CENTRE UNIVERSITAIRE
BELHADJ BOUCHAÏB
D'AIN TEMOUCHENT

INSTITUT DES SCIENCES & DE LA TECHNOLOGIE
DEPARTEMENT DE GENIE ELECTRIQUE

Polycopié de Cours

AUTOMATISME

Automates Programmables
Fonctions Logiques
Programmation
GRAFCET

Dr. ZEBENTOUT ABDEL-DJAWAD BOUMEDIENE
(MAITRE DE CONFERENCES AU C.U.A.T)

Année Universitaire 2014 / 2015


بسم الله الرحمن الرحيم

Table des Matières

Avant-propos 3

CHAPITRE I

ARCHITECTURE DES AUTOMATES PROGRAMMABLES

I-1- Introduction 4
I-2- Description d'un système automatisé 4
I-3- Définition et type d'automate programmable 7
I-4- Les systèmes de commande 9
I-5- Structure et description des différents éléments d'un A.P 11
I-5-a- L'Unité Centrale 11
I-5-b- Les cartes d'entrées et de sorties 12
I-5-c- Les coupleurs 15
I-5-d- Les blocs d'alimentation 16
I-5-e- Les racks 16
I-5-f- Les consoles 17
I-6- Cycle de l'automate 17
I-7- Automate et environnement 18
I-7-a- Environnement physique et mécanique 18
I-7-b- Environnement chimique 18
I-7-c- Environnement électriques 19
I-8- Choix de l'automate programmable 19

CHAPITRE II

FONCTIONS LOGIQUES DE BASE

II-1- Introduction 20
II-2- Identification d'un opérateur logique 20
II-3- Fonctions logiques 22
II-3-a- Opérateur OUI ou opérateur Egalité 22
II-3-b- Opérateur NON (ou négation ou complémentation) 23
II-3-c- Opérateur ET ou produit logique - AND 24
II-3-d- Opérateur OU (inclusif) ou somme logique - OR 25
II-3-e- Opérateur non ET (négation du ET - NAND) 26
II-3-f- Opérateur non OU (négation du OU - NOR) 27

II-3-g- Opérateur OU exclusif (XOR) 28
II-3-h- Opérateur identité (XNOR) 29
II-4- Exemple pratique 29

CHAPITRE III

NOTION DE PROGRAMMATION DES AUTOMATES

III-1- Introduction – Notion d'algorithme 32
III-2- Expression algorithmique d'une équation logique 33
III-3- Expression algorithmique des évolutions d'un système automatisé 34
III-3-a- Structure algorithmique linéaire (structure séquentielle) 34
III-3-b- Structures algorithmique alternatives 36
III-3-c- Structures algorithmique itératives 38
III-4- Langage de programmation pour API 43

CHAPITRE IV

LE GRAFCET

IV-1- Introduction : 45
IV-2- Règles d'écriture du GRAFCET 45
IV-3- Règles d'évolution du GRAFCET 46
IV-4- GRAFCET à séquence unique 48
IV-4-a- Description du système 48
IV-4-b- Fonction globale 49
IV-4-c- GRAFCET du point de vue système 50
IV-4-d- GRAFCET du point de vue partie opérative 51
IV-4-e- GRAFCET du point de vue partie commande 51
IV-4-f- Exemple d'étude d'un transtainer 52
IV-5- GRAFCET à aiguillage exclusif 55
IV-5-a- Exemple de sélection de séquence 57
IV-5-b- Exemple de saut d'étapes 58
IV-5-c- Exemple de reprise de séquence 59
IV-6- Séquences simultanées 61
IV-7- Temporisation 63
IV-8- Comptage 64

BIBLIOGRAPHIE 67

WEBOGRAPHIE 69

Avant-propos

Les automates programmables (A.P) se sont introduits pratiquement dans tous les domaines et leurs applications sont multiples et diversifiées. On peut les trouver dans les maisons, dans les bureaux, dans les usines, quasiment partout. Ils permettent de faciliter la vie de l'être humain et d'améliorer et d'accroitre sa productivité, etc.

Ce polycopié de cours constitue une synthèse de plusieurs documents sur le domaine de l'automatisme. Il a été élaboré selon le programme du canevas de la Licence d'Automatique, domaine du Génie Electrique. Il permet aux étudiants qui suivent le système classique ou LMD, plus précisément ceux qui préparent une Licence en Automatique ou Commande Electrique, de découvrir et de comprendre le fonctionnement des systèmes automatisés et assimiler les notions de base relatives à ce domaine.

Ce cours comprend quatre chapitres :

- Le premier chapitre est une présentation des automates programmables ;
- Le deuxième chapitre définit les fonctions logiques de bases avec des exemples concrets ;
- Le troisième chapitre montre quelques concepts sur la programmation des automates ;
- Et le dernier chapitre met l'accent sur le GRAFCET qui est l'un des langages le plus utilisé.

Chapitre I

Architecture des automates programmables

I-1- Introduction :

Aujourd'hui, il serait difficile de concevoir un système de production sans avoir recours aux différentes technologies et composants qui forment les systèmes automatisés. On dit qu'un système est automatisé lorsqu'il peut gérer de manière autonome un cycle de travail préétabli qui se décompose en séquences et/ou étapes. Ces systèmes sont réalisés en vue d'apporter des solutions à des problèmes de nature économique, humaine, ou technique [1],[9]. Par exemple :

- Améliorer la productivité en asservissant la machine à des critères de productions, de rendement ou de qualité (capacité de production accélérée) ;
- Eliminer des tâches dangereuses et pénibles, en faisant exécuter par la machine les tâches humaines complexes ou indésirables (aptitude à convenir à tous les milieux de production) ;
- Piloter une production variable, en facilitant le passage d'une opération à une autre (souplesse d'utilisation).

Cependant, ils présentent un certain nombre d'inconvénients :

- Le coût élevé du matériel ;
- La maintenance doit être structurée ;
- La suppression d'emplois.

I-2- Description d'un système automatisé [1, 9] :

Tout système automatisé est constitué de 03 parties distinctes qui échangent des informations. Ces trois parties sont (figure I-1) :

- La partie opérative (PO) ;
- La partie commande (PC) ou système de contrôle / commande (SCC) ;
- La partie relation (PR) qu'on peut représenter par un pupitre de plus en plus intégrée dans la partie commande.

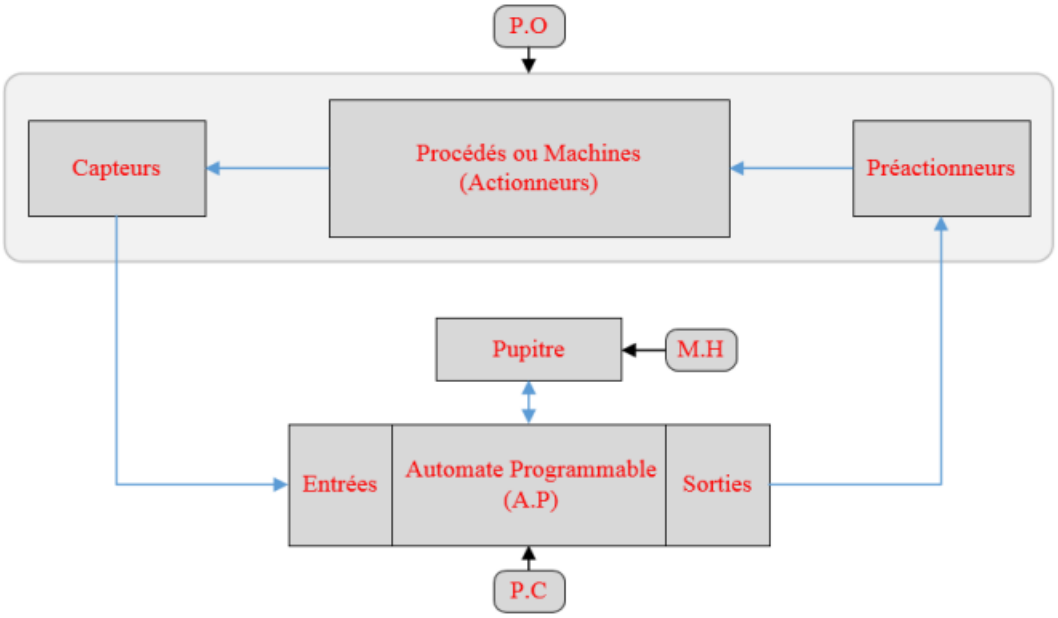


Figure I-1 : Schéma fonctionnel d'une installation automatisée [1]
(P.O : Partie Opérative ; P.C : Partie Commande ; M.H : Milieu Humain).

Le **pupitre** (fig. I-2) permet le dialogue entre l'opérateur et la partie commande. Sa complexité dépend de l'importance du système. Elle regroupe les différentes commandes nécessaires au bon fonctionnement du procédé, c'est-à-dire marche/arrêt, arrêt d'urgence, marche automatique, etc.. L'opérateur envoie des **consignes opérateur** (donne des ordres d'exécution à la partie commande) et reçoit des **informations visuelles** (élaborées par la partie commande), destinées à l'opérateur pour lui permettre de suivre la production.

Figure I-2 : Exemple de pupitre de commande [W1].

La **partie commande** (représentée par l'automate programmable " **cerveau** " du système automatisé) est chargée d'élaborer les **consignes opératives** à partir des consignes **de l'opérateur** à destination de la partie opérative en fonction de ses entrées (**comptes-rendus des capteurs / détecteurs** placés sur la partie opérative informent la partie commande de la bonne exécution des ordres donnés) et de règles de traitement (décrites par des équations logiques, un programme, un grafcet…).

La **partie opérative** agit sur la **matière d'œuvre** (assure la transformation, etc.) à partir des **consignes opératives** élaborées par la partie commande et génère des comptes rendus d'exécution à destination de celle-ci. Les **actionneurs** (vérin ou moteur par exemple) doivent être alimentés en énergie de puissance (électrique ou pneumatique) via des **préactionneurs** tels que distributeurs ou contacteurs électromagnétiques. Sur la figure I-3, on peut voir quelques composants de la partie opérative.

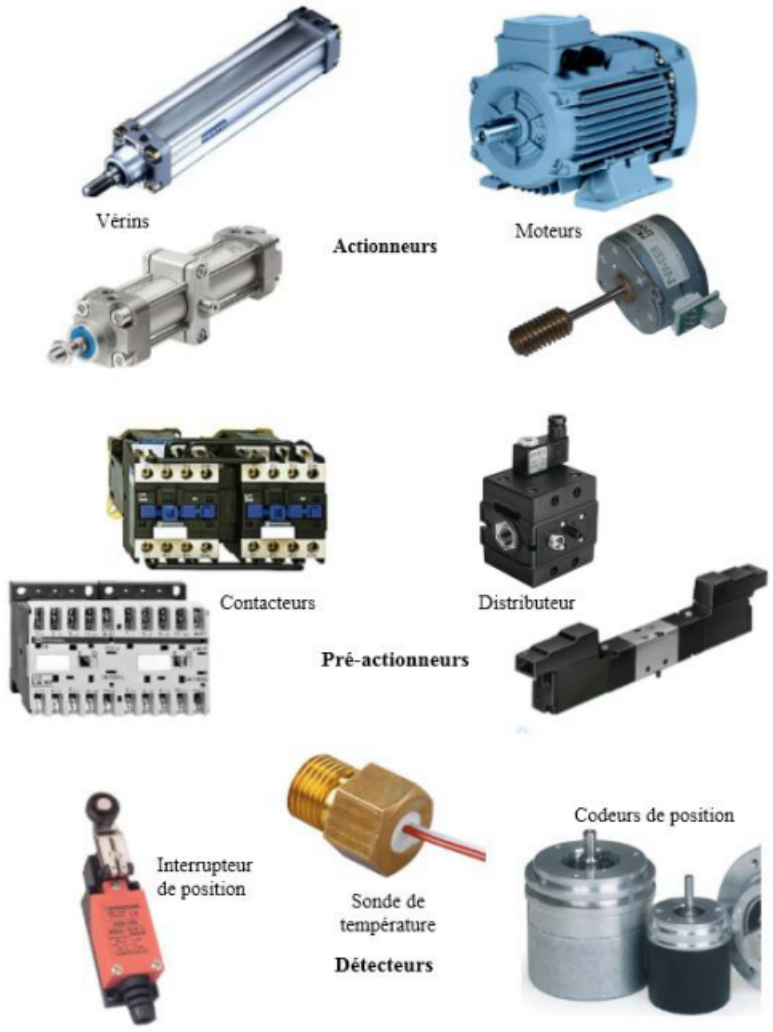


Figure I-3 : Quelques composants de la partie opérative ;
(Actionneurs, Pré-actionneurs, Détecteurs) [3].

La **matière d'œuvre** est le produit qui subit l'intervention du système. On distingue deux sortes de matière d'œuvre :

- La matière d'œuvre **entrante**, correspondant à la matière d'œuvre dans son état initial,
- La matière d'œuvre **sortante**, correspondant à la matière d'œuvre transformée.

Notons qu'après passage de la matière d'œuvre dans un système, les modifications des caractéristiques de cette dernière peuvent appartenir à l'une de ces trois familles :

- La transformation ;
- Le déplacement ;
- Le stockage.

Exemple : Pour une perceuse, la matière d'œuvre est le mur ou une pièce métallique (matière) et la modification apportée est le trou (transformation).

Dans ce cours, on s'intéressera surtout à la partie commande, c'est-à-dire l'automate programmable et à sa programmation.

I-3- Définition et type d'automate programmable [1, 3] :

Les A.P sont des **appareils électroniques** très perfectionnés qui reproduisent **sans intervention humaine**, à l'aide des outils dont ils sont équipés une **suite d'opérations** préalablement **enregistrée** par des opérateurs (utilisateur automaticien et non informaticien). Ils comportent une mémoire programmable à l'aide d'un langage adapté, pour le stockage interne des instructions composant les fonctions d'automatisme comme par exemple :

- Logique séquentielle et combinatoire ;
- Temporisation, comptage, décomptage, comparaison, etc.
- Calcul arithmétique ;
- Réglage, asservissement, etc.

Pour **commander**, **mesurer** et **contrôler** au moyen de modules d'entrées et de sorties (logique, numériques ou analogiques) différentes sortes de machines ou de processus, en environnement industriel.

L'automate trouve sa place dans les domaines les plus variés comme, dans les **chaînes de fabrication** (usinage, montage, etc.), pour les opérations de **manutention** (stockage, tri chargement, etc.) ou encore dans les **systèmes de contrôle** (installation de climatisation frigorifique, de chauffage, détection des incendies, industrie nucléaire, etc.).

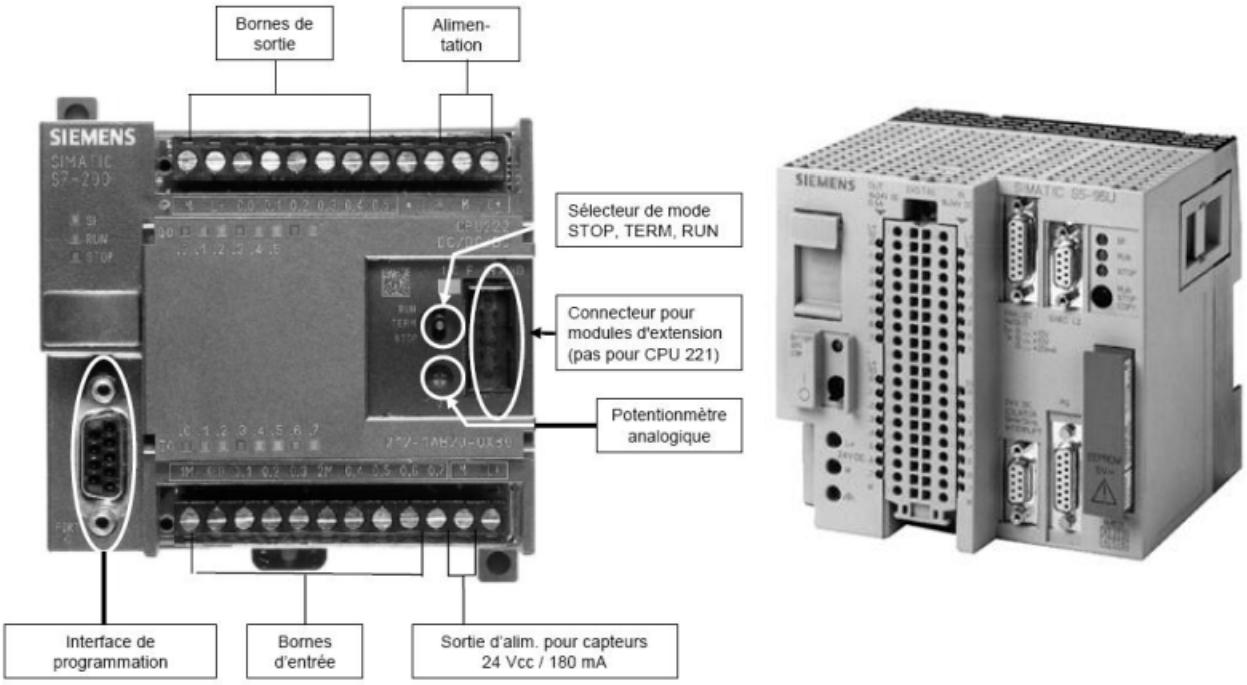
Exemple : Barrière de chemin de fer (signalisation sonore, lumineuse et action montée et descente de la barrière).

Il existe deux types d'API :

- Le type monobloc ;
- Le type modulaire.

a- Automate monobloc :

Il possède généralement un nombre d'entrées et de sorties restreint et son jeu d'instructions ne peut être augmenté (Il regroupe les principaux composants dans un espace réduit - Voir figure I-4). Il a pour fonction de résoudre des automatismes simples faisant appel à une logique séquentielle et utilisant des informations tout-ou-rien (TOR). Il faut savoir qu'il est parfois possible d'ajouter des extensions d'entrées / sorties.



(a) (b)

Figure I-4 : Automate monobloc

(a- Aspect extérieur d'un automate S7-200 CPU222 ; b- SIEMENS S5-95U) [W2].

b- Automate modulaire :

Il est adaptable à toutes situations. Selon le besoin, des modules d'entrées / sorties analogiques sont disponibles en plus de modules spécialisés tels : PID, BASIC et langage C, etc. Sa modularité permet un dépannage rapide et une plus grande flexibilité ce qui permet de le configuré individuellement selon les besoins (fig. I-5).

Figure I-5 : Automate modulaire [W2].

