République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed BOUDIAF - M'sila
Faculté de technologies
Département Génie Electrique

Module : Automatismes **Enseignant :** El Oualid. ZOUGGAR

Classe : Master 2 Maintenance Industrielle

Chapitre 05 : Architecture des APIs

2023/2024

Sommaire

Table des figures

I. CHAPITRE 05 : Architecture des APIs

I.1. Définition :

Un automate programmable est un appareil dédié au contrôle d'une machine ou d'un processus industriel, constitué de composants électroniques, comportant une mémoire programmable par un utilisateur non informaticien, à l'aide d'un langage adapté. En d'autres termes, un automate programmable est un calculateur logique, ou ordinateur, au jeu d'instructions volontairement réduit, destiné à la conduite et la surveillance en temps réel de processus industriels.

Trois caractérises fondamentales distinguent totalement l'Automate Programmable Industriel (API) des outils informatiques tels que les ordinateurs (PC industriel ou autres) :

- Il peut être directement connecté aux capteurs et pré-actionneurs grâce à ses entrées/sorties industrielles,
- Il est conçu pour fonctionner dans des ambiances industrielles sévères (température, vibrations, microcoupures de la tension d'alimentation, parasites, etc.),
- Et enfin, sa programmation à partir de langages spécialement développés pour le traitement de fonctions d'automatisme fait en sorte que sa mise en œuvre et son exploitation ne nécessitent aucune connaissance en informatique.

I.2. Architecture d'un API

La structure d'un API peut se représenter comme suit :

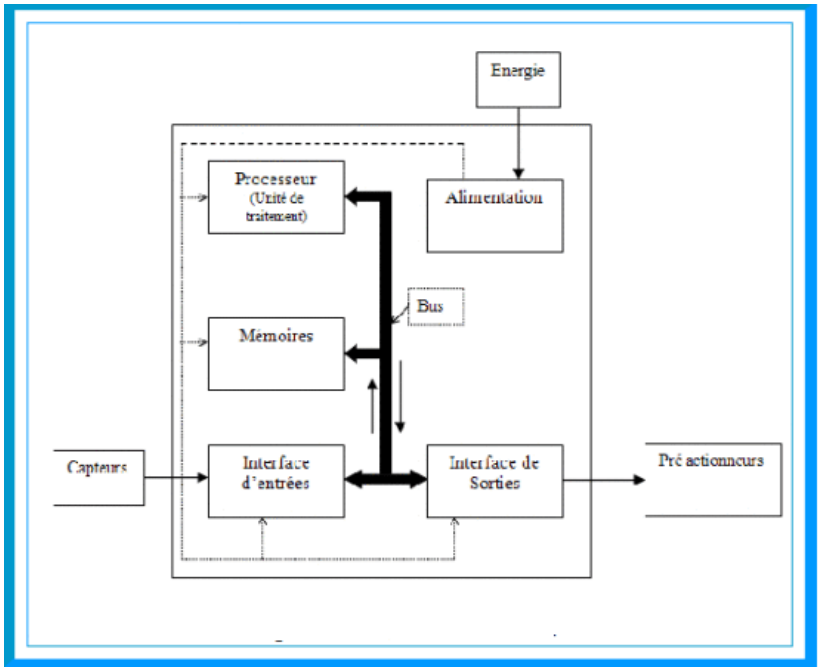

Figure 1: Architecture d'un API

L'automate programmable reçoit les informations relatives à l'état du système et puis commande les pré actionneurs suivant le programme inscrit dans sa mémoire.

Un API se compose donc de trois grandes parties :

