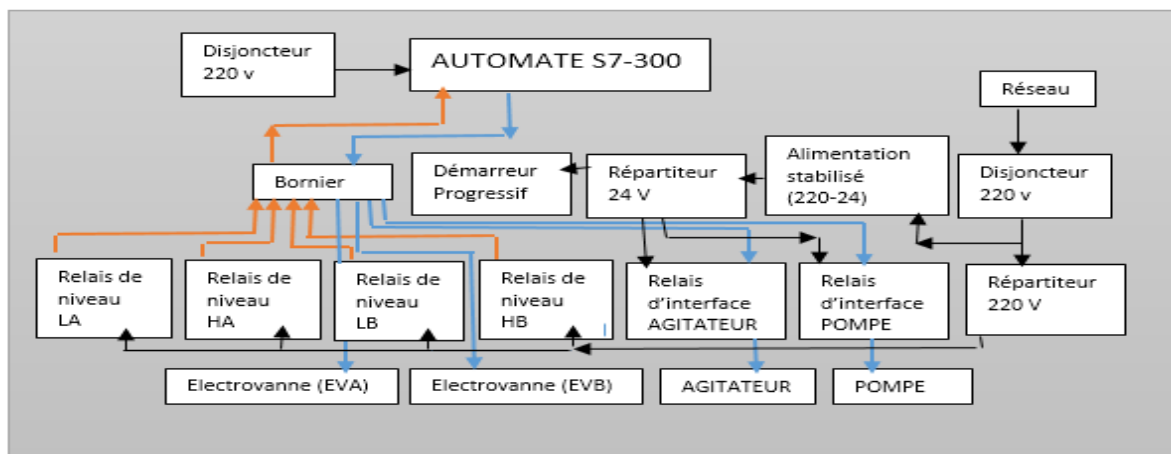


Figure IV.3 : Description du Schéma Electrique de la station

### IV.4. Eléments de commande et de supervision du Prototype de la station de pompage:

Pour cette application nous avons utilisé :

## Chapitre IV

- Un automate Siemens de la gamme SIMATIC S7-300 doté des caractéristiques suivant :
  - CPU 314 IFM, doté d'une mémoire de travail 48Ko,
  - Un module d'entrées logiques DI16/DO16;
  - Un module d'entrées/sorties analogiques AI4/AO1
  - Un port MPI.
  - Une alimentation S307 2A alimenté en monophasé 220 V et qui délivre du 24V.
- PC, système d'exploitation Windows.
- Logiciel STEP7 V 5.5
- Interface MPI pour PC (adaptateur PC)

### IV.5.Développement du projet pour gestion du prototype de la station de pompage

L'étape suivante consiste à créer et développer le programme de contrôle et de commande qui sera implémenté dans l'automate afin de gérer notre installation.

Le graphique ci-après montre ces étapes de configuration de projet et les objets à créer :

Etape	Contenu de la tache	Réalisation
1	Création du projet « Maquette de station de pompage »	<ul style="list-style-type: none"> <li>• Démarrer le SIMATIC Manager</li> <li>• Créer un nouveau projet.</li> </ul>
2	Insertion et configuration d'un matériel	<ul style="list-style-type: none"> <li>• Insérer la CPU</li> <li>• Insérer l'alimentation électrique et les module de signaux</li> <li>• adressages</li> </ul>
3	Programmer API	<ul style="list-style-type: none"> <li>• Créer des tables des mnémoniques.</li> <li>• Créer un bloc global de données.</li> <li>• Créer les différentes fonctions.</li> <li>• Appeler les blocs de programme dans le bloc principal [OB1].</li> </ul>
4	Visualisation du processus	<ul style="list-style-type: none"> <li>• Configurer le HMI Advanced.</li> <li>• Créer les vues.</li> </ul>
5	Tester en ligne	<ul style="list-style-type: none"> <li>• Tester le programme sur Plcsim</li> <li>• Tester la visualisation du processus sur WinCC flexible.</li> </ul>

**Tableau IV.1** : les étapes du développement du projet



## Chapitre IV

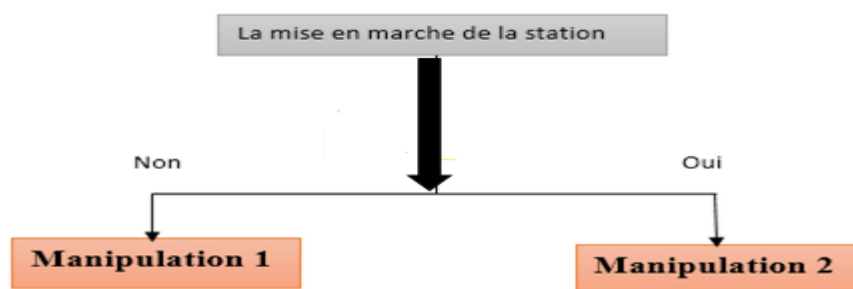
Le programme développé traduit le cahier des charges établi pour un fonctionnement optimal de notre prototype.

### IV.6. Définition des Cahiers de charge :

Le fonctionnement de l'installation est géré par l'automate, cependant l'opérateur peut intervenir pour changer sa configuration et pour se faire, l'opérateur dispose soit des boutons placés sur le pupitre soit sur le logiciel, il y aura une interface pour éventuellement changer et observer l'état de la station.

L'Automate Siemens S7-300 aura pour rôle essentiel de gérer les actionneurs tel que la pompe, les vannes motorisées et l'agitateur. Il assurera la sécurité de l'installation afin d'éviter toute fausse manipulation de la part de l'opérateur, mais aussi il permettra la communication entre les équipements et le superviseur.

Le diagramme suivant montre la manière de commutation entre les 2 manipulations :



**Figure IV.4 :** *diagramme des deux manipulations.*

#### Manipulation 1 :

Lorsque les deux réservoirs A et B sont remplis (niveau max atteints :  $H_A=H_B=1$ ) et en appuyant sur le bouton marche l'électrovanne A s'ouvre pour remplir le réservoir C et le voyant V1 s'allume, jusqu'à ce que le niveau min au niveau du réservoir A soit atteint ( $L_A=1$ ), à ce moment l'électrovanne B s'ouvre pour remplir à son tour le réservoir C et le voyant V2 s'allume, jusqu'à ce que le niveau min du réservoir B soit atteint ( $L_B=1$ ) et là, on lance une temporisation de 5 secondes avant d'actionner l'agitateur et allumer le voyant V4 pendant 5 secondes.

#### Manipulation 2 :

Le réservoir C n'étant pas vide, lorsque le réservoir A n'est pas rempli ( $\overline{H_A}=1$ ) et en appuyant sur le bouton marche, la pompe démarre et le voyant V3 s'allume ( $P=1$ ) jusqu'au remplissage du réservoir A ( $H_A=1$ ), puis on ouvre l'électrovanne EVA et on allume le voyant V1 jusqu'à ce que le réservoir B soit rempli ou le réservoir A soit vide, on lance alors une temporisation de 3 secondes et on allume le voyant V4. Après les 3 secondes et si on appuie sur le bouton marche, l'électrovanne

## Chapitre IV

EVB s'ouvre et le voyant V2 s'allume jusqu'à ce que le réservoir B soit vide (LB=1), on ce moment, on revient à l'étape initiale

La figure suivante montre les grafquets des deux manipulations :

