

Chapitre 3 : Programmation de l'API

3.1 Les langages de programmation des A.P.I :

Les langages des A.P.I. sont des langages intermédiaires entre le langage évolué et le langage machine. Ils ont l'avantage d'avoir un jeu d'instructions incluant uniquement les fonctions logiques, cela a comme conséquences, une meilleure compréhension par les automaticiens et une simplification du compilateur de la console de programmation et du logiciel constructeur.

3.1.1 Les divers types de langages pour API:

La norme IEC 1131-3 définit cinq langages qui peuvent être utilisés pour la programmation des automates programmables industriels. Ces cinq langages sont :

- Le langage LADDER LD (« **LadderDiagram** », ou **schéma à relais**): ce langage graphique est essentiellement dédié à la programmation d'équations booléennes (vraie/faux).
- Le langage mnémonique IL (« **Instruction List** », ou **liste d'instructions**): ce langage textuel de bas niveau est un langage à une instruction par ligne. Il peut être comparé au langage assembleur.
- Le langage booléen FBD (« **Function Block Diagram** », ou **schéma par blocs**): ce langage permet de programmer graphiquement à l'aide de blocs, représentant des variables, des opérateurs ou des fonctions. Il permet de manipuler tous les types de variables.
- **ST** («**StructuredText** » ou **texte structuré**): ce langage est un langage textuel de haut niveau. Il permet la programmation de tout type d'algorithme plus ou moins complexe.

3.1.2 Le langage LADDER (LD : Ladder diagramme) :

Le langage LD (ladder diagram) est une représentation graphique d'équations booléennes combinant des contacts (en entrée) et des relais (en sortie). Il permet la manipulation de données booléennes, à l'aide de symboles graphiques organisés dans un diagramme comme les éléments d'un schéma électrique à contacts. Les diagrammes LD sont limités à gauche et à droite par des barres d'alimentation. C'est le plus utilisé (voir Figure 3.1).

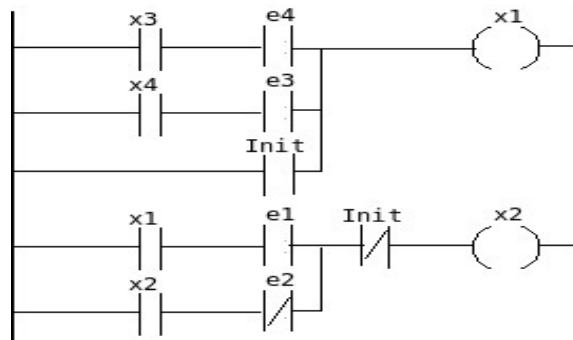


Figure 3.1: Exemples de programmation LADDER

3.1.3 Le langage Booléen (FBD : Fonction Bloc Diagramme) :

Le langage FBD (function block diagram) est un langage graphique. Il permet la construction d'équations complexes à partir des opérateurs standard, de fonctions ou de blocs fonctionnels. les fonctions sont représentées par des rectangles avec les entrées à gauche et les sorties à droites (voir Figure 3.2).

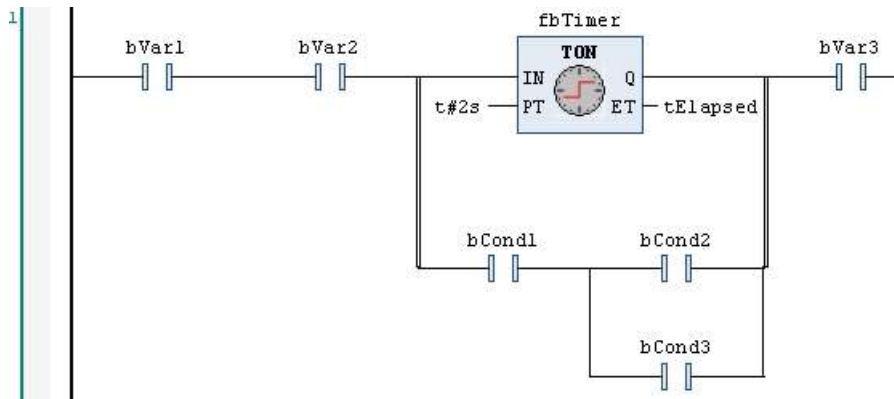


Figure 3.2: Exemple de programmation Booléenne

3.1.4 Le langage mnémonique (IL : Instruction List) :

Le langage IL (instruction list), est un langage textuel de bas niveau. Il est particulièrement adapté aux applications de petite taille. Les instructions opèrent toujours sur un résultat courant (ou registre IL). L'opérateur indique le type d'opération à effectuer entre le résultat courant et l'opérande. Le résultat de l'opération est stocké à son tour dans le résultat courant.

Un programme IL est une liste d'instructions. Chaque instruction doit commencer par une nouvelle ligne, et doit contenir un opérateur, complété éventuellement par des modificateurs et, si c'est nécessaire pour l'opération, un ou plusieurs

opérandes, séparés par des virgules (','). Une étiquette suivie de deux points (':') peut précéder l'instruction. Si un commentaire est attaché à l'instruction, il doit être le dernier élément de la ligne. Des lignes vides peuvent être insérées entre des instructions. Un commentaire peut être posé sur une ligne sans instruction.

Langage textuel de même nature que l'assembleur (programmation des microcontrôleurs) (voir Figure 3.3).

```
! %L0: LD      %I1.0
      ANDN    %M12
      OR (    %TM4.Q
      AND     %M17
      )
      AND     %I1.7
      ST      %Q2.5
! %L5: LD      %I1.10
      ANDN    %Q2.3
      ANDN    %M27
      IN      %TM0
      LD      %TM0.Q
      AND     %M25
      AND     %M000.XS
      [ %M005 := %M000+500 ]
```

Figure 3.3: Exemple de programmation mnémonique

3.1.5 Blocs fonction-Blocs opération :

En complément aux possibilités de représentation, les langages de programmation permettent l'utilisation de blocs fonctions et de blocs opérations. Ces blocs sont des fonctions préprogrammées, paramétrables, utilisables directement par le programmeur (voir Figure 3.4) Exemple de blocs fonctions : Temporisateur, Compteur.

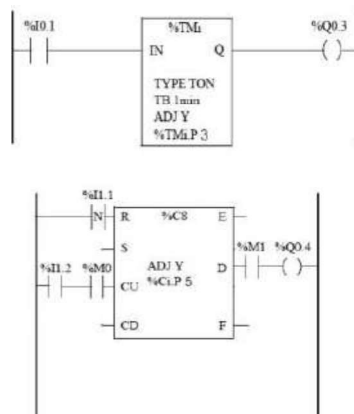


Figure 3.4: Exemple de programmation de blocs fonctions

3.1.6 Le langage GRAFCET (SFC : Séquentiel Fonction Charte) :

Le langage GRAFCET SFC (« **Sequential Function Char** »): issu du langage GRAFCET, ce langage, de haut niveau, permet la programmation aisée de tous les procédés séquentiels.

Le GRAFCET, langage de spécification, est utilisé par certains constructeurs d'automate (Schneider, Siemens) pour la programmation (voir Figure 3.5). Parfois associé à un langage de programmation, il permet une programmation aisée des systèmes séquentiels tout en facilitant la mise au point des programmes ainsi que le dépannage des systèmes. On peut également traduire un GRAFCET en langage contacts et l'implanter sur tout type d'automate.

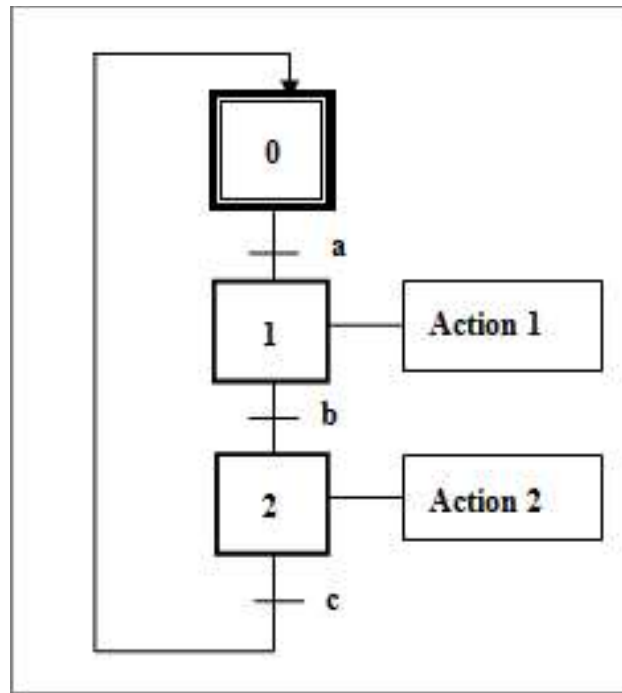


Figure 3.5: Exemples de programmation GRAFCET

3.2 Différentes types des variables

Toute expression, constante ou variable, utilisée dans un programme doit être caractérisée par un type, les types de base sont :

- **BOOL** (Booléen) : ce type donne la valeur « Vraie » ou « Faux » qui sont équivalent à « 1 » ou « 0 »).
- **DINT** (Entier) : c'est un nombre entre -2147483647 et +2147483647. Il est exprimé dans l'une des bases suivantes : décimale, hexadécimale, octale ou binaire.
- **REAL** (Réel) il prend 1 bit de signe +23 bits de mantisse +8 bits d'exposant compris entre -37 et +37.
- **TIME** (Temporisation) : c'est une valeur strictement positive et commence par T# ou TIME#.
- **STRING** (Chaîne) : elle doit être précédée et suivie par une apostrophe, et ne doit jamais excéder 255 caractères). Le caractère spécial ('\$') est utilisée pour insérer des caractères non imprimables.