I-4- Les systèmes de commande :

Pour réaliser un système de commande, il existe 2 moyens :

- Les solutions câblées ;
- Les solutions programmées.

a- Les solutions câblées [1] :

Dans ce cas la mise en œuvre nécessite uniquement l'établissement de liaisons matérielles (câblages électriques) selon un schéma établi à partir de la théorie ou de l'expérience. C'est une technologie :

- Simple, connue et maîtrisée ;
- La conception, réalisation, mise en service et maintenance ne nécessitant pas une formation spécifique.

Cependant les solutions câblées comportent des contraintes :

- Le poids et le volume des composants (non négligeable) ;
- La rentabilité financière ;
- La complexité de l'installation ;
- Les risques d'erreurs de câblages ;
- La recherche de pannes pas facile.

Et toute modification dans le choix du fonctionnement de l'installation va entrainer :

- Une intervention dans le câblage (main d'œuvre) ;
- Une augmentation du nombre de relais et de fils (redimensionnés l'armoire) ;
- Un coût élevé (main d'œuvre, relais, fils, etc.).

b- Les solutions programmées :

Dans ce cas l'élément principal s'appelle l'Automate Programmable Industriel (API). La détection est électrique. Le pilotage des actionneurs se fait par l'intermédiaire de relais ou de distributeur. Les relais auxiliaires, les relais temporisés, les pendules et toute la filerie assurant les liaisons entre ces différents éléments sont éliminées. Ainsi :

- L'encombrement se trouve réduit et la recherche de panne est facilitée ;
- La main d'œuvre réduite lors du câblage ;
- Terminal de programmation pouvant être commun à plusieurs automates ;
- Modification possibles sans intervention sur le câblage (à partir d'un terminal de programmation).

Les inconvénients de cette solution sont :

- L'utilisation d'un personnel formé à cette technologie ;
- Lorsque le fonctionnement est simple, le coût de réalisation reste élevé et le gain de place dans l'armoire électrique reste minime.

b- La mémoire : La mémoire centrale est découpée en quatre zones où l'on trouve :

- La zone mémoire programme (programme à exécuter) ;
- La zone mémoire de données (état des entrées et des sorties, valeurs des compteurs, temporisations) ;
- Une zone où sont stockés des résultats de calcul utilisés ultérieurement dans le programme ;
- Une zone pour les variables internes.

Il existe différents types de mémoires :

- **Les mémoires vives** ou RAM (Random Access Memory pour mémoire à accès aléatoires) : Ce sont des mémoires volatiles. Elles acceptent la lecture, l'écriture, les modifications, l'effacement, de façon illimitée. Cependant, elles perdent les informations en cas de coupure de l'alimentation. Pour éviter ces problèmes, certaines de ces mémoires, sont équipés de batteries qui sauvegardent les informations dans la limite de leur autonomie.
- **Les mémoires mortes :** Le contenu de ces mémoires est figé. Ce sont des mémoires à lecture. Les informations sont conservées en permanence sans source auxiliaire.

On distingue :

- **ROM :** Read Only Memory : ce sont des mémoires programmées par le fabricant et ineffaçables ;
- **PROM :** Programmable ROM : elles sont vendues vierges et se programment à partir d'un programmateur de PROM.
- **EPROM :** Erasable PROM ces mémoires sont utilisables plusieurs fois (Ecriture / Effacement). L'effacement s'effectue par rayon Ultra-violet (UV) pendant 10 à 30 mn et peuvent être reprogrammée qu'après un effacement total.
- **EEPROM :** Electrical EPROM Mémoire à effacement électrique.

I-5-b- Les cartes d'entrées et de sorties [1] :

Les modules d'entrées / sorties sont les interfaces qui permettent de communiquer avec le microprocesseur. Les cartes d'entrées reçoivent les données machines provenant des capteurs, elles permettent d'isoler électroniquement le circuit externe (entrée d'information) du circuit

de traitement et les cartes de sorties permettent d'appliquer les processus de commande par le biais de sorties TOR (contacteurs, moteurs pas à pas, électrovannes, etc.) ou sorties analogiques (boucle de régulation, débit, température et variateur de vitesse). Elles sont toutes équipées de connecteurs de raccordement permettant un montage et démontage rapide.

a- Les entrées / sorties TOR (Tout Ou Rien) :

Les automates programmables offrent une grande variété d'entrées / sortie TOR adaptées aux milieux auxquels ils sont soumis. Seuls 2 niveaux logiques sont possibles (0 ou 1). Ces entrées / sorties peuvent accepter suivant les cartes des informations en courant ou tension, alternatifs ou continus.

- **Cartes d'entrées TOR :** elles permettent le **raccordement** de l'AP aux différents **capteurs** logiques comme :
  - Boutons poussoirs ;
  - Capteurs de proximité inductifs ou capacitifs ;
  - Capteurs photo-électrique ;
  - Fibre optique ;
  - Thermostats ;
  - Fins de course ;
  - Roues codeuses ;
  - etc.

Elles assurent l'adaptation, l'isolement, le filtrage et la mise en forme des signaux électriques (capteurs vers Bus – voir fig. I-7). Une LED (Diode Electroluminescente) située sur la carte donne l'état de chaque entrée.

Le nombre d'entrées sur une carte est de 4, 8, 16, 32. Les tensions d'entrées et de sorties sont de 24, 48, 110, 220 Volts en courant continu ou alternatif [1].

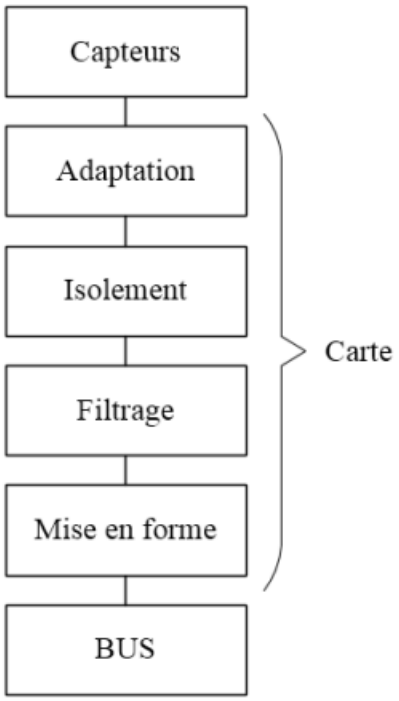


Figure I-7 : Evolution du signal d'une voie d'entrée.

- **Cartes de sorties TOR :** elles permettent le **raccordement** de l'AP aux **préactionneurs** comme :
  - Les vannes ;
  - Contacteurs ;
  - électrovannes ;
  - afficheurs et voyants
  - Relais de puissance ;
  - etc.

Les tensions de sorties usuelles sont de 5V en continue ou 24, 48, 110, 220 V en continu ou alternatif. Les courants vont de quelques milliampères à quelques ampères.

Ces cartes possèdent soit des relais, soit des triacs, soit des transistors. L'état de chaque sortie est visualisé par une LED. Notons, qu'il faut aussi prendre en considération la puissance des appareils commandés. En effet, la commande de moteurs, de résistances nécessitent des contacteurs.

En règle générale, les sorties statiques les plus utilisées sont :

- Les sorties à transistor pour les commandes en tension continue ;
- Les sorties à triac pour les commandes en tension alternative.

b- Les entrées / sorties analogiques :

Elles permettent de gérer des grandeurs analogiques en faisant varier un code numérique au sein du module.

- **Cartes d'entrées analogiques :**

Il existe trois types de cartes d'entrées analogiques :

- Haut niveau qui accepte en tension 0/10 V et en intensité 0/20 mA ou 4/20 mA.
- Pour thermocouple avec un signal d'entrée 0/20 mV, 0/50 mV, 0/100 mV.
- Pour sonde Pt 100 avec un signal d'entrée 0/100 mV, 0/250 mV, 0/400 mV.

Sur le marché, il existe des modules à 2, 4, 8 voies d'entrées. Ils disposent d'un convertisseur analogique / numérique.

- **Cartes de sorties analogiques :**

Il existe deux grands types de cartes de sorties :

- Haut niveau avec une résolution de 8 bits en tension 0 / 10 V ou en intensité, 0/20 mA ou 4/20 mA.
- Haut niveau avec une résolution de 12 bits en tension 0/10 V ; 0/5 V, ± 5 V, ± 10 V ou en intensité 0/20 mA ou 4/20 mA.

Ces modules assurent la conversion numérique / analogique. Les sorties analogiques peuvent posséder un convertisseur par voies. Le nombre de voies sur ces cartes est de 2 ou 4.

I-5-c- Les coupleurs :

Ce sont des cartes électroniques qui assurent la communication entre l'unité centrale et les entrées/sorties des périphériques ou autres.

L'échange d'information s'effectue par l'intermédiaire de Bus interne (liaison parallèle codée entre l'U.C et les modules d'entrées/sorties) et par des bus externe (liaison parallèle ou série entre l'U.C et les périphériques de l'automate).

Il existe deux types :

- Les coupleurs liaison série asynchrone,
- Les coupleurs liaison entre automates programmables.

a- Les coupleurs liaison série asynchrone assurent la communication avec des périphériques (imprimante, clavier / écran, console de dialogue, etc.) se situant à des distances plus ou moins éloignées (≈ 1500 m).

b- Les coupleurs entre automates programmables sont utilisés lors de la répartition des tâches entre plusieurs automates pour la gestion d'un processus complexe. Les informations sont alors transmises de coupleur à coupleur par l'intermédiaire d'un bus.

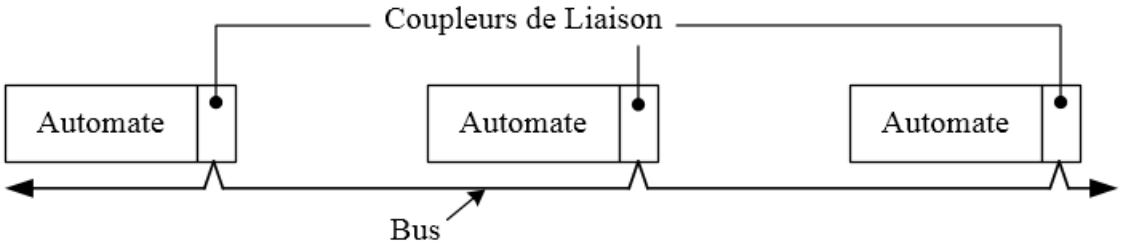


Figure I-8 : Liaison entre automates.

I-5-d- Les blocs d'alimentation :

Ils permettent de fournir à l'automate l'énergie nécessaire à son fonctionnement. A partir d'une alimentation en 220 V alternatif, ces blocs délivrent des sources de tension dont l'automate a besoin : 24, 12 ou 5 V (continu). En règle générale, un voyant sur la façade indique la mise sous tension de l'automate.

I-5-e- Les racks :

Ils sont installés en général à l'intérieur des armoires électriques et sont constitués d'une structure métallique et reçoivent les éléments tels que :

- L'Unité Centrale,
- Les cartes d'entrées / sorties,
- L'alimentation,
- Etc.

En définitif, tous les éléments s'encastrent dans un rack. On peut utiliser plusieurs racks en fonction du nombre d'entrée et de sortie.

I-5-f- Les consoles :

Ce sont des périphériques qui permettent la communication directe avec l'U.C de l'automate.

Ils permettent :

- Le paramétrage (modification des valeurs ou du cycle) ;
- Relevés d'informations (visualisation) ;
- La programmation (écriture, modification et effacement d'instruction, transfert d'un programme dans la mémoire de l'automate) ;
- Le réglage et l'exploitation (exécuter le programme pas à pas, de la visualiser, etc.).

Les consoles de programmation (terminal de programmation – fig. I-9) sont équipées d'un écran à cristaux liquides, touches de fonctions, clavier numérique, clavier alphabétique, etc.). Certaines consoles ne peuvent être utilisées que connectées à un automate, (l'automate programmable fournit l'alimentation à la console), d'autres peuvent fonctionner de manière autonome grâce à leur mémoire interne et à leur alimentation.

Figure I-9 : Console de programmation
Télémécanique Schneider TFTX 1170 [W3].

I-6- Cycle de l'automate [1] :

Le programme se trouve dans la mémoire. Il peut être constitué par exemple :

- D'une phase d'acquisition des entrées ;
- D'une phase de traitement ;
- D'une phase d'activation de sorties.

Ce qui constitue un des cycles de l'automate (cycle exécuté d'un bout à l'autre sans saut de programme).

Suivant la conception de l'automate ou de la programmation, il existe 03 cycles possibles :

- Toutes les entrées sont acquises au début du cycle, et les sorties se font après chaque résolution d'équation dans les programmes.
- Les équations sont traitées une par une en prenant uniquement la valeur des entrées concernées. Une entrée peut être appelée plusieurs fois avec des valeurs différentes pour plusieurs équations au cours du même cycle.
- Les entrées sont prises à des intervalles réguliers, le traitement s'effectue et les sorties activées à la demande (horloge interne).

Si le cycle n'est pas effectué dans une période définie par l'automate (boucle infinie involontaire) une alarme est déclenchée par l'intermédiaire du « chien de garde » (temporisation exécutée au début de chaque cycle).

Chien de garde : lorsque le programme de l'automate est en exécution, ce dispositif veille au bon déroulement du programme à chaque cycle de scrutation.

I-7- Automate et environnement [1] :

Les automates sont soumis à différents types d'environnement :

I-7-a- Environnement physique et mécanique :

Caractérisés par les vibrations, les chocs, l'humidité et la température.

- Lorsque les températures sont élevées (prés des fours, réacteurs, etc.) il faut prévoir des systèmes de ventilation forcée.
- L'humidité provoque des condensations et accélère la corrosion.
- Les vibrations et chocs agissent sur les contacts, les soudures provoquant une rupture du circuit.

I-7-b- Environnement chimique :

Les gaz corrosifs (cl_2 di chlore, H_2S sulfure d'hydrogène, SO_2 oxyde de soufre), les vapeurs d'hydrocarbures (HC), les poussières métalliques (fonderies, etc.) poussières minérales (cimenteries, etc.) entraînent une corrosion qui endommage les contacts et provoque des courts-circuits.

Pour protéger les automates les constructeurs utilisent :

- Un enduit pour recouvrir les circuits imprimés,
- Installer des filtres pour éliminer les poussières et gaz,
- Proposent des appareils totalement étanches.

I-7-c- Environnement électriques :

Les éléments perturbateurs sont :

- Les parasites d'origine électrostatiques ;
- Interférences électromagnétiques (transformateur, etc.) ;
- Certains émetteurs – récepteurs à des fréquences correspondant à l'A.P peuvent détériorer le processeur de celui-ci.

I-8- Choix de l'automate programmable :

A l'utilisateur d'établir le cahier des charges de son système, et d'utiliser l'automate le mieux adapté aux besoins, en considérant un certain nombre de critères importants :

- Le nombre d'entrées / sorties ;
- La nature des entrées (valeurs ohmiques, courant, etc.) ;
- La nature des sorties (triac, relais, etc.) ;
- La nature du traitement (temporisation, comptage, etc.) ;
- Le dialogue (la console détermine le langage de programmation) ;
- La communication avec d'autres systèmes, (automatisme complexe, utilisation de plusieurs automates) ;
- La fiabilité et la robustesse ;
- L'immunité aux parasites (blindage) ;
- Le service après-vente et la durée de garantie ;
- La formation ;
- Le coût.

Chapitre II

Fonctions logiques de base

II-1- Introduction :

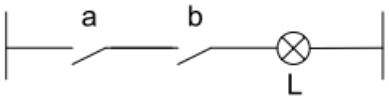
Ce chapitre [3, 9] constitue un bref rappel des notions de base sur les fonctions logiques assimilé au domaine de l'automatisme (schéma à contacts). Une fonction logique est le résultat d'une opération logique effectuée par un opérateur logique. Les opérateurs logiques constituent les blocs élémentaires des circuits logiques. Ces circuits permettent de réaliser des systèmes automatisés combinatoire. Généralement, ces derniers n'utilise aucun mécanisme de mémorisation : à une combinaison des entrées ne correspond qu'une seule combinaison des sorties. La logique associée est la logique combinatoire. Les outils utilisés pour les concevoir sont l'algèbre de Boole, les tables de vérité, etc.

II-2- Identification d'un opérateur logique :

Un opérateur logique peut se définir :

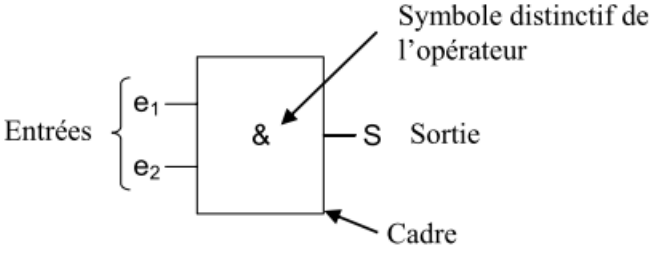
- Par la **description logique** de la fonction qu'il réalise,
- Par un **schéma à contacts** (schéma électrique) dans lequel chaque contact concrétise, par ses 02 positions, les 02 états d'une variable d'entrée.

Exemple :



Les deux contacts sont des variables d'entrées, la lampe est la variable de sortie.

- Par un **symbole logique** qui est une représentation normalisée de l'opérateur.



- Par une **table de vérité** qui indique toutes les relations nécessaires ou possibles entre les états logiques des entrées et de la sortie.
  Si n définit le nombre de variables d'entrée, la table de vérité comportera 2^n combinaisons différentes.

Variables d'entrées		Variable de sortie
e_2	**e_1**	**S**
0	0	1
0	1	0
1	0	0
1	1	0

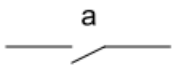
Combinaisons des états écrits suivant la numération binaire

Etat correspondant de la sortie

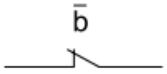
- Par une **équation logique** qui représente une identité d'états et qui permet d'exprimer en utilisant les symboles mathématique de l'algèbre de Boole la relation entre les variables d'entrées et de sorties.
- Par un **chronogramme** qui est une représentation graphique qui permet de visualiser, en fonction du temps, l'état de la sortie correspondant aux différentes combinaisons d'états logiques des entrées.