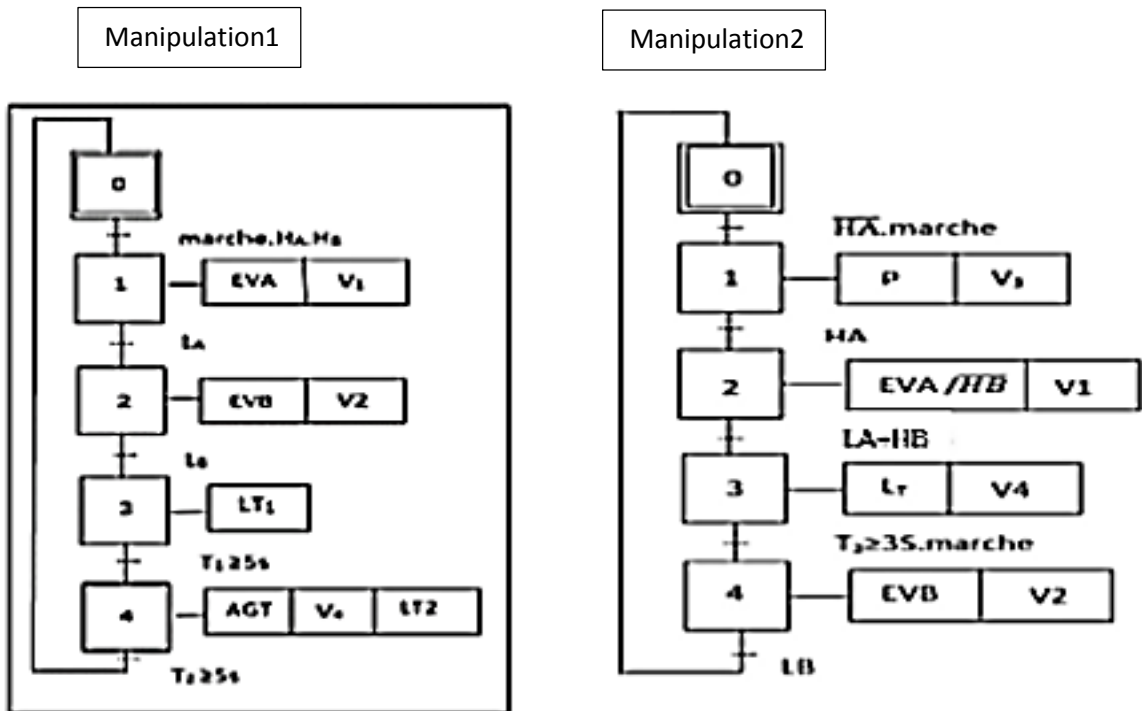


Figure IV.5: Grafquets des deux manipulation.

### Manipulation supplémentaire

On lance une manipulation supplémentaire qui sert à remplir les 2 réservoirs A et B comme une préparation pour commencer la manipulation 1

Le schéma qui suit résume ce qui était dit précédemment :

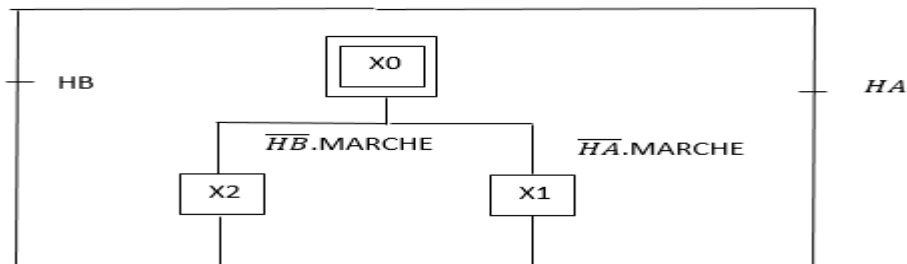


Figure IV.6: Grafquets de la manipulation supplémentaire.

## IV.7. Développement du projet S7 pour la maquette :

### IV.7.1. Insertion des stations dans SIMATIC MANAGER

## Chapitre IV

Dans ce qui suit, nous créons un nouveau projet. On insère une station SIMATIC300 en choisissant la CPU 314 IFM avec les caractéristiques citées précédemment, ensuite on insère une station SIMATIC PC pour la supervision. Telles qu'il est montré dans la figure suivante :

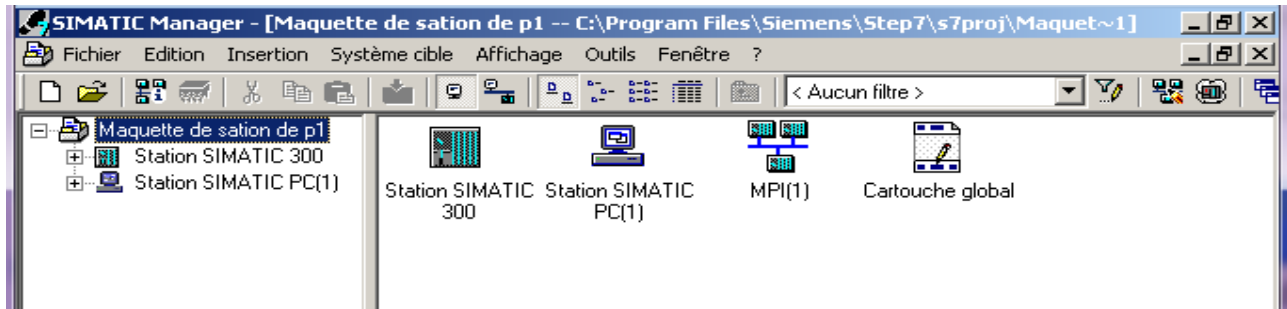


Figure IV.7 : Création du projet « Maquette de station de pompage »

#### IV.7.2. Création du programme de gestion dans STEP7 :

Avant d'éditer le programme on doit d'abord définir la table des mnémoniques comme le montre la figure suivante :

Etat	Mnémonique	Opérande	Type de données	Commentaire
1	AGT	A 124.3	BOOL	Agitateur
2	Commutateur	M 124.7	BOOL	Activer manip2(manip1 par défaut)
3	COMPLETE R...	OB 100	OB 100	Complete Restart
4	Cycle Execution	OB 1	OB 1	
5	EVA	A 124.0	BOOL	Electrovanne de Cuve A
6	EVB	A 124.2	BOOL	Electrovanne de Cuve B
7	HA	E 124.6	BOOL	
8	HB	E 125.2	BOOL	
9	LA	E 124.4	BOOL	
10	LB	E 125.0	BOOL	
11	MARCHE	E 124.2	BOOL	
12	marche_logiciel	M 124.5	BOOL	
13	P	A 124.5	BOOL	pompe
14	RESET	E 124.0	BOOL	
15	reset_logiciel	M 124.6	BOOL	
16	V1	A 125.0	BOOL	Voyant de cuve A
17	V2	A 125.2	BOOL	Voyant de cuve B
18	V3	A 125.4	BOOL	Voyant de la pompe
19	V4	A 125.6	BOOL	Voyant de l'agitateur
20	VAL1	MW 0	WORD	
21	VAL2	MW 2	WORD	
22	VAL3	MW 4	WORD	
23	VAL4	MW 6	WORD	
24	X0	M 124.0	BOOL	
25	X1	M 124.1	BOOL	
26	X2	M 124.2	BOOL	
27	X3	M 124.3	BOOL	
28	X4	M 124.4	BOOL	
29				

Figure IV.8: La table des mnémoniques.

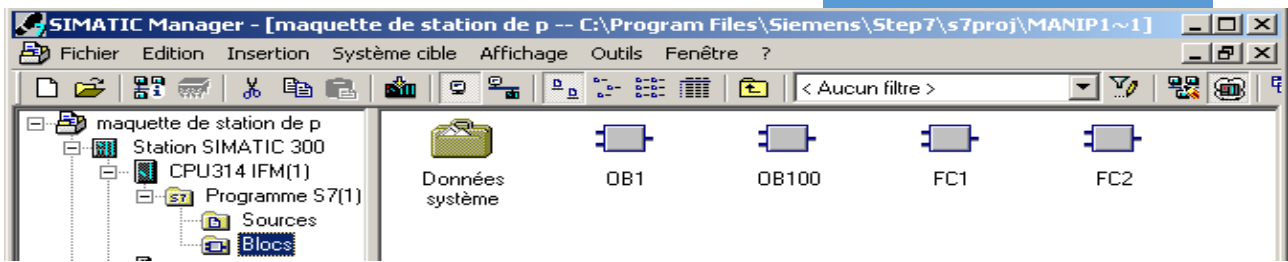
Le programme a été reparté en deux sous programmes :

- FC1 : contenant le programme de la manipulation 1.
- FC2 : contenant le programme de la manipulation 2.