3.3 Langage GRAFCET

3.3.1 Définition

GRAFCET (Graphe fonctionnel de commande des étapes et de transition) est un diagramme fonctionnel utilisé pour représenter les systèmes techniques fonctionnant selon des séquences d'opération successive, l'objectif de son utilisation est de définir la commande à partir des entités de base que sont les Etapes et les transitions.

Le GRAFCET est un langage universel, peut se câbler sur un séquenceur et programmer dans un Automate programmable industriel comme on peut le vérifier sur un ordinateur, d'autre part il facilite la conception et la maintenance et améliore la compréhension du système.

3.3.2 Concepts de base d'un GRAFCET : Les étapes d'un système de production sont représentées graphiquement par un ensemble de :

- **Etapes** auxquelles sont associées des **actions**
- **Transitions** aux quelles sont associées des **réceptivités**
- **Liaisons orientées**

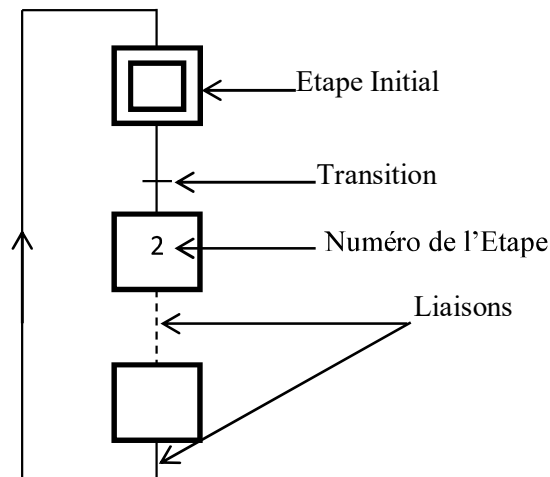


Figure 3.6: Description du programmation par GRAFCET

- **Etape initiale** : est représentée par deux carrée et toujours active au début des séquences.
- **Etape active** : les actions associées sont exécutées
- **Etape inactive** : les actions associées ne s'exécutent pas.
- **Transition** : Equation logique qui constate la bonne exécution des actions de l'étape précédente et qui fixe des conditions particulaire pour passer à l'étape suivante.
- **Transitions** : La transition est **Validé** la réceptivité qui lui est associée est prise en compte, si elle est **Non validée** la réceptivité qui lui est associée n'est pas prise en compte, La transition est **Franchissable** l'étape suivante peut être activée
- **Réceptivité** : s'il est **Vraie** la condition logique est réalisée, sinon **Fausse** la condition logique n'est pas réalisée.

3.3.3 Règles d'évolution

a) Situation initiale(Etape Initial)

- b) **Franchissement des transitions** : lorsque une étape est activée la transition qui suite est validée alors si la transitions est validée et que la réceptivité associée est vraie alors la transition doit être obligatoirement franchie. Le franchissement d'une **transition** entraine l'**activation** de toutes étapes immédiatement suivantes et la **désactivation** de toutes les étapes immédiatement précédentes.

c) **Activation est désactivation des étapes** : toutes les étapes précédentes sont désactivées et toutes les étapes suivantes sont activées

- Une étape active est activé et désactivé ou même temps, il reste active
- Plusieurs transitions simultanément franchissables sont simultanément franchies

3.3.4 Sélection des séquences

Un GRAFCET est généralement constitué de plusieurs séquences, c'est-à-dire de plusieurs suites d'étapes à exécuter les uns après les autres et il est souvent nécessaire d'effectuer une sélection exclusive d'une de ces séquences.

3.3.4.1 Divergence et convergence en "OU"

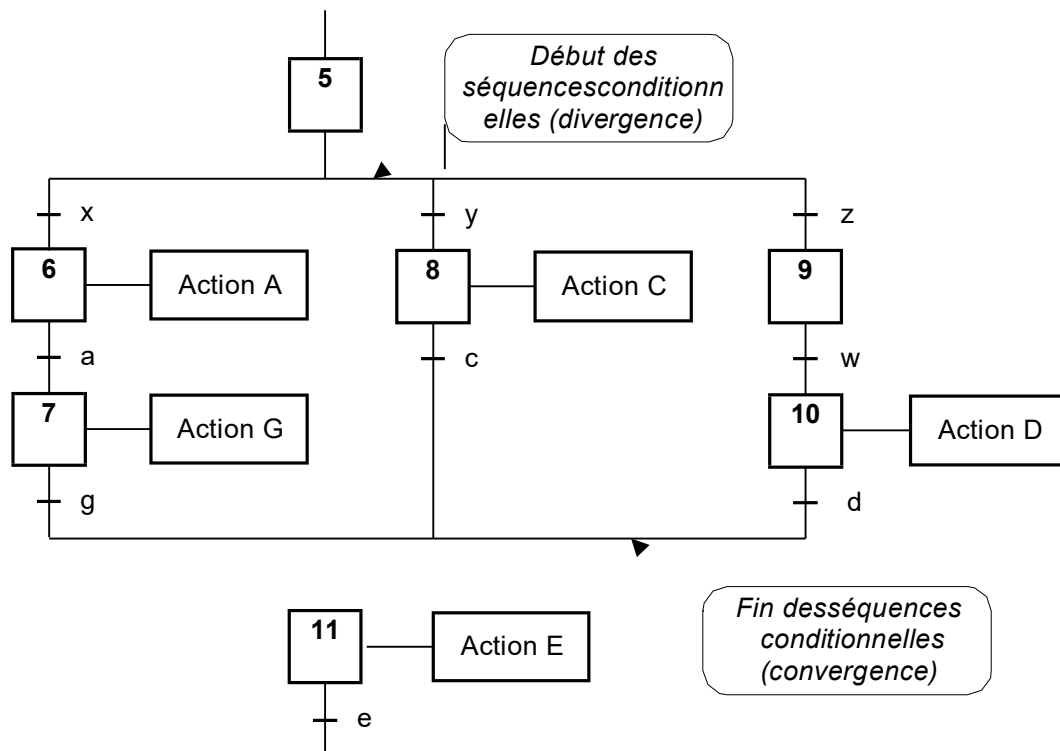


Figure 3.7: Schéma de Divergence et convergence en "OU"

Dans l'aiguillage formé par le choix de la séquence à réaliser, les différentes transitions correspondant aux réceptivités x, y et z étant simultanément validées par la même étape 5, pourraient être franchies simultanément. En pratique, l'automaticien est souvent amené à rendre des réceptivités exclusives. Il est possible également d'introduire des priorités.