Remarque : par convention la représentation des contacts est décrite comme suit :

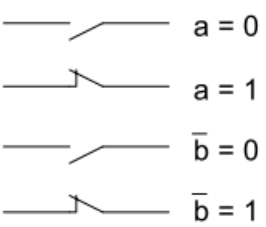
- Un contact ouvert au repos est désigné par une lettre, exemple : a



- Un contact fermé au repos est désigné par une lettre avec au-dessus une barre, exemple : $\bar{b}$



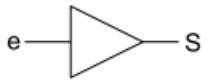
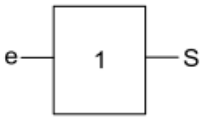
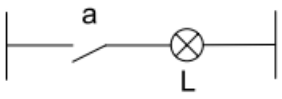
- Cas possible :

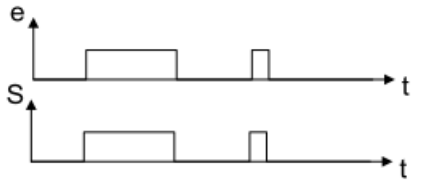


II-3- Fonctions logiques :

II-3-a- Opérateur OUI ou opérateur Egalité :

Il permet d'obtenir la sortie égale à 1 lorsque la variable d'entrée est égale à 1 (S est identique à e).

<table>
<tr><th>Symbole Logique (Américain)</th><th>Symbole Logique (AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td><table><tr><th>e</th><th>S</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>Egalité d'état entre e et S</td>
<td>$S = e$
(S égal e)</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est allumée si a est actionné</td>
<td colspan="2"> 
</td>
</tr>
</table>

Exemple : Commande d'une sonnette électrique à partir d'un bouton poussoir (Fig. II-1).

Le fonctionnement de ce circuit peut être représenté par un opérateur **OUI**. La sonnette **H** est à l'état **1** si, et seulement si, le bouton poussoir **a** est à l'état **1**.

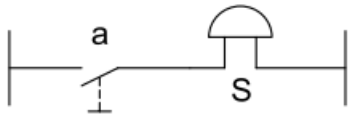
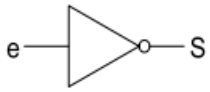
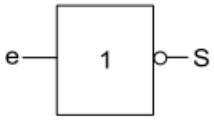
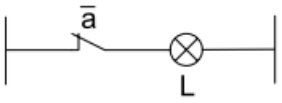
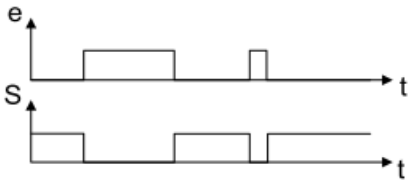


Figure II-1 : Circuit de la sonnette.

II-3-b- Opérateur NON (ou négation ou complémentation) :

Il permet d'obtenir le complément de l'état d'une variable d'entrée en sortie. La sortie est à l'état 1 si l'entrée est à l'état 0 (S est le complément de e).

<table>
<tr><th>Symbole Logique
(Américain)</th><th>Symbole Logique
(AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td><table><tr><th>e</th><th>S</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>Complémentation</td>
<td>$S = \bar{e}$
(S égal e barre)</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est allumée si $\bar{a}$ n'est pas actionné</td>
<td colspan="2"> 
</td>
</tr>
</table>

Exemple : Eclairage intérieur d'un réfrigérateur, dès que la porte s'ouvre (fig. II-2).

Le fonctionnement de ce circuit peut être représenté par un opérateur **NON**. La lampe **L** est à l'état **1** si, et seulement si, il n'y a pas d'action sur la variable $\bar{\mathbf{a}}$ (contrôle de la fermeture de la porte).

Lorsque la porte s'ouvre, plus d'action sur $\bar{\mathbf{a}}$, la lampe **L** s'allume.

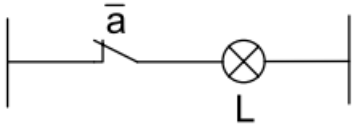
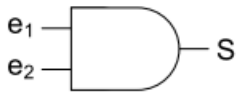
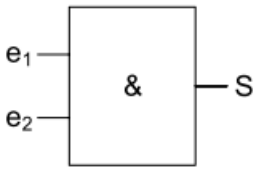
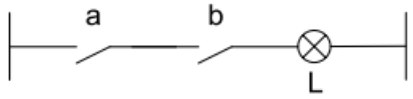
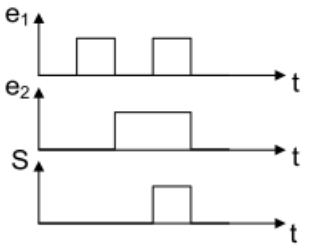


Figure II-2 : Circuit d'éclairage d'un réfrigérateur.

II-3-c- Opérateur ET ou produit logique - AND :

Il permet de traiter plusieurs variables d'entrées. Il faut que toutes les entrées soient égales à 1 pour avoir la sortie égale à 1 (un « zéro » en entrée force un « zéro » en sortie).

<table>
<tr><th>Symbole Logique
(Américain)</th><th>Symbole Logique
(AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td>
<table>
<tr><th>e2</th><th>e1</th><th>S</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
Seule la dernière combinaison affecte à S l'état 1
</td>
<td>$S = e_1 \cdot e_2$
(S égal e_1 ET e_2)</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est allumée si a et b sont actionnés</td>
<td colspan="2"> 
</td>
</tr>
</table>

Exemple : Mise en marche d'un aspirateur. Cette mise en marche ne peut se faire sans que le sac à poussière soit en place (un détecteur en contrôle la présence) (fig. II-3).

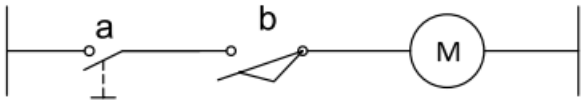
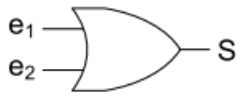
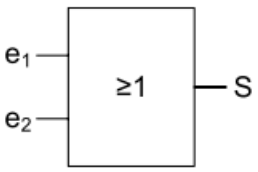
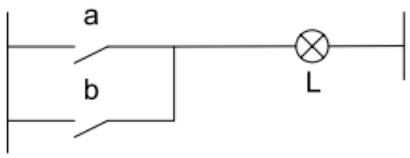


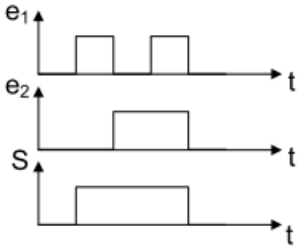
Figure II-3 : Circuit du moteur correspondant à l'aspirateur.

Le fonctionnement de ce circuit peut être représenté par un opérateur **ET**. Le moteur **M** est à l'état 1 si la variable **a** (commande " Marche ") **ET** la variable **b** (contrôle de la présence du sac) sont à l'état **1**.

II-3-d- Opérateur OU (inclusif) ou somme logique - OR :

Il permet de traiter plusieurs variables d'entrées. La sortie est égale à 1 si une variable d'entrée est égale à 1 (un « un » en entrée force un « un » en sortie).

<table>
<tr><th>Symbole Logique (Américain)</th><th>Symbole Logique (AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td>
<table>
<tr><th>e2</th><th>e1</th><th>S</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
Les 03 combinaisons affectent à S l'état 1
</td>
<td>$S = e_1 + e_2$
(S égal e_1 OU e_2)</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est éteinte si a et b ne sont pas actionnés</td>
<td colspan="2"> 
</td>
</tr>
</table>

Exemple : Commande d'une sonnette électrique depuis 02 endroits (fig. II-4).

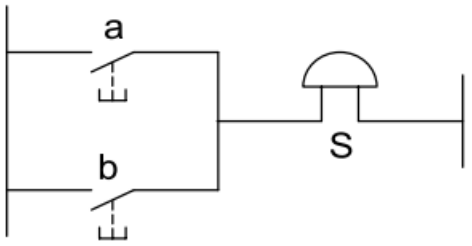
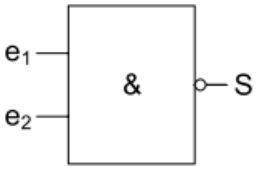
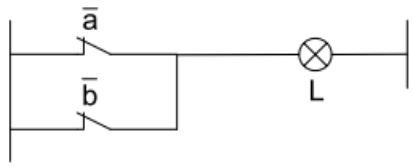


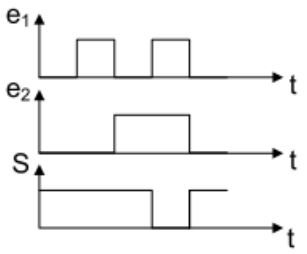
Figure II-4 : Circuit de la sonnette.

Le fonctionnement de circuit peut être représenté par un opérateur **OU**. La sonnette **H** est à l'état **1**, si la variable **a** (bouton –poussoir du premier point de commande) **OU** la variable **b** (bouton-poussoir du 2ème point de commande) sont à l'état **1**.

II-3-e- Opérateur non ET (négation du ET - NAND) :

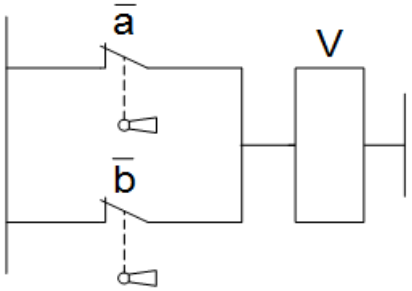
La variable de sortie est égale à 0 si les variables d'entrées sont égales à 1 (un « zéro » en entrée force un « un » en sortie).

<table>
<tr><th>Symbole Logique (Américain)</th><th>Symbole Logique (AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td>
<table>
<tr><th>e2</th><th>e1</th><th>S</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
S complémentaire aux S du ET
</td>
<td>$S = \overline{e_1 \cdot e_2}$
(S égal e_1 ET e_2 le tous barre)</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est éteinte si a et b sont actionnés</td>
<td colspan="2"> 
</td>
</tr>
</table>

Exemple : Contrôle de l'accès à la salle des coffres d'une banque (fig. II-5).

Figure II-5 : Circuit de contrôle de la salle des coffres.

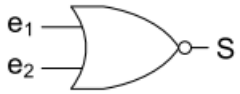
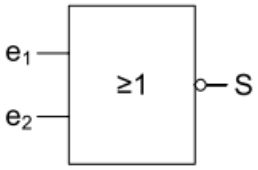
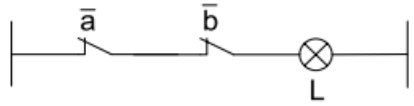
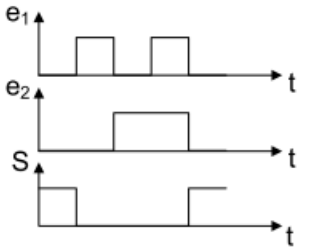


La porte de cette salle est verrouillée en permanence par un dispositif **V** qui n'autorise l'accès que lorsque le directeur et le caissier engagent simultanément leur clé de personnelle dans une serrure électronique.

Le fonctionnement de ce circuit peut être représenté par un opérateur **non ET**. Le dispositif **V** est à l'état **0** si, et seulement si, la variable $\bar{a}$ (contrôle de présence de la clé du directeur) et la variable $\bar{b}$ (contrôle de la présence de la clé du caissier) sont toutes les deux à l'état **1**.

II-3-f- Opérateur non OU (négation du OU - NOR) :

La sortie est à l'état 1 si les entrées sont à l'état 0 (un « un » en entrée force un « zéro » en sortie).

Symbole Logique (Américain)	Symbole Logique (AFNOR)	Table de Vérité	Equation Logique
		e_2 e_1 S 0 0 1 0 1 0 1 0 0 1 1 0 S complémentaire aux S du OU	$S = \overline{e_1 + e_2}$ (S égal e_1 OU e_2 le tous barre)
Schéma à Contacts		**Chronogramme**	
 La lampe L est allumée si a et b ne sont pas actionnés			

Exemple : Circuit de protection contre une tentative d'intrusion dans un local (fig. II-6).

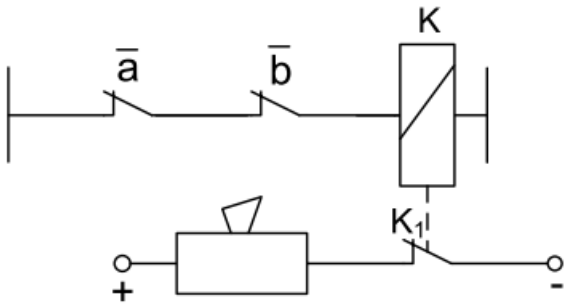


Figure II-6 : Circuit d'installation d'alarme.

Le circuit est représenté au repos. A la mise sous tension le relais K est excité, le contact K1 ouvert.

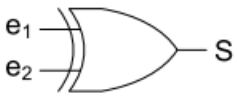
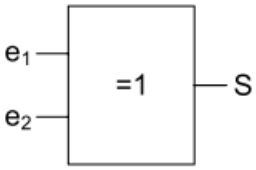
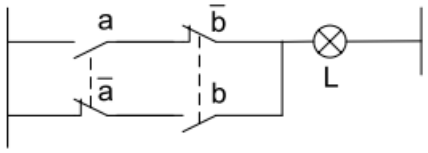
Toute intrusion, par la fenêtre ou par la porte, provoque l'interruption d'un circuit de veille, avec pour conséquence, la désexcitation d'un relais et l'émission d'un relais et l'émission d'un signal sonore.

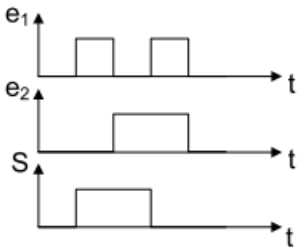
Le fonctionnement de ce circuit peut être représenté par un opérateur **non OU**. Le relais **K** de contrôle du circuit de veille est à l'état **1** si, et seulement si, la variable $\bar{a}$ (contrôle de l'ouverture de la porte) et la variable $\bar{b}$ (contrôle de l'ouverture de la fenêtre) sont toutes les deux à l'état **0** (non actionnées).

Remarque : Les fonctions de base NON, ET et OU combinées entre elles permettent de trouver tous les opérateurs logiques. Généralement, celles-ci sont utilisées pour la programmation des API.

II-3-g- Opérateur OU exclusif (XOR) :

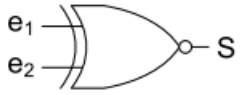
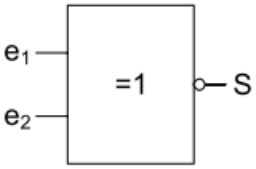
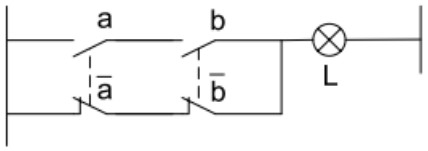
La variable de sortie est égale à 1 si et seulement si l'une des variables d'entrée est égale à 1.

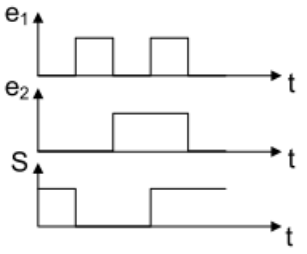
<table>
<tr><th>Symbole Logique (Américain)</th><th>Symbole Logique (AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td>
<table>
<tr><th>e2</th><th>e1</th><th>S</th></tr>
<tr><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>1</td><td>1</td></tr>
<tr><td>1</td><td>0</td><td>1</td></tr>
<tr><td>1</td><td>1</td><td>0</td></tr>
</table>
</td>
<td>$S = e_1 \oplus e_2$
$= e_1\bar{e_2} + \bar{e_1}e_2$</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est allumée si a ou b est actionné</td>
<td colspan="2"> 
</td>
</tr>
</table>

II-3-h- Opérateur identité (XNOR) :

La variable de sortie est égale à1 si et seulement si les deux entrées ont le même état logique.

<table>
<tr><th>Symbole Logique (Américain)</th><th>Symbole Logique (AFNOR)</th><th>Table de Vérité</th><th>Equation Logique</th></tr>
<tr>
<td> 
</td>
<td> 
</td>
<td>
<table>
<tr><th>e2</th><th>e1</th><th>S</th></tr>
<tr><td>0</td><td>0</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>0</td></tr>
<tr><td>1</td><td>0</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>1</td></tr>
</table>
</td>
<td>$S = e_1 \odot e_2$
$= e_1 e_2 + \overline{e_1}\,\overline{e_2}$</td>
</tr>
<tr><th colspan="2">Schéma à Contacts</th><th colspan="2">Chronogramme</th></tr>
<tr>
<td colspan="2"> 

La lampe L est allumée si a et b est actionné</td>
<td colspan="2"> 
</td>
</tr>
</table>

II-4- Exemple pratique :

Dans une usine de brique, on effectue un contrôle de qualité selon quatre critères :

Le poids, la largeur, la longueur et la hauteur.

Si la valeur d'un critère est correcte, on lui affecte la valeur 1, sinon on lui affecte la valeur 0.

Cela permet de classer les critères en trois catégories :

Qualité A : Le poids P et deux dimensions, au moins, sont correctes.

Qualité B : Le poids seul est incorrect ou le poids étant correct, deux dimensions, aux moins, sont incorrectes.

Qualité C : Le poids P est incorrect ainsi qu'une ou plusieurs dimensions.

Ainsi, pour déterminer la qualité d'une brique, on se base sur les quatre critères suivants : Le poids, la largeur, la longueur et la hauteur.

A chacun de ces critères, on va associer une variable booléenne. Ainsi, nous avons respectivement les variables p, L, l, h. Si le critère est correct, la variable correspondante prend la valeur 1, sinon elle prend la valeur 0.

1- Ecriture des fonctions f_A, f_B et f_C :

Qualité A : le poids p et deux dimensions, au moins, sont correctes.En d'autres termes : P = 1 et (Llh = 110 ou 101 ou 011 ou 111).

Ainsi, la fonction : $f_A = p \cdot (Ll\bar{h} + L\bar{l}h + \bar{L}lh + Llh)$.

Qualité B : le poids seul est incorrect ou le poids étant correct, deux dimensions, aux moins, sont incorrectes. Cela veut dire que : PLlh = 0111 ou P = 1 et (Llh = 100 ou 010 ou 001 ou 000).

Donc, la fonction : $f_B = \bar{p}Llh + p \cdot (L\bar{l}\,\bar{h} + \bar{L}l\bar{h} + \bar{L}\bar{l}\,h + \bar{L}\,\bar{l}\,\bar{h})$.

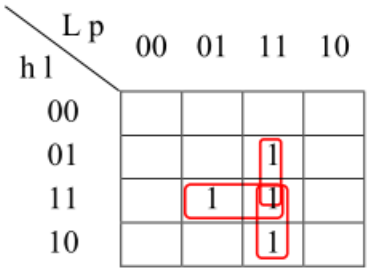
Qualité C : le poids p est incorrect ainsi qu'une ou plusieurs dimensions. En d'autre termes : p = 0 et (Llh = 011 ou 101 ou 110 ou 001 ou 010 ou 100 ou 000).