Le bloc d'initialisation OB100 :

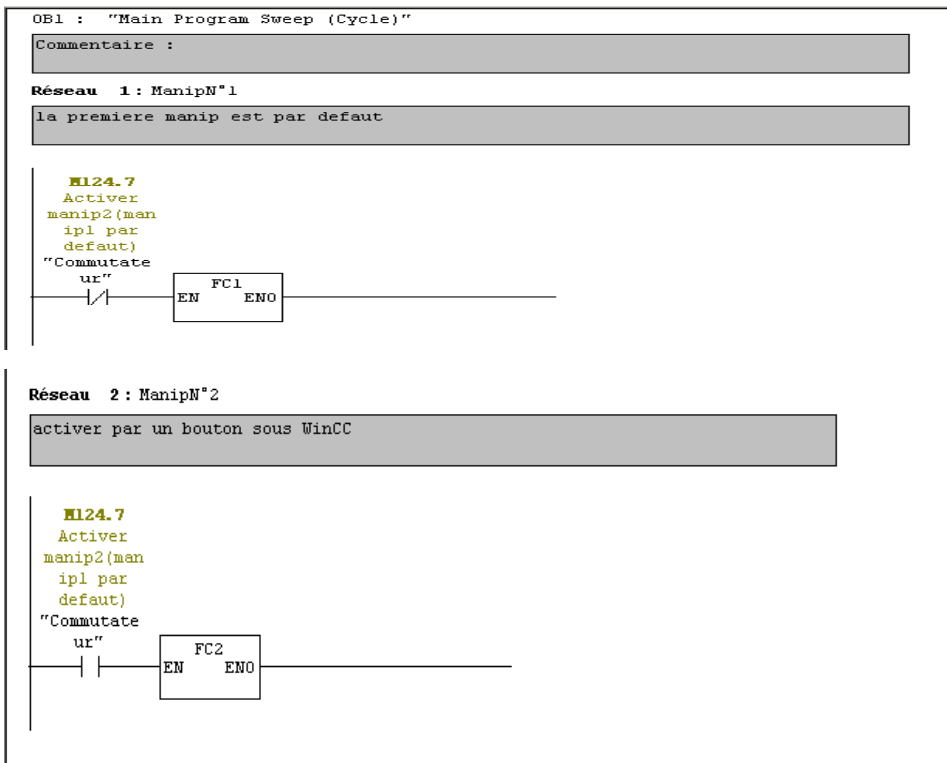
Le signal d'initialisation doit être mis à 1 lors de la mise en marche de l'automate (lorsque l'automate passe du mode STOP au mode RUN). Cela est obtenu en programmant le bloc OB100 qui s'exécute lors du changement de mode.

## Chapitre IV

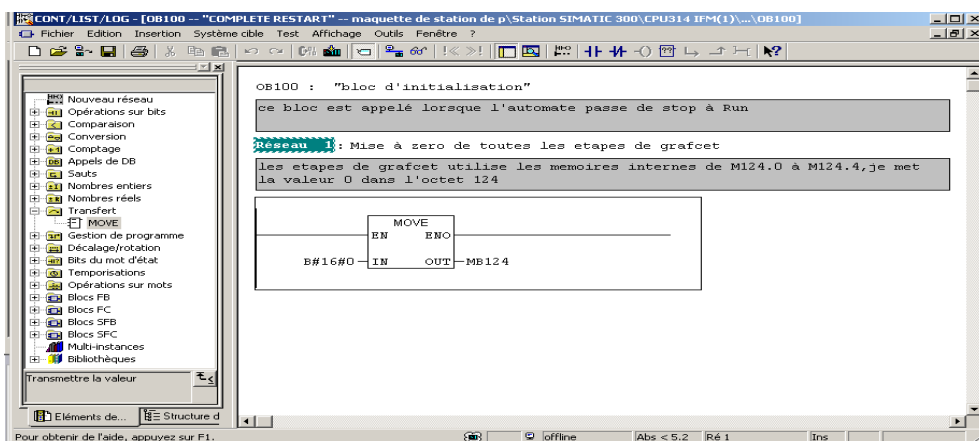


**Figure IV.9 :** Le projet S7 « maquette de station de pompage ».

Et on fait l'appel des deux fonctions FC1 et FC2 dans le bloc d'organisation OB1 comme suit :



- Le programme de bloc d'initialisation OB100 :



### IV.7.3. L'interface homme machine HMI configurée par le progiciel « Win CC flexible »

Nous allons maintenant créer le HMI « SIMATIC PC station » et réaliser la configuration des vues. Le « SIMATIC PC » est intégré dans le Projet S7.

Les vues IHM nous permettent de visualiser les différentes parties du processus :

## Chapitre IV

– Sélectionnez le "WinCC RT advanced".

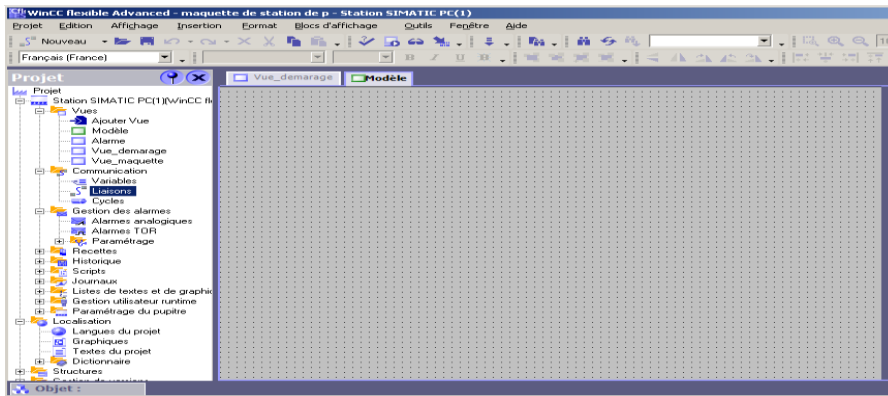


Figure IV.10 : Vue de WinCC flexible.

Pour la gestion de cette station on crée les vues suivantes :

- Une vue nommée «caractéristique» pour la gestion des composants de la station hydraulique, et la visualisation.
- Une Vue nommée «Démarrage » qui est une vue initiale pour présenter le projet (Accueil).
- Une Vue nommée « maquette» pour la supervision de toute l'installation .
- Une Vue nommée « Alarmes » qui contiennent les alarmes configurées pour signaler les éventuels problèmes au sein de la station.

### IV.8. Configuration de la connexion entre API et WinCC Advanced RT :

Pour établir la connexion entre l'API et WinCC, on utilise une connexion de type MPI à l'aide de L'adaptateur USP/MPI

Ouvrir l'éditeur de configuration de connexions en appuyant sur « connexion ».

Dans l'éditeur, on ajoute une connexion de type MPI en prenant la configuration suivante :

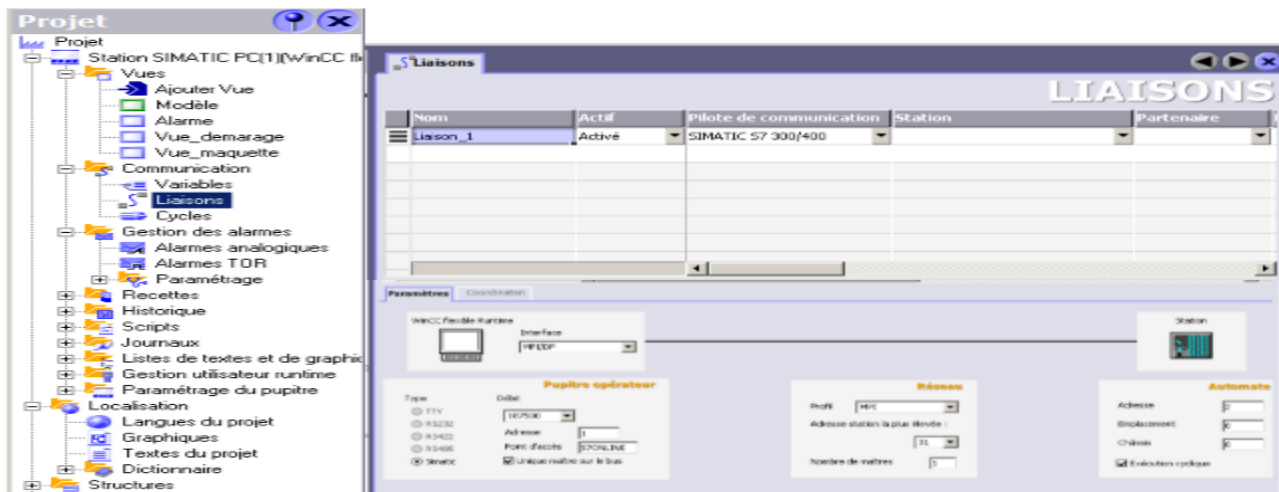


Figure IV.11 : Configuration de La liaison MPI.

### IV.9. Simulation du programme STEP 7 dans PLCSIM

Pour tester le programme, on doit démarrer la simulation dans PLCSIM. Cette simulation nous permet de tester le bon fonctionnement du programme avant de démarrer la production. Nous

## Chapitre IV

chargeons d'abord la configuration et le programme utilisateur dans le module simulé et Nous insérons des sous-fenêtres pour visualiser et forcer les sorties et mémentos.

Insérez une sous-fenêtre "Memento" pour forcer et afficher les variables API avec bits de memento dans la plage d'adresse voulu.