3.3.4.2 Divergence et convergence en "ET"

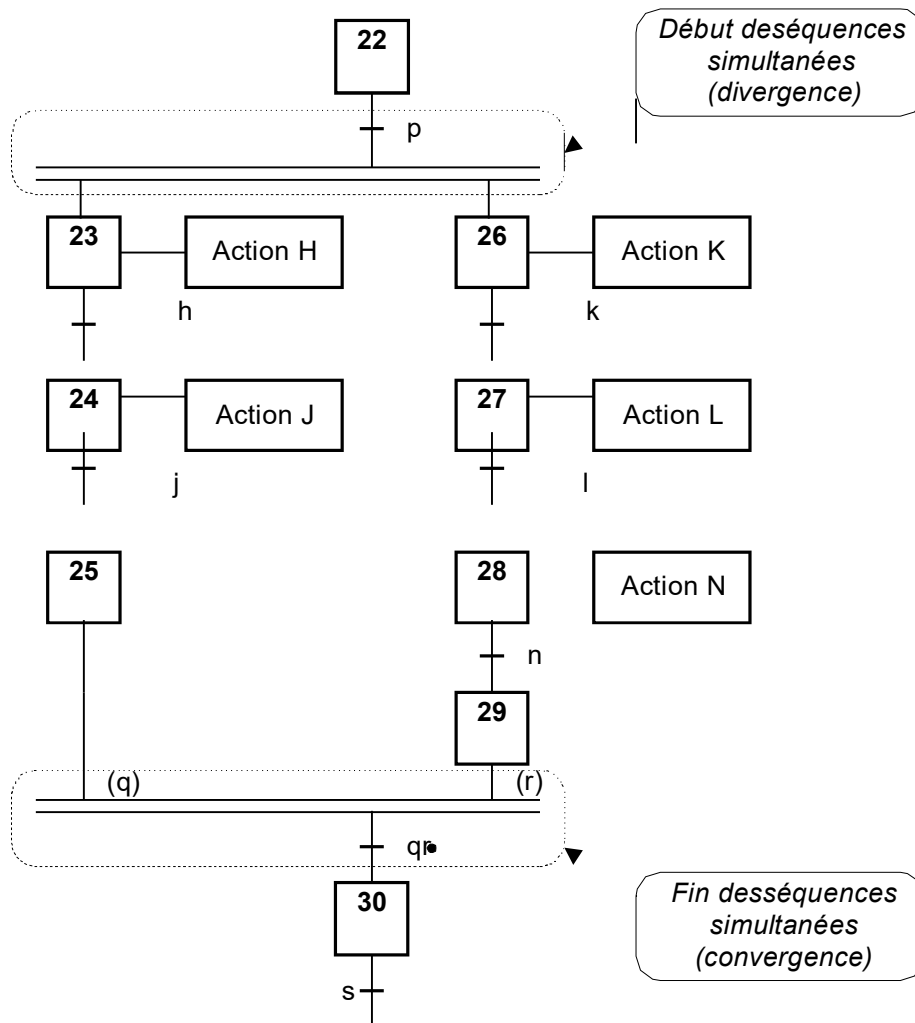


Figure 3.8: Schéma de Divergence et convergence en "ET"

Un GRAFCET peut comporter plusieurs séquences s'exécutant simultanément mais dont les évolutions des étapes actives dans chaque branche restent indépendantes.

Pour représenter ces fonctionnements simultanés, une transition UNIQUE et deux traits parallèles indiquent le début et la fin des séquences, c'est-à-dire l'activation simultanée des branches ainsi réalisées et leur attente réciproque vers une séquence commune.

A partir de l'étape 22 de la figure précédente, la réceptivité p provoque l'activation simultanée des étapes 23 et 26.

Ces deux séquences **23-24-25** et **26-27-28-29** évolueront alors de façon totalement indépendante et ce n'est que lorsque les étapes de fin de branche 25 et 29 sont actives, lorsque la réceptivité est vraie ($q.r = 1$), que la transition sera franchie. L'étape 30 devient alors active et les étapes 25 et 29 inactives.

3.3.4.3 Sauts d'étapes et reprise de séquences

Le saut conditionnel est un aiguillage particulier permettant de sauter une ou plusieurs étapes lorsque les actions à réaliser deviennent utiles, tandis que la reprise de séquences permet au contraire de reprendre une ou plusieurs fois la même séquence tant qu'une condition fixée n'est pas obtenue.

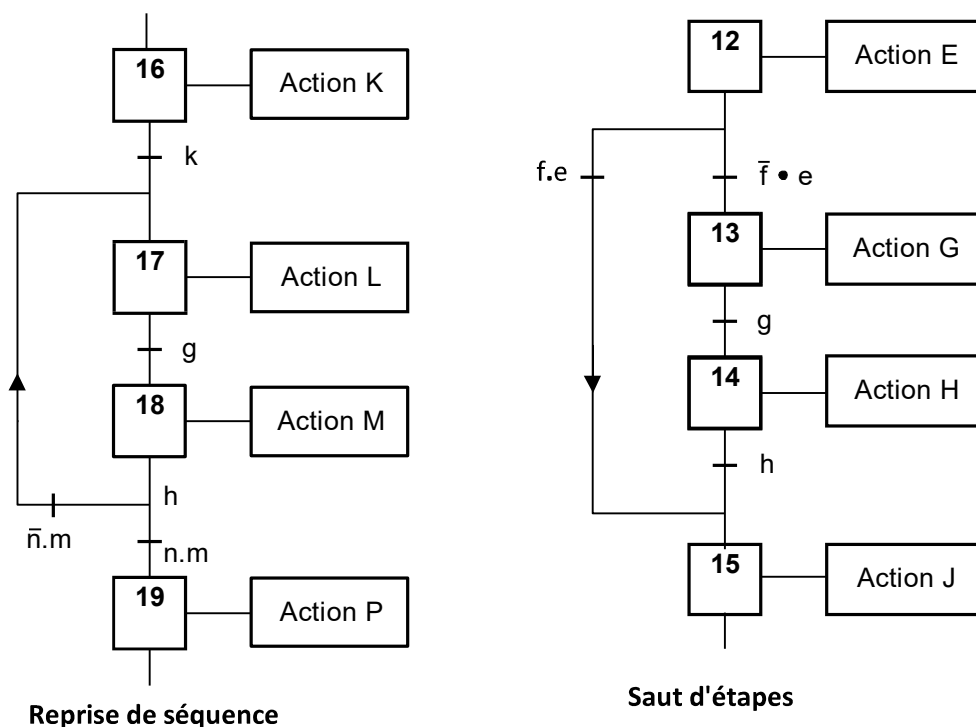


Figure 3.9: Schéma de Sauts d'étapes et reprise de séquences

3.4 Logiciel TIA portal

3.4.1 Introduction

La plateforme de développement TIA Portal de Siemens permet de faire un gain important en temps lors du développement de systèmes d'automatisation. C'est une plateforme tout en un comportant le logiciel Step7 pour la programmation d'automates et WinCC Flexible pour les interfaces homme-machine. Cette plateforme est très architecturée proposant les sections HMI pour les interfaces, réseaux et Motion pour la commande de moteurs et

variateurs. Grâce à PLC Sim, on peut simuler de manière intuitive notre projet avant de le déployé sur un contrôleur.

3.4.2 Description de TIA portal :

La plateforme « Totale Intégrates Automation Portal » est le nouvel environnement De travail Siemens qui permet de mettre en œuvre des solutions d'automatisation avec un Système d'ingénierie intègre comprenant les logiciels SIMATIC Step7 et SIMATIC Win CC.



Figure 3.10: Logo du logiciel

3.4.3 SIMATIC STEP7 :

SIMATIC STEP 7, intégré à TIA Portal, est le logiciel de configuration, programmation, vérification et diagnostic de tous les automates SIMATIC. Doté d'un grand nombre de fonctions conviviales, SIMATIC STEP 7 garantit une efficacité nettement supérieure pour toutes les tâches d'automatisation, qu'il s'agisse de la programmation, de la simulation, de la mise en service ou de la maintenance.

3.4.4 SIMATIC Win CC :

Famille intégrée d'outils d'ingénierie pour la configuration des panneaux d'opérateur SIMATIC HMI, ainsi que pour les systèmes de visualisation basés sur PC Win CC Rune time Advanced et Win CC Rune time Professional.

Win CC (TIA Portal) est basé sur le nouveau cadre d'ingénierie central Totale Intégrates Automation Portal (TIA Portal), qui offre à l'utilisateur une solution uniforme, efficace et intuitive à toutes les tâches d'automatisation.

Win CC (TIA Portal) offre également une ingénierie uniforme, du Panneau de base jusqu'aux applications SCADA.

Avec les produits STEP 7 (TIA Portal), Win CC (TIA Portal) constitue la solution optimale pour une ingénierie intégrée et efficace.