Donc, on aura la fonction : $f_C = \bar{p}(\bar{L}lh + L\bar{l}h + Ll\bar{h} + \bar{L}\,\bar{l}h + \bar{L}l\bar{h} + L\bar{l}\,\bar{h} + \bar{L}\,\bar{l}\,\bar{h})$.

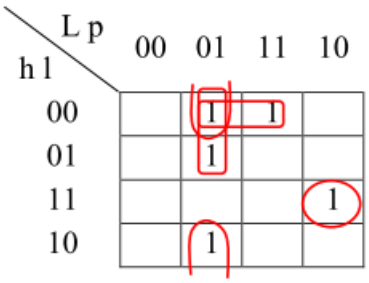
On peut mettre les différents états des fonctions f_A, f_B et f_C dans une **table de vérité** :

déc.	p	L	l	h		f_A	f_B	f_C
0	0	0	0	0				1
1	1	0	0	0			1	
2	0	1	0	0				1
3	1	1	0	0			1	
4	0	0	1	0				1
5	1	0	1	0			1	
6	0	1	1	0				1
7	1	1	1	0		1		
8	0	0	0	1				1
9	1	0	0	1			1	
10	0	1	0	1				1
11	1	1	0	1		1		
12	0	0	1	1				1
13	1	0	1	1		1		
14	0	1	1	1			1	
15	1	1	1	1		1		

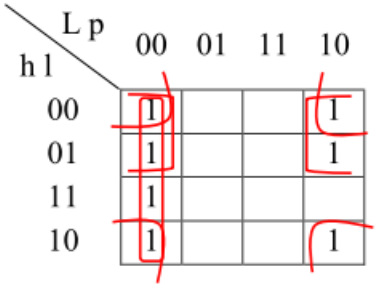
2- Simplification des fonctions f_A, f_B et f_C par le tableau de Karnaugh :



$$f_A = pLh + pLl + plh$$

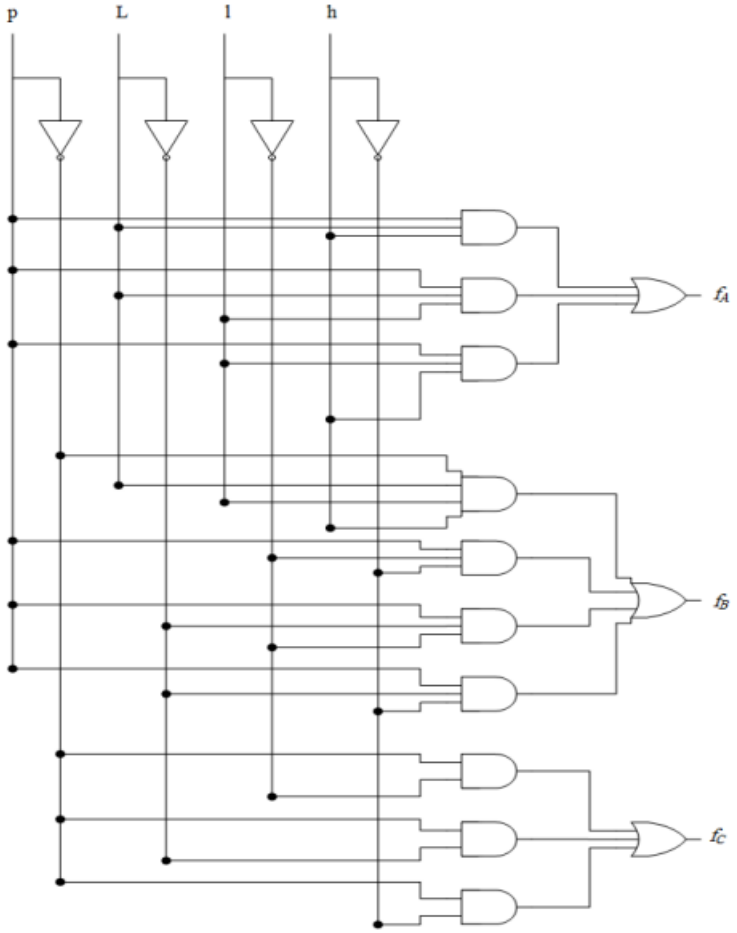


$$f_B = \bar{p}Llh + p\bar{l}\,\bar{h} + p\bar{L}\,\bar{l} + p\bar{L}\,\bar{h}$$



$$f_C = \bar{p}\,\bar{l} + \bar{p}\,\bar{L} + \bar{p}\,\bar{h}$$

3- Logigramme des fonctions f_A, f_B et f_C :



Chapitre III

Notion de programmation des automates

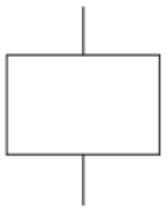
III-1- Introduction – Notion d'algorithme [3] :

Un algorithme est une règle (enchainement des actions nécessaires à l'accomplissement d'une tâche). Il s'exprime par une suite ordonnée de directives composée d'**actions** et de **décisions** qu'il faut exécuter en séquence suivant un enchaînement strict pour accomplir une **tâche** donnée, conforme à un cahier de charges. Dans un automatisme, la succession des tâches logiques constitue l'algorithme de sa fonction globale.

L'algorithme reproduit dans un langage **graphique normalisé** tous les cheminements du raisonnement logique qui détermine la composition de l'algorithme.

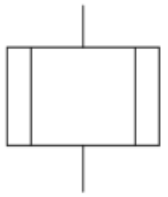
Symboles normalisés des organigrammes (utilisés dans les algorigrammes) :

Symbole **Désignation**

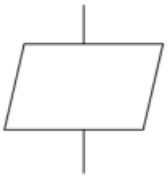


- **Symboles de Traitement :**

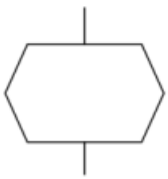
- **Symbole générale « Traitement » :** Opération ou groupe d'opérations sur des données, instructions, etc., ou opération pour laquelle il n'existe aucun symbole normalisé.



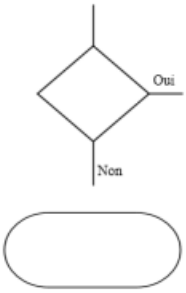
- **Sous-programme :** Portion de programme considérée comme une simple opération.



- **Entrée – Sortie (Lecture – Ecriture) :** Mise à disposition d'une information à traiter ou enregistrement d'une information traiter.



- **Préparation (action préparatoire) :** Opération qui détermine partiellement ou complétement la voie à suivre dans un embranchement ou un sous- programme.



- **Symboles Logiques :**

- **Embranchement (Test-Décision) :** Choix d'une voie parmi plusieurs, il représente une décision ou un aiguillage.

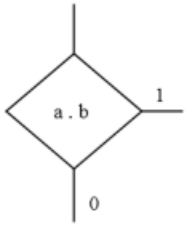
- **Symboles Auxiliaires :**

- **Terminaison :** Début, fin ou interruption d'un organigramme, point de contrôle.

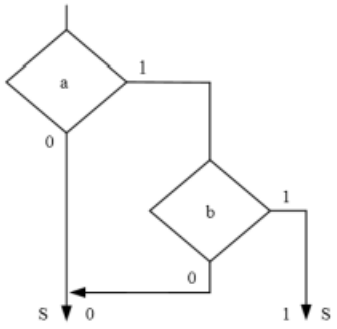
Exemples d'algorigrammes des fonctions ET et OU :

Fonction ET

Avec un seul test



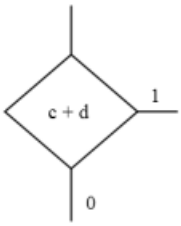
Avec deux tests successifs



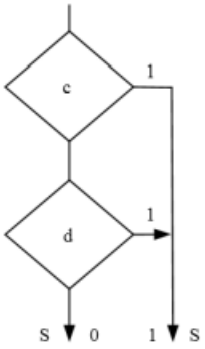
Si A = 1 et B = 1 **Alors** S = 1.

Fonction OU

Avec un seul test



Avec deux tests successifs



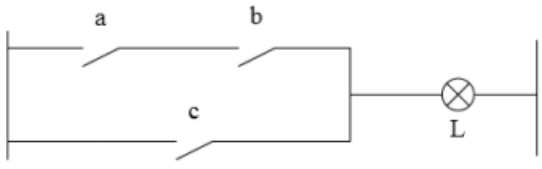
Si C = 1 ou D = 1 **Alors** S = 1.

Remarque : le sens général des lignes de liaisons doit être de haut en bas et de gauche à droite. Lorsque le sens ainsi défini n'est pas respecté, des pointes de flèches sur la ligne indiquent le sens utilisé.

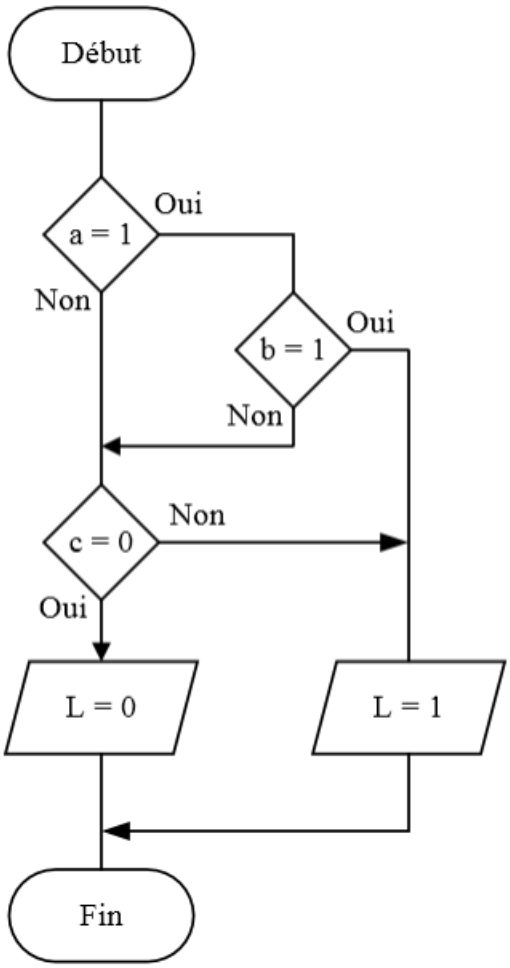
III-2- Expression algorithmique d'une équation logique :

Soit l'équation L = a·b+c.

Le schéma à contacts de cette équation est :



Algorigramme



Commentaires

La première partie de l'algorigramme est représentative de la structure du ET avec 2 tests successifs.

Chaque test a une réponse bivalente ou binaire.

A est-il égal à 1 ?

OUI : a = 1

Non : a = 0

La 2ème partie est représentative du OU. Pour ce test on peut poser la question :

C est-il égal à 0 ?

Ou

C est-il égal à 1 ?

Seules les considérations graphiques de l'algorigramme sont prises en compte pour choisir la 2ème question.

III-3- Expression algorithmique des évolutions d'un système automatisé :

L'enchainement des évolutions d'un automatisme, c'est-à-dire la suite ordonnée des évènements et des actions qui caractérisent le processus peut se décomposer en **séquences** qui sont associées et organisées suivant **03 types** de structures algorithmiques :

- La structure linéaire ;
- Les structures alternatives ;
- Les structures itératives.

III-3-a- Structure algorithmique linéaire (structure séquentielle) :

Dans ce type de structure la séquence, comprise entre un début et une fin, se traduit par la suite ordonnée des actions à exécuter.

Exemple : commande de la barrière d'accès d'un parking.

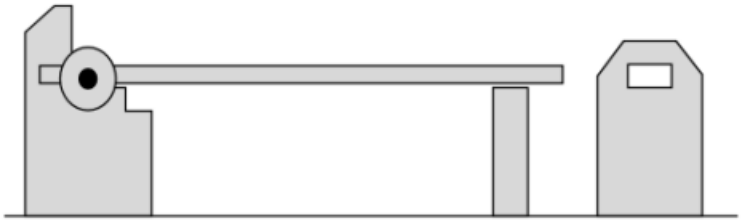
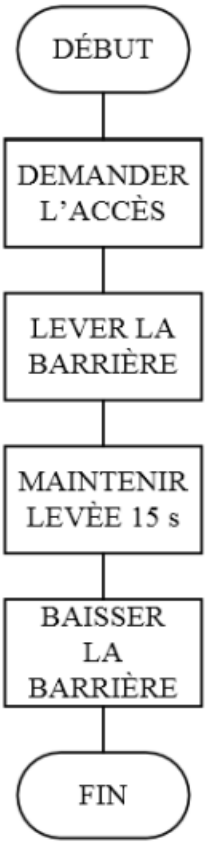


Figure III-1 : Barrière d'un parking [3].

Description : une barrière automatique contrôle l'accès d'un parc de stationnement de voitures.

Seuls certains conducteurs munis d'une clé ou d'une carte magnétique, par exemple, sont autorisés à en commander l'ouverture. Après la barrière levée l'accès est possible durant 15 secondes.

Algorigramme



Commentaires

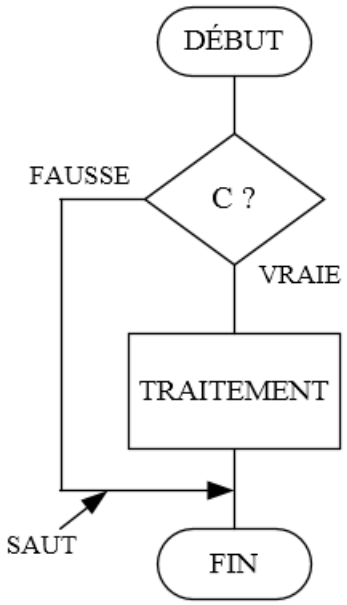
Le cycle de fonctionnement de cet automatisme se traduit par une structure linéaire dans laquelle s'enchaînent, dans leur ordre logique, les 04 actions.

III-3-b- Structures algorithmique alternatives :

Ces structures correspondent à toute situation n'offrant que 02 issues possibles s'excluant mutuellement par la **sélection** entre 02 traitements distincts.

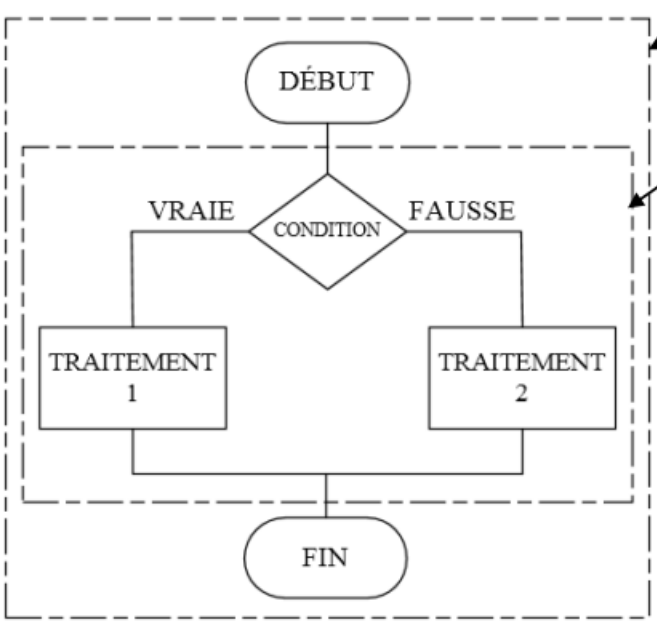
Elles se traduisent par l'expression **Si** … **Alors** … **Sinon** …

- **Structure alternative réduite :**



Dans cette structure l'une des 02 réponses n'entraîne aucun traitement, c'est un **saut**.

- **Structure alternative complète :**



Après test

Si la condition est vraie

Alors faire le traitement 1

Sinon faire le traitement 2

Exemple : Tri de sacs de 25 kg et de 50 kg pour un stockage dans 02 zones distinctes.

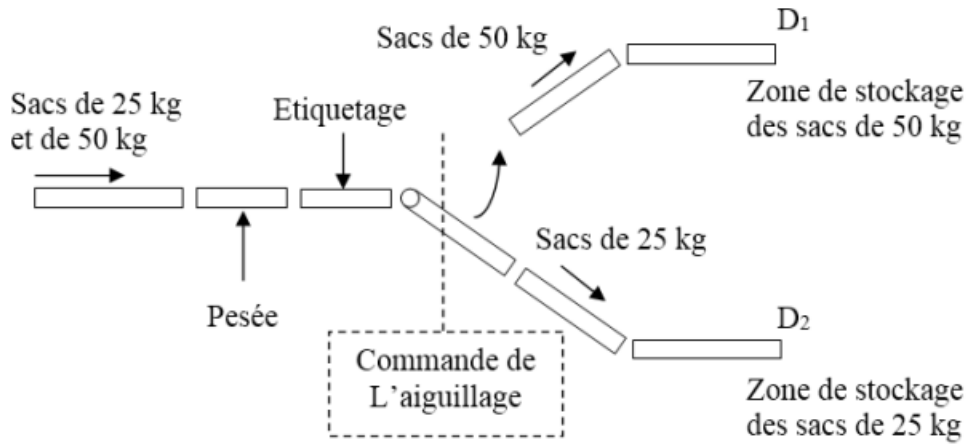
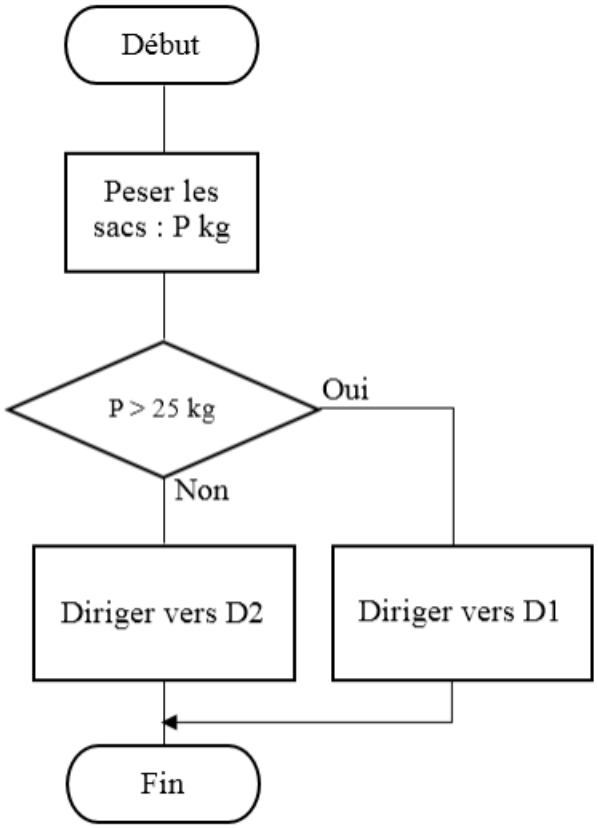


Figure III-2 : Système de tri de sacs [3].