- Le processeur ;
- La mémoire ;
- Les interfaces Entrées/sorties

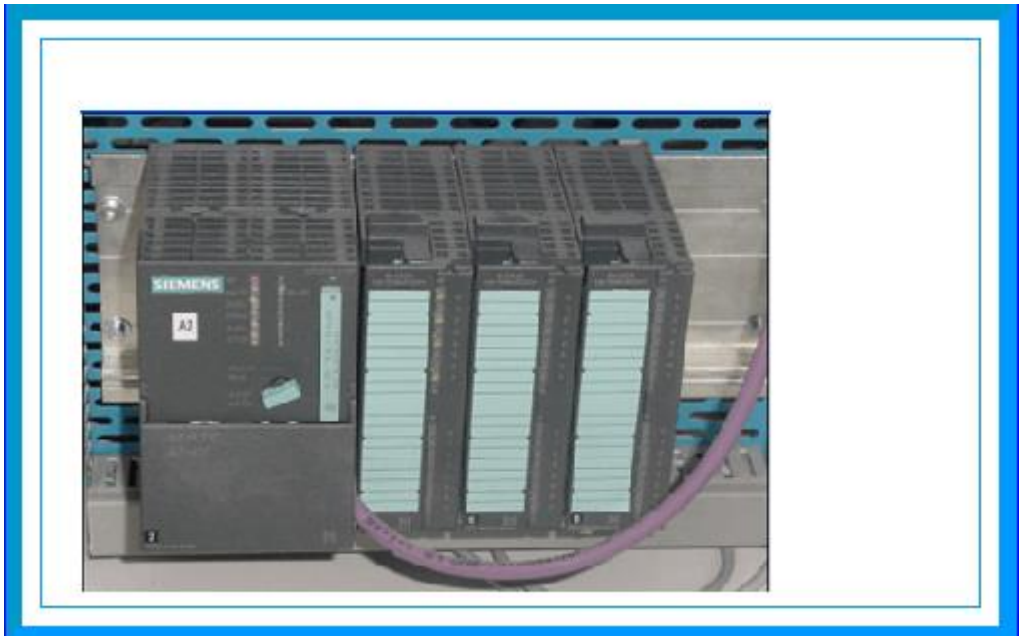

Figure 2: Exemple d'une architecture réelle d'un API S7-300 (marque de Siemens AG).

a. Le processeur

Le processeur, ou unité centrale (UC), a pour rôle principal le traitement des instructions qui constituent le programme de fonctionnement de l'application (les fonctions logiques ET, OU, les fonctions de temporisation, de comptage, de calcul PID, etc..). Mais en dehors de cette tâche de base, il réalise également d'autres fonctions :

- Gestion des entrées/sorties.
- Surveillance et diagnostic de l'automate par une série de tests lancés à la mise sous tension ou cycliquement en cours de fonctionnement.
- Dialogue avec le terminal de programmation, aussi bien pour l'écriture et la mise au point du programme qu'en cours d'exploitation pour des réglages ou des vérifications des données.

Un ou plusieurs processeurs exécutent ces fonctions grâce à un micro logiciel préprogrammé dans une mémoire de commande, ou mémoire système. Cette mémoire morte définit les fonctionnalités de l'automate. Elle n'est pas accessible à l'utilisateur.

b. La mémoire

Elle est destinée au stockage des instructions qui constituent le programme de fonctionnement de l'automatisme, ainsi que des données qui peuvent être :

- Des informations susceptibles d'évoluer en cours de fonctionnement de l'application. C'est le cas par exemple de résultats de traitements effectués par le processeur et rangés dans l'attente d'une utilisation ultérieure. Ces données sont appelées variables internes ou mots internes.
- Des informations qui n'évoluent pas au cours de fonctionnement, mais qui peuvent en cas de besoin être modifiées par l'utilisateur : textes à afficher, valeurs de présélection, etc.. Ce sont des mots constants.
- Les mémoires d'état des entrées/sorties, mises à jour par le processeur à chaque tour de scrutation du programme.

Deux familles de mémoires sont utilisées dans les automates programmables :

- Les mémoires vives, ou mémoires à accès aléatoire « Random Access Memory (RAM) ». Le contenu de ces mémoires peut être lu et modifié à volonté, mais il est perdu en cas de manque de tension (mémoire volatiles). Elles nécessitent par conséquent une sauvegarde par batterie. Les mémoires vives sont utilisées pour l'écriture et la mise au point du programme, et pour le stockage des données.
- Elles sont à lecture seule, les informations ne sont pas perdues lors de la coupure de l'alimentation des circuits. On peut citer les types suivants :

- **ROM « Read Only Memory » :** Elle est programmée par le constructeur et son programme ne peut être modifié.
- **PROM « Programmable ROM » :** Elle est livrée non enregistrée par le fabricant. Lorsque celle-ci est programmée, on ne peut pas l'effacer
- **EPROM « Erasable PROM » :** C'est une mémoire PROM effaçable par un rayonnement ultraviolet intense.
- **EEPROM « Electrically EPROM » :** C'est une mémoire PROM programmable plusieurs fois et effaçable électriquement.
- **Mémoire Flash :** C'est une mémoire EEPROM rapide en programmation. L'utilisateur peut effacer un bloc de cases ou toute la mémoire.