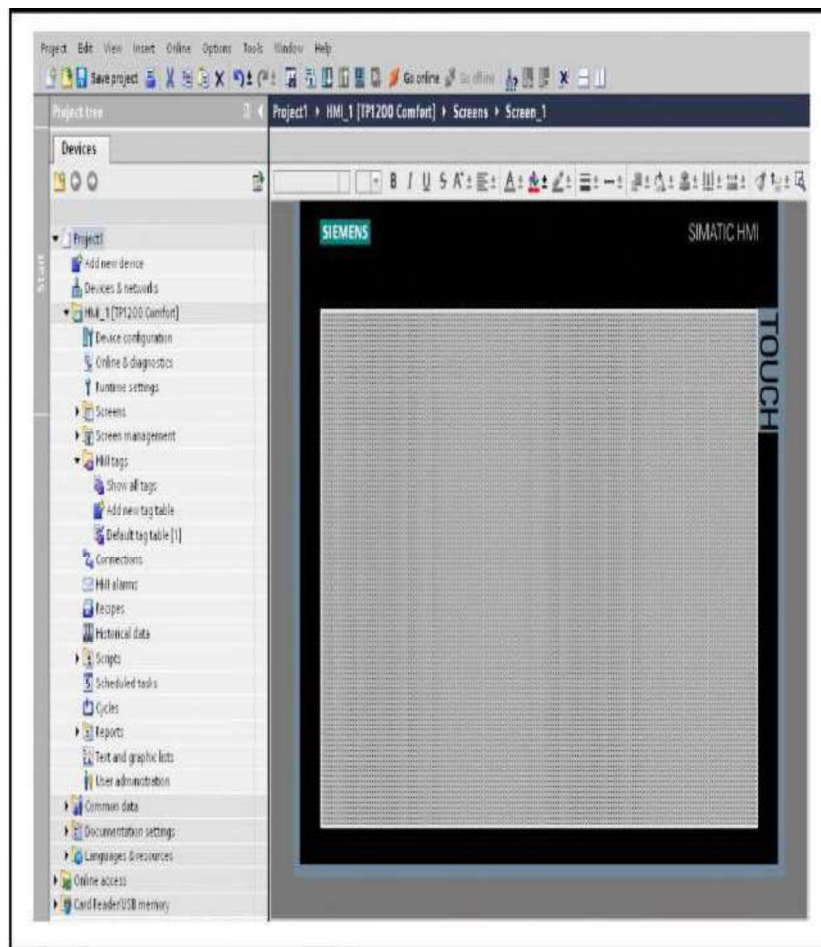


Figure 3.11: Vue du WinCC dans TIA portal

3.4.5 Logiciels de simulation auxiliaire :

a) S7-PLCSIM :

L'objectif principal de S7-PLCSIM est de contribuer à la recherche d'erreurs et à la validation d'un programme PLC unique sans avoir besoin de matériel. S7-PLCSIM vous permet d'utiliser tous les outils de recherche d'erreurs STEP 7, notamment, par exemple, les fonctions de tableau de surveillance, d'état du programme, ainsi que les fonctions en ligne et de diagnostic. S7-PLCSIM propose également des outils uniques sur S7-PLCSIM, notamment un tableau et éditeur de séquences SIM. le simulateur virtuel S7-PLCSIM fonctionne avec STEP 7 dans TIA Portal.

b) Wincc Runtime :

WinCC Runtime est le logiciel de visualisation de procédé dans Runtime, nous exécutons le projet en mode procédé.

Au Runtime, l'opérateur peut réaliser le contrôle-commande du procédé. Les tâches suivantes sont alors exécutées :

- Communication avec les automates.
- Affichage des vues à l'écran.
- Commande du procédé.
- Archivage des données de Runtime actuelles.

3.5 Vues de TIA portal :

Lorsque l'on lance TIA portal, l'environnement de travail se décompose en deux types de vue :

- **La vue du portail** : elle est axée sur les tâches à exécuter et sa prise en main est très rapide.
- **La vue du projet** : elle comporte une arborescence avec les différents éléments du projet. Les éditeurs requis s'ouvrent en fonction des tâches à réaliser. Données, paramètres et éditeurs peuvent être visualisés dans une seule et même vue.

3.5.1 Vue de portail :

La vue du portail fournit une vue d'ensemble du projet et un accès aux outils qui permettent de l'élaborer. Vous pouvez trouver rapidement ce que vous souhaitez faire, et appeler l'outil qui servira à accomplir la tâche voulue. Si vous le souhaitez, un changement vers la vue du projet s'effectue automatiquement pour la tâche sélectionnée. Cette vue simplifie donc principalement la préparation et la mise en place du projet. .

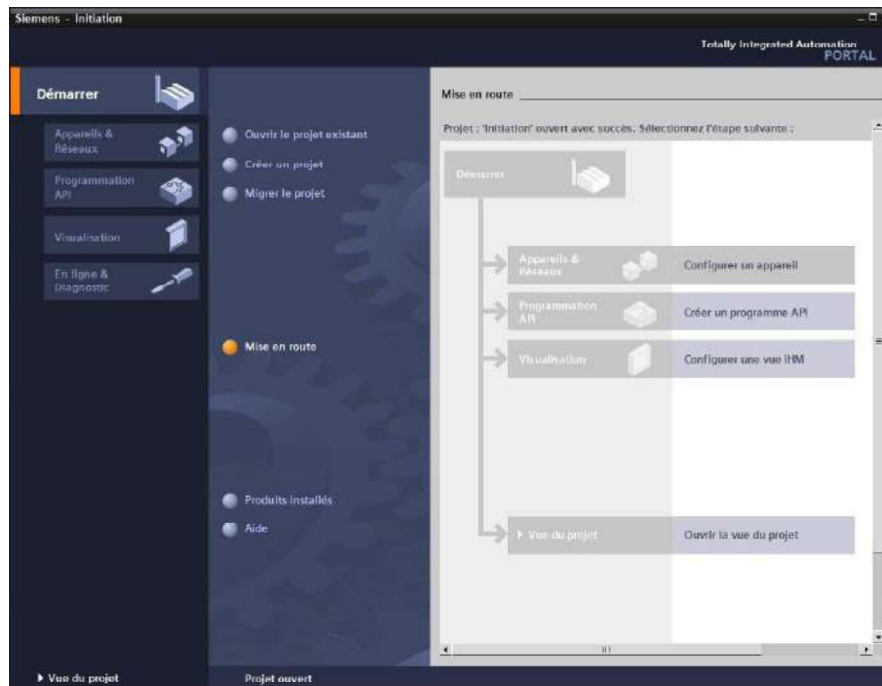


Figure 3.12: Vue portail

3.5.2 Vue projet: L'élément « Projet » contient l'ensemble des éléments et des données nécessaires pour mettre en œuvre la solution d'automatisation souhaitée.

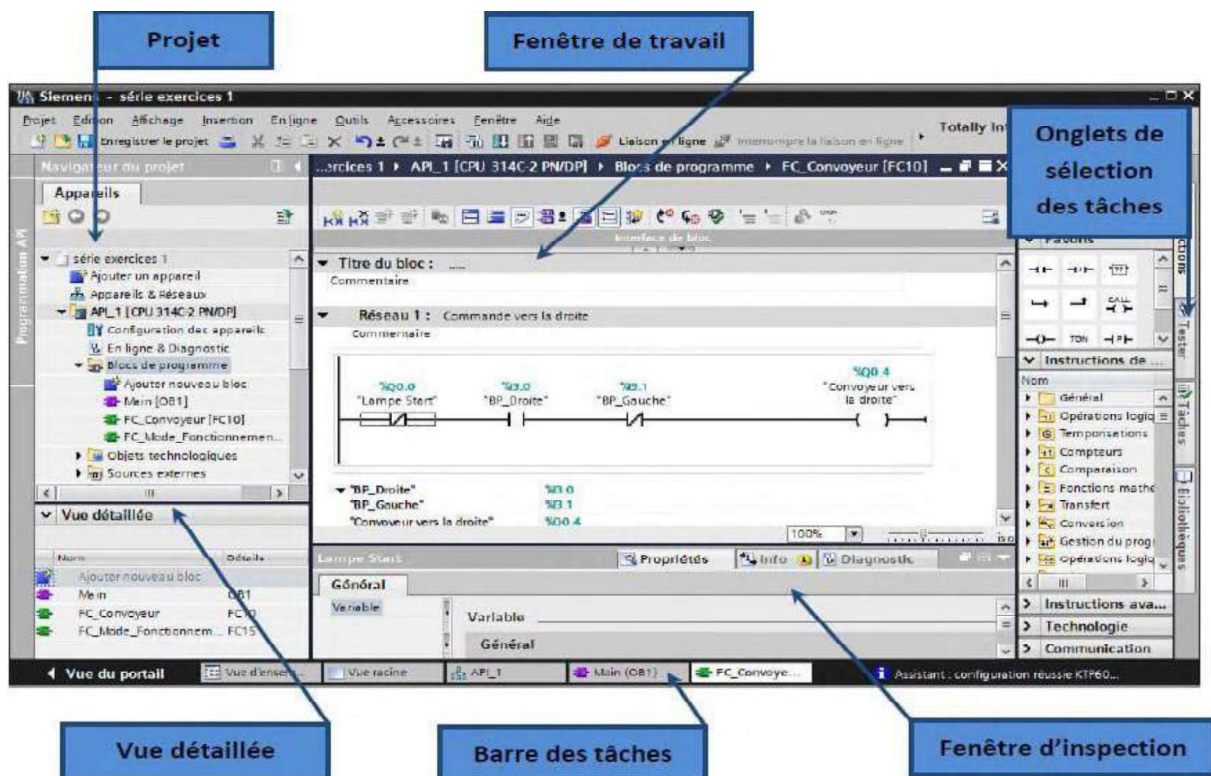


Figure 3.13: Vue de projet

La fenêtre de travail : permet de visualiser les objets sélectionnés dans le projet pour être traités. Il peut s'agir des composants matériels, des blocs de programme, des tables des variables, des HMI,...