Algorigramme



Commentaires

Le test P > 25 kg est considéré comme un aiguillage qui entraîne 02 actions différentes.

Si P > 25 kg **Alors** Aiguiller vers D1,

Sinon Aiguiller vers D2.

III-3-c- Structures algorithmique itératives :

L'itération désigne toute **répétition** de l'exécution d'un mouvement. La boucle ou reprise de reprise de séquence est bien représentative de ces structures itératives (**structure répétitives**).

Dans ces séquences, 02 cas sont à considérer :

- Le nombre d'itérations est **connu à l'avance** (il est consigné) ;
- Le nombre d'itérations est **inconnu à l'avance**, il est variable, il dépend uniquement d'un ou de plusieurs événements extérieurs.

Quel que soit le cas rencontré, un **test** effectué sur la consigne ou sur l'événement extérieur justifie l'itération.

1er Cas - Nombre d'itérations inconnu :

Cette structure peut se traduire par l'une des deux expressions :

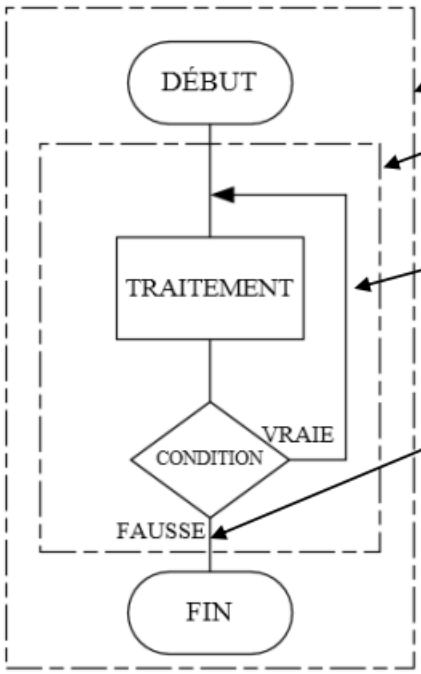
- **Répéter** … **Jusqu'à**

Ou

- **Tant que** … **Répéter**

Algorigramme Général **Commentaires**

Répéter … **Jusqu'à**



Réseau principal

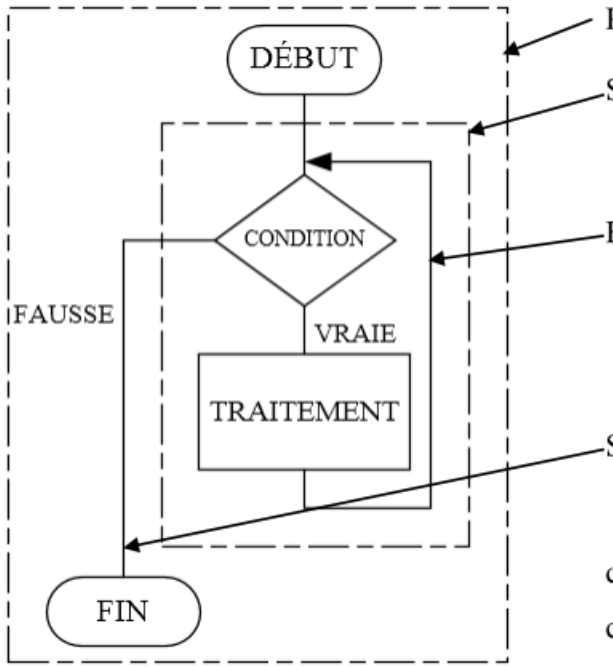
Sous-réseau

Boucle d'itération

Sortie de la boucle d'itération

Le traitement est exécuté **une première fois** dès l'entrée dans la structure, il se répète **jusqu'à** ce que la condition soit fausse.

Tant que … Répéter



Le traitement n'est exécuté que si la condition est vraie et il se répète **Tant que** la condition reste vraie.

Exemple : Soit une installation de manutention de sable. Un transporteur à benne assure le remplissage en sable d'une trémie peseuse (jusqu'à un poids P1) à partir de laquelle est alimentée une centrale à béton (voir fig.III-3).

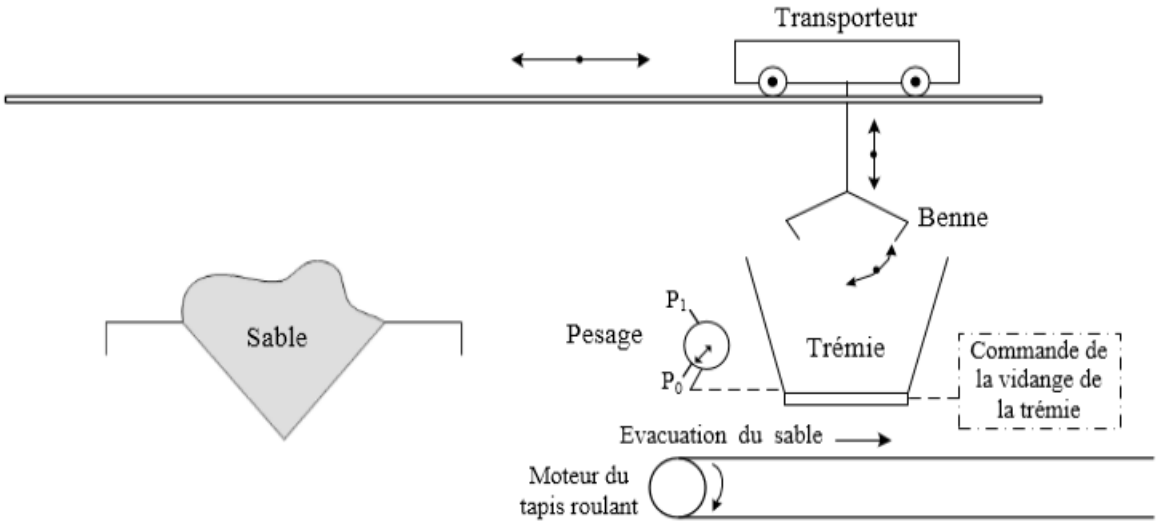
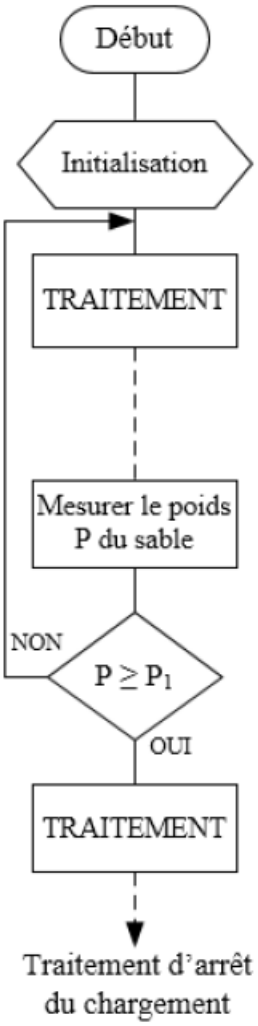


Figure III-3 : Alimentation en sable d'une centrale à béton [3].

Algorigramme



Commentaires

Dans les conditions d'initialisation doit figurer : « Afficher le poids P1 ».

La partie de l'organigramme déjà parcourue est relative :

- Au déplacement du dispositif de manutention,
- A la commande de la benne.

L'opération qui précède le test est la saisie de l'information sur laquelle se fait ce test : « poids du sable ».

Tant que ce poids n'est pas atteint le programme se répète.

Le nombre d'itération est variable.

Le poids P1 est variable et dépend du besoin en sable, son affichage ne prédétermine pas d'une façon précise le nombre de fois que la boucle sera parcourue compte tenu :

- Du niveau de remplissage de la benne,
- De l'état hygrométrique du sable.

2ème Cas - Nombre d'itérations connu :

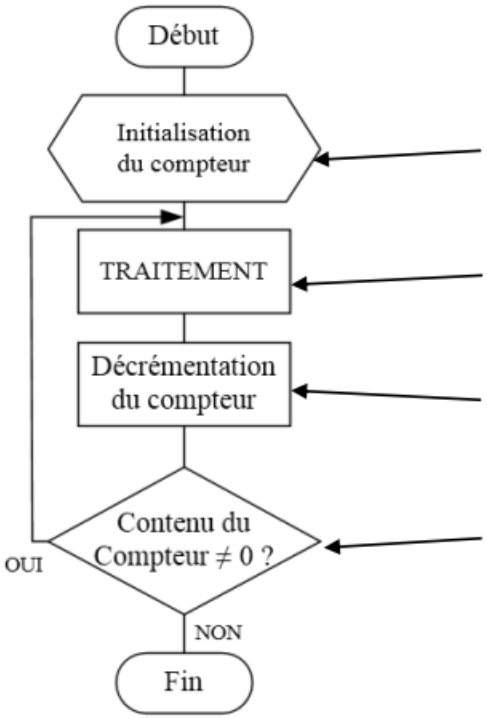
Dans cette structure la sortie de boucle d'itération s'effectue lorsque le nombre souhaité de répétition est atteint. C'est une **itération commandée** ou **automatique**, différente de la précédente qui est **conditionnelle**.

Pour contrôler l'exécution des itérations et faire en sorte qu'elle se termine lorsque le nombre prédéterminé est atteint il est possible de programmer :

- L'utilisation d'un **compteur** ;
- Ou les conditions de variation d'une **valeur numérique** affectée à une variable.

1er Cas - Utilisation d'un compteur :

Algorigramme **Commentaires**



Affectation du nombre de répétition à une variable choisie comme compteur.

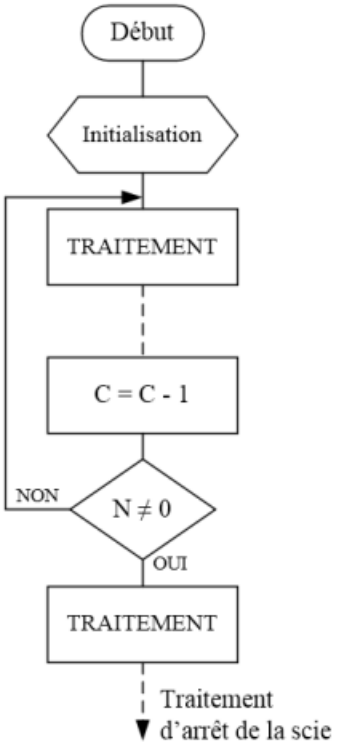
Suite ordonnée d'instructions qui doivent être répétées.

Contrôle de l'exécution des répétitions.

Test de fin de répétitions.

Exemple : Une scie doit débiter un lot de 05 pièces et s'arrêter (nombre d'itération connu).

Algorigramme



Commentaires

Dans les conditions d'initialisation doit figurer :
« Afficher N = 05 au compteur ».
La partie d'organigramme déjà parcourue est relative :

- à la mise en position de la barre,
- à son serrage,
- à la descente de la scie,
- au sciage,
- à la remontée de la scie,
- au desserrage de la barre.

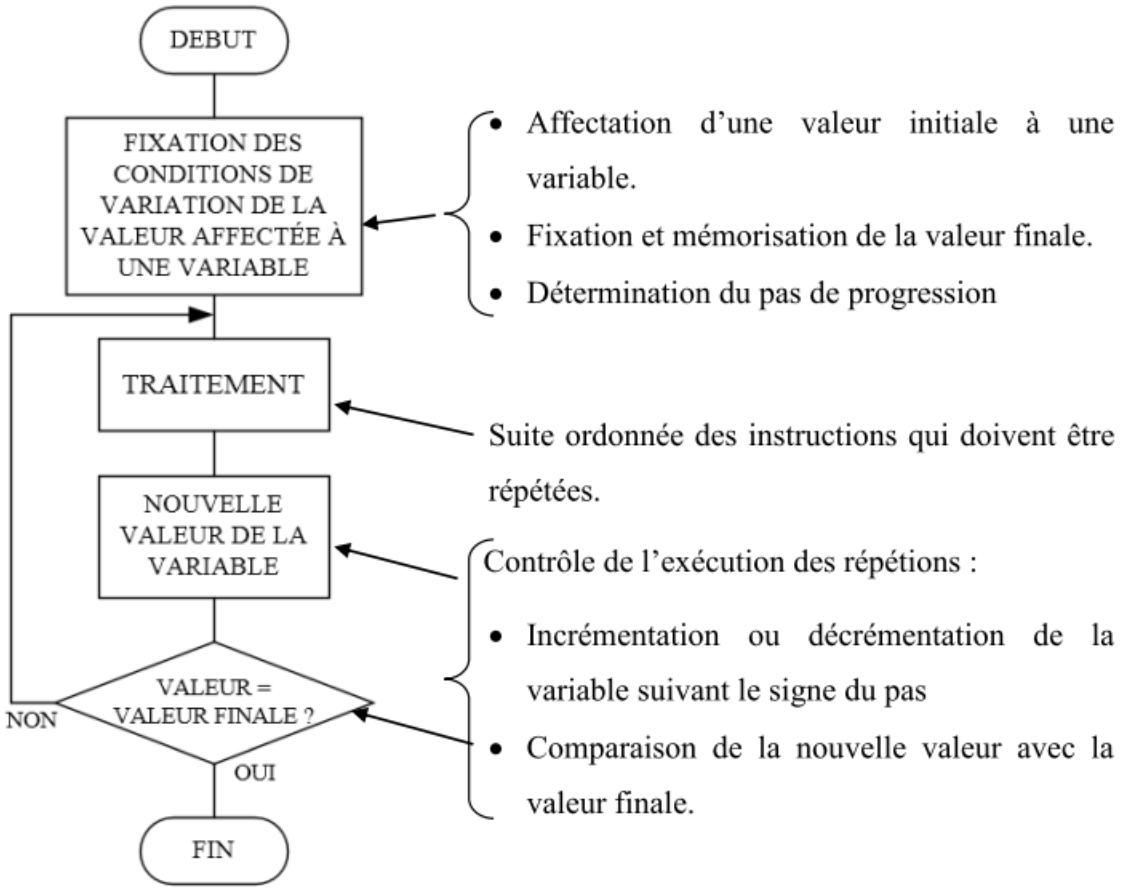
Le nouveau contenu du compteur est égal à l'ancien contenu moins1 (N-1 ; décrémentation)

Tant que la 5ème coupe n'est pas effectuée le programme se répète.

Avant d'entrée dans la boucle on sait qu'elle sera parcourue cinq fois.

2ème Cas - Utilisation d'une variable :

Algorigramme **Commentaires**



La variable retenue est dite **variable d'itération**. Elle varie :

- D'une valeur initiale V_I ;
- Jusqu'à une valeur finale V_F ;
- Avec un pas de variation P.

Le nombre d'itérations est : $\frac{V_I - V_F}{P}$

L'expression de cette structure est :

POUR … À … RÉPÉTER …

Exemple :

Pour le poste de sciage, **POUR** compteur = 5 **À** compteur = 0 **RÉPÉTER** le sciage d'une pièce.

III-4- Langage de programmation pour API [12] :

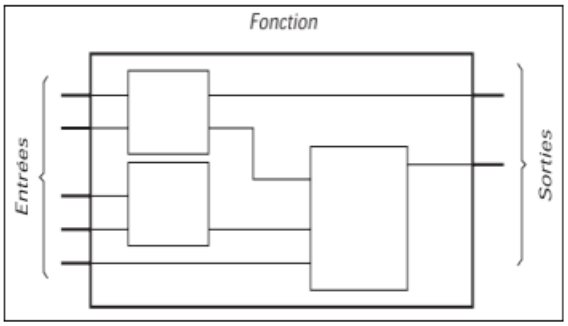
Pour un système donné et à partir de son logigramme il est possible :

- D'associer, d'interconnecter entre eux les opérations logiques qui le composent et de réaliser ainsi une solution en logique câblée,
- Ou d'écrire un programme, un logiciel dont les instructions concernent à la fois :
  - La saisie des informations d'entrée,
  - La nature des opérations logiques qui leur sont affectées,
  - La relation entre le résultat du traitement logique et les sorties concernées.

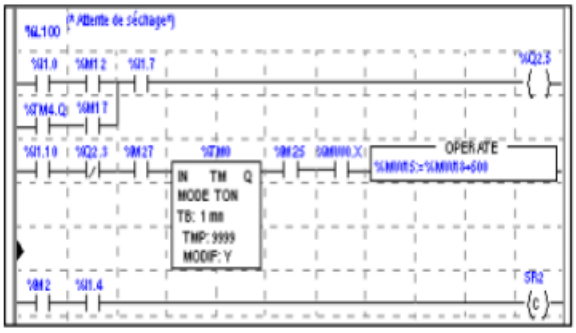
Dans ce cas la solution retenue relève de la logique programmée.

Un langage de programmation est défini par sa série d'instruction. Chaque automate possède son propre langage. Mais par contre, les constructeurs proposent tous une interface logicielle répondant à la norme CEI[1] 61131-3. Cette norme définit cinq langages de programmation d'applications d'automates programmables industriels (API) utilisables, [12] qui sont :

- **FBD** (Function Block Diagram, ou schéma par blocs) : Ce langage permet de programmer **graphique**ment à l'aide de **blocs** (rectangles), représentant des variables, des opérateurs ou des fonctions. Il permet de manipuler tous les types de variables. Les blocs sont programmés (bibliothèque) ou programmables ;



- **LD** (Ladder Diagram, ou schéma à relais) : Ce langage **graphique** est essentiellement dédié à la programmation d'équations booléennes (true/false). Il utilise les **symboles** tels que : contacts, relais et blocs fonctionnels et s'organise en réseaux (labels). C'est le plus utilisé ;



[1] Commission Electrotechnique Internationale

- **ST** (Structured Text ou texte structuré) : Ce langage est un langage **textuel** de **haut niveau** de même nature que le Pascal. Il utilise les fonctions comme *if ... then .... Else ....* Il permet la programmation de tout type d'algorithme plus ou moins complexe ;

```
IF %M0 THEN
    FOR %MW99 := 0 TO 31 DO
        IF %MW100 [%MW99]<> 0 THEN
            %MW10 := %MW100 [%MW99];
            %MW11 := %MW99;
            %M1 := TRUE;
            EXIT;                (*Sortie de la boucle FOR*)
        ELSE
            %M1 := FALSE;
        END_IF;
    END_FOR;
ELSE
    %M1 := FALSE;
END_IF;
```

- **IL** (Instuction 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 (programmation des microcontrôleurs) ;

```
!  %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:X5
        [ %MW15 :=  %MW18+500]
```

- **SFC** (Sequential Function Chart) : Issu du langage **GRAFCET**, ce langage, de haut niveau, permet la programmation aisée de tous les procédés séquentiels ; dans le chapitre suivant, nous allons présenter les bases du langage Grafcet objet de ce cours.