La mémoire morte est destinée à la mémorisation du programme après la phase de mise au point. La mémoire programme est contenue dans une ou plusieurs cartouches qui viennent s'insérer sur le module processeur ou sur un module d'extension mémoire.

c. Les interfaces entrées/sorties :

Les entrées/sorties TOR (Tout ou Rien) assurent l'intégration directe de l'automate dans son environnement industriel en réalisant la liaison entre le processeur et le processus. Elles ont toutes, de base, une double fonction :

- Une fonction d'interface pour la réception et la mise en forme de signaux provenant de l'extérieur (capteurs, boutons poussoirs, etc.) et pour l'émission de signaux vers l'extérieur (commande de pré-actionneurs, de voyants de signalisation, etc.). La conception de ces interfaces avec un isolement galvanique ou un découplage opto-électronique assure la protection de l'automate contre les signaux parasites.
- Une fonction de communication pour l'échange des signaux avec l'unité centrale par l'intermédiaire du bus d'entrées/sorties.

Le fonctionnement de l'interface d'entrée (figure 3) peut être résumé comme suit :

Lors de la fermeture du capteur ;

- La « **Led 1** » signale que l'entrée de l'API est actionnée.
- La « **Led D'** » de l'optocoupleur « **Opto 1** » s'éclaire.
- Le phototransistor « **T'** » de l'optocoupleur « **Opto 1** » devient passant.
- La tension Vs=0V.

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

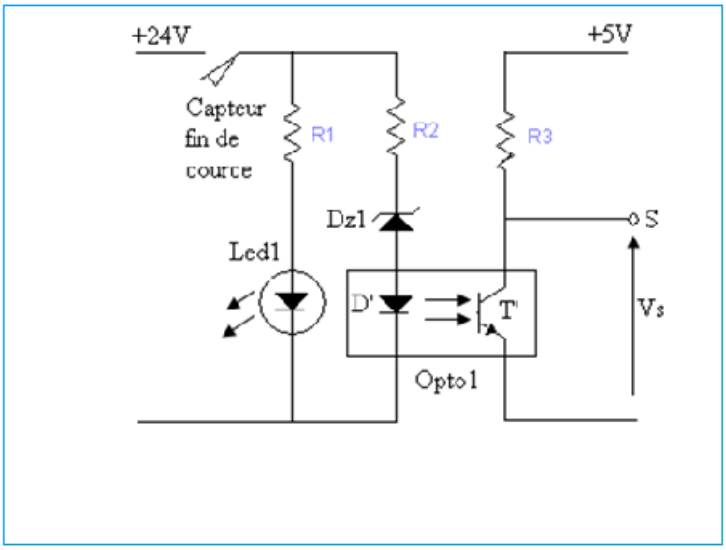

Figure 3: Principe de fonctionnement de l'interface d'entrée

Le fonctionnement de l'interface de sortie (figure 4) peut être résumé comme suit :

Lors de commande d'une sortie automate ;

- L'unité de commande envoi un « **1** » logique (5V).
- « **T1** » devient passant, donc la « **Led D'** » s'éclaire
- Le photo-transistor « **T'** » de l'optocoupleur « **Opto1** » devient passant.
- La « **Led1** » s'éclaire.
- « **T2** » devient passant.
- La bobine « **RL1** » devient sous tension et commande la fermeture du contact de la sortie « **Q0.1** ».