La fenêtre d'inspection : permet de visualiser des informations complémentaires sur un objet sélectionné ou sur les actions en cours d'exécution (propriété du matériel sélectionné, messages d'erreurs lors de la compilation de blocs de programme,...)

Les onglets de sélection de tâches : ont un contenu qui varie en fonction de l'objet sélectionné (configuration matérielle bibliothèques des composants, blocs de programme, instructions de programmation).

Cet environnement de travail contient énormément de données. Il est possible de masquer ou réduire certaines de ces fenêtres lorsque l'on ne les utilise pas. Il est également possible redimensionner, réorganiser, désancrer les différentes fenêtres.

3.5.3 Les avantages du logiciel TIA portal :

- programmation intuitive et rapide : avec des éditeurs de programmation nouvellement développés SCL, CONT, LOG, LIST, et GRAPH
- efficacité accrue grâce aux innovations linguistiques de STEP7 : programmation symbolique uniforme, Calculate Box, ajout de blocs durant le fonctionnement, et bien plus encore
- performance augmentée grâce à des fonctions intégrées : simulation avec PLCSIM, télémaintenance avec TeleService et diagnostic système cohérent.
- technologie flexible : fonctionnalité motion control évolutive et efficace pour les automates S7-1500 et S7-1200.
- sécurité accrue avec Security Integrated : protection du savoir-faire, protection contre la copie, protection d'accès et protection contre la falsification.
- environnement de configuration commun avec pupitres IHM et entraînements dans l'environnement d'ingénierie TIA portal.

3.5.4. Les étapes d'application de TIA portal :

La simulation d'un projet dans TIA PORTAL nécessite à quatre étapes principales :

3.5.4.1. Création d'un projet :

Pour créer un projet dans la vue de portail, il faut sélectionner l'action « créer un projet ». On peut donner un nom au projet, choisir un chemin où il sera enregistré, indiquer un commentaire ou encore définir l'auteur du projet.

Une fois que ces informations sont entrées, il suffit de cliquer sur le bouton « créer ».

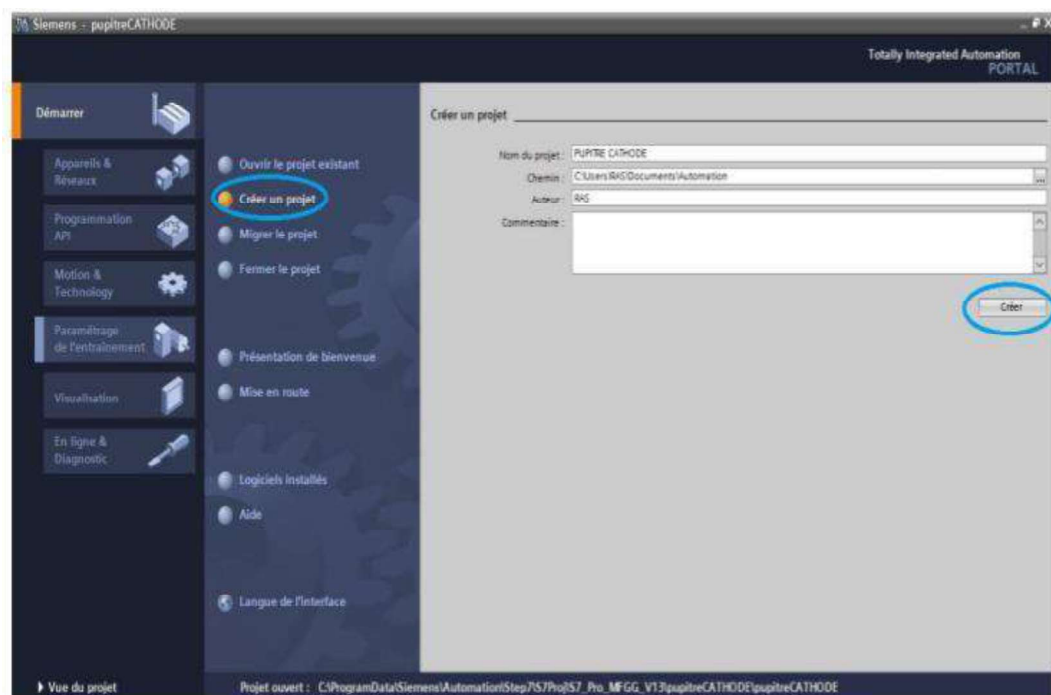


Figure 3.14: Création de nouveau projet

3.5.4.2. Configuration paramétrage matérielle :

Une fois votre projet créé, on peut configurer la station de travail. La première étape consiste à définir le matériel existant. Pour cela, on peut passer par la « vue du projet » et cliquer sur « ajouter un appareil » dans le navigateur du projet.

La liste des éléments que l'on peut ajouter apparaît (API, IHM, système PC). On commencera par faire le choix de notre CPU pour ensuite venir ajouter les modules complémentaires (alimentation, E/S TOR ou analogiques, module de communication...Etc.).

Les modules complémentaires de l'API peuvent être ajoutés en utilisant le catalogue. Si l'on veut ajouter un écran ou un autre API, il faut repasser par la commande « ajouter un appareil » dans le navigateur du projet.

La figure ci-dessous représente la configuration et paramétrage du matériel.

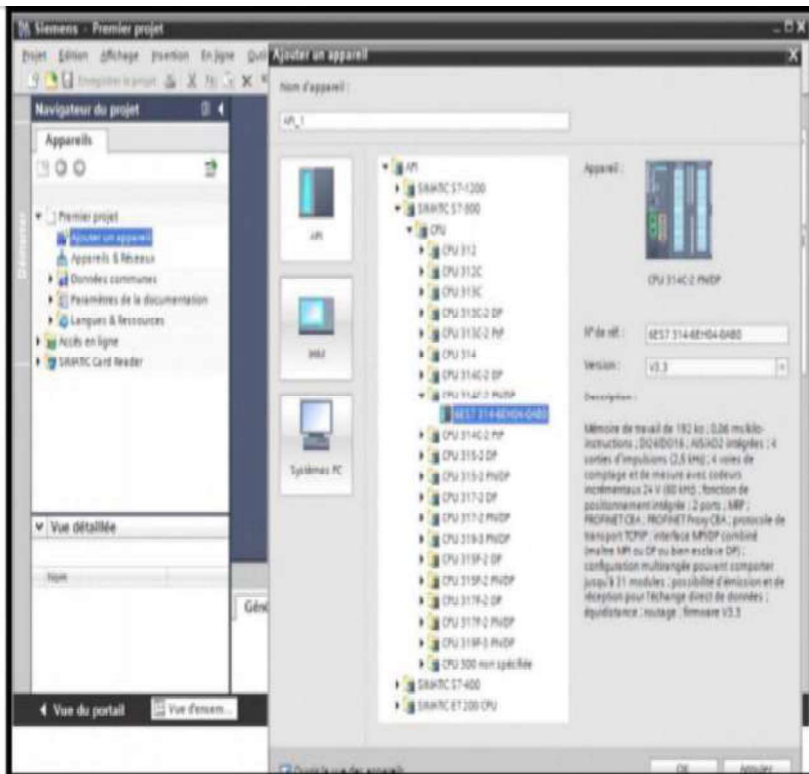


Figure 3.15: Configuration matériel

3.5.4.3. Établissement de la liaison :

Il faut maintenant charger la configuration de l'automate dans celui-ci. Pour cela, il faut tout d'abord connecter l'automate au PC en utilisant l'interface Simatic S7 PC USB adapté. Ensuite, après avoir sélectionné la vue « En ligne et diagnostique », sélectionnez les options suivantes :

- Mode : MPI
- Interface PG/PC : Pc Adapter

Il ne reste plus qu'à cliquer sur le bouton « liaison en ligne » pour vous connecter à l'automate.

3.5.4.4. Adressage des entrées/sorties :

Pour connaître l'adressage des entrées et sorties présentes dans la configuration matériel, il faut aller dans « appareil et réseau » dans le navigateur du projet.

Dans la fenêtre de travail, on doit s'assurer d'être dans l'onglet « Vue des appareils » et de sélectionner l'appareil voulu.