On insère une sous-fenêtre "Sortie" pour délivrer les sorties dans la plage d'adresse des module

On mette la simulation de la CPU en mode RUN.

Nous avons configuré la simulation de la CPU dans PLCSIM. Une fois en RUN, les deux LED "RUN" (mode de fonctionnement) et "DC" (alimentation) sont allumées en vert.

On laisse la fenêtre PLCSIM ouverte. Nous allons maintenant nous servir de la simulation pour tester l'exemple de projet "maquette de station de pompage" avec les fonctions en ligne.

### IV.10. Visualisation du processus dans WinCC Advanced Runtime

La fonctionnalité de la visualisation peut être testée avec le logiciel de simulation "WinCC Runtime Advanced". Cette simulation nous permet de tester le bon fonctionnement de la visualisation avant de démarrer du processus. On procède comme suit :

1. On clique droit avec la souris sur le HMI RT dans la navigation du projet et on démarre la simulation du Runtime dans le menu contextuel.

Les éléments configurés du HMI RT sont automatiquement compilés avant le démarrage du Runtime. L'état de compilation s'affiche dans la fenêtre des erreurs et avertissements.

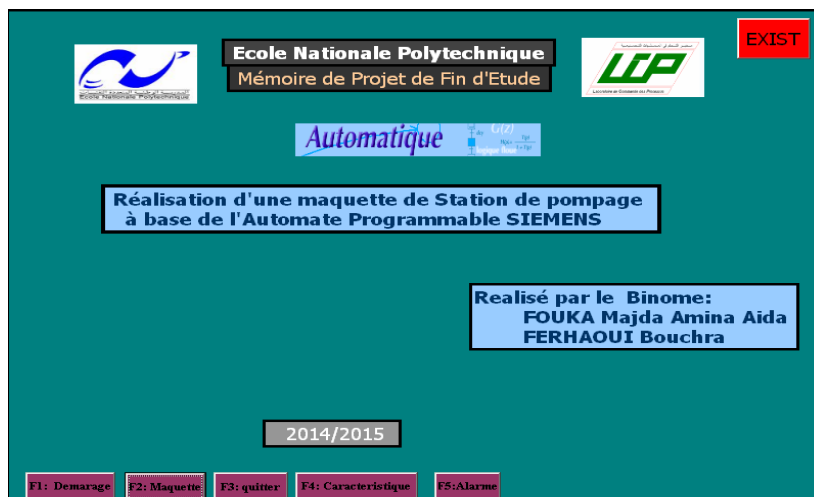


Figure IV.12: La vue « démarrage ».

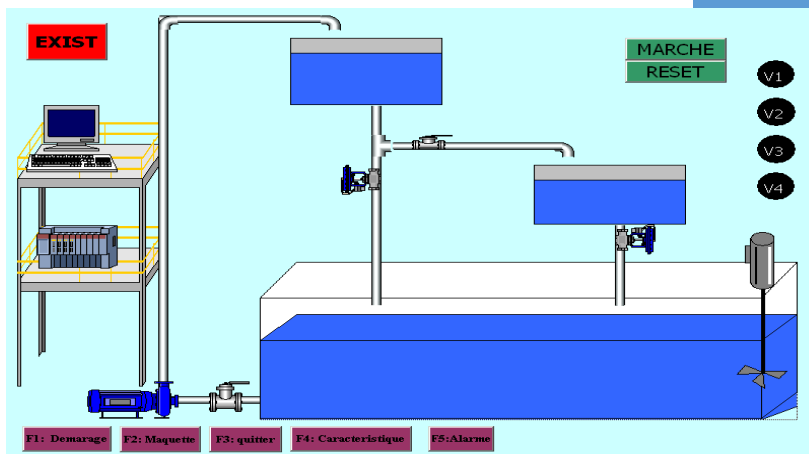


Figure IV.13 : La vue « maquette »

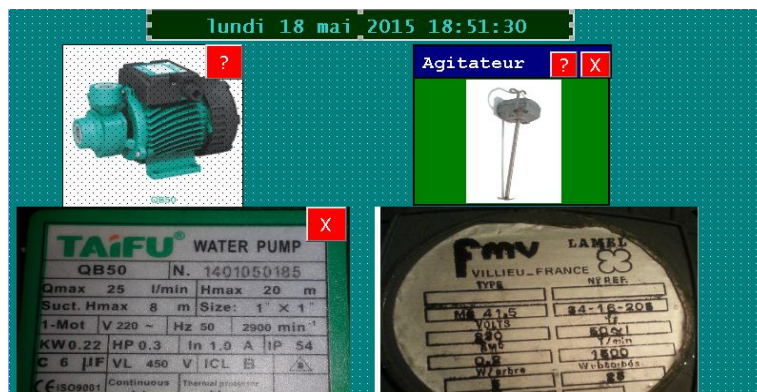


Figure IV.14 : La vue « caractéristique »

### IV.11 Chargement dans le système cible à partir de la PG/PC

Le chargement du programme dans le système cible se fait sous certaines conditions :

- Une liaison est établie entre la PG et la CPU du système cible (via l'interface MPI ou PROFIBUS).
- L'accès au système cible est possible.
- La compilation du programme à charger doit se faire sans erreur.
- La CPU doit se trouver dans un état de fonctionnement autorisant le chargement (STOP ou RUN-P).

Si la syntaxe est correcte, le bloc est ensuite compilé en code machine, enregistré et chargé par la Commande **Système cible > Charger**, ou en sélectionnant tous les blocs et charger par la fonction de chargement dans la barre d'outils du SIMATIC Manager.

### IV.12 Conclusion

Nous avons pu réaliser une maquette de station de pompage à base de l'API SIEMENS S7-300. On a programmé le fonctionnement de la maquette à l'aide de logiciel STEP7 ainsi que la supervision de processus à travers l'interface graphique réalisé à l'aide de WinCC flexible.

# Conclusion Générale et Perspectives



## **Conclusion générale et Perspectives**

Un élève ingénieur ne doit pas se satisfaire d'acquérir des connaissances théoriques, il est plutôt primordiale qu'il complète sa formation par une bonne expérience dans le domaine de la pratique, c'est dans ce but qu'on a choisi un projet de fin d'étude qui porte sur la réalisation, la gestion et la supervision d'un prototype de station de pompage.

La commande du prototype est assurée par un automate Siemens de la série S7-300, qui forme une unité de traitement et de commande d'une grande flexibilité, et qu'on a pu exploiter d'une manière très optimale grâce au logiciel de programmation STEP7. Pour la supervision de la station, on a utilisé une station PC comme une interface homme machine programmée par le logiciel WinCC Flexible et connectée à l'automate via un réseau MPI.

Pendant cette réalisation, on a appris de nouvelles compétences telles que : le travail en groupe, la planification, la gestion du budget, ainsi que des principes de bases dans le câblage électrique et l'instrumentation.

Ce projet était une occasion formidable d'appliquer nos connaissances acquises durant notre formation et de les confronter à des problèmes qu'un ingénieur affronte dans le monde industriel, d'autre part, cela nous a permis d'apprendre une méthodologie rationnelle à suivre pour l'élaboration d'un projet tout en optimisant les coûts et les performances.

En fin, nous espérons que notre travail sera utile aux étudiants en automatique, pour appliquer ce qu'ils prennent au cours de l'informatique industrielle, et qu'il participe à forger des ingénieurs automaticien avec beaucoup d'expérience qui leurs permette de transiter vers une nouvelle phase où ils seront obligés à manipuler des systèmes réels.

### **Perspectives**

La maquette qu'on a réalisé se base totalement sur des entrées/sorties logiques, Il serait intéressant de manipuler des entrées/sorties analogiques, en ajoutant par exemple:

- un débitmètre pour mesurer le débit
- un capteur de niveau analogique
- une vanne proportionnelle ou bien un variateur de fréquence pour la pompe pour la régulation (débit et/ou niveau)

## ***BIBLIOGRAPHIE :***

- [1] M. BERTRAND, « Automates programmables industriels », Techniques de l'ingénieur, Vol. S 8015.
- [2] P.JARGOT, « Langages de programmation pour API. Norme IEC 1131-3 », Techniques de l'ingénieur, Vol. S 8030.
- [3] G. MICHEL, « Les A.P.I Architecture et application des automates programmables industriels », Edition DUNOD, 1987
- [4] Jacky MONTMAIN, « SUPERVISION HOMME-MACHINE » Techniques de l'Ingénieur, Vol. S 7620
- [5] E.BAJIC et B.BOUARD, « Réseau PROFIBUS », Techniques de l'ingénieur, Vol. S 8160.
- [6] A.ABRICHE, « Réalisation et gestion d'un prototype de station de pompage à base d'automates programmables industriels SIEMENS », projet de fin d'étude, Ecole Nationale Polytechnique 2007.