Donc pour commander un API, l'unité de commande doit envoyer :

- Un « **1** » logique pour actionner une sortie API
- Un « **0** » logique pour stopper la commande d'une sortie API

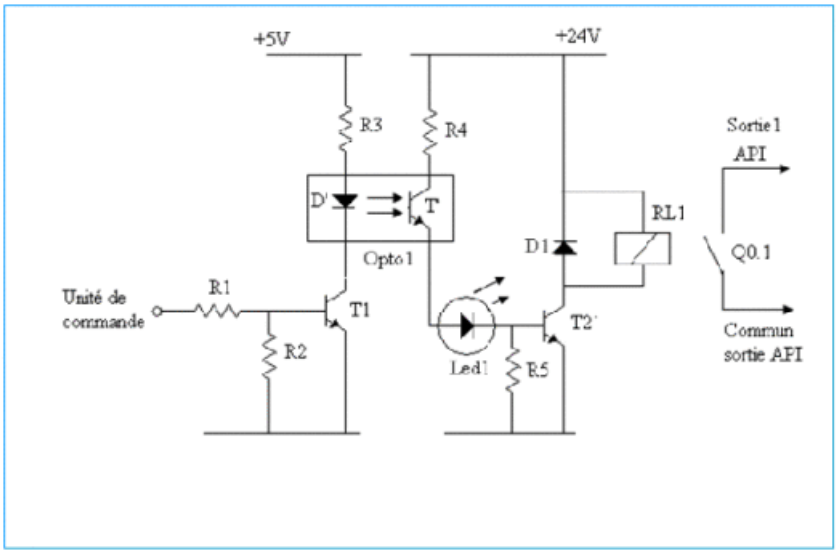

Figure 4: Principe de fonctionnement de l'interface de sortie

La figure 5 donne une idée concrète sur un module TOR industriel.

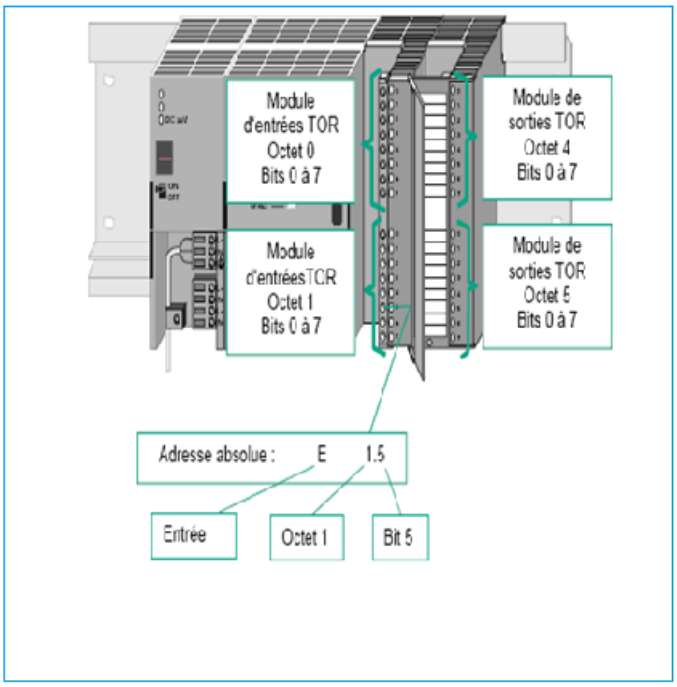

Figure 5: Schéma de principe d'un module TOR

d. Le Bus

C'est un ensemble de conducteurs qui réalisent la liaison entre les différents éléments de l'automate. Dans un automate modulaire, il se présente sous forme d'un circuit imprimé situé au fond du bac et supporte des connecteurs sur lesquels viennent s'enficher les différents modules : processeur, extension mémoire, interfaces et coupleurs.

Le bus est organisé en plusieurs sous ensembles destinés chacun à véhiculer un type bien défini d'informations :

- Bus de données.
- Bus d'adresses.
- Bus de contrôle pour les signaux de service tels que tops de synchronisation, sens des échanges, contrôle de validité des échanges, etc..
- Bus de distribution des tensions issues du bloc d'alimentation.

e. Alimentation

Elle élabore à partir d'un réseau 220V en courant alternatif, ou d'une source 24V en courant continu, les tensions internes distribuées aux modules de l'automate.

A fin d'assurer le niveau de sûreté requis, elle comporte des dispositifs de détection de baisse ou de coupure de la tension réseau, et de surveillance des tensions internes. En cas de défaut, ces dispositifs peuvent lancer une procédure prioritaire de sauvegarde.

Les schémas de principe et raccordement sont, respectivement, illustrés par la figure 5 et la figure 6.