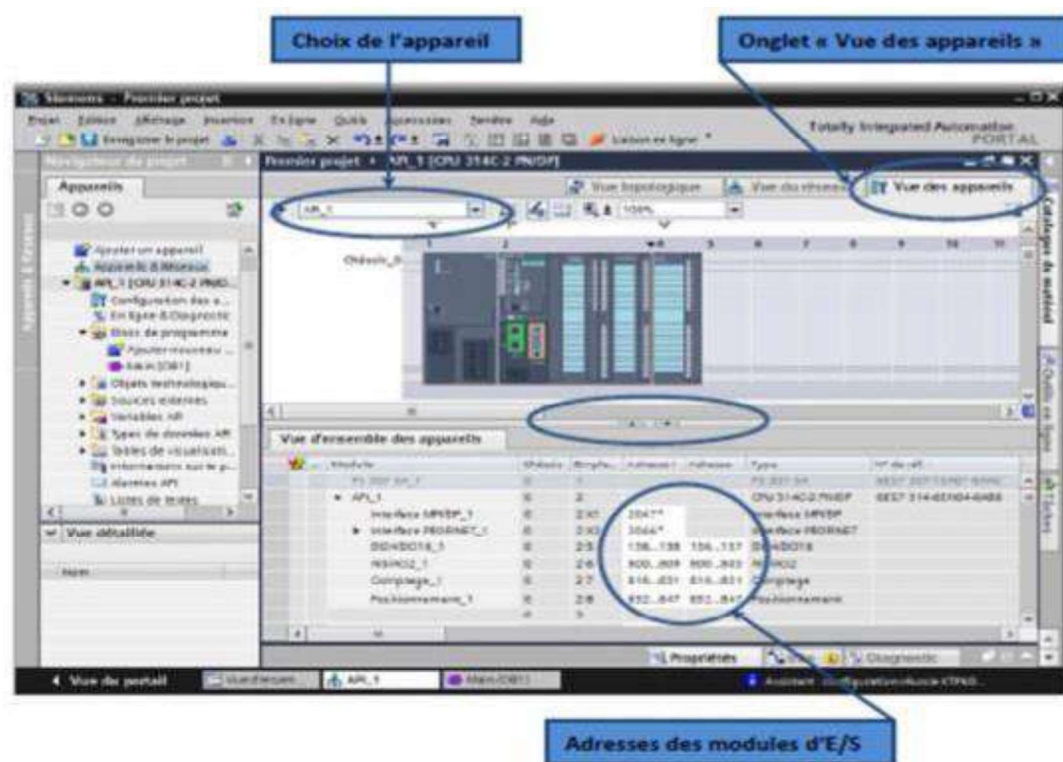


Figure 3.16: Adressage E/S

3.5.4.5. Adresse Ethernet de la CPU :

Toujours dans les propriétés de la CPU, il est possible de définir son adresse Ethernet. Un double clic sur le connecteur Ethernet de la station fait apparaître la fenêtre d'inspection permettant de définir ses propriétés.

Pour établir une liaison entre la CPU et la console de programmation, il faut affecter aux deux appareils des adresses appartenant au même réseau. On utilisera comme adresse pour l'automate 192.168.0.n° de l'automate.

3.5.4.6. Compilation et chargement de configuration matérielle :

Une fois la configuration matérielle réalisée, il faut la compiler et la charger dans l'automate. La compilation se fait à l'aide de l'icône « compiler » de la barre de tâche. On sélectionne l'API dans le projet puis cliquer sur l'icône « compiler ». Une fois utilisant cette manière, on effectue une compilation matérielle logicielle.

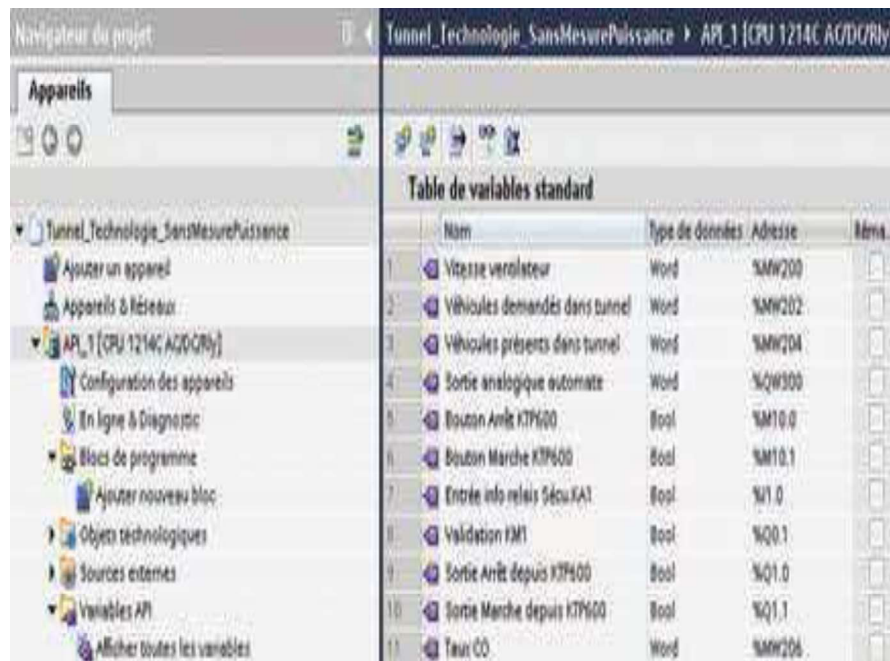
Une autre solution pour compiler est de faire un clic droit sur l'API dans la fenêtre du projet et de choisir l'option « compiler.... Configuration matérielle »

3.5.4.7. Programmation des API :

La programmation est l'utilisation d'un ensemble logique d'éléments et de constructions de langage de programmation nécessaires au traitement afin des signaux destinés à commander une machine ou un processus.

➤ La table de variables :

Dans le but de faciliter la programmation, il est nécessaire de créer une table de variables afin de présenter les différentes entrées/sorties (instrumentation) nécessaire pour le



	Nom	Type de données	Adresse	Remarques
1	Vitesse ventilateur	Word	%MW200	
2	Véhicules demandés dans tunnel	Word	%MW202	
3	Véhicules présents dans tunnel	Word	%MW204	
4	Sortie analogique automate	Word	%QW300	
5	Bouton Arrêt KTP600	Bool	%M10.0	
6	Bouton Marche KTP600	Bool	%M10.1	
7	Entrée info relais SécuXA1	Bool	%I1.0	
8	Validation FMI	Bool	%Q0.1	
9	Sortie Arrêt depuis KTP600	Bool	%Q1.0	
10	Sortie Marche depuis KTP600	Bool	%Q1.1	
11	Taux CO	Word	%MW206	

fonctionnement de notre système.

Figure 3.17: Table de variable d'API

➤ Bloc de programmation :

Blocs des programmes : l'automate met à disposition différents types de blocs qui contiennent les programmes et les données correspondants. Selon les exigences et la complexité du processus, il est possible de structurer le programme en différents blocs : OB, FB et FC.

- OB : bloc d'organisation : les OB sont appelés par le système d'exploitation en liaison avec l'événement d'exécution du programme.
- FC : fonction : ce sont des blocs de code sans mémoire.
- FB : bloc fonctionnel : Ce sont des blocs de code qui mémorisent durablement leurs paramètres d'entrées/sorties dans des blocs de données.
- DB : bloc de données : Les blocs de données du programme utilisateur qui contiennent des données utilisateur.



Figure 3.18: Fenêtre d'ajout de nouveau bloc

Exercice (01): Alarme de signalisation :

En cas d'un défaut l'alarme s'allume et après la réparation en remarque la signalisation de l'alarme (en utilisant mémoire de cadence) ce qui prouve qu'il ya un défaut est réparé, l'impulsion sur reset pour réinitialise l'alarme.