### **Manuels:**

- [7] Siemens, « STEP 7 Pour une transition facile de S5 à S7 » SIMATIC, Edition 12/2002
- [8] Siemens, « Automate programmable S7-200 Manuel système » SIMATIC, Edition 08/2008.
- [9] Siemens, « Mise en route avec le S7-1200 » SIMATIC, Edition 11/2009.
- [10] Siemens, « SIMATIC S7-1500 une nouvelle génération automates » SIMATIC, Edition 2012.
- [11] Siemens, « Systèmes intégrés compacts C7-623, C7-624 » SIMATIC, Edition 1995.
- [12] Siemens, « Calculateur industriel M7-300 Installation et configuration –Caractéristiques des CPU » SIMATIC Edition 1997.
- [13] Siemens, « Faites connaissance avec le S7-300 » SIMATIC, Edition 1996.
- [14] Siemens, « S7-PLCSIM » version 5.4, SIMATIC, 07/2011.
- [15] Siemens, « Mise en route WinCC flexible débutants » SIMATIC HMI, Edition 2008
- [16] Siemens, « SIEMENS/fast industrie ; Automation and Drives-SCE », Edition 02/2006.
- [17] Siemens, « Programmer avec STEP 7 » version 5.2, SIMATIC, 05/2010.

[18] Siemens, « STEP 7, getting started», version 5.1, SIMATIC, Edition 08/2000.

[19] Siemens, « Programmation d'automate avec SIMATIC S7-300-Notion de base » Edition : 03/2001.

[20] Siemens, « Programmation pas à pas du S7-GRAPH » Edition : 02/2002.

[21] Siemens, « SIMATIC WinCC, supervision de process avec Plant Intelligence » Brochure .Avril 2012

### **Site Internet:**

[22] [www.siemens.com](http://www.siemens.com)

[23] [www.wikipedia.com](http://www.wikipedia.com)

[24] [www.SITELEC.org](http://www.SITELEC.org)

### **Aide logiciel :**

[25] WinCC flexible 2008 SP2, WinCC flexible Information System.

## ANNEXE A

### Description des éléments de l'application « maquette de station de pompage à base de l'API SIEMENS »

---

#### A.1 Démarreur progressive pour moteur asynchrone

#### A.2 RELAIS DE NIVEAU

#### A.3 pompe

#### A.4 Alimentation stabilisée

#### A.5 Electrovanne

#### A.6 relais d'interface

### A.1 Démarreur progressive pour moteur asynchrone



Figure A.1 : démarreur progressif ATS 01N103FT

#### A.1.1 Présentation :

Le démarreur progressif Altistart 01 est soit un limiteur de couple au démarrage, soit un démarreur progressif ralentisseur pour les moteurs asynchrones.

L'utilisation de l'Altistart 01 améliore les performances de démarrage des moteurs asynchrones en permettant un démarrage progressif sans à-coup et contrôlé. Son utilisation permet la suppression des chocs mécaniques cause d'usure, d'entretien et de temps d'arrêt de production.

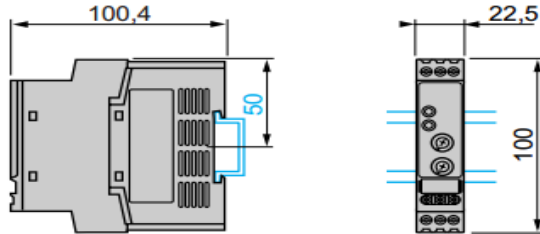
L'altistart 01 limite le couple de décollage et les pointes de courant au démarrage, Il est destiné aux applications simples suivantes :

Convoyeurs, tapis transporteurs, pompes, ventilateurs, compresseurs, portes automatiques, les petits portiques, machines à courroies...

#### A.1.2 Encombrements :

## ATS 01N103FT, ATS 01N106FT

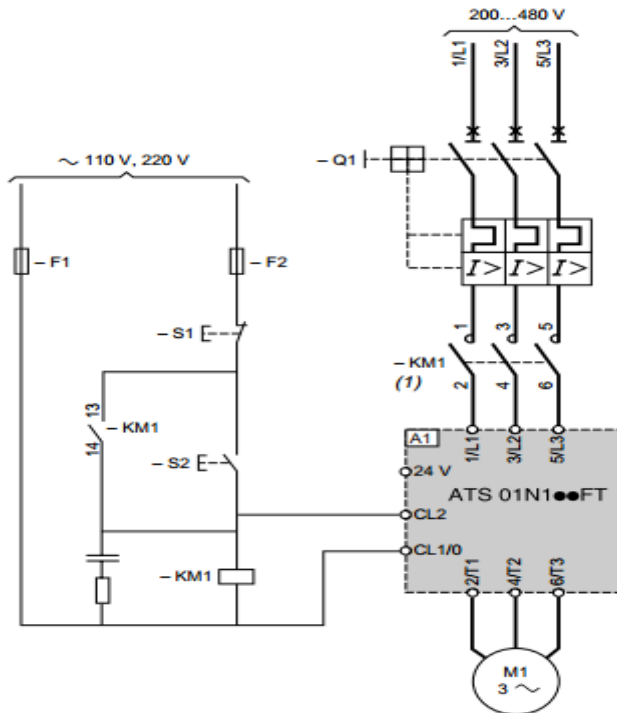
Montage sur profilé  $\sqsubset$  (35 mm) ou sur profilé  $\sqsubset$  avec l'adaptateur RHZ 66



### A.1.3 Schémas

#### Démarrateurs progressifs ATS 01N1●●FT

Alimentation monophasée ou triphasée



#### Repère Désignation

- A1 Démarreur progressif
- Q1 Disjoncteur GV2 ME
- KM1 LC1 ppp + LA4 DA2U
- F1, F2 Fusibles de protection de commande
- S1, S2 Boutons poussoirs XB4 B ou XB5 B

Pour moteur monophasé, on utilise l'ATS 01N1ppFT sans connecter la 2ème phase 3/L2, 4/T2.

## A.2 RELAIS DE NIVEAU

Les relais de contrôle de niveaux de liquides sont utilisés pour la protection marche à sec des pompes ou comme régulateur de niveau des liquides. Ils fonctionnent à l'aide de détecteurs, qui mesurent la conductibilité. A cette fin un détecteur pour la hauteur de remplissage

maximale et un détecteur pour la hauteur de remplissage minimale sont nécessaires. Un troisième détecteur sert comme potentiel de masse.



**Figure A.2 :** relais de niveau

## **A.3 pompe**

### **A.3.1 Description d'une pompe électrique**



Une pompe électrique est une pompe à eau qui fonctionne à partir de l'énergie électrique. Elle transforme l'énergie électrique reçue en énergie mécanique servant à activer la pompe. L'énergie mécanique sera ensuite transmise au liquide où elle est convertie en énergie cinétique « débit » puis en énergie potentielle « pression ». C'est à partir de ces énergies que le fonctionnement de la pompe à eau électrique s'opère à travers des échanges et consommations d'énergies dans les circuits d'installation.

### **A.3.2 L'utilité du surpresseur d'eau :**

Peu importe l'endroit, lorsque la pression au niveau des installations ne suffit pas, le surpresseur s'avère nécessaire. Ce matériel assure alors une meilleure pression du circuit peu importe les besoins. Le surpresseur d'eau est dans certains cas pourvu d'un variateur de fréquence ou d'un ballon tampon. Il faut souligner que le surpresseur est un dispositif

facultatif que vous pouvez installer après votre compteur d'eau. A la différence du réducteur de pression, le surpresseur vise à remédier à un problème de pression faible.

