



الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique



UNIVERSITE DES FRERES MENTOURI CONSTANTINE 1
Institut Des Sciences Et Techniques Appliquées

جامعة الاخوة منتوري قسنطينة 1
معهد العلوم والتقنيات التطبيقية

Polycopié de cours: Bases de l'automatisme

Ce cours destiné aux étudiants de la première année LMD de l'institut des sciences et techniques appliquées (ISTA).

Réalisé par :

Adel BOUCHAHED

Docteur en Électromécanique - Maitre de conférences B

Années universitaires 2016-2021

Objectifs :

Au terme de ce cours, l'apprenant capable de:

- ✓ Connaître les bases de l'automatisme ;
- ✓ Connaître la structure d'un système automatisé et ses composants de bases ;
- ✓ Modéliser un système combinatoire sous une forme d'expressions booléennes puis effectuer une simplification ;
- ✓ Mettre en œuvre un ensemble d'équations de commande sous la forme de logique câblée et/ou programmée ;
- ✓ Identifier un système séquentiel.

Présentation du module

Ce cours est une partie d'Unité d'enseignement Découverte (UE1.2) du Génie Électrique, destiné aux étudiants de la première année licence professionnelle de l'Institut des Sciences et Techniques Appliquées (ISTA), spécialité Génie Industriel et Maintenance (GIM).

Compétences visées :

Choisir, mettre en place et assurer la mise au point de systèmes automatisés.

Responsable du module:

Dr. BOUCHAHED Adel (bureau ISTA)

Email : adel.bouchahed@gmail.com, bouchahed.adel@umc.edu.dz

Heures de réception : Lundi 11h.00 à 12h.30 et Jeudi de 09H30 à 12h.30.

Prérequis :

Baccalauréat ou équivalent.

Volume horaire:

6h CM, 9h TD, 15h TP / semestre 1.

Coefficient :

- 2.

Contenus:

Outils initiaux de l'automatisme : algèbre de Boole, numération, simplifications, logique combinatoire et séquentielle, GRAFCET, GEMMA. Structure fonctionnelle d'un système automatisé, partie opérative & partie commande. Capteurs, actionneurs et systèmes d'identification pour l'automatisme. Initiation au principe de fonctionnement d'un automate programmable, éléments de langage de programmation.

Modalités de mise en œuvre :

Cours et TD définitions des systèmes automatisés et leurs parties essentielles, présentations des installations de mécanique automatisées industrielles, exercices d'applications sur la programmation des automates programmables industriels par différents

langages en prenant des exemples réels. Ce qui concerne les TP l'étudiant doit être réalisé ses travaux pratiques au niveau des ateliers comme suit :

- ✓ TP sur les portes logiques (utilisation des circuits logiques et des logiciels comprenant les portes logiques) ;
- ✓ TP sur les capteurs et les actionneurs ;
- ✓ TP sur les langages de programmation des API ;
- ✓ TP sur les platines didactiques de câblage (pour maîtriser la logique câblé);
- ✓ TP sur les systèmes automatisés composées d'un automate programmable (cellule pneumatique commandée par API et démarrage deux sens de rotation du moteur asynchrone à cage d'écureuil commandé par API).

5 TP de 3 h par permutation circulaire soit 15 h de TP.

Table des Matières

République algérienne démocratique et populaire.....	I
Ministère de l'enseignement supérieur et de la recherche scientifique.....	I
Université frères Mentouri Constantine 01.....	I
Institut des sciences et techniques appliquées.....	I
Département électromécanique.....	I
Objectifs.....	II
Présentation du module.....	III
Contenus.....	III
Modalités de mise en œuvre.....	III
I. Définition.....	01
I.1. Qu'est qu'un système ?.....	01
I.2. Système technique.....	01
I.3. Système automatisé.....	01
I.4. Objectif de l'automatisation.....	01
I.5. Les parties d'un système automatisé.....	01
I.5.1. La partie opérative (PO)	02
I.5.2. La partie commande (PC)	04
I.5.3. La partie relation (PR)	04
I.6. Différents types de commande des systèmes automatisés.....	05
I.6.1. Système automatisé combinatoire.....	05
I.6.2. Système automatisé séquentiel.....	05
II. Système de numération.....	05
II.1. Définition.....	05
II.2. Différents systèmes de numération.....	05
II.2.1. Système décimal.....	05
II.2.2. Système binaire.....	06
II.2.3. Système octal.....	06
II.2.4. Système hexadécimal.....	07
II.3. Transformations (conversions)	08
II.3.1. Décimal / Binaire.....	08
II.3.2. binaire/décimale.....	10
II.3.3. Binaire /Octale - Binaire / Hexadécimale.....	11
II.4. Codage.....	11
II.4.1. Codes numériques.....	11
II.4.1.1. Code binaire naturel.....	11
II.4.1.2. Code binaire réfléchi (Code GRAY)	12
II.4.1.3. Code DCB 8421 (Décimal Codé Binaire)	13

II.4.1.4. Code à excès de trois (ou code de STIBITZ).....	13
II.4.1.5. Le code Aïken.....	14
II.4.1.6. Les codes « 2 parmi 5 »	15
III. Algèbre de Boole et simplification des fonctions logiques.....	16
III.1. Définition.....	16
III. 2. Propriétés de base de l'algèbre de Boole.....	16
III.3. Table de vérité et équation logique.....	17
III.4. Table de KARNAUGH.....	18
III.4.1. Définition.....	18
III.4.2. Table de KARNAUGH à trois variables.....	18
III.4.3. Table de KARNAUGH à quatre variables.....	18
III.4.4. Exemples d'applications.....	18
IV. Programmation des automates programmables par l'application de la logique combinatoire et séquentielle.	20
IV.1. Définition.....	20
IV.2. Différents langages de programmation.....	21
IV.3. Langage à contact.....	21
IV.3.1. Représentation des éléments principaux pour un réseau LD (LADDER).....	21
IV.3.2. Exemples d'applications sur langage à contact (Ladder).....	23
IV.4. Langage liste instruction.....	26
IV.4.1. La fonction égalité.....	26
IV.4.2. La fonction négation ou complémentation.....	27
IV.4.3. La fonction intersection ou multiplication logique.....	28
IV.4.4. La fonction réunion ou addition logique.....	29
IV.4.5. La fonction NON ET (NAND)	30
IV.4.6. La fonction NON OU (NOR)	31
IV.4.7. La fonction ou exclusif XOR.....	32
IV.4.8. La fonction coïncidence (L'OU-EXCLUSIF-NON).....	33
IV.4.9. Représentation des éléments principaux pour langage liste instruction.....	34
IV.4.10. Exemples d'applications sous langage liste.....	35
V. Langage grafcet.....	37
V.1. Définition.....	37
V.2. Types du grafcet.....	38
V.2.1. Grafcet de niveau 01 (grafcet-partie système).....	38

V.2.2. Grafcet de niveau 02 (grafcet-partie opérative).....	38
V.2.3. Grafcet de niveau 03 (grafcet/ partie commande).....	38
V.3. Exemples d'applications sur le grafcet unique.....	38
V.4. Reprise d'une séquence.....	50
V.5. Saut d'étape.....	50
V.6. Aiguillage en OU.....	51
V.6.1. Exemples d'applications sur le grafcet en OU.....	51
V.6.2. Aiguillage en ET.....	54
VI. Guide d'étude des modes de marches et d'arrêts (GEMMA).....	59
VI.1. Définition.....	59
VI.2. Représentation graphique du GEMMA.....	59
VI.3. Structuration du GEMMA.....	60
VI.3.1. Partie commande hors énergie (PZ).....	60
VI.3.2. Partie commande sous énergie.....	60
VI.3.2.1. Procédures de fonctionnement (F).....	61
V.3.2.2. Procédures en défaillances (D)	61
V.3.2.3. Les procédures d'arrêts (A).	62
VI.4. Les rectangles états.....	63
Bibliographie	67
Sites Web	68
Liste des tableaux	70
Liste des figures	71

I. Définition

Actuellement, les systèmes de production automatisés fonctionnent en haute technologie qui permet aux entreprises industrielles d'améliorer la qualité de produit, augmenter la production, diminuer le risque d'accident, éliminer les tâches pénibles et aussi d'améliorer la sécurité de l'entreprise... etc. Un système de production automatisé (SAP) constitue par plusieurs éléments et machines liés entre eux, commandés par un programme d'automate programmable, ce dernier permet une dynamique élevée des éléments en interaction.

I.1. Qu'est-ce qu'un système ?

C'est un ensemble d'éléments en interaction dynamique, organisé en fonction d'un but.

Pour cela :

- Un système est organisé,
- Un système est finalisé,
- Un système est constitué d'éléments et de leurs relations.

I.2. Système technique : La norme NFE 90-001 définit un système technique comme un ensemble d'éléments interconnectés de façon logique, qui se coordonnent pour réaliser une tâche précise".

I.3. Système automatisé : Un système est dit automatique s'il réalise la fonction seul, sans intervention humaine.

I.4. Objectifs de l'automatisation

- **Visant le personnel :** Améliorer ses conditions de travail en supprimant les tâches les plus pénibles et en augmentant la sécurité;
- **Visant le produit :** Améliorer sa faisabilité, sa qualité par rapport au cahier des charges, sa fiabilité dans le temps;
- **Visant l'entreprise :** Améliorer sa compétitivité (en diminuant les coûts de production), sa productivité, la qualité de production, la capacité de contrôle, de gestion, de planification.

I.5. Les parties d'un système automatisé

Un système automatisé peut, pour faciliter l'analyse, se représenter sous la forme d'un schéma identifiant trois parties (P.O ; P.C ; P.P) du système et exprimant leurs interrelations (Informations, Ordres, Comptes rendus, Consignes).

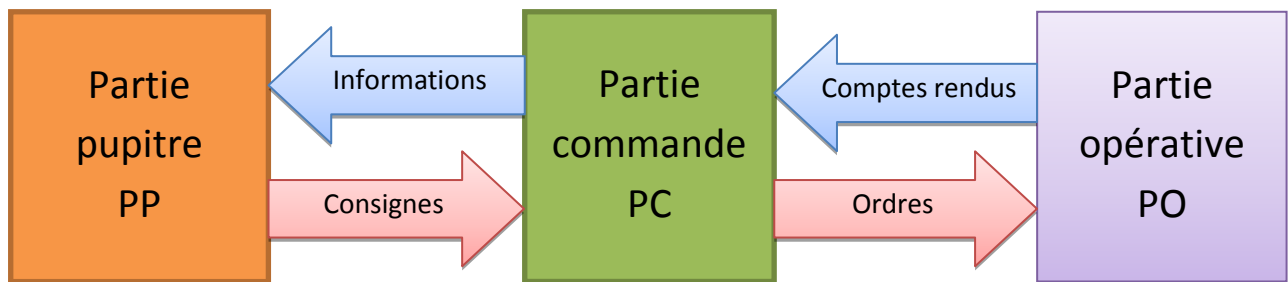


Figure 1. Les parties d'un système automatisé.

I.5.1. La partie opératives (PO)

C'est la partie visible du système. Elle comporte les éléments mécaniques du mécanisme avec :

Des *pré-actionneurs* (distributeurs, contacteurs), lesquels reçoivent des ordres de la partie commande; voir les figures suivantes :



Figure 2.a. Pré-actionneur pneumatique



Figure 2.b. Pré-actionneur électrique

Figure 2. Les pré-actionneurs.

Des *actionneurs* (vérins-moteurs) qui ont pour rôle d'exécuter ces ordres. Ils transforment l'énergie pneumatique (air comprimé), hydraulique (huile sous pression) ou électrique en énergie mécanique. Ils se présentent sous différentes formes comme :

Moteurs: hydraulique, pneumatique, électriques, vérins : linéaires (1 ou 2 tiges) rotatifs, sans tige.



Figure 3.a. Actionneur pneumatique (Vérin)

Figure 3.b. Actionneur électrique (MAS 3~)

Figure 3. Les actionneurs.

Des *capteurs* qui informent la partie commande pour l'exécution du travail. Ils existent sous différents types comme :

Capteurs mécaniques, pneumatiques ou électriques;

Capteurs magnétiques montés sur les vérins,

Capteurs pneumatiques à chute de pression.



Figure 4. Les capteurs.

Dans un système automatisé de production, ce secteur de détection représente le service de surveillance et renseignement du mécanisme. Il contrôle, mesure, surveille et informe la PC sur l'évolution du système.

I.5.2. La partie commande (PC)

Ce secteur de l'automatisme gère dans la suite logique le déroulement ordonné des opérations à réaliser. Il reçoit des informations en provenance des capteurs situés dans la PO, et les restitue vers cette même PO en direction des pré-actionneurs (distributeurs).

L'outil de description s'appelle GRAFCET (Graphe de Commande Étape et Transaction).



Figure 5. Automate programmable associe avec ordinateur.

I.5.3. La partie Relation (PR)

Sa complexité et sa taille dépendent de l'importance du système. Il regroupe les différentes commandes nécessaires au bon fonctionnement du procédé : marche-arrêt, arrêt d'urgence, marche automatique, marche cycle/cycle...

L'outil de description s'appelle « GEMMA » (Guide d'Étude des Modes de Marches et Arrêts).

Ces outils graphiques (GRAFCET et GEMMA) sont utilisés également par les techniciens et les ingénieurs de maintenance, pour la recherche des pannes sur les SAP (Système Automatisé de Production).

Pendant le fonctionnement, un dialogue continu s'établit entre les trois secteurs du système, permettent ainsi le déroulement correct du cycle défini dans le cahier de charges.



Figure 6. Pupitre de commande.

I.6. Différents types de commande des systèmes automatisés

Dans les systèmes automatisés de production il existe plusieurs types de commande

I.6.1. Système automatisé combinatoire

Les systèmes automatisés combinatoires sont des systèmes simples à utiliser, parce que ces derniers n'ont pas de mémoires. Ils ont fonctionné à l'aide de la logique combinatoire où nous avons utilisé l'algèbre de Boole qui est l'outil mathématique le plus utilisé pour concevoir les tables de Karnaugh et les tables de vérité à partir des équations logiques des actionneurs, dans les systèmes combinatoires une combinaison des variables d'entrées fait activer un seul état de sortie. Actuellement ils ne sont pas beaucoup dans le domaine industriel à cause de ne peuvent pas faire plusieurs actions.

I.6.2. Système automatisé séquentiel

Actuellement dans le domaine industriel la plupart des systèmes automatisés sont de type séquentiel. Ils ont effectué les opérations étape par étape, par ailleurs les entrées de ces systèmes dans lesquelles les sorties sont déterminées au même temps, cela signifie que la même combinaison d'entrée ne commande pas toujours les mêmes sorties.

Pour commander les systèmes automatisés séquentiels on utilise deux types de logique :

- La logique câblée : peut-être électrique ou pneumatique ;
- La logique programmée : peut-être par un programme d'automate, microprocesseur, ...etc.

II. Système de numération

II.1. Définition

Le traitement de l'information (Image, mot, nombre,...etc.) est effectué par l'ordinateur ou calculateur électronique. Les circuits électroniques qui les constituent ne peuvent prendre que deux états représentés par **0** et **1**. Donc, le langage utilisé dans ces machines est appelé système de numération binaire (**suite de 0 et de 1**).

II.2. Différents systèmes de numération

Il existe quatre systèmes de numération qui sont :

- Système décimal ;
- Système binaire ;
- Système octal ;
- Système hexadécimal.

II.2.1. Système décimal

C'est un système utilisé dans le monde extérieur, les chiffres utilisés sont des entiers composés par des valeurs de 0 à 9. La base de ce système est 10.

Exemple - 01

1995 : ce nombre s'écrit dans le système à base 10 sous la forme :

$$1 * 10^3 + 9 * 10^2 + 9 * 10^1 + 5 * 10^0$$

II.2.2. Système binaire

C'est le système qui est utilisé par la machine (l'ordinateur), il est composé seulement par des suites binaires (**1 et 0**). La base de ce système est 2.

Exemple - 02

1001101 : la valeur décimale de cette suite est :

$$1 * 2^6 + 0 * 2^5 + 0 * 2^4 + 1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = 77$$

Chaque chiffre binaire (**0 ou 1**) est appelé bit qu'on note par b, et toutes combinaisons binaires composées de huit bits est appelée Octet.

Exemple – 03: 11100100

$$B_0 = 0, b_1 = 0, b_2 = 1, b_3 = 0, b_4 = 0, b_5 = 1, b_6 = 1, b_7 = 1$$

Seize (16) bits est appelée « mot-mémoire » ou bien mot machine

Exemple – 04 : 1110001010100110

$$B_0 = 0 \quad b_1 = 1 \quad b_2 = 1 \quad b_3 = 0 \quad b_4 = 0 \quad b_5 = 1 \quad \dots \quad b_{14} = 1 \quad b_{15} = 1$$

II.2.3. Système octal

Le système décimal est celui qui est le plus utilisé par l'homme, mais le système binaire est plus utilisé dans les ordinateurs.

Le système binaire présente l'inconvénient d'être difficile à manipuler par l'homme car les nombres sont représentés par une suite de 0 et 1 qu'il est difficile de lire ou d'interpréter sans erreur. Il est donc plus commode d'écrire les nombres codés en binaire sous une forme plus compacte, dans un système dont la base est une puissance de 2, de façon à permettre une conversion facile avec le système binaire.