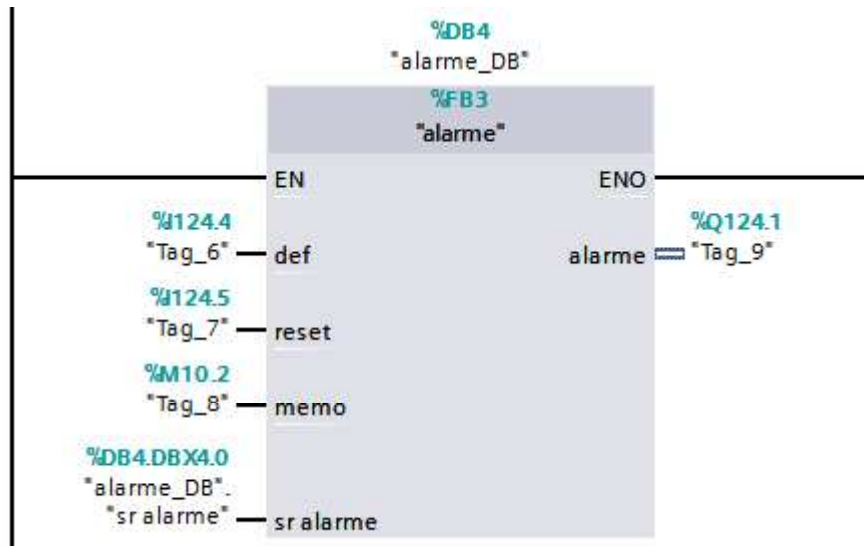


Figure 3.19: Schéma de l'Alarme dans le bloc d'organisation OB

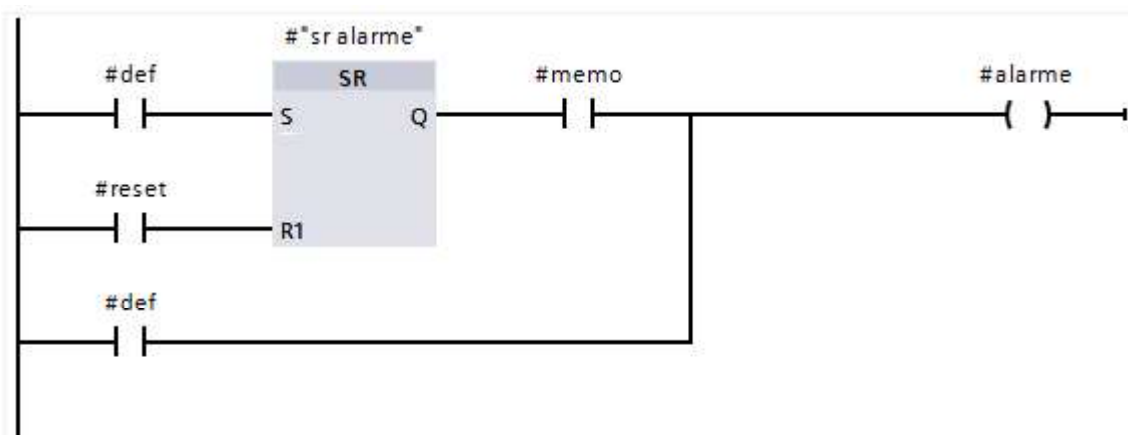


Figure 3.20: Schéma s'une alarme de signalisation sous logiciel TIAPORTAL

Exercices (2): Mode de fonctionnement Manuelle ou Automatique

Tous les systèmes automatisé ayant deux modes de fonctionnement qui sont le mode Manuelle et Automatique le choix entre les deux modes ce fait par un sélecteur.

Si le contact **aut-man** fermé avec l'impulsion sur le contact de validation **val** on à en mode automatique sinon en mode manuelle

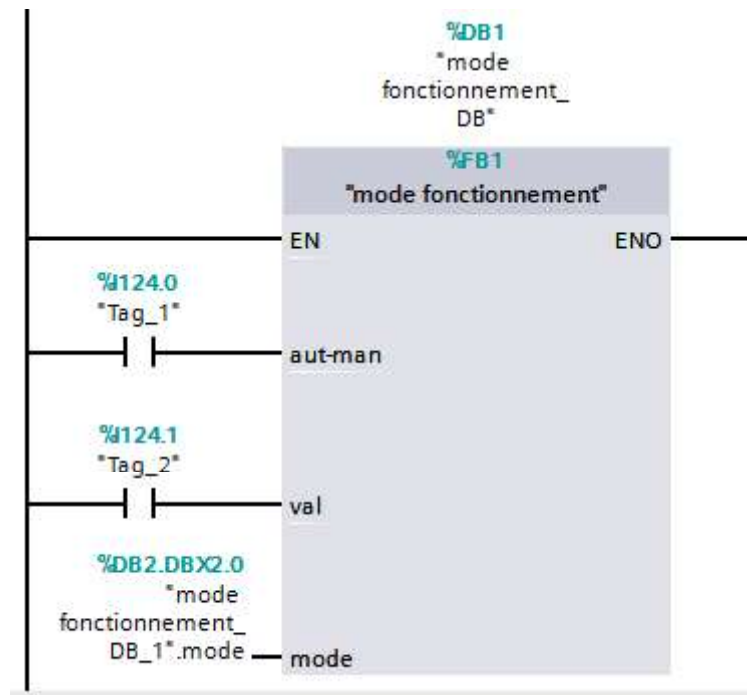


Figure 3.21: Schéma de mode de fonctionnement dans le bloc d'organisation OB

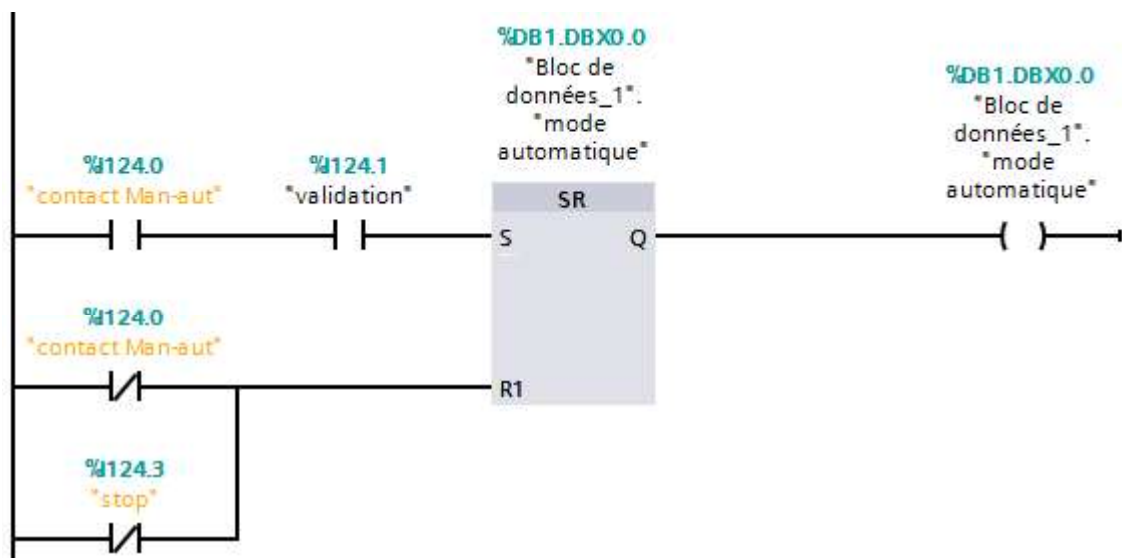


Figure 3.22: Mode de fonctionnement Automatique

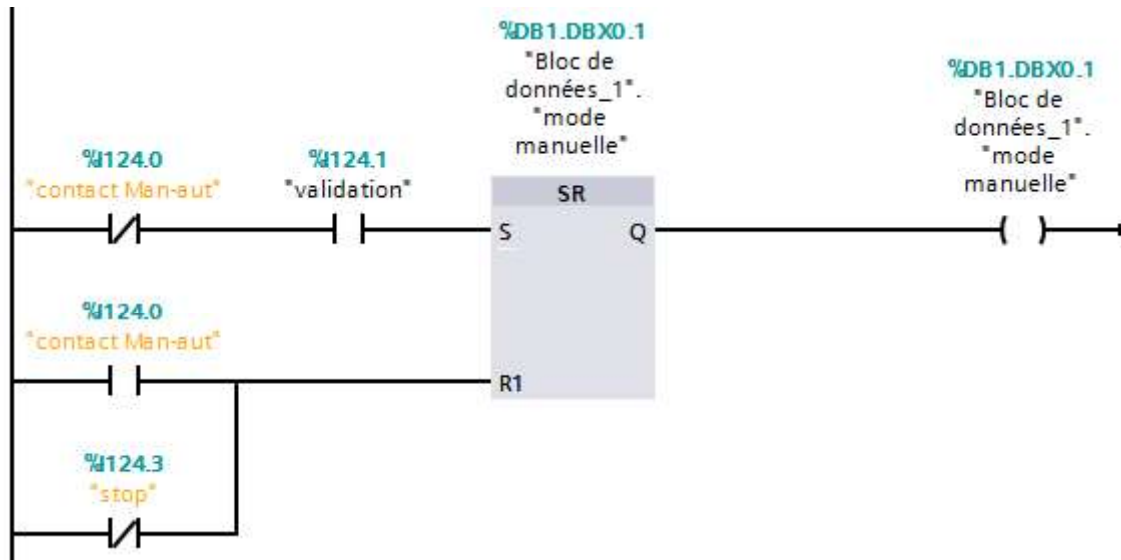


Figure 3.23: Mode de fonctionnement Manuelle

Schéma de mode fonctionnement sous logiciel TIAPORTAL dans le bloc fonctionnel **FB**