Chapitre IV

Le GRAFCET

IV-1- Introduction :

L'AFCET (**A**ssociation **F**rançaise pour la **C**ybernétique **E**conomique et **T**echnique) et l'ADEPA (**A**gence **N**ationale pour le **D**éveloppement de la **P**roduction **A**utomatisée) ont mis au point et développé une représentation graphique qui traduit sans ambiguïté, l'évolution des cycles d'un automatisme séquentiel [6].

Ce diagramme fonctionnel : le GRAFCET (**GRA**phe **F**onctionnel de **C**ommande **E**tapes **T**ransitions) permet de décrire les comportements attendus de l'automatisme de commande au niveau :

- Du traitement des **informations** délivrées par la partie opérative,
- Et des **ordres** transmis à cette même partie opérative.

En définitif, le GRAFCET est un langage graphique normalisé au plan international qui utilise des symboles.

IV-2- Règles d'écriture du GRAFCET [3] :

Le GRAFCET se compose d'un ensemble (fig. IV-1) :

- D'**étapes** auxquelles sont associées des **actions** ;
- De **transitions** auxquelles sont associées des **réceptivités** ;
- De **liaisons orientées** reliant les étapes aux transitions et les transitions aux étapes.

L'écriture du GRAFCET doit respecter certaines règles :

- L'association d'une étape avec ses actions se représente :
  - Par un carré repéré numériquement pour l'étape,
  - Relié à un ou plusieurs rectangles dans lesquels les actions sont décrites (elle comporte un verbe à l'infinitif écrit en majuscule),
- Deux étapes dont les actions doivent s'enchainer sont reliées entre elles et un trait perpendiculaire à cette liaison confirme cette possibilité d'évolution, ce petit trait est une transition,

- A la possibilité d'enchaînement d'actions successives doit obligatoirement être associée la condition préalable à cette évolution, c'est-à-dire la nature des évènements qui l'autorisent ou l'interdisent, cette condition obligatoirement associée à chaque transition est nommée réceptivité, elle s'écrit sous la forme de proposition logique (écrite en minuscule).

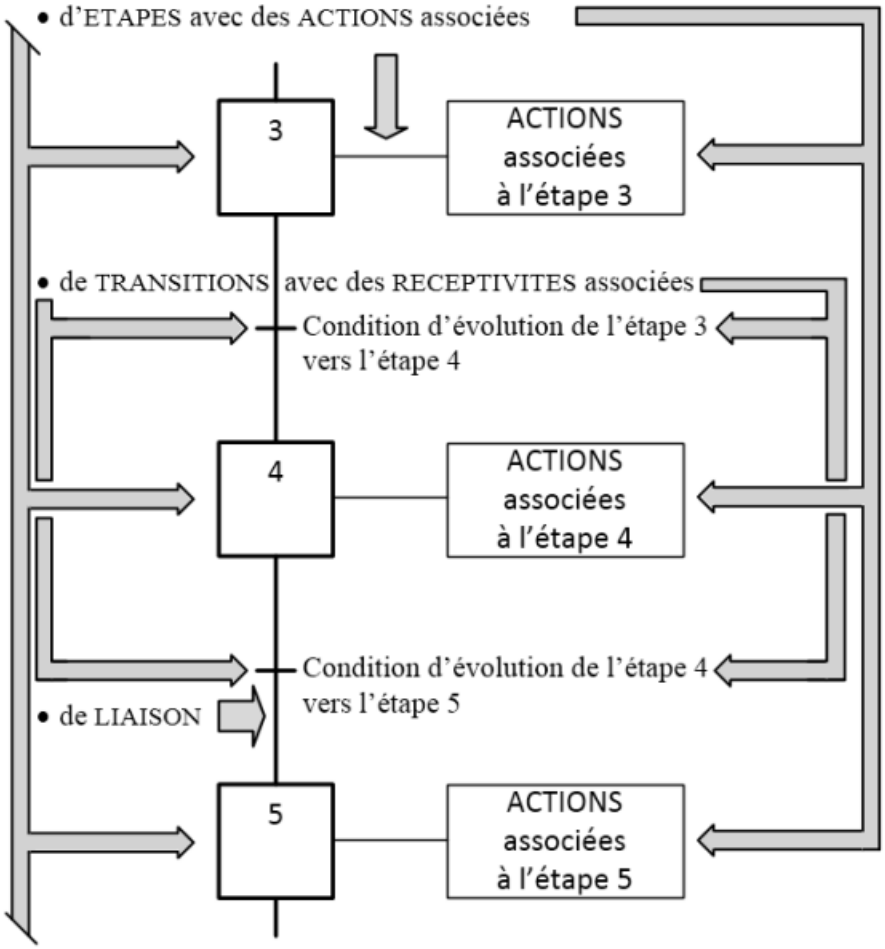


Figure IV-1 : Règles d'écriture d'un grafcet [3].

IV-3- Règles d'évolution du GRAFCET :

Les règles d'évolution précisent les conditions dans lesquelles le GRAFCET peut évoluer, c'est-à-dire les conditions dans lesquelles les étapes sont actives ou inactives.

- **Initialisation** (Règle 1) :

L'**étape initiale** d'un GRAFCET est représentée par un **double carré** (fig. IV-2). Elle est considérée comme **initialement active** ce qui permet l'évolution de l'automatisme dès que la réceptivité associée à la première transition devient vraie.

Après la dernière transition cette étape est de nouveau activée par la **boucle de retour à l'état initial** encore désignée **boucle de reprise** et représentée par une **liaison orientée** (fig. IV-2).

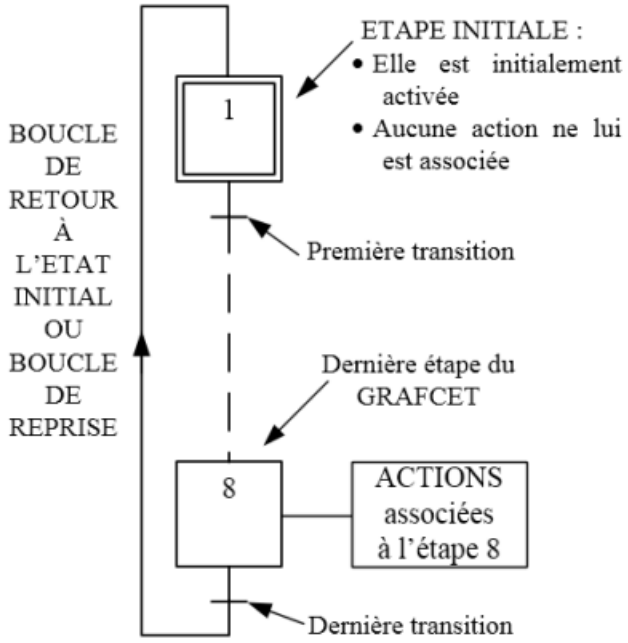


Figure IV-2 : Etape initiale et boucle de retour à l'état initial [3].

- **Franchissement d'une transition** (Règle 2) :

Le franchissement d'une transition s'effectue (fig. IV-3) :

- Si la **transition est validée**, c'est-à-dire si **l'étape précédente est active**,
- Et si la **réceptivité** associée à la transition est **vraie**.

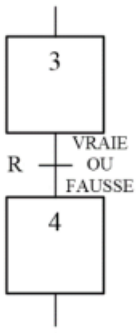
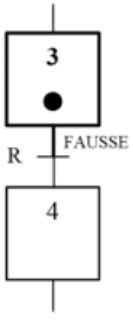
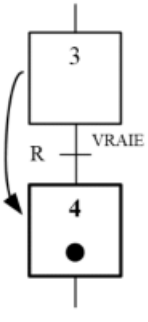
		
Cas 1 : l'étape 3 n'est pas active, que la réceptivité soit vraie ou fausse, la transition n'est pas franchie.	**Cas 2 :** l'étape 3 est active, mais la réceptivité est fausse, la transition n'est pas franchie.	**Cas 3 :** la réceptivité devient vraie, la transition est franchie, l'étape 3 est désactivée, l'étape 4 est activée.

Figure IV-3 : Franchissement de la transition entre les étapes 3 et 4 [3].

- **Evolution des étapes actives** (Règle 3) :

Le **franchissement** d'une transition entraîne (fig. IV-3) :

- L'**activation de l'étape suivante**,
- Et la **désactivation de l'étape précédente**.

Le point dans le carré de l'étape indique l'activation de l'étape.

IV-4- GRAFCET à séquence unique [3, 5] :

Un automatisme est représenté par un GRAFCET à **séquence unique** lorsqu'il peut être décrit par un ensemble de plusieurs étapes formant **une suite** dont le déroulement s'effectue toujours dans le **même ordre**. Soit l'exemple d'application d'un système automatisé séquentiel, cas d'une barrière automatique contrôlant l'accès d'un parking.

IV-4-a- Description du système (Expression du cahier de charge) :

Une barrière automatique contrôle l'accès d'un parc de stationnement de voitures. Seuls certains conducteurs munis d'une clé ou d'une carte magnétique, par exemple, sont autorisés à en commander l'ouverture. Après la commande de l'ouverture l'accès est possible durant 15 secondes.

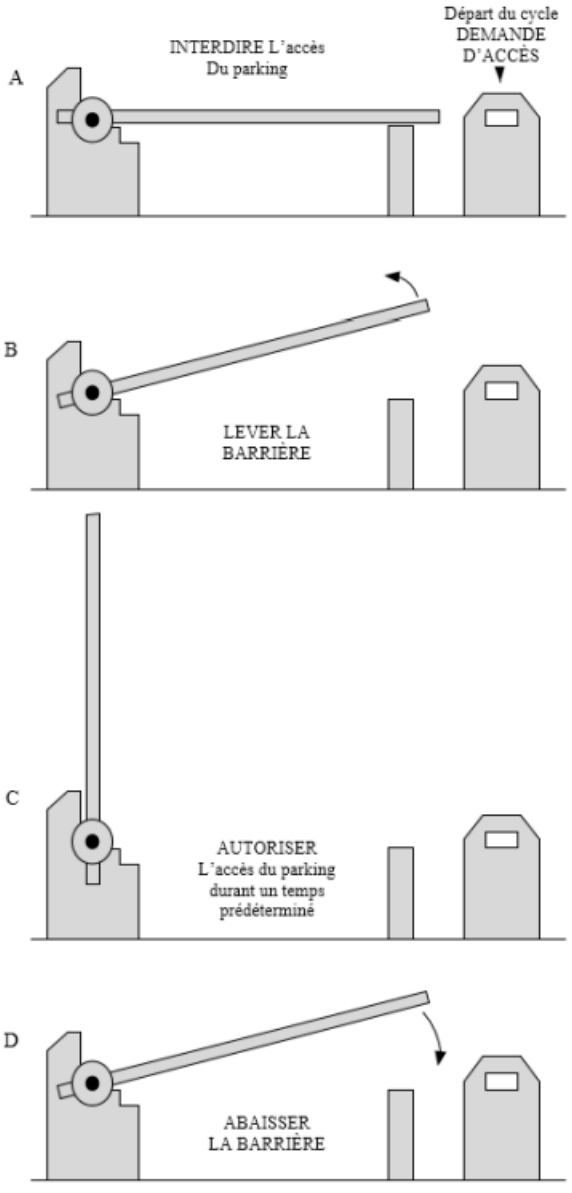


Figure IV-4 : Description du fonctionnement de la barrière automatique [3].

IV-4-b- Fonction globale :

On définit la fonction globale par **diagramme d'activité** (actigramme – fig. IV-5) qui s'identifie par un **verbe d'action** (activité faire sur les données d'entrées). Il crée et génère une ou plusieurs **données de sortie** :

- Après avoir transformé, modifié, changé l'état d'une ou plusieurs **variables d'entrées** en leur conférant une **valeur ajoutée**,
- Par l'utilisation des potentialités d'un ou plusieurs **supports d'activité** qui peuvent être des moyens techniques ou humains,
- Et en respectant **des données ou des contraintes** de commande, de contrôle ou d'exploitation.

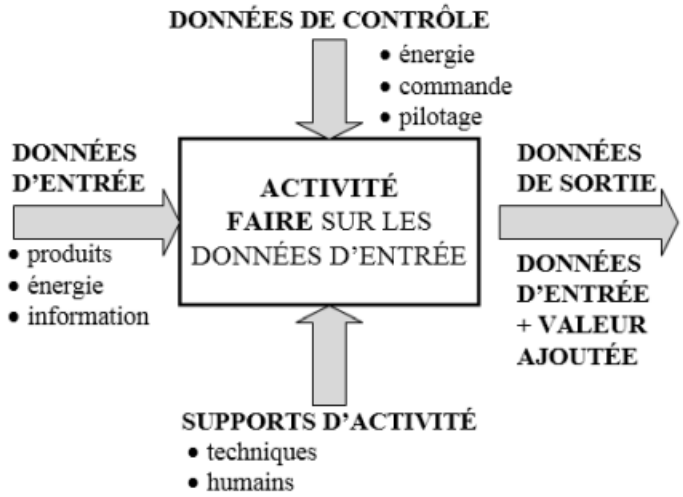


Figure IV-5 : Modèle général d'actigramme [3].

En fait un actigramme s'écrit en répondant aux questions (fig. IV-6) :

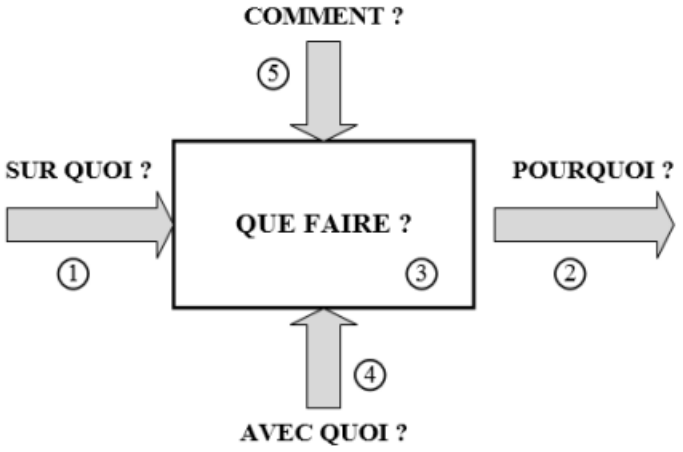


Figure IV-6 : Les cinq questions de l'actigramme [3].

- **sur quoi ?** sur les entrées,
- **pourquoi ?** pour leur donner une valeur ajoutée,
- **que faire ?** une certaine activité,
- **avec quoi ?** en utilisant des moyens,
- **comment ?** en respectant certaines contraintes.

Pour le cas de la barrière automatique, l'actigramme est représenté sur la figure IV-7.

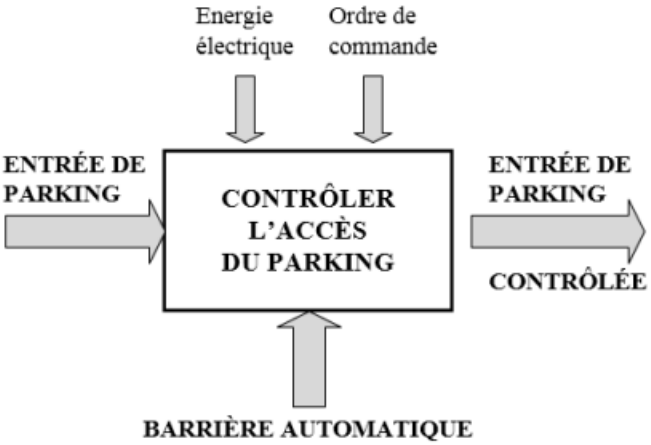


Figure IV-7 : Fonction globale de la barrière automatique [3].

IV-4-c- GRAFCET du point de vue système :

Le GRAFCET de la figure IV-8 traduit le fonctionnement du système tel qu'il est décrit **sans présager** (indiquer) **des moyens techniques** qui seront mis en œuvre pour le réaliser.

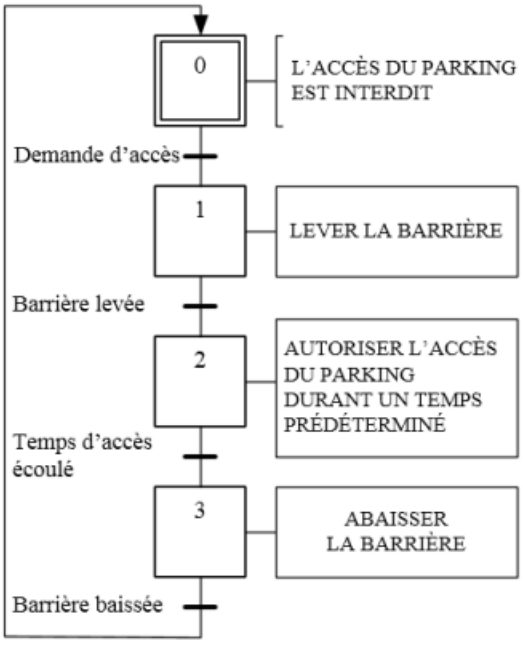


Figure IV-8 : GRAFCET selon un point de vue système.

Ce GRAFCET établi selon un point de vue système indique la **coordination des tâches principales** nécessaires, pour satisfaire la fonction globale (contrôler l'accès du parking par exemple – fig. IV-7) et pour la valeur ajoutée à l'entrée de parking.

Les différentes situations A, B, C et D (fig. IV-4) peuvent être observées par toute personne, utilisatrice ou non du parking.

IV-4-d- GRAFCET du point de vue partie opérative :

Ce type de GRAFCET (fig. IV-9) décrit le **comportement de la partie commande** pour obtenir une évolution correcte de la partie opérative **sans tenir compte de la technologie** qui sera retenue pour les capteurs, les préactionneurs et les actionneurs.