Nous définissons dans ce paragraphe le système octal, et plus loin le système hexadécimal.

Certains calculateurs numériques utilisent le système octal qui est un système de numération dont la base 8.

On dispose de huit symboles pour représenter un nombre dans le système octal. Ce sont les chiffres successifs allant de 0 à 7. Chaque chiffre d'un nombre octal a une pondération égale à la valeur du chiffre multipliée par la puissance de 8 correspondant au rang qu'il occupe dans le nombre.

Exemple – 05 : $N = 536_8$

- Chiffre	5	3	6
- Puissance	8^2	8^1	8^0
- Pondération	320	24	6

L'addition des pondérations donne l'équivalent décimal du nombre octal considéré.

$$320+24+6 = 350$$

Soit $536_8 \Rightarrow 350_{10}$

II.2.4. Système hexadécimal

Ce système est utilisé sur la plupart des nouveaux calculateurs numériques. Sa base est égale à 16. On dispose de 16 symboles pour représenter un nombre hexadécimal. Ces symboles sont les **10 digits** du système décimal auxquels on a ajouté les 6 premières lettres de l'alphabet. Les 16 symboles sont alors :

0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F.

Chaque signe d'un nombre hexadécimal a une pondération qui s'obtient en multipliant la valeur numérique du symbole par la puissance de 16 correspondant au rang qu'il occupe dans le nombre.

Exemple – 06 : $N = 4CA2_{16}$

- Signes	4	C	A	2
- Puissance	16^3	16^2	16^1	16^0
- Pondérations	16384	3072	160	2

L'addition des pondérations donne l'équivalent décimal du nombre hexadécimal considéré :

$$16384+3072+160+2 = 19618$$

$$\text{Soit } 4AC2_{16} \Rightarrow 19618_{10}$$

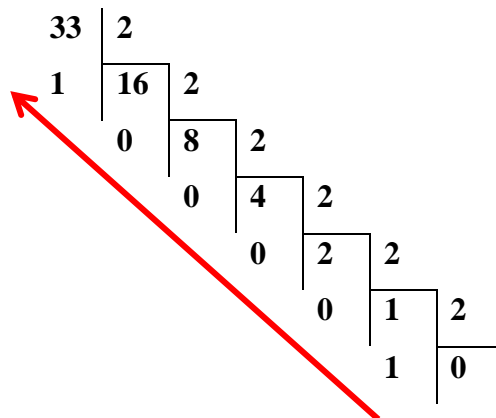
L'intérêt de ces deux systèmes vient de ce que **8 et 16** ont des **puissances entières de** groupe de **3 et 4 bits** et par conséquent, on peut **effectuer les conversions** Octale/Décimale ainsi que Hexadécimale/Décimale. (**Voir Tableau**) :

Nombre décimal	Binaire	octal	Hexadécimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	10001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10
17	10001	21	11
18	10010	22	12
19	10011	23	13
20	10100	24	14

II.3. Transformations (conversions)

II.3.1. Décimal / Binaire

Pour convertir un nombre décimal en un nombre binaire, diviser continûment le nombre décimal par la valeur 2, retenir la suite des restes de chaque division en commençant de bas en haut pour avoir le nombre équivalent.

Exemple :

$$(33)_{10} = (100001)_2$$

Dans le cas où le nombre donné dispose d'une partie fractionnaire, l'opération de conversion se fait de la manière suivante :

- Multiplier la partie, fractionnaire du nombre donné par 2 et retenir la partie entière du résultat trouvé.
- Continuer à multiplier successivement par la valeur 2 les parties fractionnaires obtenues pendant chaque opération tout en retenant leurs parties entières, jusqu'à ce qu'on obtienne une partie fractionnaire nulle ou déjà obtenue, sinon ce sera une suite binaire infinie et dans ce cas , on peut se limiter à quelques bits (problème d'arrondi).
- Le résultat de cette conversion est l'ensemble, des parties entières conservées à chaque étape de multiplication.

Exemple - 07 :

Transformer le numéro 23 en binaire ?

$$(23)_{10} = (10111)_2$$

Exemple – 08 :

Convertir le nombre (23.625) de la base 10 à la base 2.

La partie entière 23 est déjà déterminée, son résultat est : $(23)_{10} = (10111)_2$.

Pour la partie fractionnaire 0.625, on procède comme suit :

$$0.625 * 2 = 1.250$$

$$0.250 * 2 = 0.500$$

$$0.500 * 2 = 1.00$$

On constate que la dernière partie fractionnaire est nulle, donc le résultat sera :

$$(0.625)_{10} = (0.101)_2$$

$$\text{Le résultat final : } (23.625)_{10} = (10111.101)_2$$

Prenons le cas infini par exemple $(23.56)_{10}$

Pour $(23)_{10}$ c'est toujours (10111) : mais pour la partie fractionnaire $(0.56)_{10}$, on aura :

$$0.56 * 2 = 1.12$$

$$0.12 * 2 = 0.24$$

$$0.24 * 2 = 0.48$$

$$0.48 * 2 = 0.96$$

$$0.96 * 2 = 1.92$$

$$0.92 * 2 = 1.84 \text{ etc.,}$$

On remarque, si on continue l'opération de multiplication de la partie fractionnaire par la valeur 2, on ne tombe jamais sur le critère d'arrêt de cette opération, cela veut dire que la suite binaire est infinie et nous pouvons la tronquer en nous limitant à un certain nombre de bits.

D'où le résultat final de : $(23.56)_{10}$ est $(10111.100011 \dots)_2$.

II.3.2. binaire/décimale

Pour convertir une suite binaire en système décimal, on procède de la manière suivante

Faisons l'addition de tous les termes de la forme $(\text{bit} * 2^i)$, en commençant de droite à gauche pour les parties entières et de gauche à droite pour les parties fractionnaires.

« bit » : peut prendre les valeurs 0 ou 1. Le symbole " i " indique les puissances de 2 qui sont positives pour les parties entières et vont de 0 à $(n-1)$ et négatives pour les parties fractionnaires et vont de la valeur (-1) jusqu'à $(-m)$ avec n et m sont respectivement les longueurs de la suite binaire avant et après la virgule.

Exemple - 09 :

$$(10110)_2 = 0 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 1 \cdot 2^4 = (22)_{10}$$

$$(10110.1011)_2 = ?$$

- Pour la partie entière $(10110)_2 = (22)_{10}$
- Pour la partie fractionnaire :

$$(0.1011)_2 = (1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 1 \cdot 2^{-4})$$

Le résultat final :

$$(10110.1011)_2 = (22.6875)_{10}$$

II.3.3. Binaire /Octale - Binaire / Hexadécimale

La conversion d'une suite binaire en système octal et en système hexadécimal, se fait respectivement par le regroupement de trois et de quatre bits, en commençant de droite à gauche, chaque-bloc de bits sera convertible en système demandé

Exemple – 10 :

$$(1010110111)_2 = (?)_8 \text{ et } (?)_{16}$$

En base 8 (regroupement par des blocs de trois bits):

$$(111)_2 = 7 ; (110)_2 = 6 ; (010)_2 = 2 ; (001)_2 = 1 \text{ D'où } (1010110111)_2 = (1267)_8$$

- En base 16 (regroupement par des blocs de quatre bits), |

$$(0111)_2 = 7 ; (1011)_2 = B ; (0010)_2 = 2 \text{ D'où } (1010110111)_2 = (2B7)_{16}$$

II.4. Codage

Pour traiter les informations il existe deux codes numériques qui sont :

- Les codes numériques qui utilisent dans le codage des nombres.
- Les codes alphanumériques qui utilisent dans le traitement d'une information quelconque comme les lettres, les chiffres, les symboles, les images et les programmes... etc.

II.4.1. Codes numériques**II.4.1.1. Code binaire naturel**

Le code binaire naturel est utilisé pour représenter un nombre dans la base de numération binaire.

✓ **Inconvénients du code binaire naturel:**

- La longueur du nombre (nécessite une grande quantité de bits) ;
- Les variables changent d'état entre deux combinaisons successives ;
- Lors du codage ce type de système est non adapté à la correction automatique des erreurs.

II.4.1.2. Code binaire réfléchi (Code GRAY)

Dans ce type de code un seul bit change de valeur entre deux codages successifs. Il se compose de trois bits et quatre bits, comme l'indique sur les deux tableaux :

Tableau 01: Conversion binaire naturel en code binaire réfléchi sur 3bits

Code binaire Naturel.....	..Code binaire réfléchi
Sur 3 bits	
000.....	..000
001.....	..001
010.....	..011
011.....	..010
100.....	..110
101.....	..111
110.....	..101
1 11.....	..100

Tableau 02 : Conversion binaire naturel en code binaire réfléchi sur 4bits

Code binaire Naturel.....	..Code binaire réfléchi
Sur 4 bits	
0	0000..... ..0000
1	0001..... 0001
2	0010..... ..0011
3	0011..... ..0010
4	0100..... ..0110
5	0101..... ..0111
6	0110..... ..0101
7	0111..... ..0100
8	1000..... ..1100
9	1001..... ..1101

10	1010.....	..1111
11	1011.....	..1110
12	1100.....	..1010
13	1101.....	..1011
14	1110.....	..1001
15	1111.....	..1000

II.4.1.3. Code DCB 8421 (Décimal Codé Binaire)

Dans ce code chaque chiffre décimal de 0 à 9 est représenté par 4 bits du binaire pur, avec dans l'ordre les poids 8, 4, 2, 1, c'est-à-dire $8=2^3$

Pour le bit N^0 3, $4=2^2$ pour le bit N^0 2, $2=2^1$ pour le bit N^0 1, $1=2^0$. Pour le bit N^0 0.

Exemple - 11 :

$$1011_{\text{DCB}} = 1*8+0*4+1*2+1*1 = 11_{10}$$

$$32_{10} = 00110010_{\text{DCB}}$$

Exemple – 12 :

$$(1297)_{10} = (0001\ 0010\ 1001\ 0111)_{\text{BCD}}$$

II.4.1.4. Code à excès de trois (ou code de STIBITZ)

Le code à excès de trois (Excès 3 noté XS 3 en abrégé) s'obtient en ajoutant 3 à chaque mot-code du code BCD. Tout comme le BCD, le code à excès de 3 est un code décimal, son tableau de conversion ne concerne donc que les chiffres de 0 à 9. Comme en BCD, pour coder un nombre en code à excès de trois, il faudra

Concaténer une succession de tétrades, traduisant chacune un chiffre du nombre à coder.

Tableau 03 : Conversion décimal en code excès de 3

Décimal	Code à excès de 3	Exemple de codage en XS 3					
0	0011						
1	0100		1	9	8	2	En décimal
2	0101		↓	↓	↓	↓	
3	0110	devient	0100	1100	1011	0101	En XS 3
4	0111						
5	1000						
6	1001		1000	0011	1010	0110	En XS 3
7	1010		↓	↓	↓	↓	
8	1011	devient	5	0	7	3	En décimal
9	1100						

✓ Propriété du code XS 3

Le code à excès de trois a été créé pour permettre la réalisation simple des opérations de soustraction. Le complément à 1 d'un mot-code représente le complément à 9 dans l'ensemble source : les codes possédant cette propriété sont appelés des codes auto-complémentaires.

Exemple avec le chiffre 7 :

En XS 3, le chiffre 7 se code 1010. En XS 3 le complément à 1 de 7 est donc 0101 (complément de chacun des bits). En décimal le complément à 9 de 7 est 2 ($9-7=2$). On remarque que 0101 est bien le mot-code de 2 en XS 3. Et cette propriété d'auto-complémentarité peut être vérifiée pour les 10 chiffres : **Le complément à 1 d'un mot-code en XS 3 correspond au complément à 9 du chiffre en décimal.**

Remarque : le code XS 3 est un code **auto-complémentaire**, mais il n'est pas pondéré.

II.4.1.5. Le code Aïken

Le code Aïken regroupe les deux propriétés des codes BCD et XS 3 précédents : c'est un code décimal pondéré et auto-complémentaire. Les poids des éléments binaires sont **2 4 2 1**. La différence entre le code Aïken et le code BCD est le poids du premier bit à gauche : il valait 8 en BCD alors qu'il vaut 2 en code Aïken.

Tableau 04 : Conversion code décimal en code Aïken

Décimal	Code Aïken	Exemple de codage en code Aïken				
0	0000					
1	0001		1	9	8	2
2	0010		↓	↓	↓	↓
3	0011	devient	0001	1111	1110	0010
4	0100					
5	1011					
6	1100		1011	0100	1101	1100
7	1101		↓	↓	↓	↓
8	1110	devient	5	4	7	6
9	1111					

✓ **Propriété d'auto-complémentarité du code Aïken :**

Exemple avec le chiffre 8 :

- En décimal le complément à 9 de 8 est 1
- En Aïken le complément à 1 de 1110 (mot-code du 8) est 0001 (mot-code du 1) Et cette propriété d'auto-complémentarité peut être vérifiée pour les 10 chiffres : **Le complément à 1 d'un mot-code en code Aïken correspond au complément à 9 du chiffre en décimal.**

II.4.1.6. Les codes « 2 parmi 5 »

Avec les codes « 2 parmi 5 », les mots-code comprennent 5 bits dont 2 sont à 1 (et les 3 autres à 0). Il existe plusieurs codes « 2 parmi 5 », et les plus utilisés sont le code « 8 4 2 1 0 » et le code « 7 4 2 1 0 ». Ces deux codes sont pondérés, la liste des poids figurant dans la dénomination du code.

Tableau 05 : Conversion code décimal en code "2 parmi 5"

décimal	Code "2 parmi 5" 84210	Code "2 parmi 5" 74210
0	10100	11000
1	00011	00011
2	00101	00101
3	00110	00110
4	01001	01001
5	01010	01010
6	01100	01100

7	11000	10001
8	10001	10010
9	10010	10100

Les codes « 2 parmi 5 » font partie des codes spécialement conçus pour la transmission de l'information et pour la détection des erreurs. En effet, si on reçoit un nombre codé en « 2 parmi 5 », pour détecter une éventuelle erreur dans ce nombre il suffit de compter le nombre de 1 logique présente dans chacun des groupes de 5 bits. Si un groupe ne présente pas deux 1 logiques, on peut en déduire avec certitude qu'il est erroné.

Remarque : contrairement à d'autres codes plus perfectionnés (code de Hamming par exemple), les codes « 2 parmi 5 » permettent de détecter une erreur, mais ne permettent pas de la corriger. De plus, si lors de la transmission, 2 bits de valeurs différentes changent simultanément d'état, aucune erreur ne pourra être détectée à l'arrivée.

III. Algèbre de Boole et simplification des fonctions logiques :

III.1. Définition

Dans cette partie on va étudier la logique utilisée dans les systèmes automatisés (traitement des données et des signaux digitaux dans la partie commande). Cette logique est découverte en **1847** par **George Boole** : mathématicien britannique (**1815-1864**), sa **logique** est basée sur deux variables binaires **0 et 1** ou signal digital, la variable binaire est représentée aussi par des états particuliers comme :

- Arrêt/ marche ;
- ouvert /fermé ;
- avant/ arrière ;
- vrai /faux...etc.

III. 2. Propriétés de base de l'algèbre de Boole

Le tableau ci-dessous représente les propriétés de base de l'algèbre de Boole

Involution	$\bar{\bar{a}} = a$
Idempotence	$a + a = a$ $a . a = a$
Complémentarité	$a . \bar{a} = 0$ $a + \bar{a} = 1$
Éléments neutres	$a = a . 1 = 1 . a = a$ $a + 0 = 0 + a = a$
Absorbants	$a + 1 = 1$ $a . 0 = 0$
Associativité	$(a . b) . c = a . (b . c)$ $(a + b) + c = a + (b + c)$

Distributivité	$a.(b + a) = a.b + a.c$ $a + (b.c) = (a + b).(a + c)$
Règles de Morgan	$\overline{a + b} = \bar{a}.\bar{b}$ $\overline{a.b} = \bar{a} + \bar{b}$
Optimisation	$a + \bar{a}b = a + b$ $a + bc = (a + b)(a + c)$

III.3. Table de vérité et équation logique :

À partir de l'équation algébrique on remplit la table de vérité, voir les exemples représentés sur le tableau suivant :

Équation logique	Table de vérité			
Soit : $f(a, b) = 1$ si $a = 0$ et $b = 1$ sinon $f(a, b) = 0$	a	b	f(a, b)	
	0	0	0	
	0	1	1	
	1	0	0	
	1	1	0	
Soit : $f(a, b) = a.\bar{b} + \bar{a}.b$	a	b	f(a, b)	
	0	0	1	
	0	1	0	
	1	0	1	
	1	1	0	
Soit : $f(a, b, c) = abc + abc\bar{c} + \bar{a}bc\bar{c} + \bar{a}\bar{b}c$	a	b	c	f(a, b, c)
	0	0	0	1
	0	0	1	0
	0	1	0	1
	0	1	1	0
	1	0	0	0
	1	0	1	0
	1	1	0	1
	1	1	1	1

III.4. Table de KARNAUGH

III.4.1. Définition

Pour la simplification des fonctions logiques, le tableau de *Karnaugh* est le moyen le plus utilisé dans la réduction des expressions booléennes.