### **A.3.3 Le mode de fonctionnement du surpresseur d'eau :**

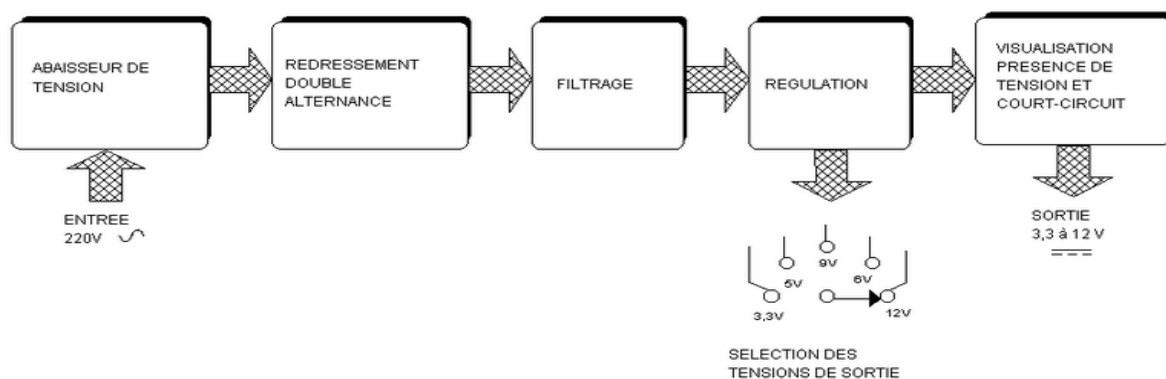
Le surpresseur d'eau sert principalement à faciliter la montée de l'eau. Celui-ci va alors aspirer l'eau pour la faire venir jusqu'à la surface et ce, avec une pression appréciable. Cet appareil est constitué d'un moteur produisant la puissance utile au pompage. Cette puissance sera ensuite émise à la partie hydraulique qui à son tour s'occupe de faire parvenir cette puissance jusqu'à l'eau pour ainsi enclencher son ascension. Dans le cas d'un puits, le tuyau du surpresseur d'eau est inséré au fond du puits, puis le moteur est activé et transmet une grande source d'énergie contribuant alors à l'ascension de l'eau. Le surpresseur s'avère également utile pour les eaux de pluies. Celui-ci maintient les eaux puis les utilise à nouveau notamment pour arroser le jardin.

## **A.4 Alimentation stabilisée**



### **A.4 .1 Définition :**

Une alimentation stabilisée est un dispositif électronique, utilisant une régulation, permettant le réglage de la tension et du courant nécessaires au fonctionnement d'un appareil électrique ou électronique. Il est généralement conçu pour fonctionner en générateur idéal de tension et de courant.



**Figure A.3 :** Schéma de principe de l'alimentation stabilisée

## A.5 Electrovanne

### A.5.1 Définition

Une électrovanne ou électrovalve est un dispositif commandé électriquement par un électroaimant permettant d'autoriser ou d'interrompre par une action mécanique la circulation d'un fluide dans un circuit. Une électrovanne est composée de deux parties élémentaires : une tête électromagnétique (électro-aimant) et son noyau mobile (plongeur), une vanne dont l'orifice est obturé par un clapet ou un pointeau. L'ouverture et la fermeture de la vanne sont fonction de la position du noyau mobile qui se déplace sous l'effet du champ magnétique engendré par la mise sous tension de la bobine. Il existe deux types d'électrovanne : "tout ou rien" et "proportionnelle". Elles sont utilisées usuellement dans : l'industrie alimentaire, les industries chimiques et pétrochimiques...

### A.5.2 Type des Electrovanes

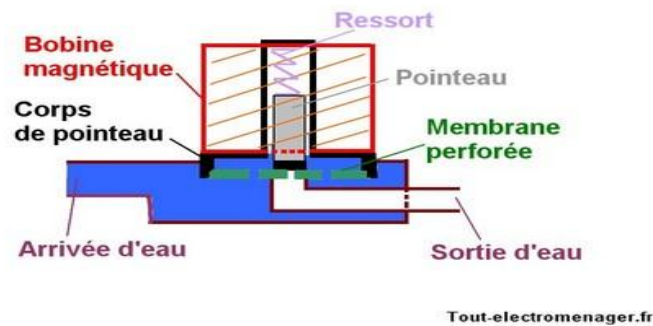
Il existe deux types d'électrovannes : « tout ou rien » et « proportionnelle ».

1)-Les électrovannes dites de « tout ou rien » ne peuvent s'ouvrir qu'en entier ou pas du tout. L'état change suivant qu'elles sont alimentées électriquement ou non. Il existe deux sortes d'électrovannes tout ou rien :

2)-Les électrovannes proportionnelles sont celles qui peuvent être ouvertes avec plus ou moins d'amplitude en fonction du besoin. Selon les types de vannes l'ouverture peut être proportionnelle au courant électrique de l'alimentation, ou à la tension électrique de l'alimentation. Ce type d'électrovanne est généralement piloté par l'intermédiaire d'une commande.



## Schéma d'une électrovanne



## A.6 relais d'interface

### A.6.1 Définition

C'est un appareil composé d'une bobine (électroaimant) qui lorsqu'elle est parcourue par un courant électrique agit sur un ou plusieurs contact.

Un relais électromagnétique est un organe électrique permettant de dissocier la partie puissance de la partie commande : Il permet l'ouverture/fermeture d'un circuit électrique par un second circuit complètement isolé (isolation galvanique) et pouvant avoir des propriétés différentes.

### A.6.2 Description

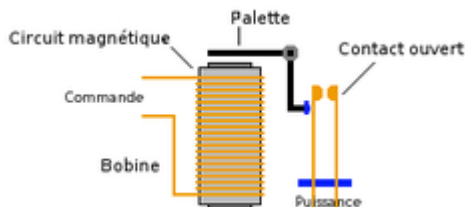


Figure A.4: Schéma d'un relais électromécanique

- Un relais est composé principalement d'un électroaimant, qui lorsqu'il est alimenté, transmet une force à un système de commutation électrique : les contacts.
- L'électroaimant peut être, suivant les spécifications et besoins, alimenté en TBT (Très Basse Tension) (12 V, 24 V, 48 V) continu ou alternatif ou en BT (Basse Tension) (230 V, 400 V).
- Le système de commutation peut être composé d'un ou plusieurs interrupteurs simple effet appelés contacts normalement ouverts (NO) ou normalement fermés (NF), d'un ou plusieurs inverseurs (contacts repos-travail RT). Ces commutateurs sont adaptés aux courants et à la gamme de tensions à transmettre à la partie puissance.

- Dans les systèmes mettant en œuvre une certaine puissance, on appelle les relais des contacteurs.
- Divers systèmes mécaniques ou pneumatiques peuvent créer un retard à l'enclenchement ou au relâchement.

## ANNEXE B

Les programmes de l'application «maquette de station de pompages à base d'automates programmable SIEMENS »

---

### B.1 Le bloc d'organisation OB1

### B.2 La fonction de gestion de la première manipulation FC1

### B.3 La fonction de gestion de la deuxième manipulation FC2

### B.4 Le bloc d'initialisation OB100

Le programme qui suit concerne l'application : « Réalisation et gestion d'un prototype de station de pompage à base d'API SIEMENS », on peut voir dans l'ordre le bloc d'organisation principale OB1 et OB100, ainsi qu'aux fonctions qu'il appelle.

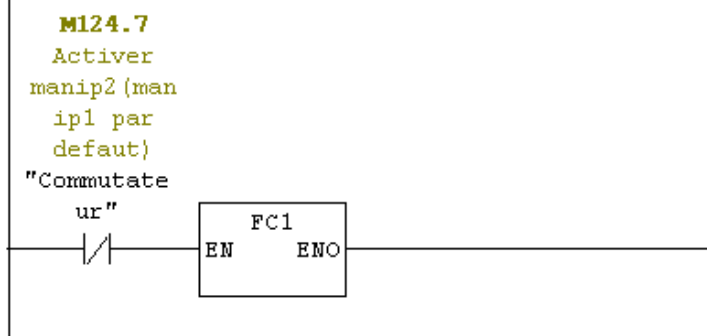
### B.1 Le bloc d'organisation OB1

OB1 : "Main Program Sweep (Cycle)"

Commentaire :

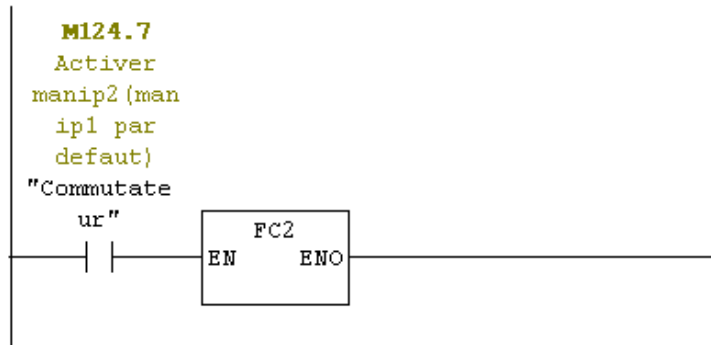
Réseau 1 : manip1

Commentaire :