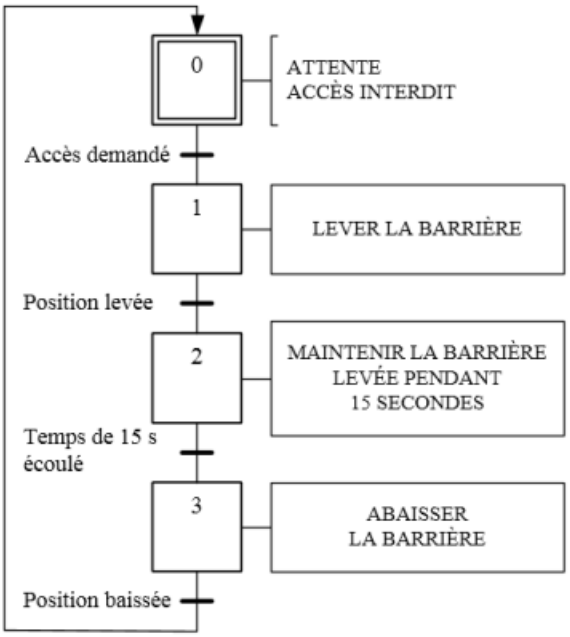


Figure IV-9 : GRAFCET selon un point de vue partie opérative.

IV-4-e- GRAFCET du point de vue partie commande :

Ce GRAFCET **prend en compte les choix technologiques** et l'ensemble des échanges de la partie commande avec la partie opérative et le dialogue avec l'opérateur (fig. IV-10).

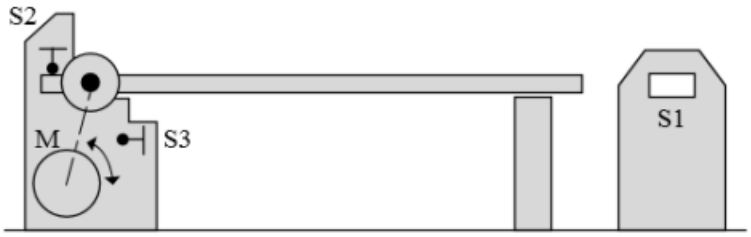


Figure IV-10 : Identification de l'actionneur et des capteurs [3].

C'est ainsi que pour le système barrière de parking il est choisi :

- Un **moteur M** à deux sens de marche pour la **L**evée ou l'**A**baissement de la barrière,
- Un **capteur-lecteur** de carte Magnétique **S1** pour la demande d'ouverture,
- **Deux capteurs mécaniques de position S2 et S3** pour le contrôle de la position de la barrière,
- Un dispositif de **temporisation T** réglé à 15 secondes.

Le GRAFCET du point de vue commande (fig. IV-11) décrit cette dernière selon le point de vue du **réalisateur**.

- **INFORMATIONS**

S1 : demande d'ouverture

S2 : barrière en position basse

S3 : barrière en position haute

- **ORDRES ET ACTIONS**

M.L : lever

T = 15 s : déclencher une temporisation de 15 s

M.A : abaisser

- **INFORMATION GÉNÉRÉE PAR LA PARTIE COMMANDE**

t/2/15 s : fin de la temporisation de 15 s déclenchée à l'étape 2

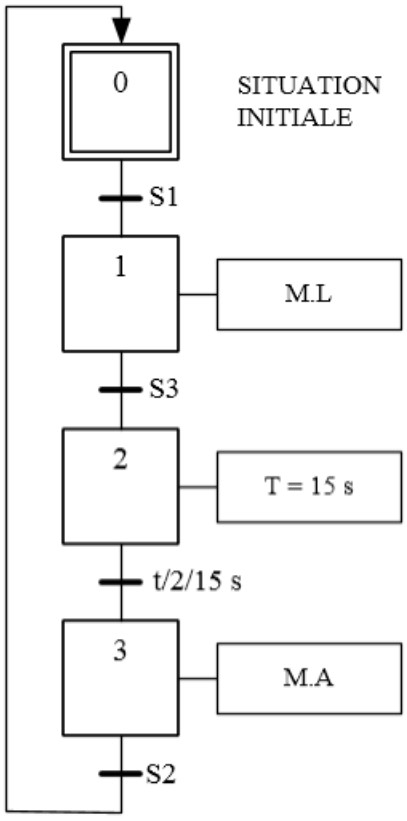


Figure IV-11 : GRAFCET du point de vue partie commande.

IV-4-f- Exemple d'étude d'un transtainer [5] :

Un transtainer (fig. IV-12) est un système chargé de transborder des conteneurs d'un train ou d'un camion à un autre. Il est constitué d'un portique mobile sur lequel se déplace un chariot qui commande une pince.

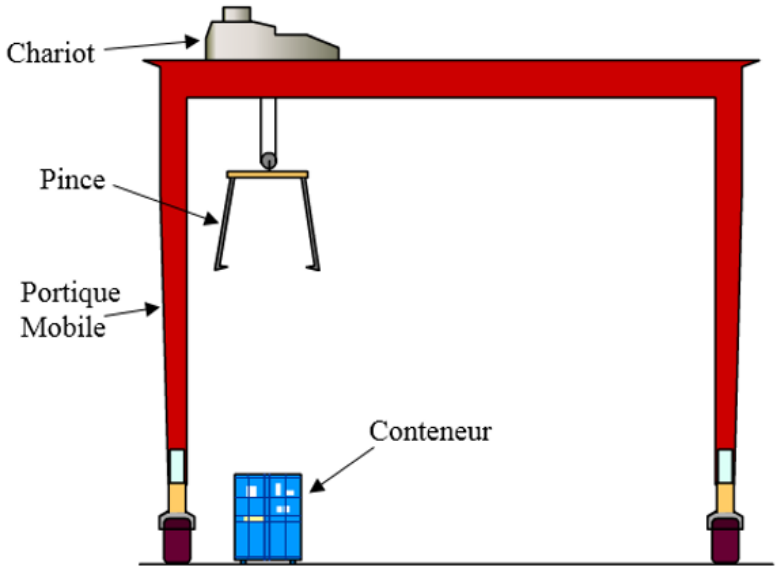


Figure IV-12 : Représentation d'un transtainer [5].

Le fonctionnement du transtainer est décrit par le grafcet proposé ci-contre [5] :

Voir la solution pendant la séance de TD

Le système attend que les **conditions de départ** soient remplies.

Le début du cycle est représenté par un double carré numéroté. Cette étape s'appelle **étape initiale**. 1

Le transtainer est en attente. Il faut définir toutes les informations que la partie commande doit recevoir pour passer à l'étape 2 qui donne l'ordre « DESCENDRE la pince ».

Parmi les huit propositions suivantes, on doit choisir cinq (dc et cp et sh et sd et cg).

dc = départ de cycle	cp = container présent	sb = pince en bas
cg = chariot à gauche	cd = chariot à droite	sh = pince en haut
sv = pince verrouillée	sd = pince déverrouillée	

Ainsi, si toutes les conditions qui lui sont associées sont remplies, la réceptivité est vraie et la transition franchie, l'étape 2 est dite « activée » et l'étape 1 « désactivée » (simultanément).

Le grafcet peut maintenant se dérouler étape par étape pour transférer un container.

Lorsque l'étape 9 est activée, la dernière action du cycle est exécutée et le chariot est reculé. Si un nouveau container arrive alors que le déroulement de ce grafcet est terminé ! Le grafcet doit décrire une **boucle** en reliant la dernière transition à la première étape.

Les traits qui relient les étapes et les transitions ensemble s'appellent des **liaisons orientées**.

IV-5- GRAFCET à aiguillage exclusif [5] :

Dans de nombreuses applications, le grafcet doit être en mesure de commander des cycles différents selon des **conditions** extérieures ou des **choix** de l'opérateur. La structure en aiguillage exclusif (divergence exclusive) apporte la solution.

On distingue trois structures fondamentales :

- **Sélection de séquence (aiguillage) :** c'est la structure de base constituée d'autant de transitions qu'il y a de choix (**divergence en OU**).
- **Saut d'étape :** c'est une structure particulière qui permet d'éviter le passage par certaines étapes du grafcet (c'est un aiguillage en OU dans lequel une des branches ne comporte **pas d'étape**).
- **Reprise de séquence :** c'est une structure particulière qui permet d'exécuter à nouveau une partie du grafcet (la même séquence peut être reprise une ou plusieurs fois tans qu'une **condition** n'a pas été obtenue).

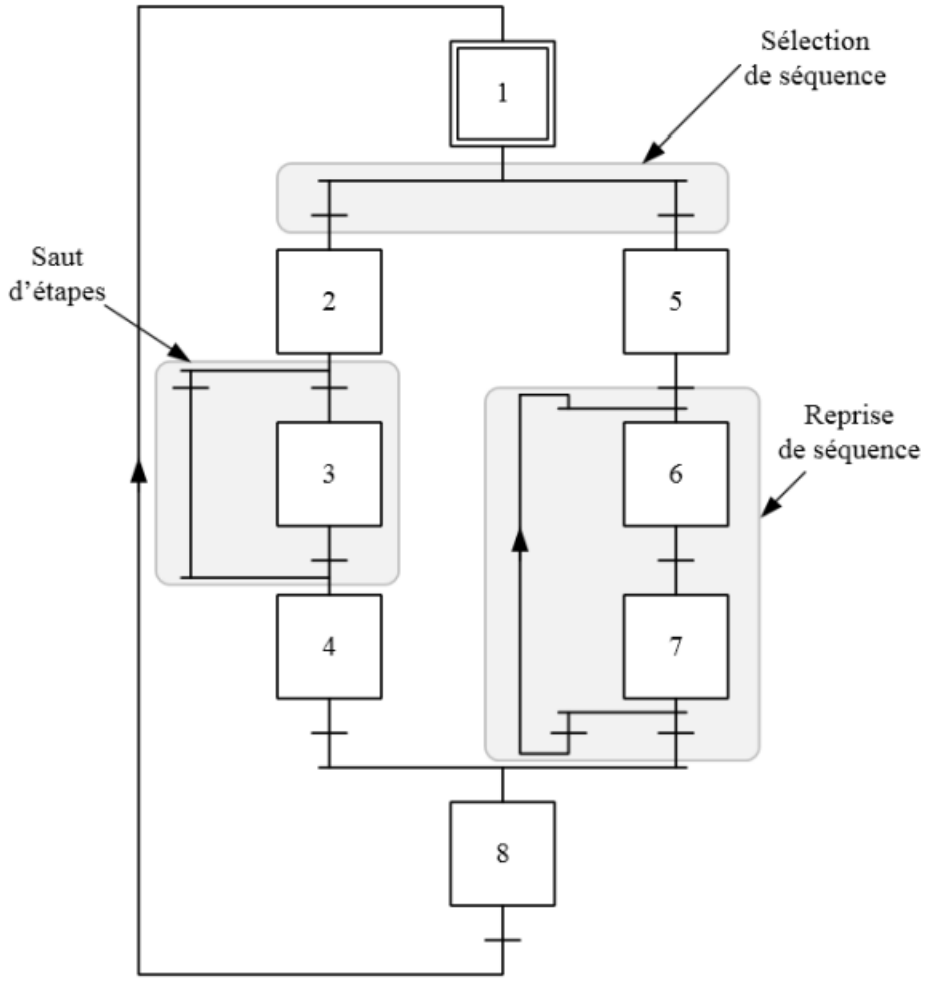


Figure IV-13 : Structure fondamentale d'aiguillage exclusif [5].

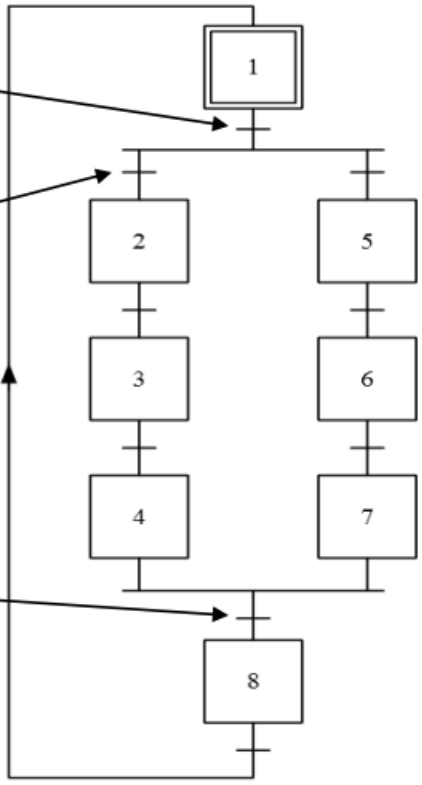
Exercice : Le grafcet suivant comporte plusieurs erreurs, on doit trouver les éléments qui semblent faux ou mal placés.

La première transition n'a pas sa place car il ne peut y avoir qu'une seule transition entre deux étapes.

Comme il s'agit d'une sélection de séquences, les transitions sont placées dans les séquences.

L'avant dernière transition n'a pas sa place ici.

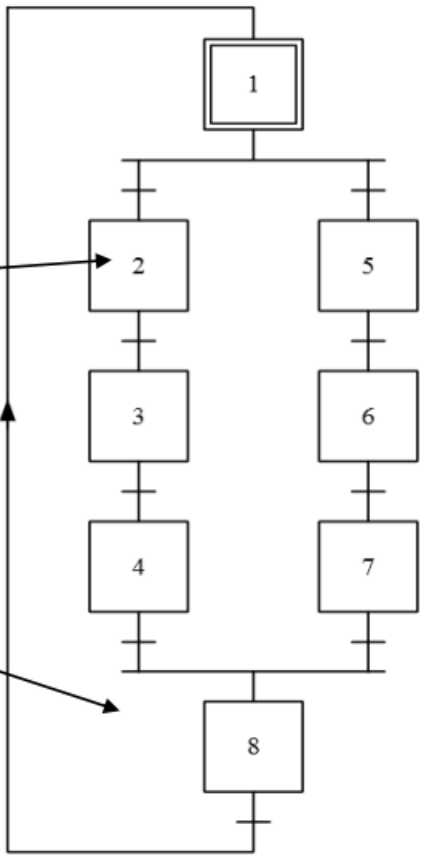
Il y a en amont une convergence en OU après sélection de séquences, les transitions doivent être placées dans les séquences.



Solution :

Donc, chaque séquence débute par une transition et une transition ne peut être commune à deux séquences différentes.

Aussi, chaque séquence doit se terminer par une transition placée avant le symbole de **convergence en OU**.



IV-5-a- Exemple de sélection de séquence [5] :

Un chef de gare commande un poste d'aiguillage ferroviaire. A partir du **tableau de commande** (pupitre), l'opérateur peut envoyer des **consignes** à la partie commande à laquelle il est relié. Cette dernière envoie alors des ordres d'action à la partie opérative, ce qui permet de diriger le train dans la bonne voie.

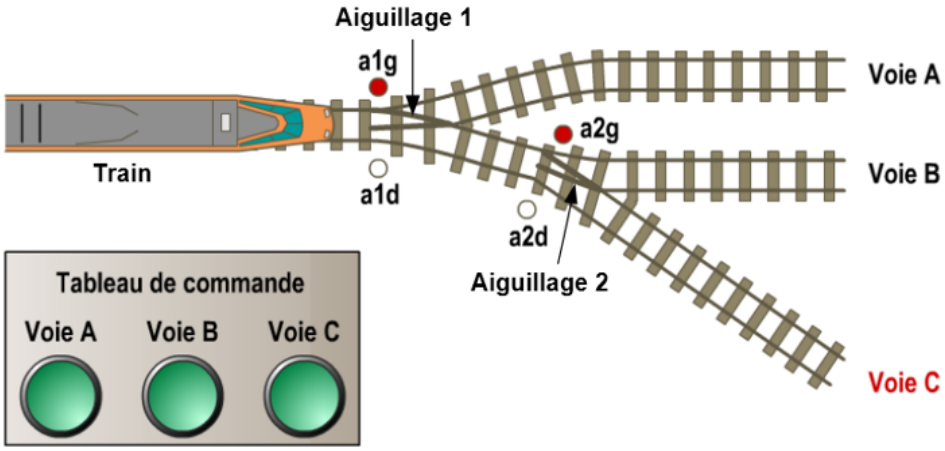


Figure IV-14 : Aiguillage de train à partir d'un tableau de commande [5].

Le grafcet qui gère cet aiguillage est le suivant :

Voir la solution pendant la séance de TD

PA1G : POSITIONNER l'aiguillage 1 à gauche,

PA1D : POSITIONNER l'aiguillage 1 à droite,

a1g : aiguillage n°1 à gauche, **a1d** : aiguillage n°1 à droite,

CI : conditions initiales – considérées comme remplies.

- En définitif, un grafcet est dit à **sélection de séquence** lorsque plusieurs déroulements sont possibles en fonction des transitions proposées ;
- Lorsque l'étape qui précède la **divergence** (l'aiguillage) est active, toutes les transitions qui suivent sont validées ;
- Lorsque l'une des réceptivités associées à ces transitions devient vraie, alors la transition correspondante est franchie et sa branche est exécutée. Les autres branches ne peuvent plus être exécutées puisque l'étape 1 est désactivée ;
- Une structure en sélection de séquence se termine également par une réunion des branches grâce à une **convergence**.

IV-5-b- Exemple de saut d'étapes [5] :

Soit une machine de peinture et d'étiquetage de boîtes :

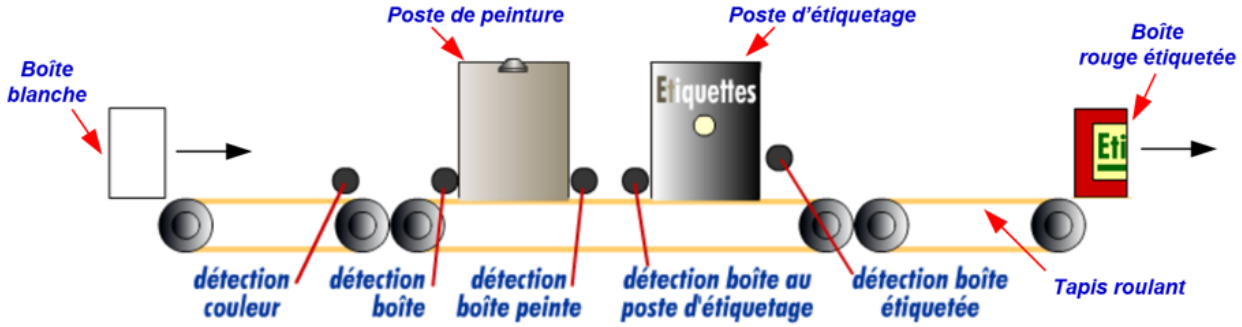


Figure IV-15 : Unité de peinture et d'étiquetage de boîtes [5].

Le grafcet proposé correspond à un saut d'étape. Lorsque les boîtes sont rouges, elles ne sont pas peintes une seconde fois. Si par contre, les boîtes sont blanches, elles sont peintes en rouge. Toutes les boîtes sont ensuite étiquetées.

Le saut d'étapes est une structure particulière de la sélection de séquence permettant, selon les cas, de ne pas exécuter certaines actions.