III.4.2. Table de KARNAUGH à trois variables :

S	c	ab	00	01	11	10
		0				
	1					

Avec a, b, c sont les variables d'entrées et S est la variable de sortie

III.4.3. Table de KARNAUGH à quatre variables

S	c d	ab	00	01	11	10
		00				
	01					
	11					
	10					

Avec a, b, c, d sont les variables d'entrées et S est la variable de sortie

III.4.4. Exemples d'applications

Équation algébrique

$$f(a, b) = \bar{a} \cdot \bar{b} + a \cdot b$$

$$f(a, b) = (\bar{a} + b) (a \cdot \bar{b})$$

Tables de KARNAUGH

b	0	1
	a	
0	1	0
1	0	1

$$f(a, b) = \bar{a}.\bar{b} + \bar{a}.b + a.b$$

Équation simplifiée à l'aide de la table de KARNAUGH

$$f(a, b) = \bar{a} + b$$

	<i>b</i>	0	1
<i>a</i>			
0		1	1
1		0	1

$$f(a, b, c) = \bar{a}.\bar{b}.\bar{c} + \bar{a}.b.c + \bar{a}.b.\bar{c} + a.b.\bar{c}$$

Équation simplifiée à l'aide de la table de KARNAUGH

$$f(a, b, c) = \bar{a}.\bar{c} + \bar{a}.b + b.\bar{c}$$

	<i>bc</i>	00	01	11	10
<i>a</i>					
0		1	0	1	1
1		0	0	0	1

$$f(a, b, c, d) = \bar{a}.\bar{b}.\bar{c}.\bar{d} + \bar{a}.b.\bar{c}.\bar{d} + a.\bar{b}.c.d + a.\bar{b}.\bar{c}.\bar{d}$$

Équation simplifiée à l'aide de la table de KARNAUGH

$$f(a, b, c, d) = \bar{a}.\bar{c}.\bar{d} + a.\bar{b}.c$$

	<i>cd</i>	00	01	11	10
<i>ab</i>					
00		1	0	0	0
01		1	0	0	0
11		0	0	0	0
10		0	0	1	1

$$f(a, b, c, d) = \bar{a}.\bar{b}.\bar{c}.\bar{d} + \bar{a}.\bar{b}.\bar{c}.d + \bar{a}.b.\bar{c}.\bar{d} + \bar{a}.b.\bar{c}.d$$

Équation simplifiée à l'aide de la table de KARNAUGH

$$f(a, b, c, d) = \bar{a}.\bar{c}$$

	<i>cd</i>	00	01	11	10
<i>ab</i>					
00		1	1	0	0
01		1	1	0	0
11		0	0	0	0
10		0	0	0	0

IV. Programmation des automates programmables par l'application de la logique combinatoire et séquentielle.

IV.1. Définition

L'automate programmable est un dispositif électronique utilisé dans les entreprises industrielles, les usines, les hôpitaux, les ascenseurs,...etc. Son objectif est automatisé un processus contient plusieurs éléments à titre d'exemple : la commande automatisée d'un tour et d'une fraiseuse, commande d'un convoyeur à bande, la commande automatisée d'une cellule pneumatique et hydraulique,...etc. Il constitue par un module d'entrée, module de sortie, une unité centrale de traitement et un bloc de communication. Son fonctionnement est basé sur la réception des informations délivrées par les capteurs vers ses entrées, celles-ci sont traitées par un programme au niveau de l'unité centrale de programmation (CPU) dans laquelle les sorties de L'API reçoivent les ordres pour actionner les bobines des pré-actionneurs. La figure ci-dessous représente un automate programmable avec ces blocs principaux.

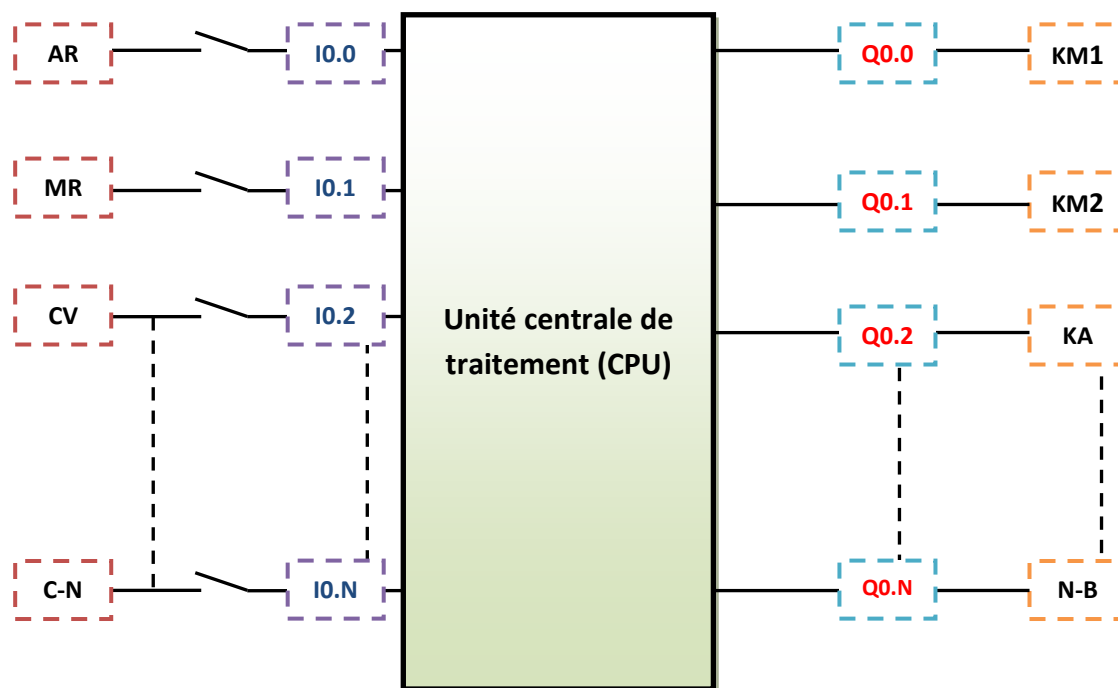


Figure 7. Automate programmable.

Les entrées de l'automate

AR : bouton poussoir arrêt prend l'entrée **I0.0** ;

MR : bouton poussoir marche prend l'entrée **I0.1**;

CV : capteur de vitesse prend l'entrée **I0.2**;

C-N : N capteurs prend l'entrée **I0.N**;

Les sorties de l'automate

KM1 : bobine du contacteur 1 prend la sortie **Q0.0** ;

KM2 : bobine du contacteur 2 prend la sortie **Q0.1** ;

KA : bobine du distributeur prend la sortie **Q0.3** ;

N-B : N bobines prend la sortie **Q0.N**.

IV.2. Différents langages de programmation

Pour la programmation des API on a plusieurs types de langages parmi lesquels :







- Langage à contact (Ladder) ;
- Langage de programmation où nous avons utilisé un logigramme (liste d'instruction) ;
- Langage de programmation à partir d'un **GRAFCET**.

IV.3. Langage à contact

Ce langage est basé sur la logique câblée où nous avons utilisé des circuits de commande et de puissance qui comportent les contacteurs, relais thermiques, boutons poussoirs, capteurs, temporisateurs, pré-actionneurs, actionneurs, ...etc. À partir de ces circuits on va concevoir un schéma à contact dans un logiciel de programmation des API.

IV.3.1. Représentation des éléments principaux pour un réseau LD (LADDER).

Le tableau ci-dessous représente les différents éléments de programmation en langage LADDER.

Symbole	Désignation
Les entrées	
	Contact normalement fermé.
	Contact normalement ouvert.
	Contact fermé au front montant.
	Contact fermé au front descendant.
Les connexions	
	Connexion horizontale.
	Connexion verticale.

Les sorties



Bobine.



Bobine inverse.

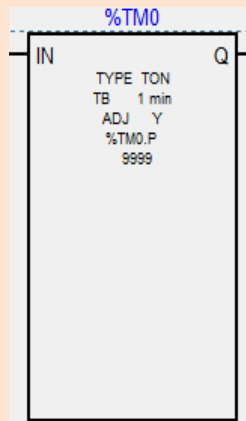
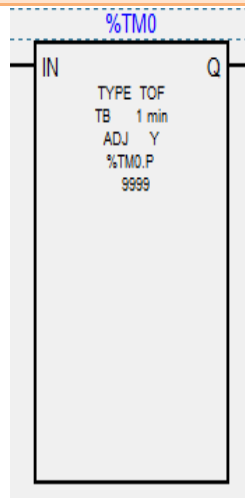
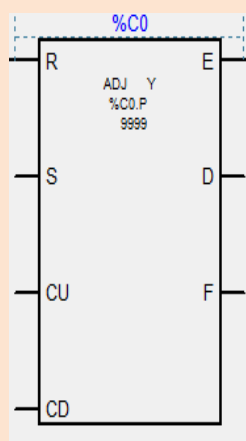


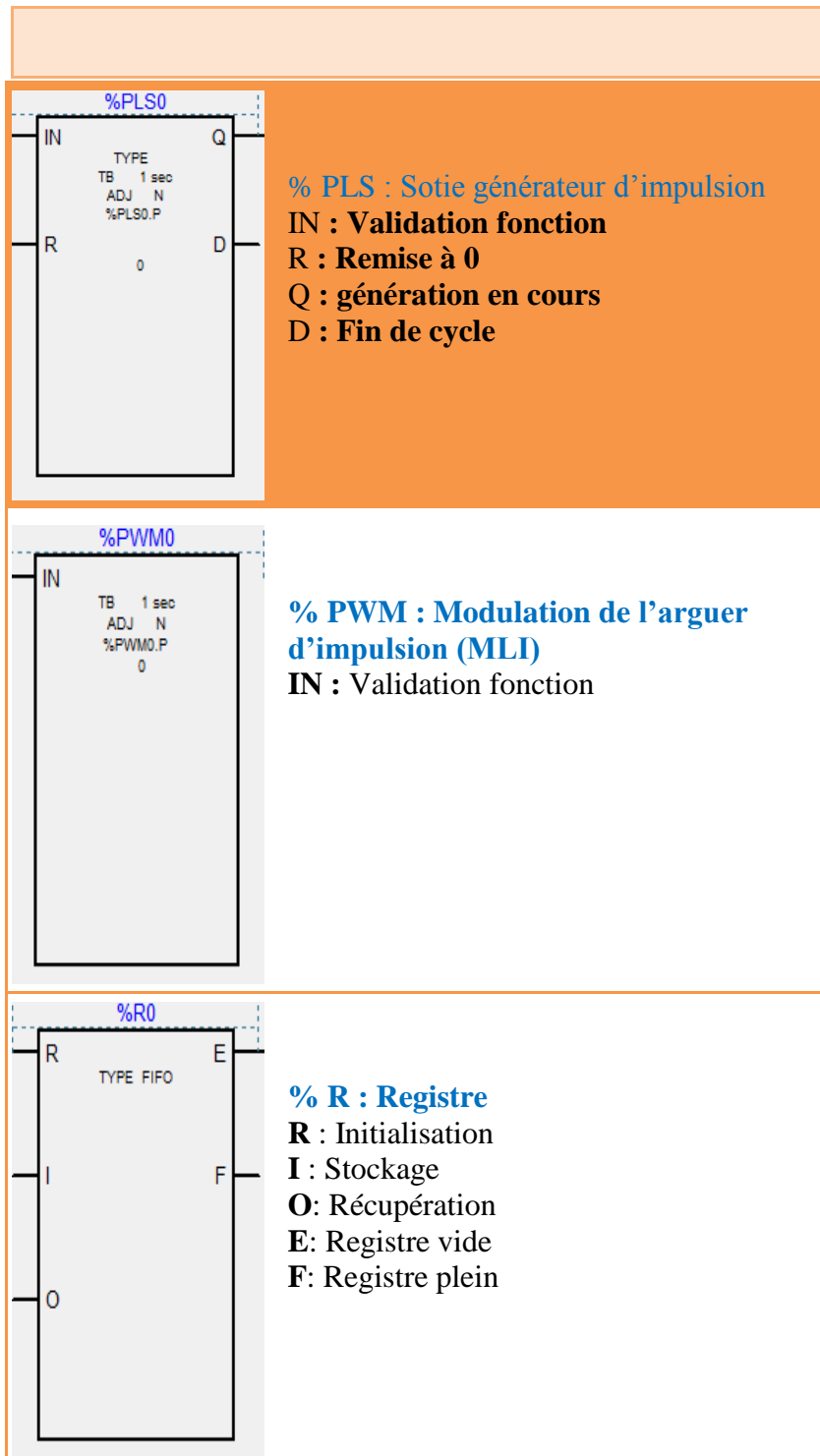
Bobine mise à 1 une fois actionnée.



Bobine reset remise à 0 la bobine "S".

Les blocs fonctionnels

**%TMO : temporisateur (TON)****IN** : Validation d'une fonction**Q** : Sortie temporisation (par exemple vers une bobine direct)**%TMO : temporisateur (TOF)****IN** : Validation d'une fonction**Q** : Sortie temporisation (par exemple vers une bobine direct)**%C0 : Compteur****R** : Remise à 0**S** : Présélection**CU** : Comptage**CD** : Décomptage**E** : Débordement**D** : Présélection**F** : Débordement



IV.3.2. Exemples d'applications sur langage à contact (Ladder)

Exemple d'application -1

Un chariot de marchandise se déplace du point **FC1** vers le point **FC** (voir la figure 8) et revient à sa position initiale.

- Réaliser le programme en langage à contact (ladder) sous **Twidosuite** et indiquer les entrées et les sorties sur le programme.

Condition :

Le chariot se déplace au point FC mais il doit rester 10 seconds.

Liste des éléments constituant le programme en langage à contact

Appareil	Fonction	Symbole	Entrée automate	Sortie automate
Relais thermique	Protection du moteur contre les surcharges	F1	I0.0	
Contacteur 1	Marche avant du chariot	KM1		Q0.0
Contacteur 2	Marche arrière du chariot	KM2		Q0.1
Capteur de fin de course 1	Indique la fin de l'aller du chariot	FC	I0.3	
Capteur de fin de course 2	Indique le retour du chariot et fin de cycle	FC1	I0.4	
Bouton poussoir marche	Démarré le chiot	MR	I0.2	
Bouton poussoir arrêt	Arrête le chariot	AR	I0.1	
Temporisateur	Permet au chariot de rester 10 seconds pour faire le chargement des produits	% TM0		

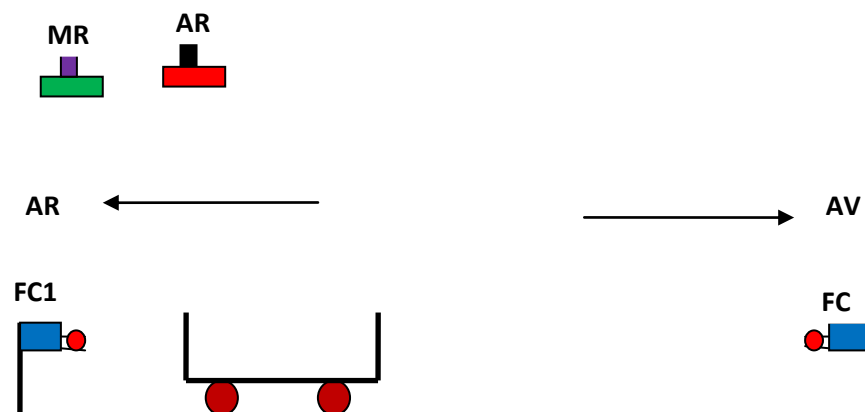
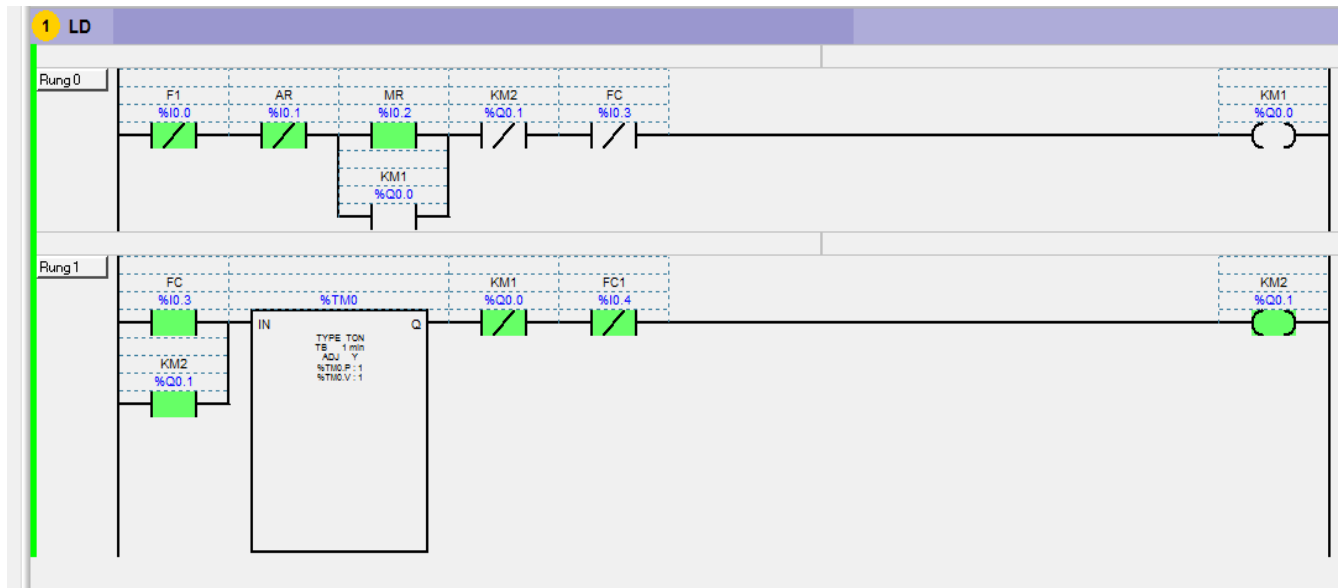


Figure 8. Chariot de marchandé automatisé

Solution-1

- Langage à contact sur Twidosuite avec l'appellation d'entrées et des sorties.