## Réseau 2 : manip2

Commentaire :

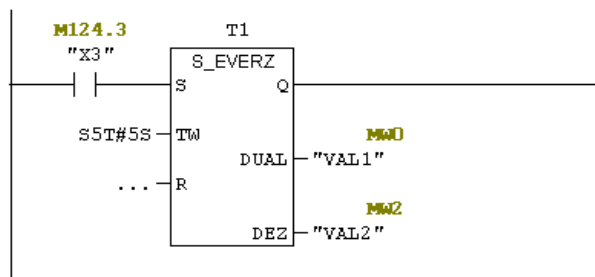


On détail dans ce qui suit les deux fonctions FC1 et FC2 « Gestion de la station »

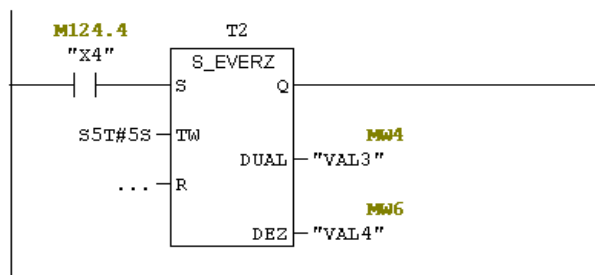
### B.2 La fonction de gestion de la première manipulation FC1