Voir la solution pendant la séance de TD

IV-5-c- Exemple de reprise de séquence [5] :

Soit la représentation schématique d'un chargement de plateau (fig. IV-16). Dès que 05 caisses sont traitées, le chargement est terminé.

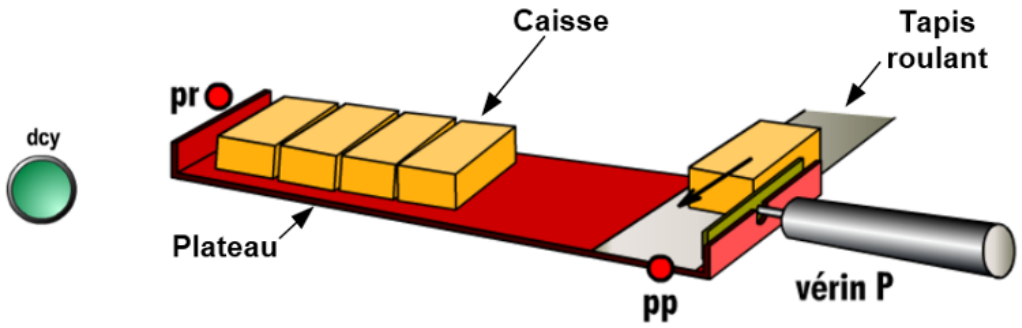


Figure IV-16 : Système de chargement de plateau [5].

Le tapis est mis en mouvement pour déplacer une caisse jusqu'à ce que le **capteur pp** soit actionné.

La tige du vérin P sort et pousse la caisse sur le plateau. Dès que la tige est totalement sortie, l'ordre lui est donné de rentrer.

Tant que le **capteur pr** n'est pas actionné, le cycle d'aménagement (acheminement) de caisses reprend pour apporter une nouvelle caisse.

Dès que **pr** est actionné, la partie commande est informée que le chargement du plateau est terminé.

Le grafcet qui gère le chargement de plateau avec 05 caisses par exemple est donné comme suit :

Voir la solution pendant la séance de TD

- Le premier élément à placer dans ce grafcet est l'étape initiale (1 par exemple) ;
- Suit la première transition ; L'équation logique à écrire prend en compte plusieurs éléments :
  - Tout d'abord, la **tige du vérin P** doit être **rentrée** et l'opérateur doit appuyer sur le **bouton de départ de cycle** (tige P rentrée et dcy) ;
  - Mais, il convient également de vérifier qu'**aucune caisse** n'est déjà devant le **capteur pp** et qu'aucune caisse n'est déjà devant le **capteur pr**.
  - L'équation complète est : tige P rentrée et dcy et non pp et non pr.
- Le tapis va être en mouvement ;
- Puis suit la réceptivité pp ;
- La tige du vérin P va sortir pour pousser la caisse ;
- Puis réceptivité tige P sortie ;
- Ensuite, il faut que la tige du vérin P rentre ;
- A ce moment, y a-t-il 5 caisses de placées ou moins de 5 ?

  Pour permettre le traitement des 2 cas, on place une sélection de séquences à l'issue de l'étape 4.
  - Dans le cas où il y a moins de 5 caisses (branche gauche), la réceptivité est : non pr et tige P rentrée (pr n'est pas vraie).
  - Par contre, s'il y a 5 caisses (branche de droite), la réceptivité est : pr et tige P rentrée (pr est vraie)
- Dans le cas où la transition de la branche gauche sera franchie (mois de 5 caisses), il faut une **liaison orientée** pour exécuter à nouveau une partie des actions du cycle. La transition doit être reliée à l'étape 2, ce qui permettra d'exécuter à nouveau les étapes 2 à 4.
- Pour la branche de droite, il nous faut là encore une liaison orientée pour exécuter à nouveau le cycle complet (transition reliée à l'étape initiale).

IV-6- Séquences simultanées [5] :

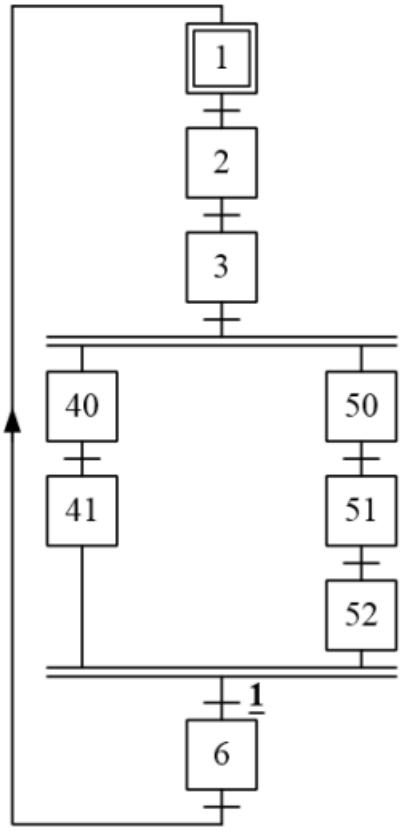
Un automatisme est représenté par un grafcet à séquences **simultanées** ou **parallélisme** lorsque cet automatisme possède plusieurs séquences qui se **déroulent en même temps**.

Le symbole grafcet qui représente les séquences simultanées est : ═══

Le grafcet ci-contre est un grafcet à séquences simultanées car dès que la réceptivité après l'étape3 par exemple est vraie, les étapes 40 et 50 sont activées simultanément (**divergence en ET**) et les 2 séquences doivent être terminées pour activer l'étape 6.

Les 2 séquences **évoluent indépendamment**. Il est donc nécessaire qu'elles s'attendent. C'est le rôle des étapes 41 et 52 (se sont des étapes d'**attente** ou de **synchronisation**).

Dès qu'elles sont toutes les deux actives, la transition qui suit doit être franchie. Pour cela, la réceptivité doit être **vraie**, on la note par **1**.



Exemple : Soit la représentation schématique d'un palettiseur (fig. IV-17). Le système traitant le grafcet à reprise de séquence (chargement de plateau – fig. IV-16) est à présent complété d'un ensemble de 03 vérins T, C et M permettant le déchargement du plateau et le transfert des caisses vers une palette.

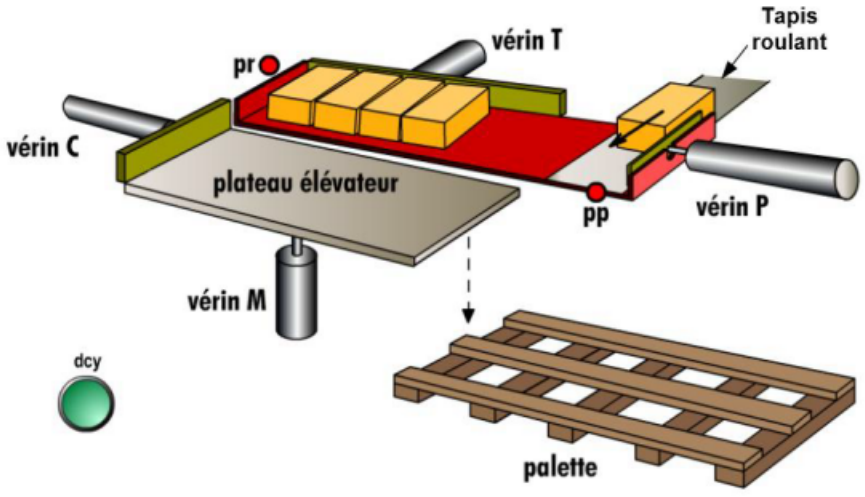


Figure IV-17 : Système de palettisation de caisses [5].

Lorsque 5 caisses sont préparées, le vérin T les pousse sur le plateau élévateur.

Le vérin M amène alors le plateau au niveau de la palette et le vérin C peut pousser l'ensemble de caisses sur la palette.

Le grafcet proposé pour le chargement de plateau peut être développé par l'utilisation d'un grafcet à séquences simultanées (ci-dessous).

Voir la solution pendant la séance de TD

A la première transition, l'équation logique prend en compte :

- L'état du **capteur pp** (non pp) ce qui permettra l'acheminement des caisses par le tapis roulant,
- **Table en haut** (plateau élévateur en position haute),
- L'opérateur doit appuyer sur le bouton de **départ de cycle** (dcy).

Nous avons ensuite, deux étapes : SORTIR et RENTRER la tige du vérin T ; Ce qui permettra de vider le plateau et de charger le plateau élévateur.

Une fois la réceptivité **T rentrée** est vraie, on passe aux séquences simultanées (**divergence en ET**). La branche de gauche permettra le chargement du plateau ; et celle de droite la descente et la montée du plateau élévateur tout en le vidant des caisses par la tige C.

Les deux branches sont terminées par les étapes d'attentes (33 et 44). A la suite de ces 2 étapes, on peut placer un symbole de synchronisation (**convergence en ET**) suivie d'une transition toujours vraie 1 pour franchir les 2 branches.

Ainsi, les **étapes d'attente synchronisent** la fin des séquences simultanées. Tant que les 2 étapes d'attente 33 et 44 ne sont pas **actives en même temps**, la transition qui suit n'est pas franchie.

IV-7- Temporisation [5] :

La temporisation (**fonction retard**) est une fonction dans laquelle intervient le temps de façon que la transition soit retardée pendant une durée définie.

Exemple : Soit un système de tri de pièces en fonction de leur couleur (fig. IV-18).

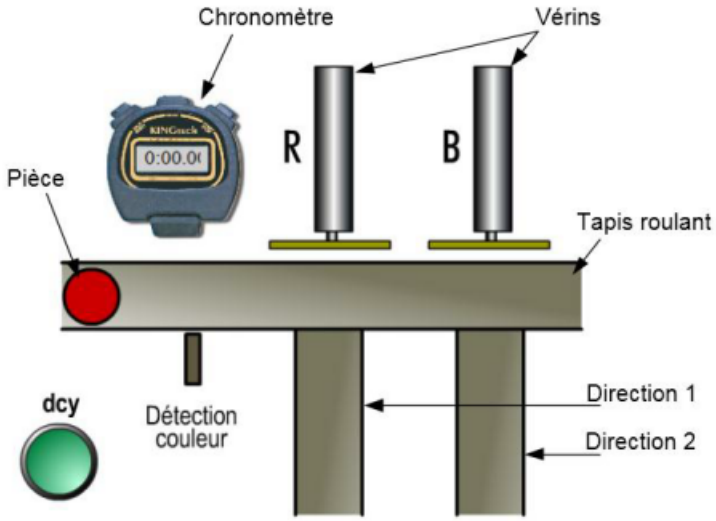


Figure IV-18 : Système de tri de pièce [5].

Ce système permet le tri de pièce de couleur rouge ou bleue en utilisant un détecteur de couleur. Une fois la pièce rouge arrive au niveau du détecteur, le tapis roulant va déplacer la pièce pendant 3 s et s'arrêter. Ce temps est nécessaire pour qu'elle arrive au niveau du vérin R. Ce dernier poussera la pièce vers la direction 1. Si la pièce est bleue le tapis roulant s'arrêtera après 6 s pour qu'elle arrive au niveau du vérin B qui la poussera vers la direction 2.

Le grafcet proposé introduit une sélection de séquence (divergence en OU) avec temporisation.

Voir la solution pendant la séance de TD

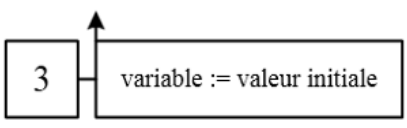
Dans la branche gauche (transition : pièce rouge), on a une temporisation de 3 s à l'étape 30. Par contre, Dans la branche droite (transition : pièce bleue), on a une temporisation de 6 s à l'étape 40. La dernière transition sera franchie que lorsque la durée de 2 s sera écoulée.

Sur le grafcet, X est le repère de l'étape. X3 par exemple, signifie que l'automatisme a donné l'ordre de lancer la temporisation à l'activation de l'étape 3.

Pour séparer les 2 parties de la notation (**durée** et **étape** de lancement), on utilise la barre de fraction /.

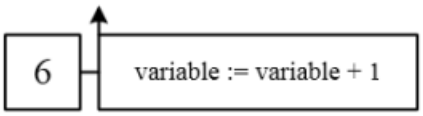
IV-8- Comptage [5] :

- Pour réaliser un comptage dans un grafcet, il faut définir une **variable numérique** qui est **initialisée** dans une **action à l'activation**. L'initialisation consiste à fixer une **valeur de départ**. Cette valeur est souvent zéro « 0 », mais peut être un tout autre nombre.
- La **flèche placée sur le cadre** indique qu'il s'agit d'une **action à l'activation**.
- La variable est définie par une opération d'**affectation** dont le symbole est **:=**

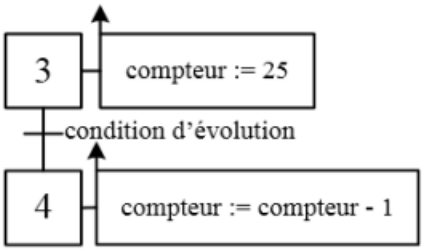


- L'étape d'initialisation peut être suivie d'une **transition toujours vraie** (mais ce n'est pas une obligation). Cette écriture peut être employée pour indiquer que si l'étape est active, la transition qui suit est automatiquement franchie.

- Lorsque les conditions sont remplies, le compteur est **incrémenté** dans une **action à l'activation**. Cela signifie que chaque fois que l'étape est activée, le compteur est incrémenté d'une valeur. L'incrément porte généralement la valeur 1, mais il pourrait être de n'importe quelle valeur : variable := variable + 5.



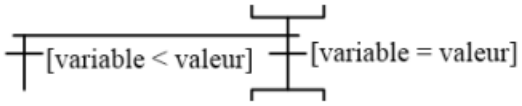
Si l'application le nécessite, il est tout aussi possible de **décrémenter** un compteur qui aurait été initialisé à une valeur supérieure à 0.



- La valeur de la variable numérique est testée (**exploitation** du résultat du comptage) dans une réceptivité :

[nom de la variable opérateur de test valeur de test]

Le test est écrit entre **deux crochets**. Souvent, on retrouve ces tests dans des **sélections de séquence**. Chaque transition permet d'orienter l'évolution du grafcet en fonction de la valeur du compteur.



- Les opérateurs de tests les plus fréquents sont :

[variable	**=**	**valeur]**
[variable	**≠**	**valeur]**
[variable	**>**	**valeur]**
[variable	**<**	**valeur]**
[variable	**≥**	**valeur]**
[variable	**≤**	**valeur]**

La réceptivité associée à la transition est vraie lorsque la condition sur la variable numérique est vraie (compteur = 10 ou compteur > 10, etc.)

Tant que cette condition n'est pas remplie, la réceptivité reste fausse.

Exemple : Le grafcet suivant est relatif à une machine qui évacue les pièces issues d'une chaîne de production. Les pièces sont évacuées par lots de 10 unités.

L'étape 2 de ce grafcet est l'étape d'initialisation d'un compteur dont l'écriture est Compteur := 10 (**affectation**).

Après l'étape 4, on a une sélection de séquence. Pour passer à l'étape 5, le compteur doit contenir la valeur 0 (Compteur = 0).

Dans l'étape 4, on doit procéder au **décompte** des pièces qui sont passées. Etant donnée la valeur d'initialisation du compteur, on doit écrire Compteur := Compteur – 1.

Pour le test de la branche de gauche qui provoque le rebouclage de l'étape 3, on doit mettre Compteur > 0, c'est-à-dire que tant que le compteur est supérieur à 0 on aura une reprise de séquence.

Voir la solution pendant la séance de TD

(Ctrl)

Bibliographie

[01]- P. JACQUARD, S. SANDRE, " Automates programmables industriels ", Ed. PYC, Paris, Mai 1993.

[02]- A. SIMON, " Automates programmables industriels ", Ed. L'Elan – EYROLLES, Paris, 1991.

[03]- A. RIDEAU & al. , " La technologie des systèmes automatisés ", Ed. DELAGRAVE, Paris, Novembre 1992.

[04]- S. MORENO, E. PEULOT, " Le GRAFCET : Conception-Implantation dans les automates programmables industriels ", Ed. CASTEILLA, France, Avril 2009.

[05]- T. SCHANEN, " Guide des automatismes ", Ed. POS Industry, France, 2007.

[06]- J. PERRIN & al. , " Automatique et informatique industrielle ", Ed. Nathan Technique, Paris, Septembre 2009.

[07]- J. HÉNG, " Pratique de la maintenance préventive ", Ed. Dunod, Paris, 2002.

[08]- F. CASTELLAZZI & al. , " MÉMOTECH – Maintenance industrielle ", Ed. CASTEILLA, Paris, Septembre 1998.

[09]- J. L. FANCHON, " Guide des sciences et technologies industrielles ", Ed. Nathan, Paris, Avril 2001.

[10]- R. PRIGENT, M. AUCLERC, " Aide-mémoire – Régulation et automatisme des systèmes frigorifiques ", Ed. Dunod, Paris, 2010.

[11]- D. DUPONT, D. DUBOIS, " Réalisation technologique du GRAFCET ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle, S 8032, Mars 2002.

[12]- N. JOUVRAY, " Langage de programmation pour systèmes automatisés : Norme CEI 61131–3 ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle S 8030v2, Mars 2008.

[13]- M. BERTRAND, " Automates programmables industriels ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle S 8015v2, Décembre 2010.

[14]- P. VIDAL, " Informatique industrielle - Introduction ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle S 7000, Mars 2000.

[15]- P. BRARD, " Outil de description des automatismes séquentiels : le GRAFCET ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle R 7250, Janvier 1988.

[16]- J-J. DUMÉRY, " GRAFCET – Concept de base ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle S 7240, Mars 2007.

[17]- J-J. DUMÉRY, " GRAFCET – Structuration des descriptions. Applications ", Ed. Les Techniques de l'Ingénieur, traité informatique industrielle S 7241, Juin 2007.

Webographie

[W1]- " Photographie d'un pupitre de commande " ; prise du site officiel d'EMG (Entreprise de la Mécanique Générale) ; consulté le 14/03/2015.

http://www.emg61.fr/automatisme.html

[W2]- " Photographies d'automates programmables monobloc et modulaire " ; prises du site Technologue pro – Ressources pédagogiques pour l'enseignement Technologique - cours Automatisme et Informatique Industrielle ; consulté le 24/01/2015.

http://www.technologuepro.com/cours-automate-programmable-industriel/Les-automates-programmables-industriels-API.htm

[W3]- " Photographie d'une console de programmation " ; prise du site BS'Automates ; consulté le 27/03/205.

http://www.obsautomates.fr/product.php?id_product=16