**Exemple d'application -2**

Soit le système pneumatique suivant

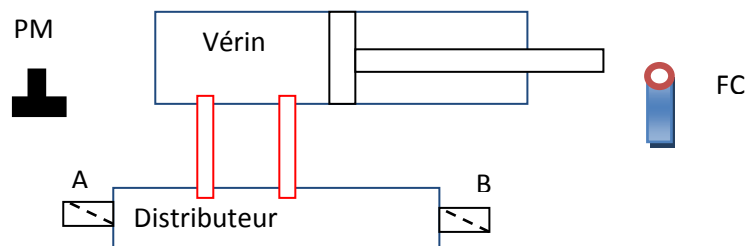


Figure 9. Commande d'un vérin double effet par automate programmable.

La figure ci-dessus représente un système pneumatique qui constitue par un distributeur ayant deux bobines A et B, un vérin double effet et un capteur de fin de course mécanique.

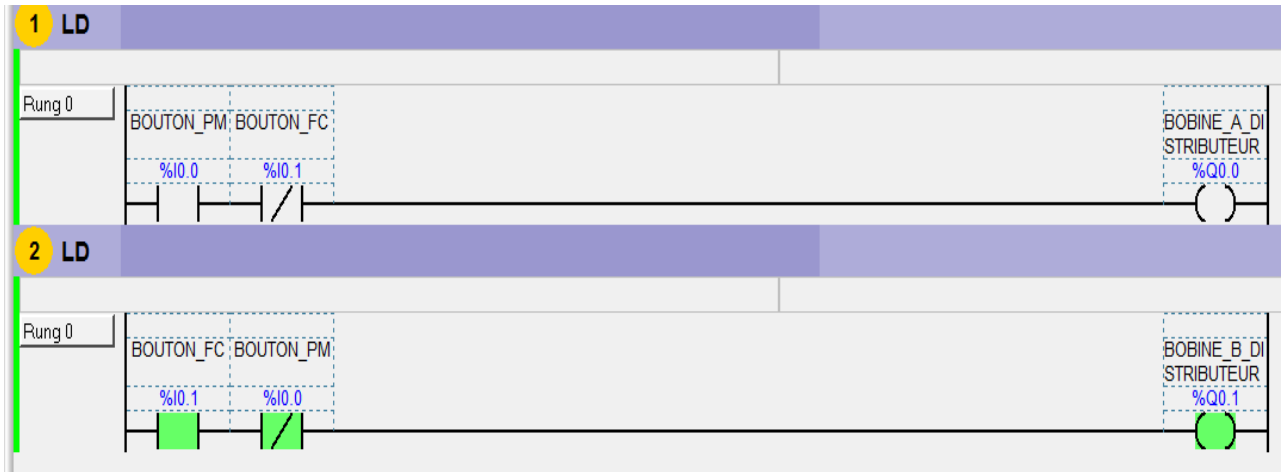
Liste des éléments constituant le programme en langage à contact

Appareil	Fonction	Symbole	Entrée automate	Sortie automate
Bouton poussoir	Démarrage du système pneumatique	PM	I0.0	
Bobine de distributeur A	Permet de sortir la tige du vérin en avant	A		Q0.0
Bobine de distributeur B	Permet le retour de la tige du vérin	B		Q0.1

- On demande le programme en langage à contact sous Twidosuite.

Solution-2

- Langage à contact sous Twidosuite du système pneumatique



IV.4. Langage liste instruction

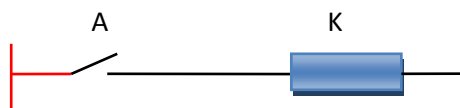
Pour programmer par ce langage on va définir les opérateurs de base de l'Algèbre de Boole. Il existe huit fonctions logiques élémentaires qui sont :

- La fonction égalité ;
- La fonction négation ;
- La fonction multiplication logique.
- La fonction addition logique.
- La fonction NON ET (NAND)
- La fonction NON OU (NOR)
- La fonction OU Exclusif
- La fonction Coïncidence

IV.4.1. La fonction égalité

Cette fonction est appelée **oui** ou fonction **on**, son principe est comme suit :

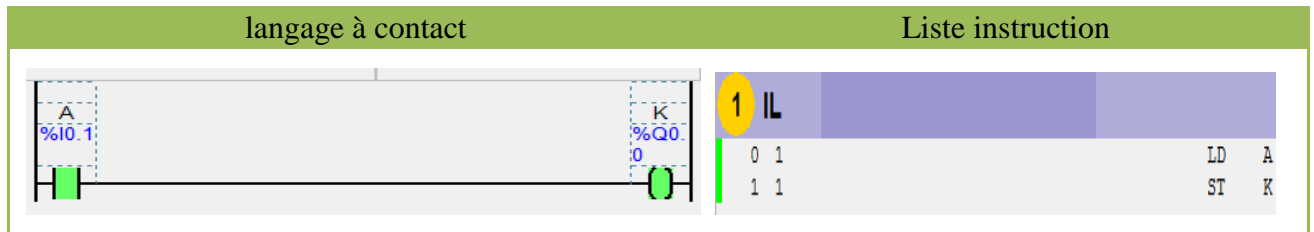
On prend la bobine **K** et l'interrupteur **A** :



Si A = 0	K = 0
Si A = 1	K = 1

On peut écrire $A = K$

- Représentation de la fonction égalité sous langage ladder et liste instruction

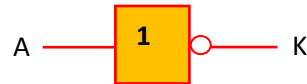


IV.4.2. La fonction négation ou complémentation

Le complément ou l'inverse d'une variable binaire A , notée \bar{A} , réalise le **NON** de cette variable ou **NO** en anglais (\bar{A} se lit A barré ou encore non A).

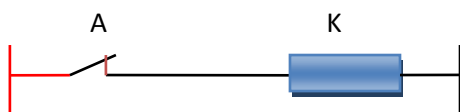
$\bar{A} = 1$ si et seulement si $A = 0$

L'opérateur qui réalise cette fonction d'inversion logique s'appelle un inverseur dont le symbole est le suivant :



Exemple :

On prend toujours l'interrupteur et la bobine K

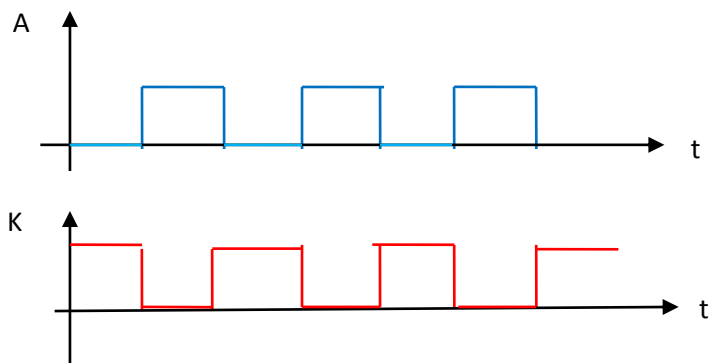


$\bar{A} = 1$	$K = 1$
$\bar{A} = 0$	$K = 0$

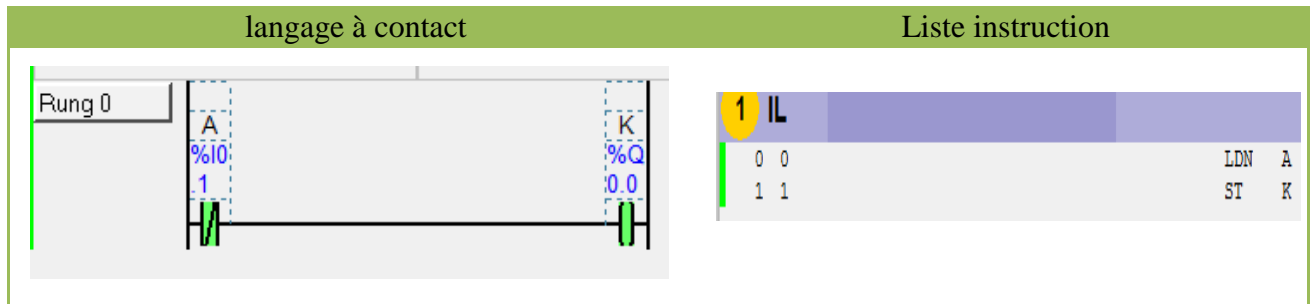
On peut écrire $K = \bar{A}$

- Table de vérité et chronogramme

A	K
0	1
1	0

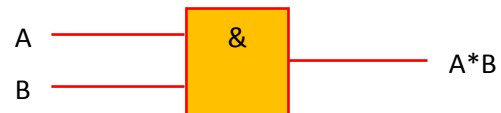


- Représentation de La fonction négation ou complémentation sous langage ladder et liste instruction

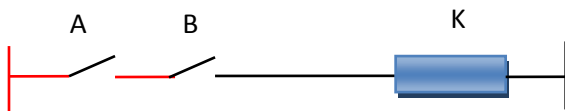


IV.4.3. La fonction intersection ou multiplication logique

Pour réaliser la fonction **ET**, on a besoin de deux variables binaires A et B, le produit de ces deux variables donne la fonction ET ou fonction AND. Le symbole qui réalise cette fonction est comme suit :



Exemple :

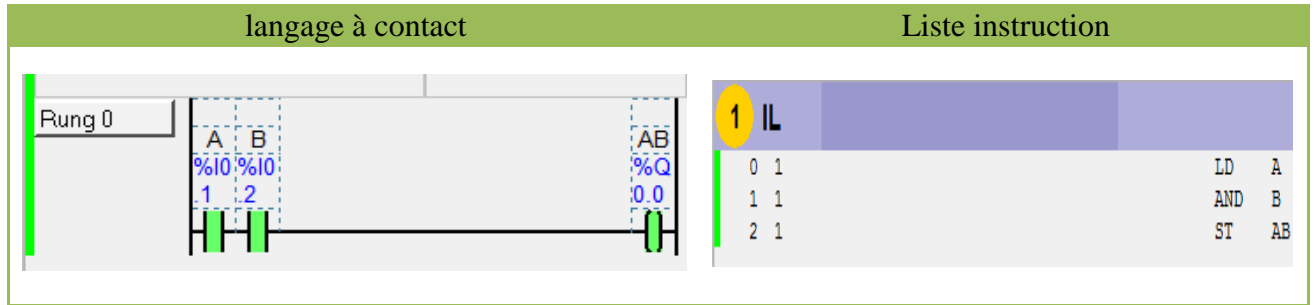


Si $A = 1$ et $B = 1$, $K = 1$, sinon $K = 0$

- Table de vérité et chronogramme

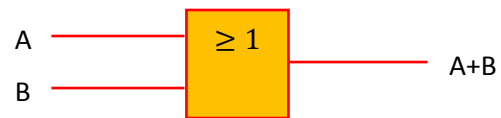
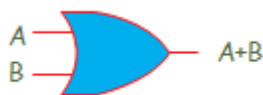
A	B	A*B
0	0	0
0	1	0
1	0	0
1	1	1

- Représentation de la fonction AND ou ET sous langage ladder et liste instruction

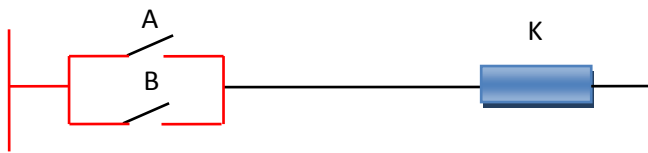


IV.4.4. La fonction réunion ou addition logique

Pour réaliser la fonction **OU**, on a besoin de deux variables binaires A et B, l'addition de ces deux variables donne la fonction **OU**, ou fonction **OR** à la sortie de la porte logique. Le symbole qui réalise cette fonction est comme suit :



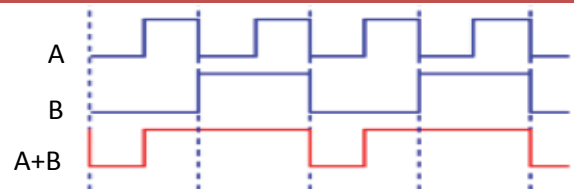
Exemple :



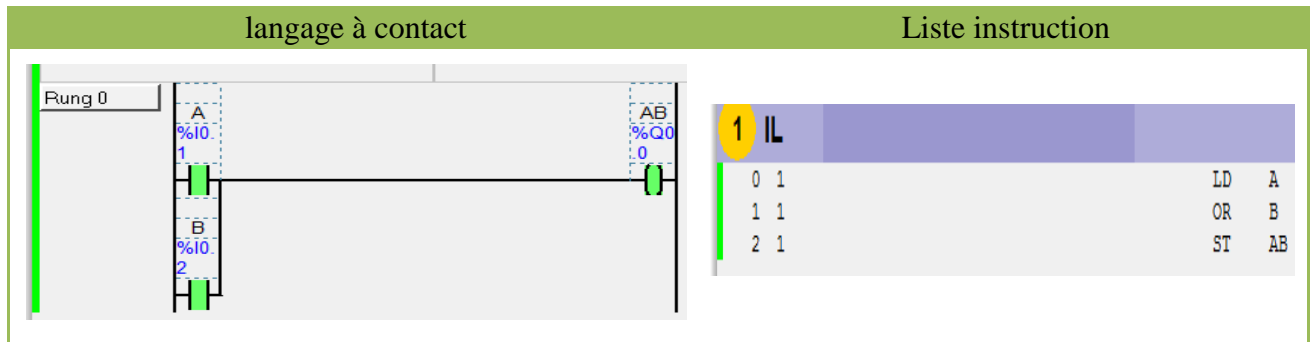
Si $A = 1$ ou $B = 1 \Rightarrow K = 1$
 Si $A = 1$ et $B = 1 \Rightarrow K = 1$

- Table de vérité et chronogramme

A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1



- **Représentation de la fonction OR sous langage ladder et liste instruction**



IV.4.5. La fonction NON ET (NAND)

La fonction **NAND** ou **NON ET** est la fonction **ET** inversée, sa sortie égale 0 si A =1 et B=1, donc $\overline{A \cdot B} = 0$. Le symbole qui réalise cette fonction est le suivant :



- **Table de vérité et chronogramme**

A	B	$K = \overline{a \cdot b}$
0	0	1
0	1	1
1	0	1
1	1	0

- **Constitution de la porte NAND**

Pour construire la porte NAND on utilise le bloc NOT en anglais ou NON en français.

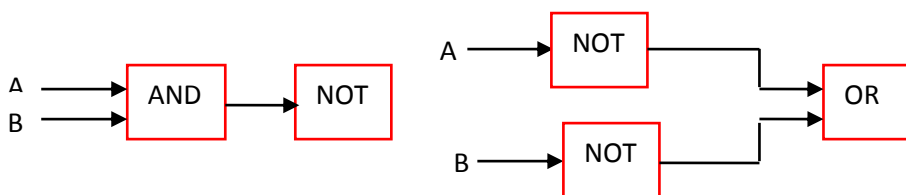


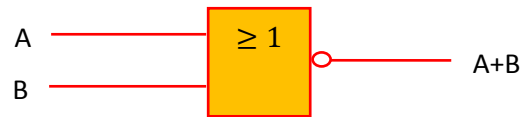
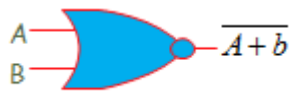
Figure 10. Porte logique NAND

- **Représentation de la fonction NAND sous langage ladder et liste instruction**

langage à contact	Liste instruction																
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #92d050;"> <th style="width: 5%;">1</th> <th style="width: 10%;">IL</th> <th style="width: 75%;"></th> <th style="width: 10%;"></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>LDN A</td> <td>A</td> </tr> <tr> <td>1</td> <td>0</td> <td>ANDN B</td> <td>B</td> </tr> <tr> <td>2</td> <td>1</td> <td>ST K</td> <td>K</td> </tr> </tbody> </table>	1	IL			0	0	LDN A	A	1	0	ANDN B	B	2	1	ST K	K
1	IL																
0	0	LDN A	A														
1	0	ANDN B	B														
2	1	ST K	K														

IV.4.6. La fonction NON OU (NOR)

La fonction **NOR** ou **NON** est la fonction **OU** inversée, sa sortie égale 1 si A =0 et B = 0, donc $\overline{A + B} = 1$. Le symbole qui réalise cette fonction est le suivant.