FC1 : Manipulation1

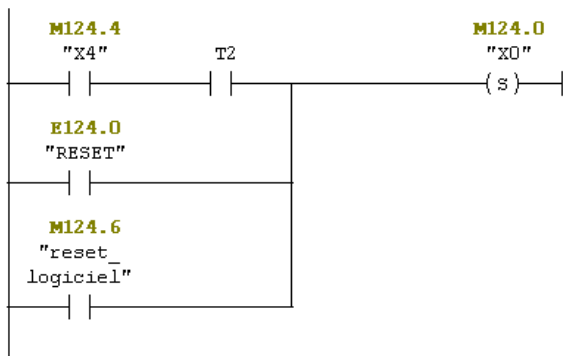
Réseau 1 : Timer 1



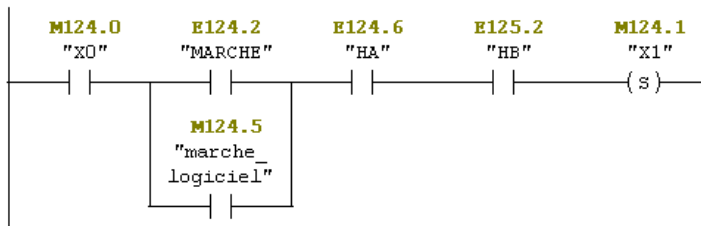
Réseau 2 : Timer 2



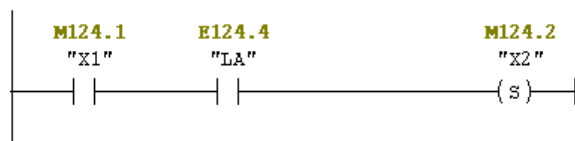
Réseau 3 : S0



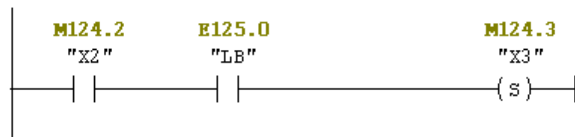
Réseau 4 : S1



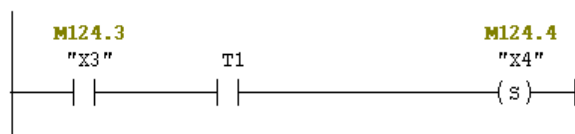
Réseau 5 : S2



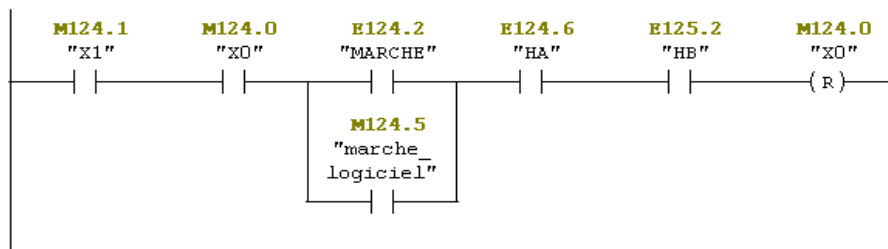
Réseau 6 : S3



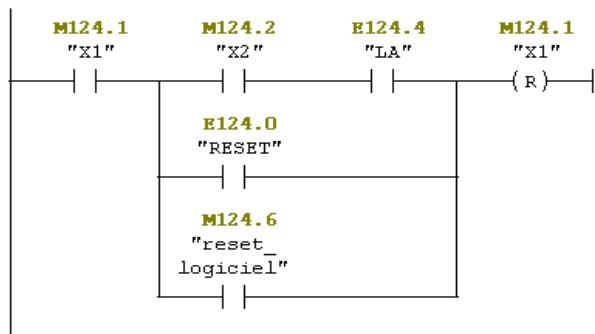
Réseau 7 : S4



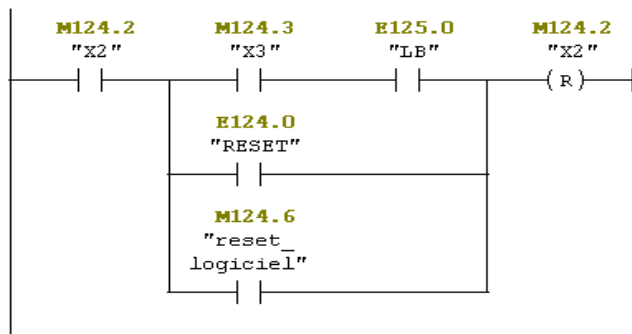
Réseau 8 : R0



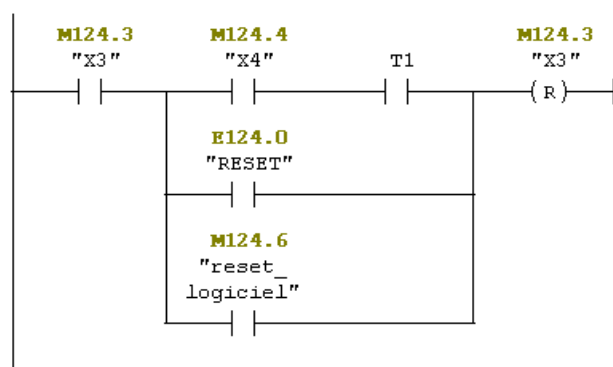
Réseau 9 : R1



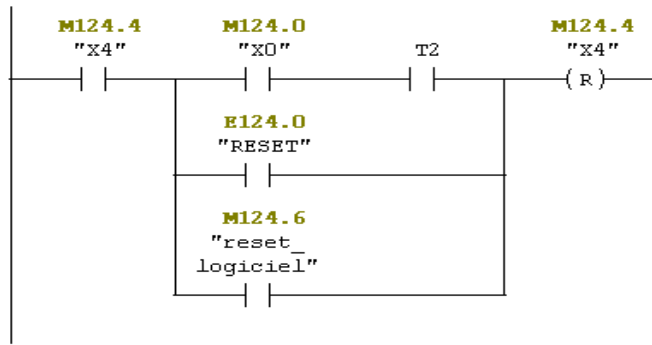
Réseau 10 : R2



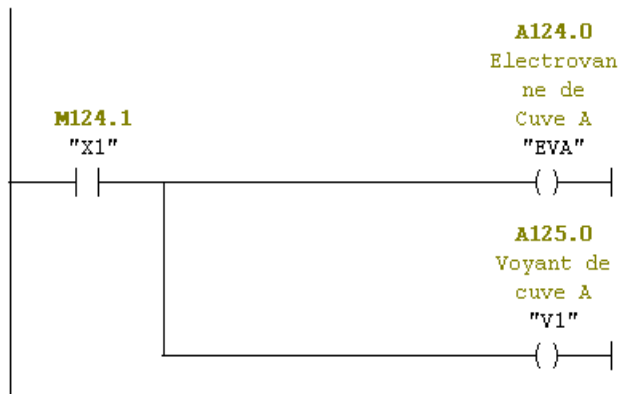
Réseau 11 : R3



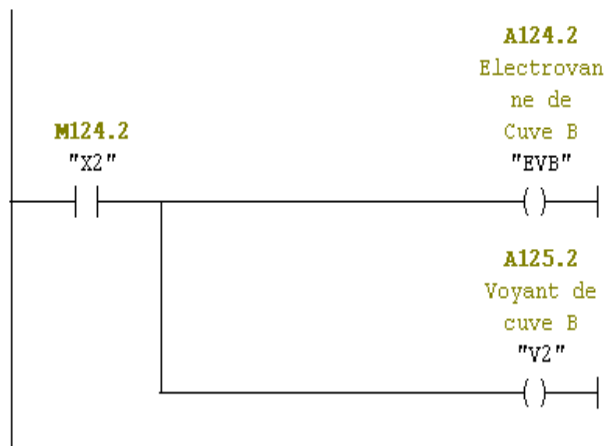
Réseau 12 : R4



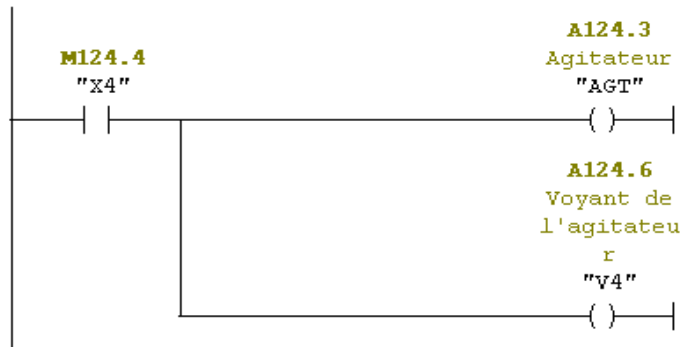
Réseau 13 : EVA et V1



Réseau 14 : EVB et V2



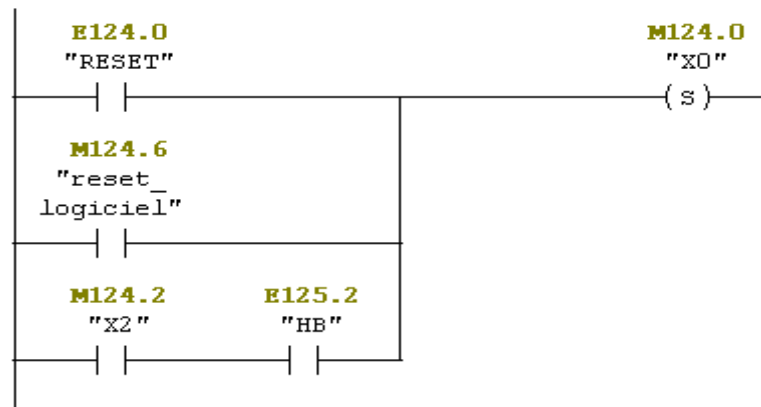
Réseau 15 : AGT et V4



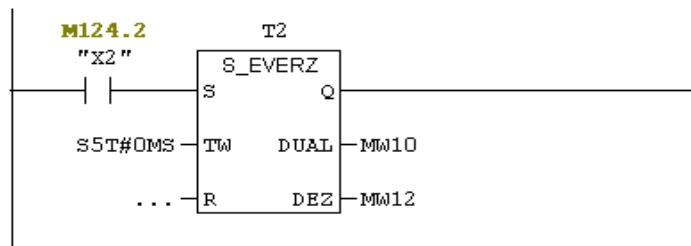
### B.3 La fonction de gestion de la deuxième manipulation FC2

FC2 : Manipulation2

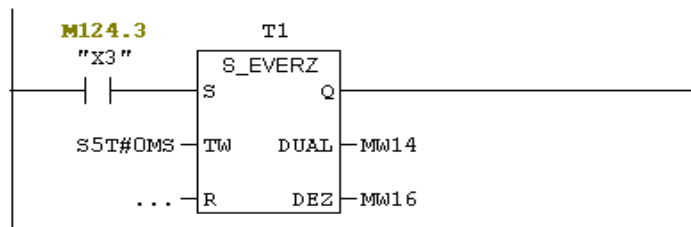
Réseau 1 : S0



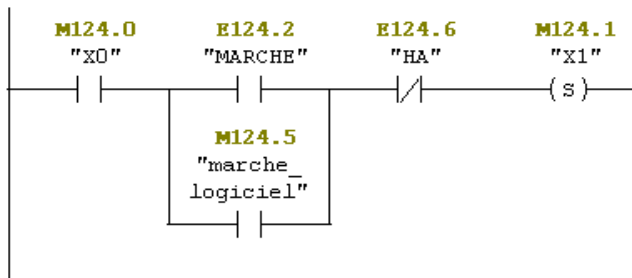
Réseau 2 : Titre :



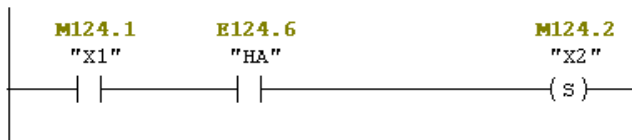
Réseau 3 : Titre :



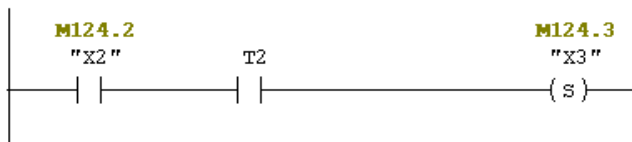
Réseau 4 : S1



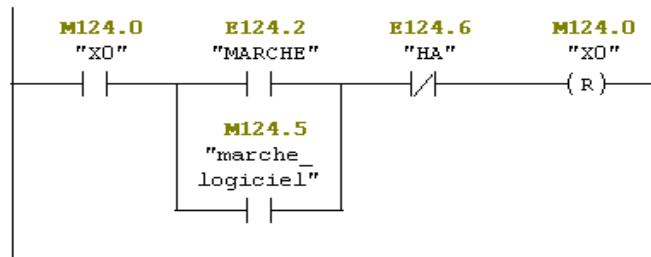
Réseau 5 : S2



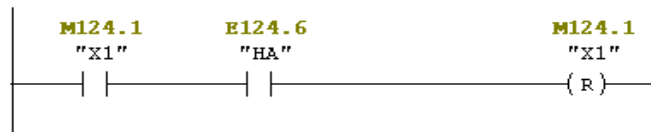
Réseau 6 : S3



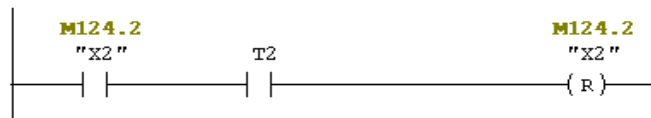
Réseau 7 : R0



Réseau 8 : R1

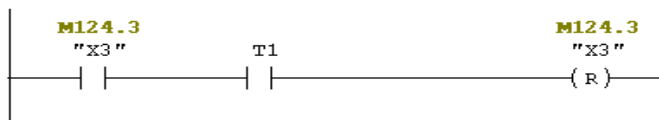


Réseau 9 : R2

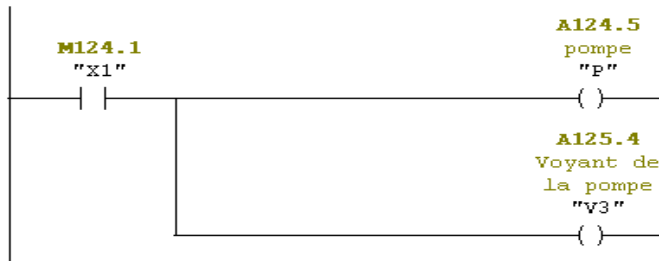




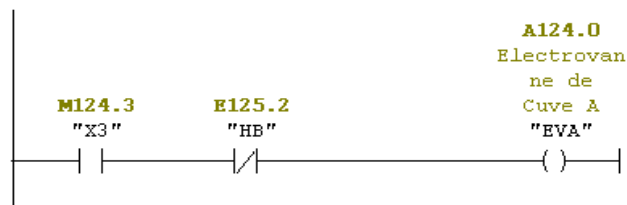
**Réseau 10 : R3**



**Réseau 11 : pompe et Voyon V3**



**Réseau 12 : EVA**



**Réseau 13 : V1**



## B.4 Le bloc d'initialisation OB100

OB100 : "bloc d'initialisation"

ce bloc est appelé lorsque l'automate passe de stop à Run

**Réseau 1** : Mise à zero de toutes les étapes de grafcet

les étapes de grafcet utilise les memoires internes de M124.0 à M124.4, je met la valeur 0 dans l'octet 124