Exercice (3) Démarrage d'un moteur selon le mode de fonctionnement

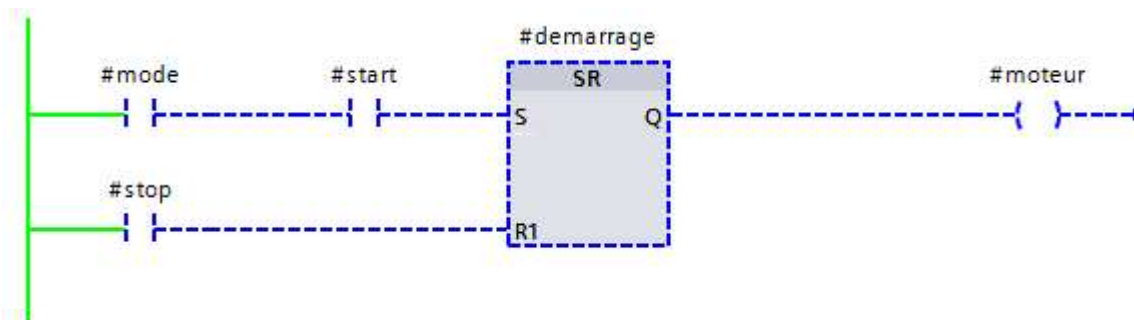


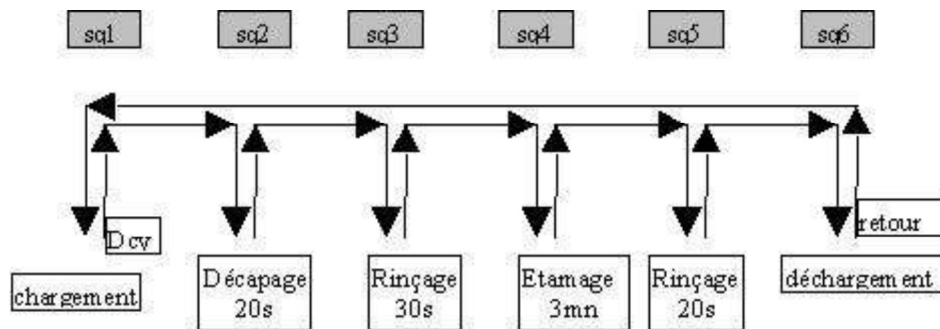
Figure 3.24: Schéma de Démarrage d'un moteur à un sens de rotation

3.4 Exercices

3.4.1 Traitement de surface

Une installation de traitement de surface comprend un chariot automoteur des servant 4 bacs, un poste de chargement et un poste de déchargement. Des capteurs _sq1_ à _sq6_ permettent le positionnement au-dessus des différents postes (voir schéma de simulation ci-dessous).

L'opérateur après avoir accroché les pièces à traiter sur le cadre situé au point de chargement, en position basse, donne l'ordre de départ cycle. Le chariot doit alors effectuer le cycle suivant :

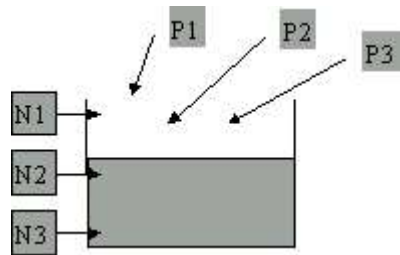


Au poste de déchargement l'opérateur décroche les pièces et renvoie le chariot avec l'ordre retour.

- Donner le GRAFCET de cet automatisme.

3.4.2 Commandes de pompes

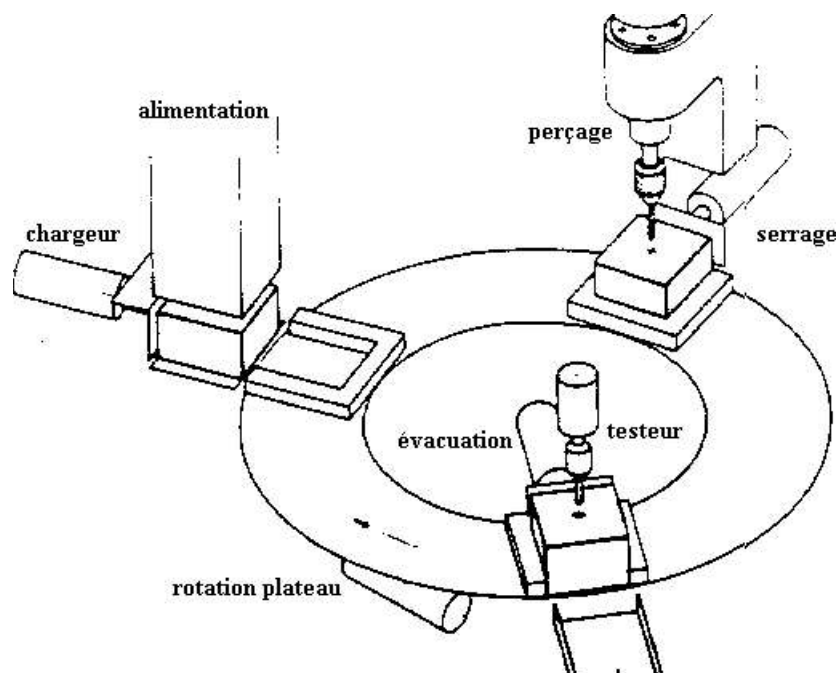
- Le niveau de liquide contenu dans un réservoir est contrôlé par trois détecteurs N1, N2, N3.
- L'alimentation de ce réservoir s'effectue par trois pompes P1, P2, P3 de la façon suivante :
 - si le niveau N1 est découvert (N1=0), une première pompe est mise en marche
 - si le niveau N2 est découvert (N1=0, N2=0), une deuxième pompe est mise en marche
 - si le niveau N3 est découvert (N1=0, N2=0, N3=0), la troisième pompe est mise en marche.



Le nombre de pompes en service sera égal au nombre de niveaux découverts. De plus afin d'équilibrer l'usure des pompes, celles-ci seront permutées à tour de rôle.

- Donner le GRAFCET de commande des pompes.

3.5.3 Poste de perçage



Un plateau tournant dessert 3 postes de travail :

- un premier poste de chargement
- un deuxième de perçage
- un troisième de contrôle et d'évacuation des pièces percées.

Un vérin permet la rotation de 120° du plateau supportant les pièces à usiner et son indexation, c'est à dire son blocage précis après chaque rotation.

Le contrôle du perçage s'effectue par un testeur qui doit descendre en position basse, si le trou est correctement percé. Si cela n'est pas réalisé, tout le système se bloque, testeur en position haute, de façon à ce que l'opérateur puisse enlever la pièce défectueuse avant de réarmer manuellement le système.

- Donner le GRAFCET décrivant le fonctionnement de cet automatisme.

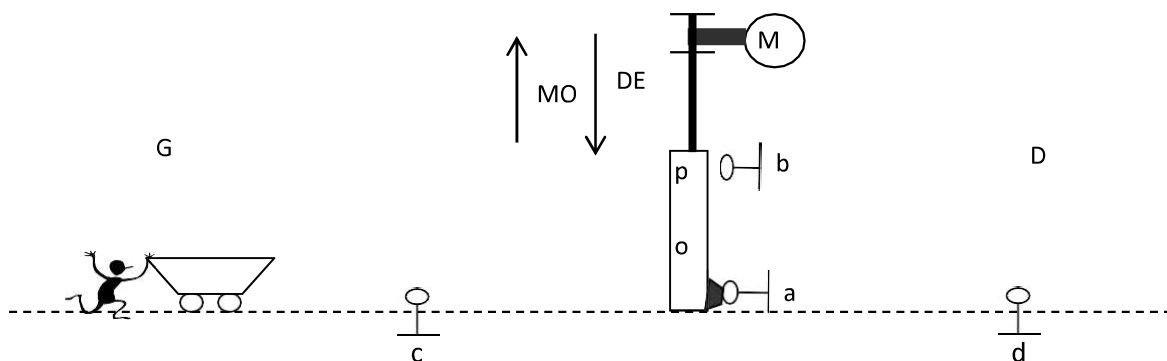
3.4.4 Commande d'ouverture de porte

Une porte soulevée par l'intermédiaire d'un groupe moto-réducteur M (à deux sens de rotation) permet le passage d'un chariot (un seul chariot existe) guidé sur rails et poussé par une personne. Le chariot doit avoir:

- Entrer à gauche (G) et sortir à droite(D)
- Entrer à droite (D) et sortir à gauche(G)
- Entrer à gauche (G) et sortir à gauche(G)
- Entrer à droite (D) et sortir à droite(D)

En venant de gauche (G), le chariot agit sur un contact c (impulsion) : la porte s'ouvre et le moteur effectue l'action MO. En venant de droite (D), le chariot agit sur un contact d (impulsion) : la porte s'ouvre également et le moteur s'effectue l'action MO.

En sortant le chariot agit sur c ou sur d ; la porte se ferme et le moteur effectue l'action DE. Les contacts a et b limitent les déplacements de la porte ; a=1 : porte en bas et b=1 : porte en haut



- Donner le Grafcet de ce système