- **Table de vérité et chronogramme**

A	B	$K = \overline{a + b}$	
0	0	1	
0	1	0	
1	0	0	
1	1	0	

- **Constitution de la porte NOR**

Pour construire la porte NOR on utilise le bloc NOT en anglais ou NON en français.

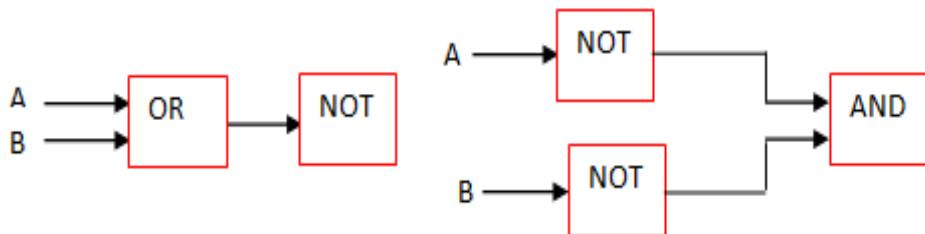
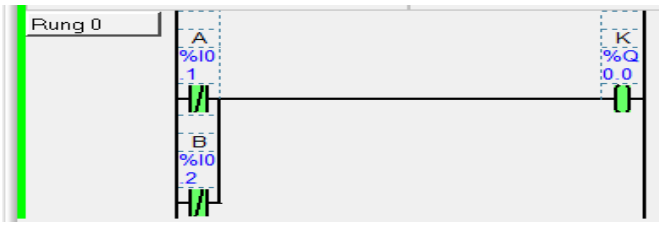


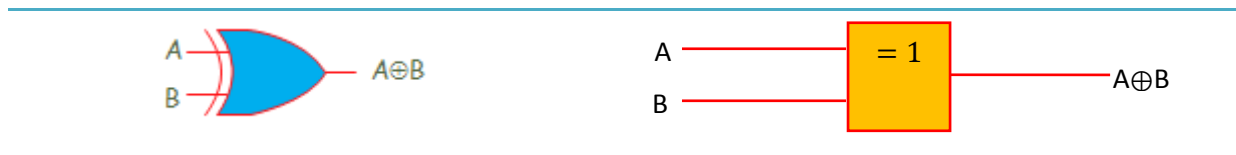
Figure 11. Porte logique NOR

- **Représentation de la fonction NOR sous langage ladder et liste instruction**

langage à contact	Liste instruction																
	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr style="background-color: #92d050;"> <td style="width: 5%; text-align: center;">1</td> <td style="width: 15%; text-align: center;">IL</td> <td style="width: 60%;"></td> <td style="width: 20%;"></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td></td> <td style="text-align: center;">LDN A</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td></td> <td style="text-align: center;">ORN B</td> </tr> <tr> <td style="text-align: center;">2</td> <td style="text-align: center;">1</td> <td></td> <td style="text-align: center;">ST K</td> </tr> </table>	1	IL			0	0		LDN A	1	0		ORN B	2	1		ST K
1	IL																
0	0		LDN A														
1	0		ORN B														
2	1		ST K														

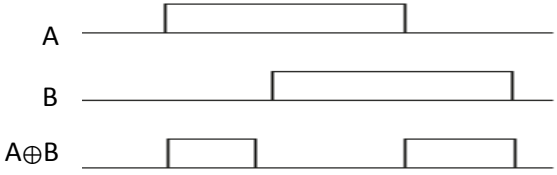
IV.4.7. La fonction ou exclusif XOR

La fonction ou exclusif ou **XOR** prend la valeur 1 si l'un des deux variables binaires prend 1, pour tous les autres cas prend la valeur 0. Le symbole qui représente cette fonction est le suivant :



- **Table de vérité et chronogramme**

A	B	$K = A \oplus B = \bar{A} \cdot B + \bar{B} \cdot A$	
0	0	0	A
0	1	1	B
1	0	1	A ⊕ B
1	1	0	



- **Constitution de la porte XOR**

Pour construire la porte XOR on utilise le ploc NOT, les portes AND et la porte OR.

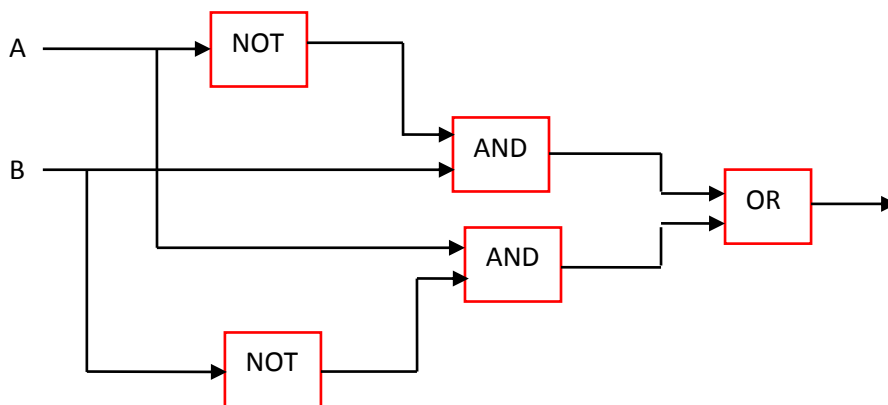
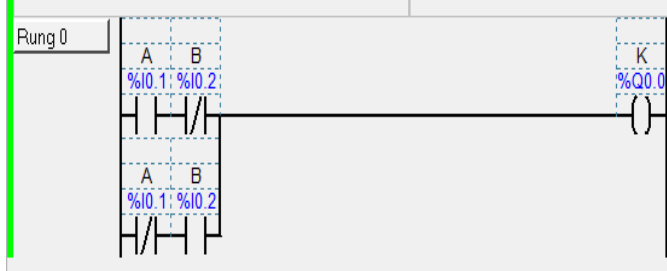


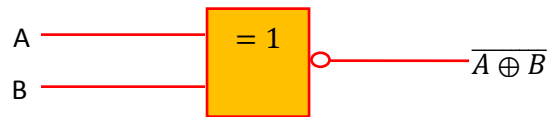
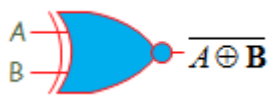
Figure 12. Porte logique XOR

- **Représentation de la fonction XOR sous langage ladder et liste instruction**

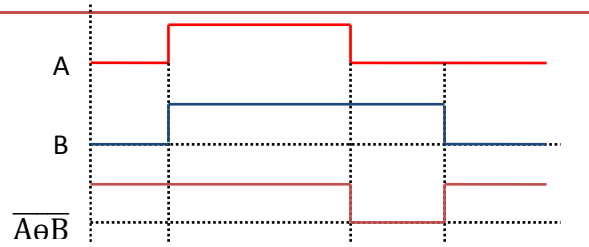
langage à contact	Liste instruction																												
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #92d050;"> <th style="width: 5%;">1</th> <th style="width: 15%;">IL</th> <th style="width: 10%;"></th> <th style="width: 70%;"></th> </tr> </thead> <tbody> <tr><td>0</td><td>LD</td><td>A</td><td></td></tr> <tr><td>1</td><td>ANDN</td><td>B</td><td></td></tr> <tr><td>2</td><td>OR (N</td><td>A</td><td></td></tr> <tr><td>3</td><td>AND</td><td>B</td><td></td></tr> <tr><td>4</td><td>)</td><td></td><td></td></tr> <tr><td>5</td><td>ST</td><td>K</td><td></td></tr> </tbody> </table>	1	IL			0	LD	A		1	ANDN	B		2	OR (N	A		3	AND	B		4)			5	ST	K	
1	IL																												
0	LD	A																											
1	ANDN	B																											
2	OR (N	A																											
3	AND	B																											
4)																												
5	ST	K																											

IV.4.8. La fonction coïncidence (L'OU-EXCLUSIF-NON)

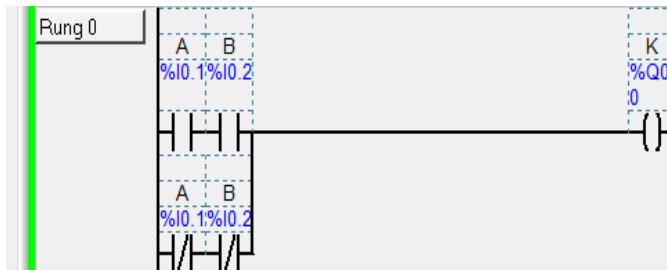
La fonction **ou exclusif-non** prend la valeur **1** si et seulement les deux variables binaires **A** et **B** prennent la même valeur, pour tous les autres cas prend la valeur 0. L'opérateur qui réalise cette fonction est le suivant :



- **Table de vérité et chronogramme**

A	B	$K = \overline{A \oplus B} = AB + \overline{A} \cdot \overline{B}$	
0	0	1	
0	1	0	
1	0	0	
1	1	1	

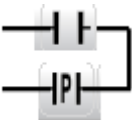
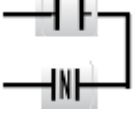
- **Représentation de la fonction NXOR sous langage ladder et liste instruction**

langage à contact	Liste instruction																												
	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #92d050;"> <th style="width: 5%;">1</th> <th style="width: 15%;">IL</th> <th style="width: 10%;"></th> <th style="width: 70%;"></th> </tr> </thead> <tbody> <tr><td>0</td><td>LD</td><td>A</td><td></td></tr> <tr><td>1</td><td>AND</td><td>B</td><td></td></tr> <tr><td>2</td><td>OR (N</td><td>A</td><td></td></tr> <tr><td>3</td><td>ANDN</td><td>B</td><td></td></tr> <tr><td>4</td><td>)</td><td></td><td></td></tr> <tr><td>5</td><td>ST</td><td>K</td><td></td></tr> </tbody> </table>	1	IL			0	LD	A		1	AND	B		2	OR (N	A		3	ANDN	B		4)			5	ST	K	
1	IL																												
0	LD	A																											
1	AND	B																											
2	OR (N	A																											
3	ANDN	B																											
4)																												
5	ST	K																											

IV.4.9. Représentation des éléments principaux pour langage liste instruction.

Le tableau ci-dessous représente les différents éléments de programmation en langage liste instruction.

Symbole Langage Liste	Symbole Langage à contact	Désignation
Les entrées		
LD		Contact normalement fermé. Prend la valeur 1 pendant la commande.
LDN		Contact normalement ouvert. Prend la valeur 0 pendant la commande
LDR		Contact fermé au front montant, actif pendant un cycle de scrutation où le bit de commande a un front montant.
LDF		Contact fermé au front descendant, actif pendant un cycle de balayage où le bit de commande a un front descendant.
Les sorties		
ST		Le résultat de fonction logique active la bobine.
STN		Le résultat inverse de la fonction logique active la bobine.
S		Le résultat de la fonction logique active le relais (met le verrou).
R		Le résultat de la fonction logique désactive le relais (réinitialise le verrou)
Instruction de base de la porte logique AND		
AND		Le résultat précédent de l'opération logique avec la porte AND
ANDN		L'inverse du résultat précédent de l'opération logique avec la porte AND.
ANDR		Le résultat précédent de l'opération logique du front montant avec la porte logique AND.
ANDF		Le résultat précédent de l'opération logique du front descendant avec la porte logique AND
Instruction de base de la porte logique OR		
OR		Le résultat précédent de l'opération logique avec la porte OR
ORN		L'inverse du résultat précédent de l'opération logique avec la porte OR.

ORR		Le résultat précédent de l'opération logique du front montant avec la porte logique OR.
ORF		Le résultat précédent de l'opération logique du front descendant avec la porte logique OR.

IV.4.10. Exemples d'applications sous langage liste

Soit le système suivant

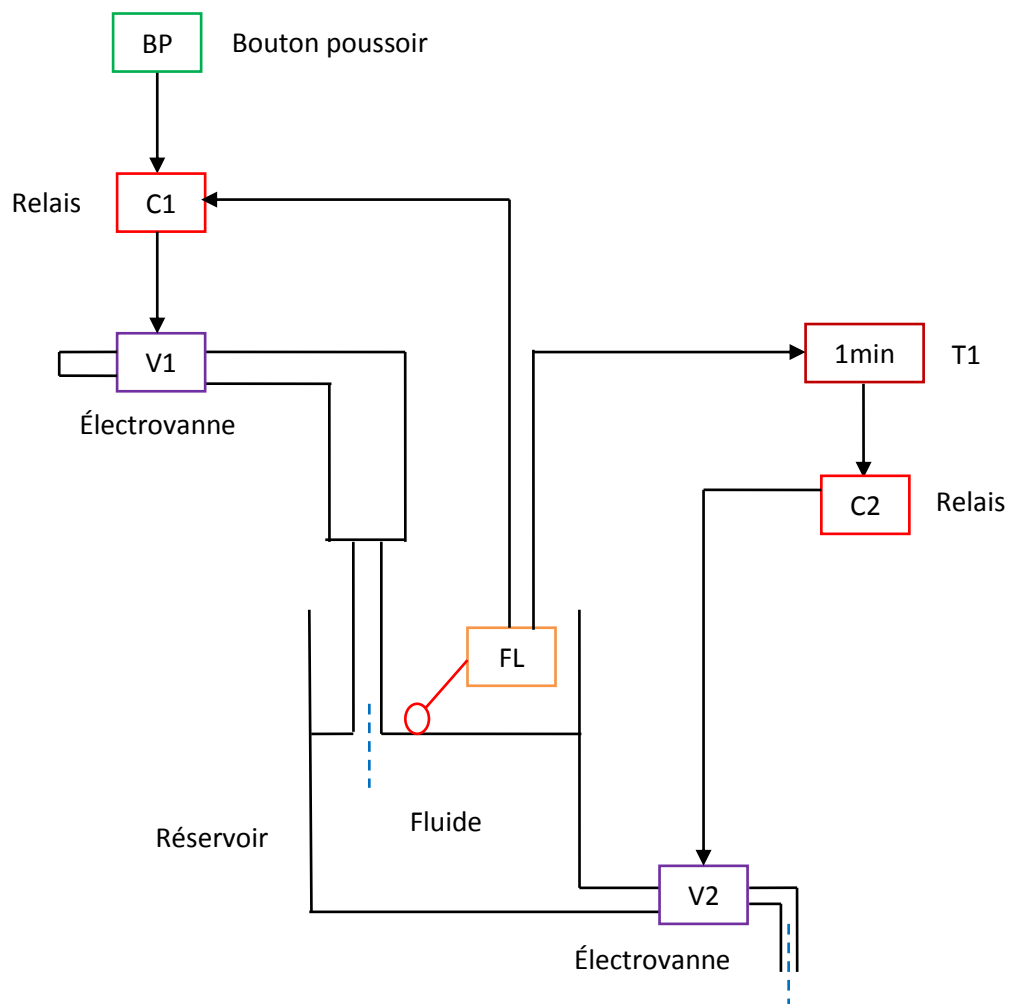


Figure 13. Remplissage d'un réservoir d'eau

Avec :

BP : bouton poussoir ;

C1 et C2 relais de commande ;

V1 et V2 électrovannes ;

FL : interrupteur flotteur ;

Description du cycle

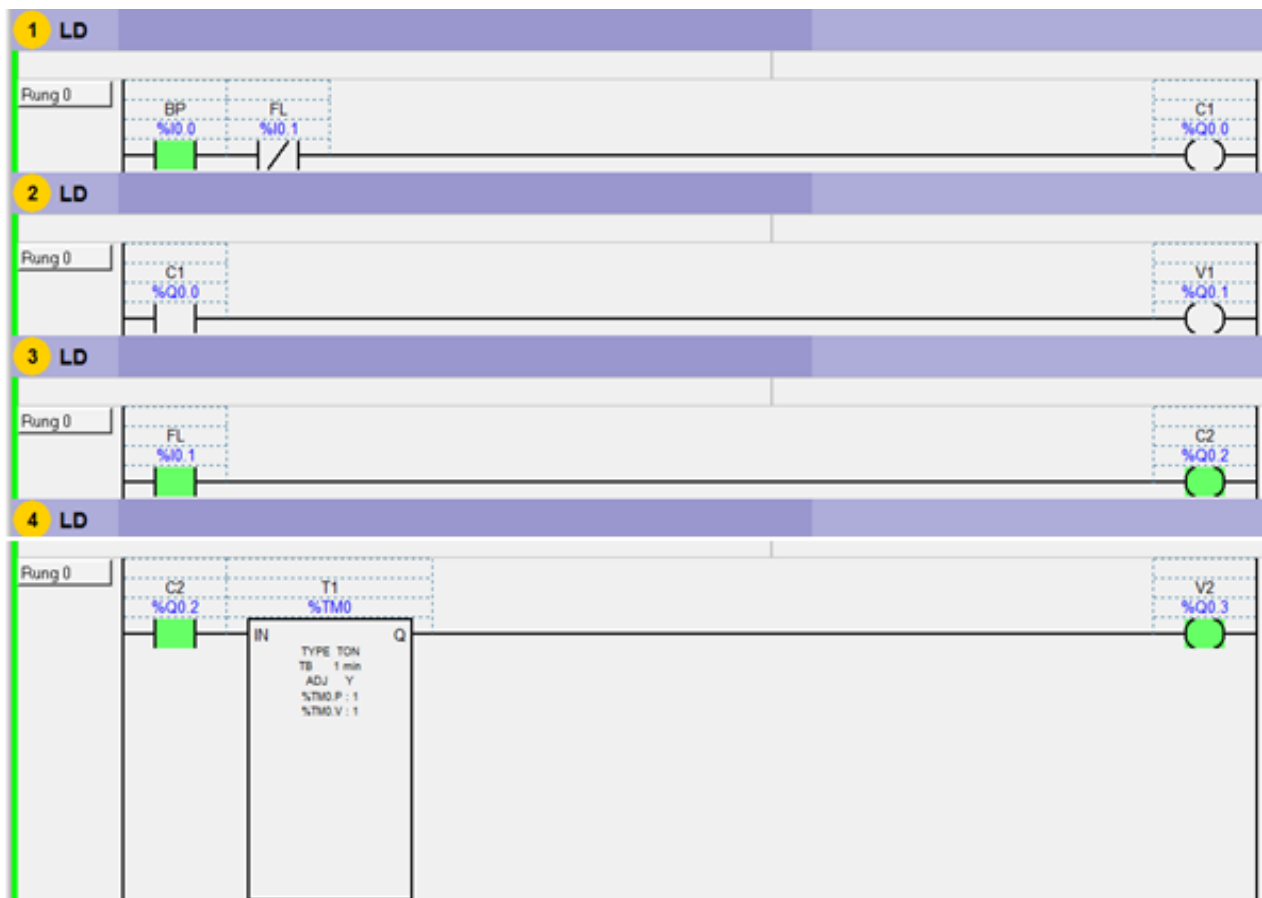
Une impulsion sur le bouton poussoir BP provoque l'excitation du relais de commande C1. Ce qui permet d'ouvrir l'électrovanne V1 et commence le remplissage du réservoir jusqu'au le plein (FL=1). Pour vider le réservoir, l'électrovanne vanne V2 reste 01 minute avant le commencement de la vidange du réservoir.