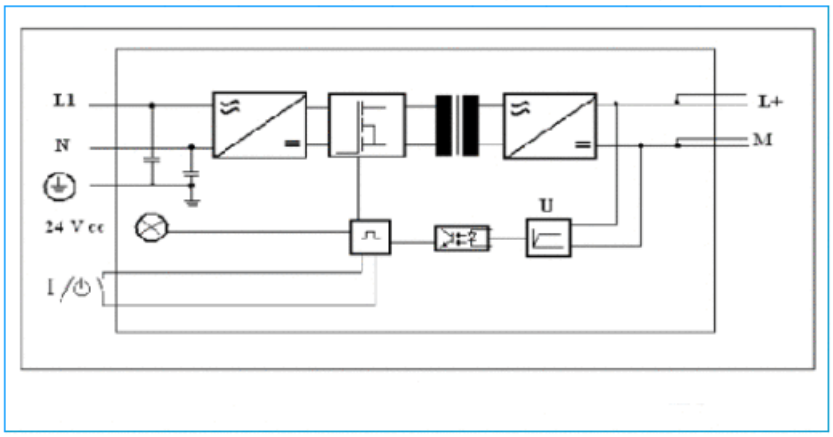

Figure 6: Schéma de principe d'une alimentation d'un API

I.3. Choix d'un API

Le choix d'un API est fonction de la partie commande à programmer. On doit tenir compte de plusieurs critères.

- Nombres d'entrées/sorties : le nombre de cartes peut avoir une incidence sur le nombre de racks dès que le nombre d'entrées / sorties nécessaires devient élevé.
- Type de processeur : la taille mémoire, la vitesse de traitement et les fonctions spéciales offertes par le processeur permettront le choix dans la gamme souvent très étendue.
- Nombre de compteurs.
- Nombre de temporisateurs
- Fonctions de communication : l'automate doit pouvoir communiquer avec les autres systèmes de commande (API, supervision ...) et offrir des possibilités de communication avec des standards normalisés (Profibus ...).

I.4. Principaux automates programmables industriels :

La programmation de ces automates se fait soit à partir de leur propre console, soit à partir du logiciel de programmation propre à la marque.

a. OMRON :

- **CQM1 – CPU 11/21/41**

- E - 192 Entrées/Sorties (à relais, à triac, à transistors ou TTL) ;
- 32 K RAM data on Board;
- Structure multifonction ;
- Structuration multitâche ;

- SYSWIN 3.1, 3.2 … 3.4 et CX Programmer (Littéral, Ladder) ;
- Communication sur RS 232 – C ;
- Programmation sur IBM PC/PS.

b. TELEMECANIQUE :

✻ **TSX 17/20 :**

- Nombre d'entrées et de sorties variable : 20 à 160 E/S.
- Microprocesseur 8031.
- Langage de programmation PL7.2.

✻ **TSX 67.20 :** La compacité d'un automate haut de gamme, à E/S déportables par fibre optique :

- 1024 E/S en six bacs de huit modules ;
- Extension de bacs à distance par fibre optique à 2000 m ;
- 16 coupleurs intelligents ;
- 24 K RAM data on Board;
- 32 K RAM / EPROM cartouche utilisateur ;
- Structure multifonctions ;
- Structuration multitâche ;
- Langage PL7.3 (Grafcet, Littéral, Ladder);
- Programmation sur IBM PC/PS.

c. FESTO :

Architecture modulaire : carte de base ; carte processeur ; carte de mémorisation ; carte E/S.

- FPC 202 :
- 16 entrées 24 V DC ;
- 16 sorties 24 V DC - 1 A ;
- 8 K RAM, 8 K EPROM ;
- Interface série, 20 mA boucle de courant pour imprimante ;
- Console de programmation externe : console ou IBM PC ;
- Programmation : grafcet, langage Festo, schéma à relais.

d. SIEMENS :

✻ **S7 – 200.**

- 64 entrées 24 V DC ;

- 64 sorties 24 V DC - 1 A ;
- 8 Entrées analogiques AEW0
- AEW14 ; - 8 Sorties analogiques AAW0
- AAW6 ; - interface série,
- Console de programmation externe : PG 702 ;
- Programmation STEP7 : schéma à relais, Ladder.