Question

Réaliser le programme en utilisant langage ladder et liste instruction.

Solution

- Langage ladder



- Langage liste

0	1	BLK	T1
1	1	LD	C2
2		IN	
3		OUT_BLK	
4	*	LD	Q
5	1	ST	V2
6		END_BLK	

V. Langage grafcet

V.1. Définition:

Le grafcet (**G**raphe **F**onctionnel de **C**ommande **É**tapes et **T**ransitions) est un diagramme fonctionnel. Il permet de représenter par un graphe le fonctionnement d'un système automatisé, ce dernier se compose par trois parties essentielles qui sont :

Partie pupitre ;

Partie commande ;

Partie opérative.

- **Le grafcet** est une représentation graphique alternée des étapes et des transitions. Une seule transition doit séparer deux étapes. Le diagramme ci-dessous représente un *grafcet*.

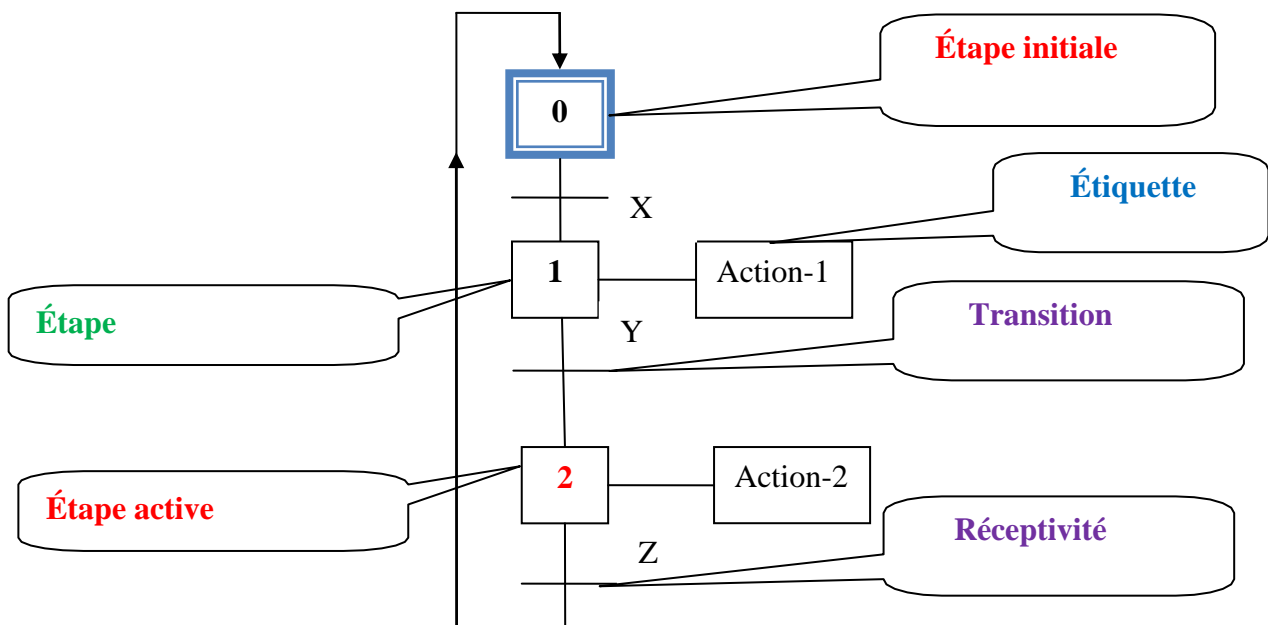


Figure 14. Représentation du grafcet séquence unique

V.2. Types du grafcet

Il existe trois types de *grafcet* :

V.2.1. Grafcet de niveau 01 (grafcet-partie système)

Ce type de grafcet est basé sur la représentation de toutes les parties du système automatisé avant l'existence de ce dernier (système automatisé). Par ailleurs le grafcet de niveau 1 est un grafcet de coordination des données et des actions.

V.2.2. Grafcet de niveau 02 (grafcet-partie opérative)

Le grafcet de niveau-2 est basé sur la technologie des actionneurs (moteurs électriques, vérins, ...etc.) et capteurs, ces derniers nous permettent de réaliser un diagramme séquentielle qui définit le comportement de la partie commande d'un système automatisé.

V.2.3. Grafcet de niveau 03 (grafcet/ partie commande)

Ce grafcet prend le matériel existant (automates programmables, contacteurs, boutons poussoirs, ...etc.) pour réaliser la partie commande.

Le grafcet de niveau 03 est basé sur la programmation des automates programmables en utilisant par exemple le langage LADDER (langage à contact) dont les entrées (%I0.0) et les sorties (%Q0.0).

V.3. Exemples d'applications sur le grafcet unique

Exemple d'application -01

Dans un atelier de mécanique situé au niveau de L'ISTA, il existe un poste de perçage automatisé qui nous a permis le perçage et le serrage des pièces mécaniques, **voir la figure.2**. Son fonctionnement est basé sur le mouvement de rotation du moteur électrique dans les deux sens (marche avant et marche arrière). Le serrage et le desserrage des pièces mécaniques sont effectués par le **vérin-V1**, mais pour les capteurs de positions, son rôle est fixé l'arrivé et le retour du **vérin-V1** ainsi que du moteur **électrique M**.

Description du cycle

La pièce mécanique est arrivée en **C5** (capteur mécanique de position), l'étou est ouvert (le capteur **C4** indique l'ouverture de l'étou), le moteur tourne le foret et reste en position haute (le capteur **C1** indique la position du moteur). L'étou serre la pièce mécanique (le capteur **C4** indique le serrage de la pièce mécanique), la perceuse descend et fait le perçage (le capteur **C2** indique la descente de la perceuse), la perceuse remonte (le capteur **C1** indique la remonte de la perceuse jusqu'à la position initiale). L'étou se desserre la pièce mécanique (le capteur **C3** indique le retour de la tige du vérin à sa position initiale), la pièce est évacuée (le capteur **C6** indique la sortie de la pièce).

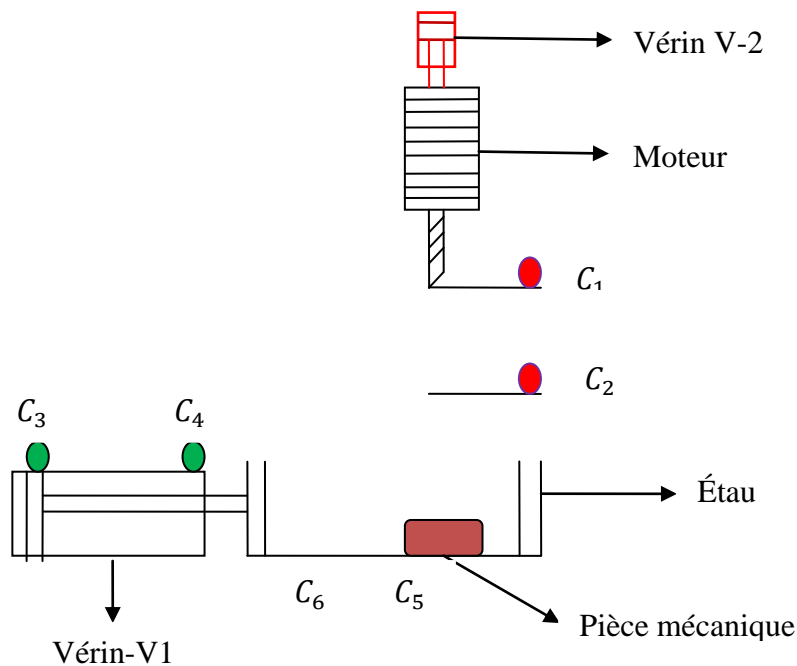


Figure 15. Poste de perçage automatisé

Remarque :

L'avancement du moteur électrique est assuré par le vérin V2

Fiche des données techniques

Entrées automate programmable		Sorties automate programmable	
I0.0	C1	Q0.0	M-
I0.1	C2	Q0.1	M+
I0.2	C3	Q0.2	V1-
I0.3	C4	Q0.3	V1+
I0.4	C5	Q0.4	
I0.5	C6	Q0.5	

Questions

- Réaliser le grafcet de niveau-01 ou grafcet de point de vue partie système ;
- Réaliser le grafcet de niveau-02 ou grafcet de point de vue partie opérative ;

- Réaliser le grafcet de niveau-03 ou grafcet de point de vue partie commande.

Solution-01

- **Grafcet de niveau-01**

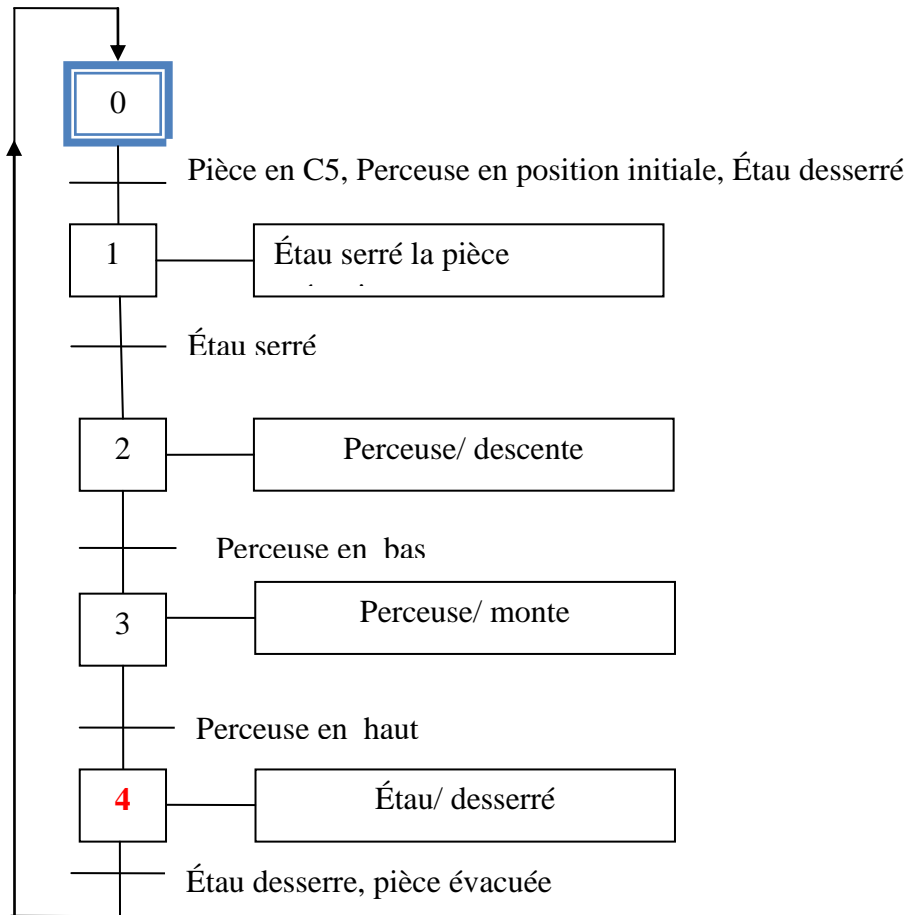


Figure 16. Grafcet de niveau-01 du poste de perçage automatisé

- **Grafctet de niveau-02**

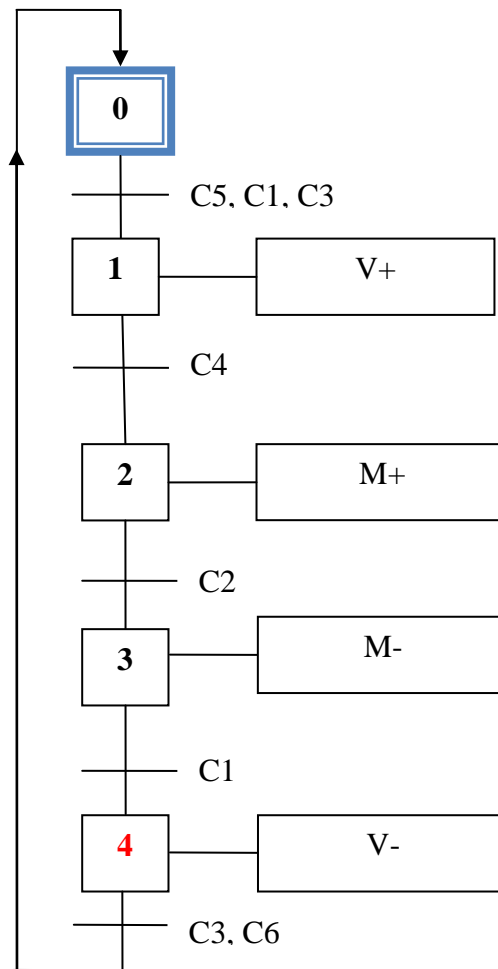


Figure 17. Grafctet de niveau-02 du poste de perçage automatisé

- **Grafctet de niveau-03**

Pour réaliser ce type de grafctet on définit les entrées et les sorties de l'automate programmable.

Le **I** désigne l'entrée de l'automate programmable et le **Q** désigne une sortie de l'automate programmable.

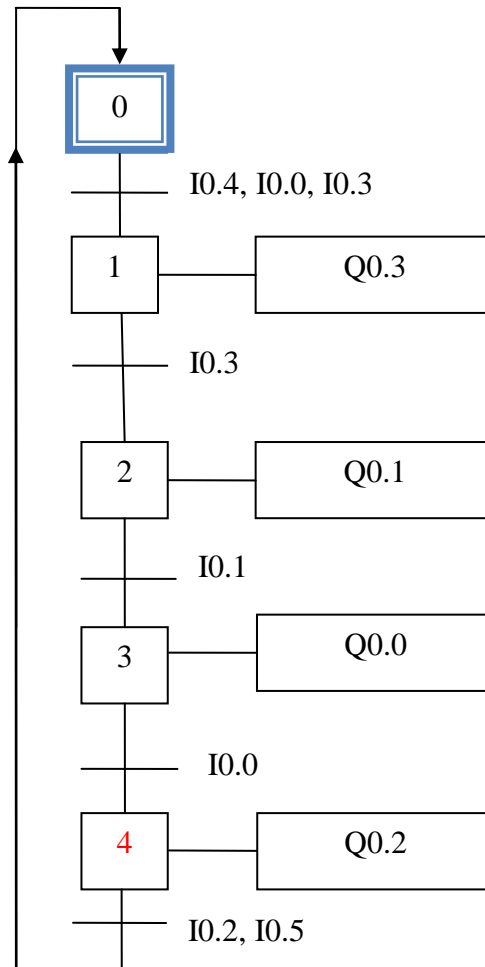


Figure 18. Grafcet de niveau-03 du poste de perçage automatisé

Exemple d'application -02

A- Un chariot de marchandise se déplace du point **C1** vers le point **C2** (voir la figure.6) et revient à sa position initiale.

- Réaliser le grafcet de niveau-01, niveau-02 et niveau-03

Condition :

Le chariot ne peut pas se déplacer si le bouton poussoir **M** non actionné.

Fiche des données techniques

Entrées automate programmable		Sorties automate programmable	
I0.0	C1 (capteur mécanique)	Q0.0	AR
I0.1	C2 (capteur mécanique)	Q0.1	AV
I0.2	M		

B- Le chariot se déplace au point C2 mais il doit rester 10 seconds.

- Établir le grafcet de niveau-01, niveau-02 et niveau-03.

Entrées automate programmable		Sorties automate programmable	
I0.0	C1 (capteur mécanique)	Q0.0	AR
I0.1	C2 (capteur mécanique)	Q0.1	AV
I0.2	M		
I0.3	T1/10S	Q0.2	T

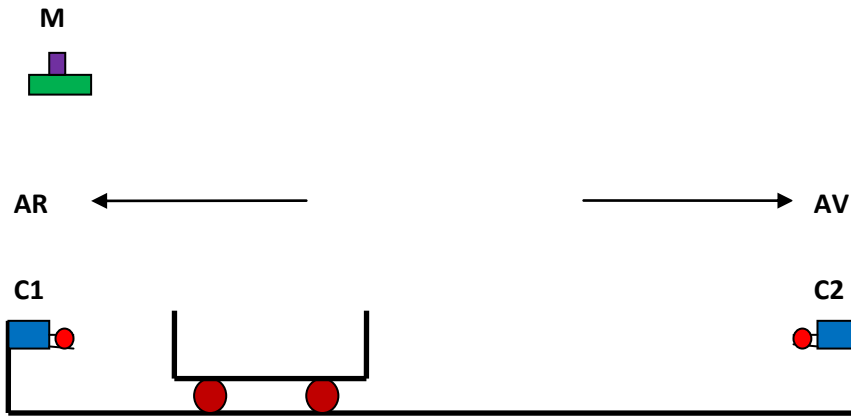


Figure 19. Chariot de marchandise automatisé

Solution-A

- Grafcet de niveau-01

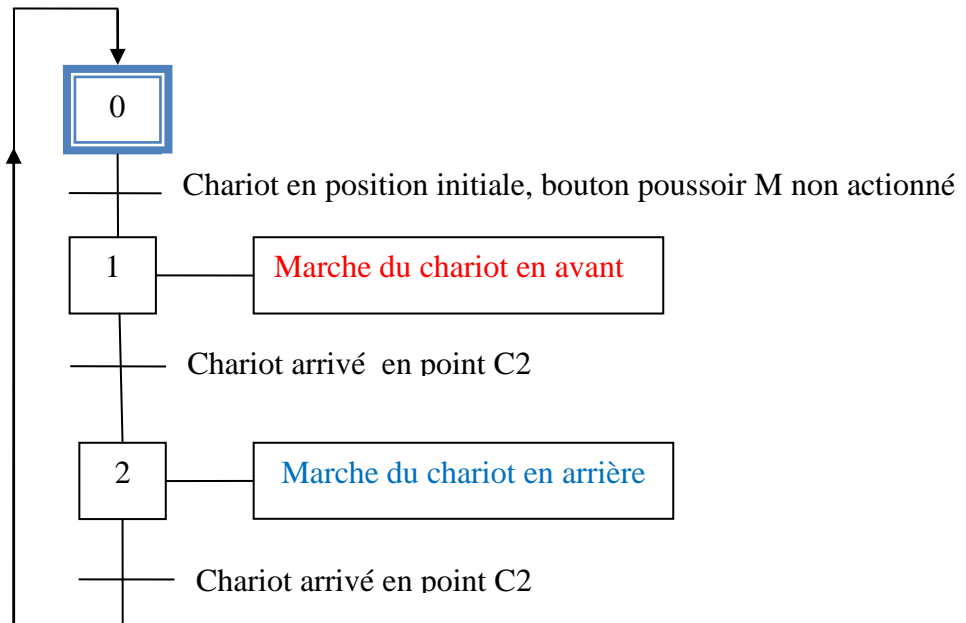
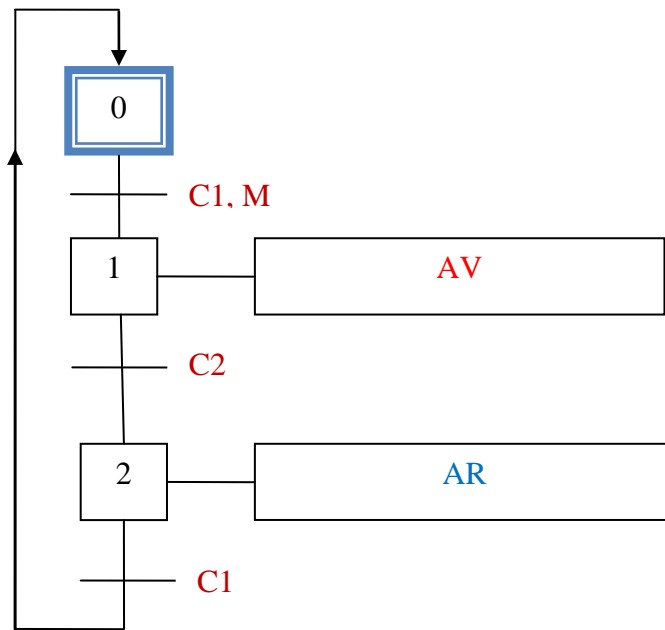
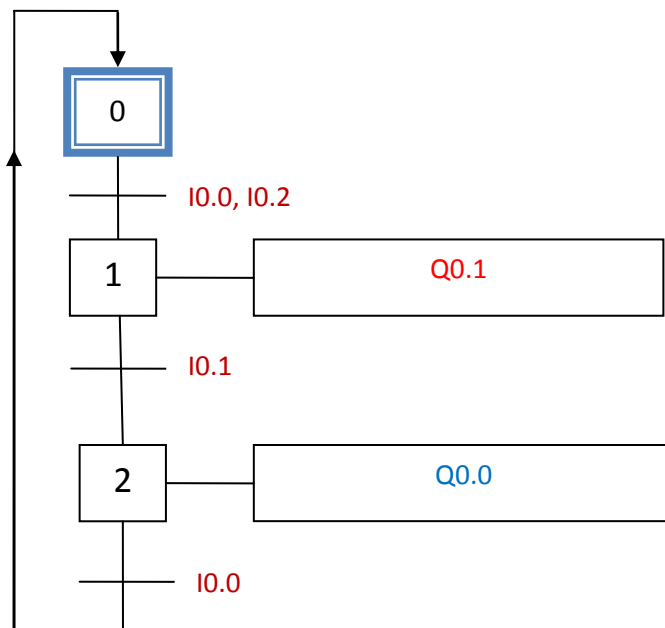
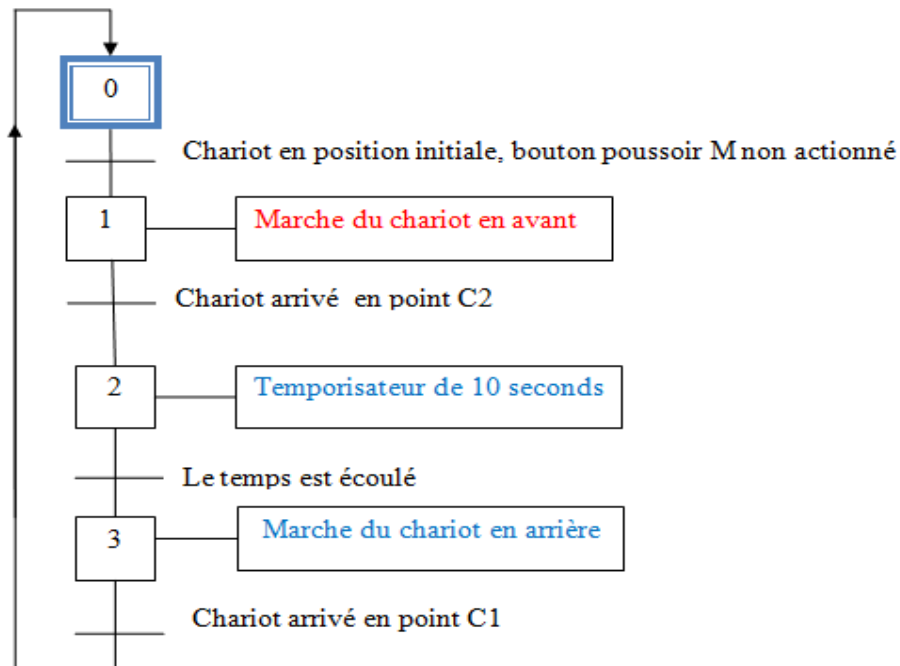
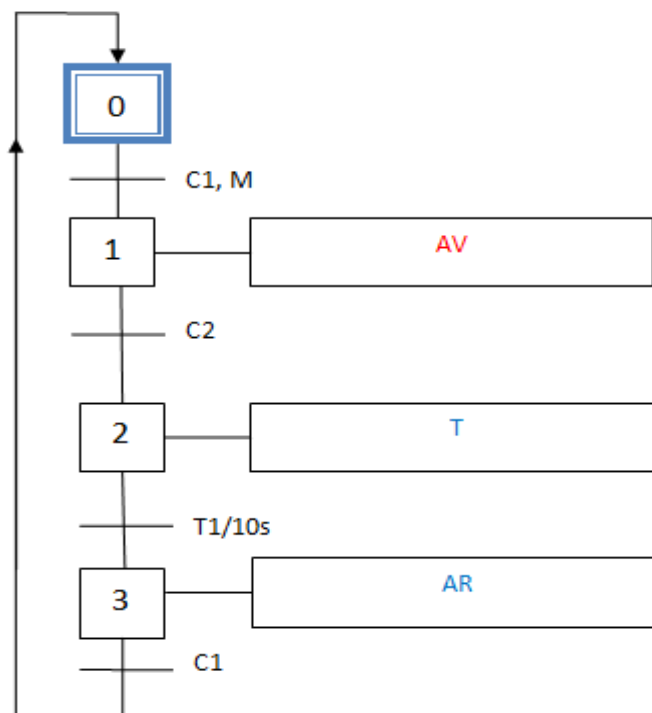


Figure 20. Grafcet de niveau-01 du chariot de marchandise automatisé

- Grafcet de niveau-02**Figure 21.** Grafcet de niveau-02 du chariot de marchandise automatisé**- Grafcet de niveau-03****Figure 22.** Grafcet de niveau-03 du chariot de marchandise automatisé

Solution-B**- Grafcet de niveau-01****Figure 23.** Grafcet de niveau-01 du chariot de marchandise automatisé avec temporisation**- Grafcet de niveau-02****Figure 24.** Grafcet de niveau-02 du chariot de marchandise automatisé avec temporisation

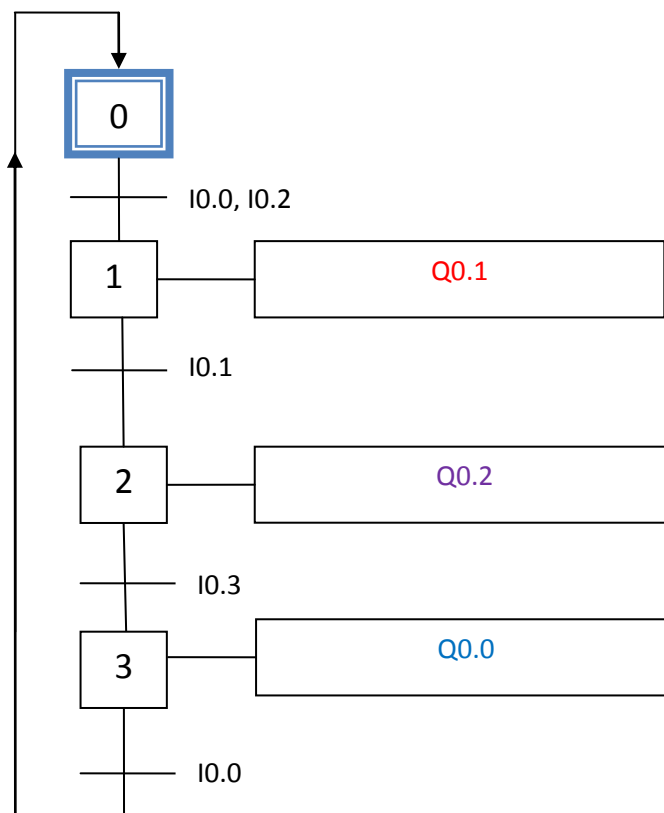
- Grafcet de niveau-03

Figure 25. Grafcet de niveau-03 du chariot de marchandise automatisé avec temporisation

Exemples d'application -03

Dans une usine de fabrication des tôles, il existe un pont roulant utilisé pour prendre les déchets métalliques et qui se déplace du point C1 au point C2 (voir la figure.13). Pour le chargement et le déchargement des déchets métalliques ce pont fait monter et descendre la charge.

Les capteurs de position C1 et C2 sont des capteurs mécaniques et les capteurs A, B, C3 et C4 sont des capteurs magnétiques de position.

Installation électromécanique

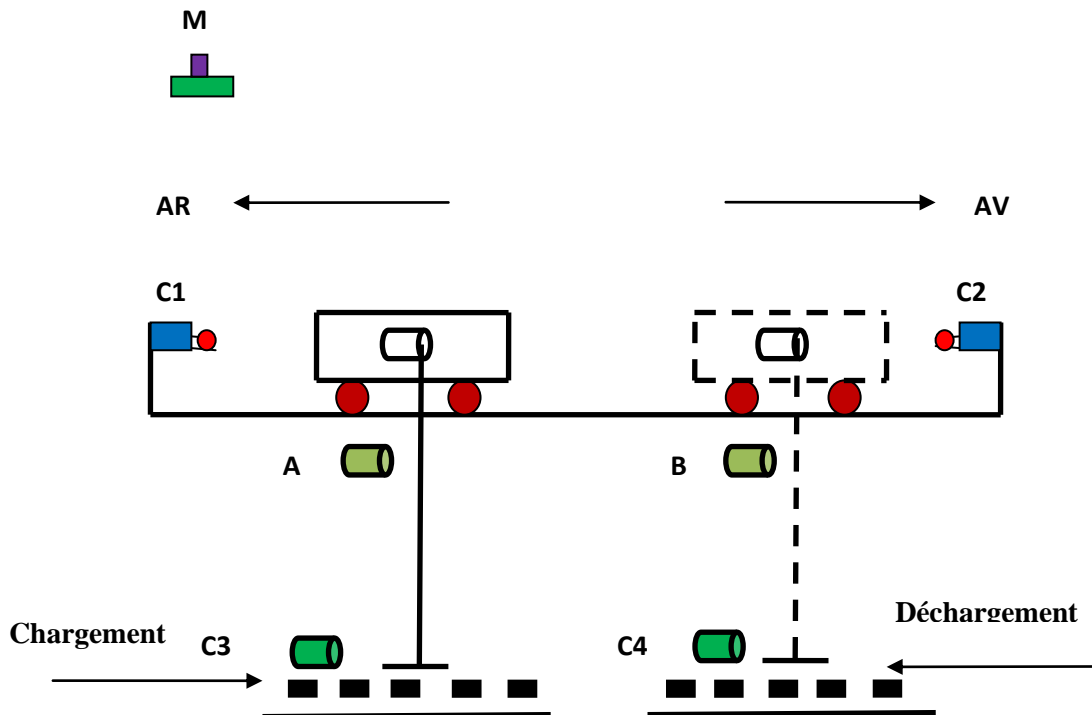
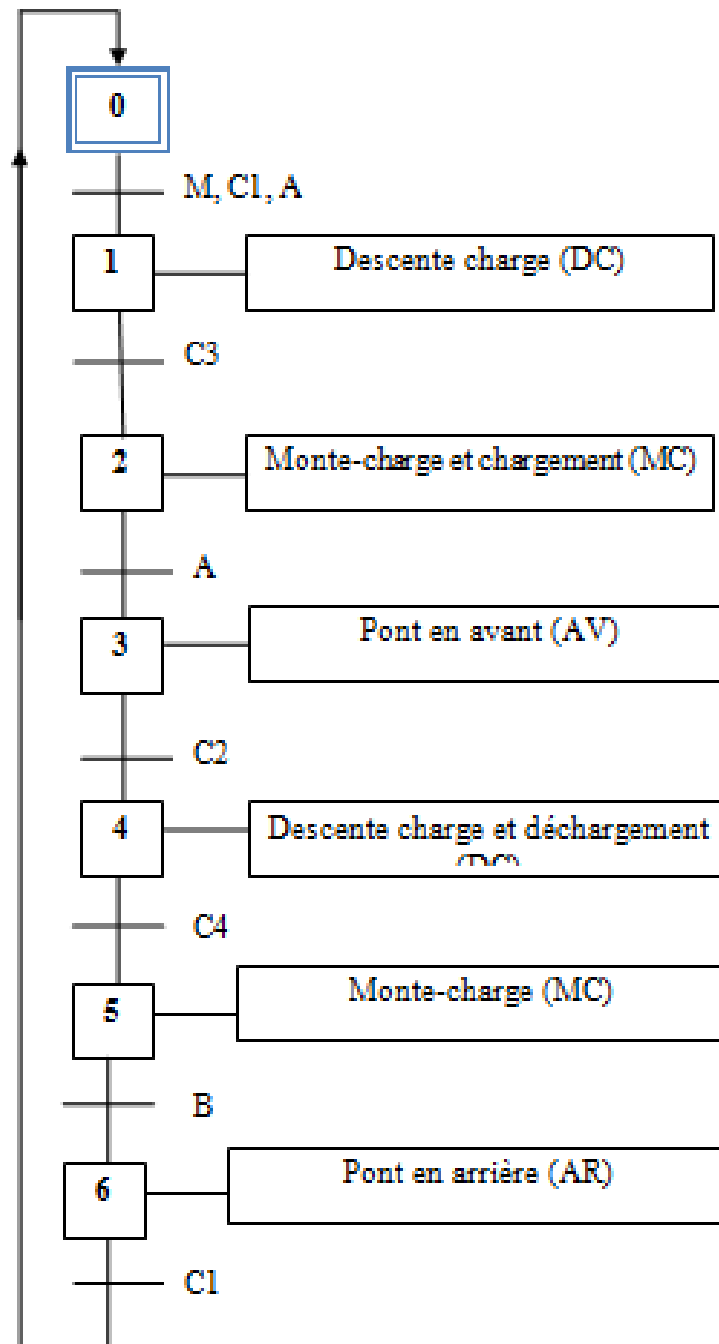


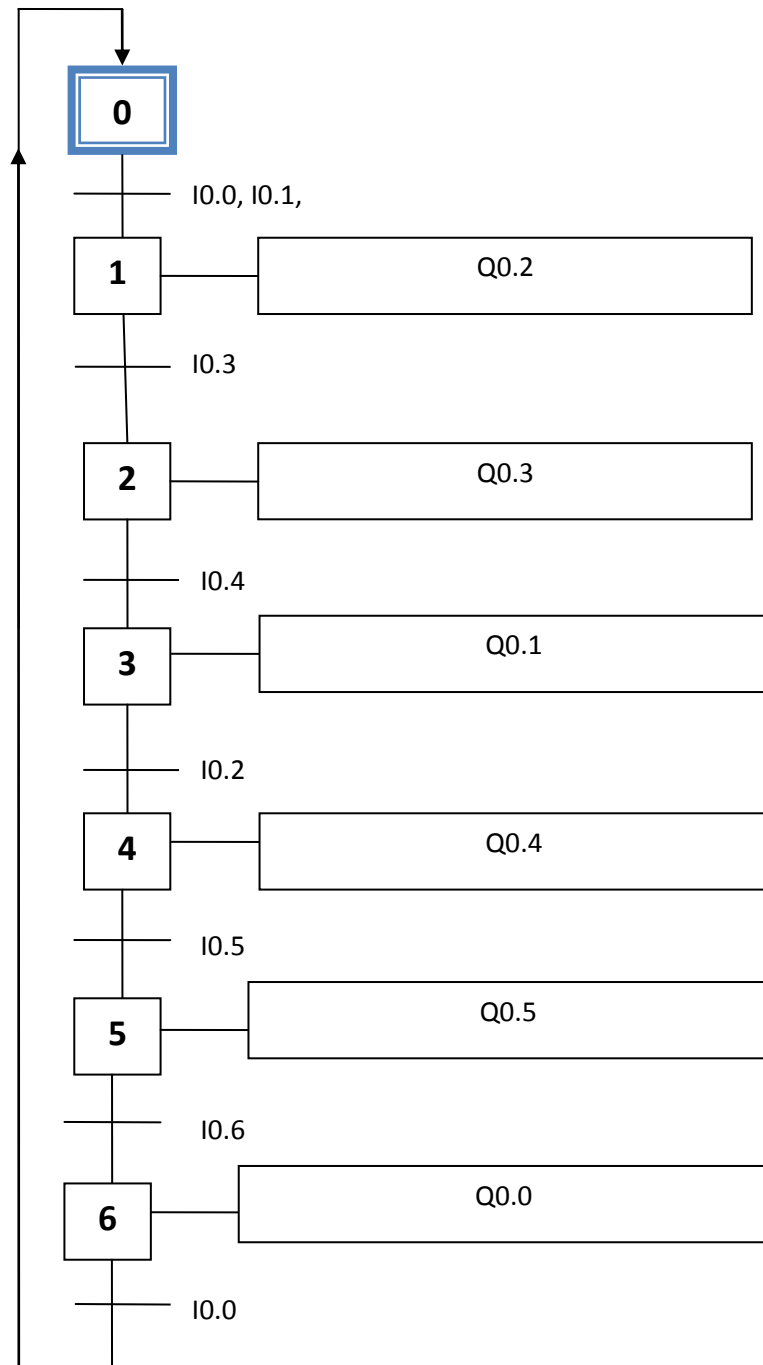
Figure 26. Pont roulant automatisé

Question:

Réaliser le grafcet de niveau-02 et niveau-03.

Entrées automate programmable		Sorties automate programmable	
I0.0	M		
I0.1	C1 (capteur mécanique)	Q0.0	AR
I0.2	C2 (capteur mécanique)	Q0.1	AV
I0.3	C3 (capteur magnétique)	Q0.2	Descente charge (DC)
I0.4	A (capteur magnétique)	Q0.3	Monte-charge (MC)
I0.5	C4 (capteur magnétique)	Q0.4	Descente charge (DC)
I0.6	B (Capteur magnétique)	Q0.5	Monte-charge (MC)

Solution-03**- Grafcet de niveau-02****Figure 27.** Grafcet de niveau-02 du pont roulant automatisé

- Grafcet de niveau-03**Figure 28.** Grafcet de niveau-03 du pont roulant automatisé

V.4. Reprise d'une séquence

Afin de comprendre la reprise d'une séquence on prend le chariot de marchandise représenté par la figure.19. Ce type de grafctet nous a permis de ne pas continuer le cycle si les actions sont répétitives.

Grafctet correspondant

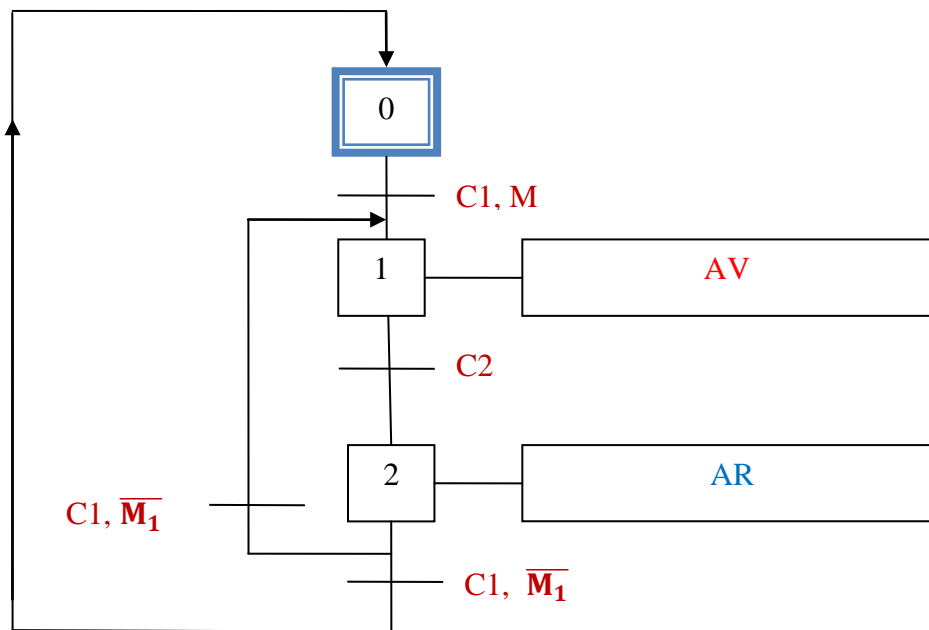


Figure 29. Grafctet de niveau-02 (reprise d'une séquence)

D'après la figure 16 on remarque que le cycle possède deux possibilités de travail :

- 1- Le cycle revient à l'étape 1 et ne continue pas vers l'étape 3.
- 2- Le cycle peut passer directement vers l'étape 3.

V.5. Saut d'étape

Ce type de grafctet nous a permis de sauter une ou plusieurs étapes. Pour réaliser ce grafctet on prend l'exemple du chariot représenté par la figure 19.

Conditions de fonctionnement

Si le chariot est en position initiale (C1), alors une impulsion sur M le chariot fait un avancement en avant et en arrière.

Si initialement le chariot n'est pas en position initiale (C1), alors une impulsion sur M provoque le déplacement du chariot au point (C1).

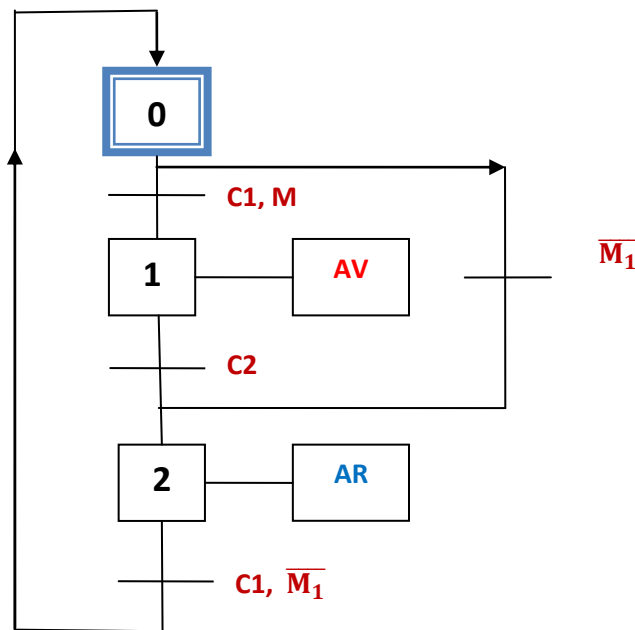


Figure 30. Grafcet de niveau-02 (saut d'étape)

D'après la figure 20 on remarque que le cycle possède deux possibilités de fonctionnement :

- 1- Le cycle continue de travailler jusqu'à l'étape 2 ;
- 2- Le cycle ignore l'étape 1 et continue directement vers l'étape 2

V.6. Aiguillage en OU

Ce type de grafcet est basé sur l'étape initiale "0" c'est-à-dire à la sortie de cette étape on a le choix entre plusieurs séquences. Le choix que nous avons préféré dépend aux différentes transitions ainsi que ces dernières dépendent aux réceptivités.

V.6.1. Exemples d'applications sur le grafcet en OU

Dans un atelier de mécanique situé au niveau de L'ISTA, il existe un poste de perçage automatisé qui nous a permis le perçage et le serrage des pièces mécaniques, voir la figure.31. Son fonctionnement est basé sur le mouvement de rotation du moteur électrique dans les deux sens (marche avant et marche arrière). Le serrage et le desserrage des pièces mécaniques sont effectués par le vérin-V1, mais pour les capteurs de positions son rôle est fixé l'arrivée et le retour du vérin-V1 ainsi que du moteur électrique M.

Conditions :

- Le système automatisé ne peut pas fonctionner si les deux boutons poussoirs **M** et **M1** non actionnés.
- Le bouton poussoir **M** et actionné avant **M1**

Remarque :

L'avancement du moteur électrique est assuré par le vérin V2

Fiche des données techniques

Entrées automate programmable		Sorties automate programmable	
I0.0	C1	Q0.0	M-
I0.1	C2	Q0.1	M+
I0.2	C3	Q0.2	V1-
I0.3	C4	Q0.3	V1+
I0.4	C5		
I0.5	C6	Q0.5	P+
I0.6	M		
I0.7	M1		

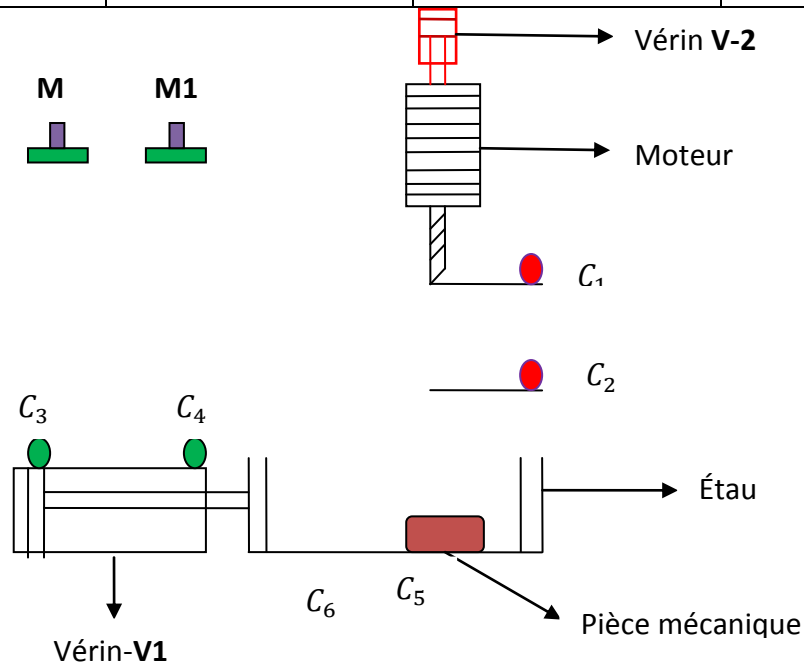


Figure 31. Poste de perçage automatisé (aiguillage en OU)

Questions

- Réaliser le grafcet de point de vue partie système ;
- Réaliser le grafcet de point de vue partie opérative ;
- Réaliser le grafcet de point de vue partie commande.

Solution

- **Grafcet de niveau-01**

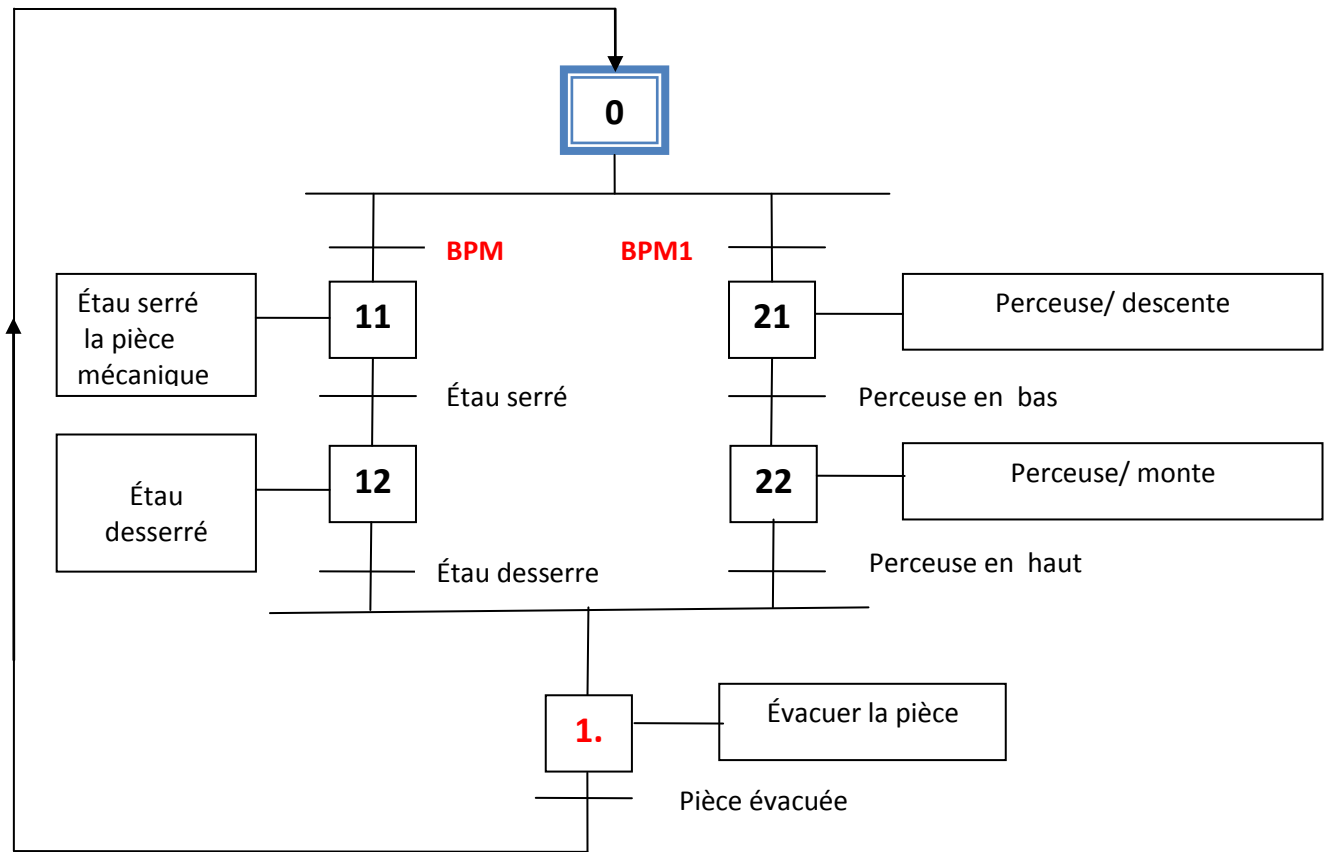


Figure 32. Grafctet de niveau-01(aiguillage en OU)

- Grafctet de niveau-02

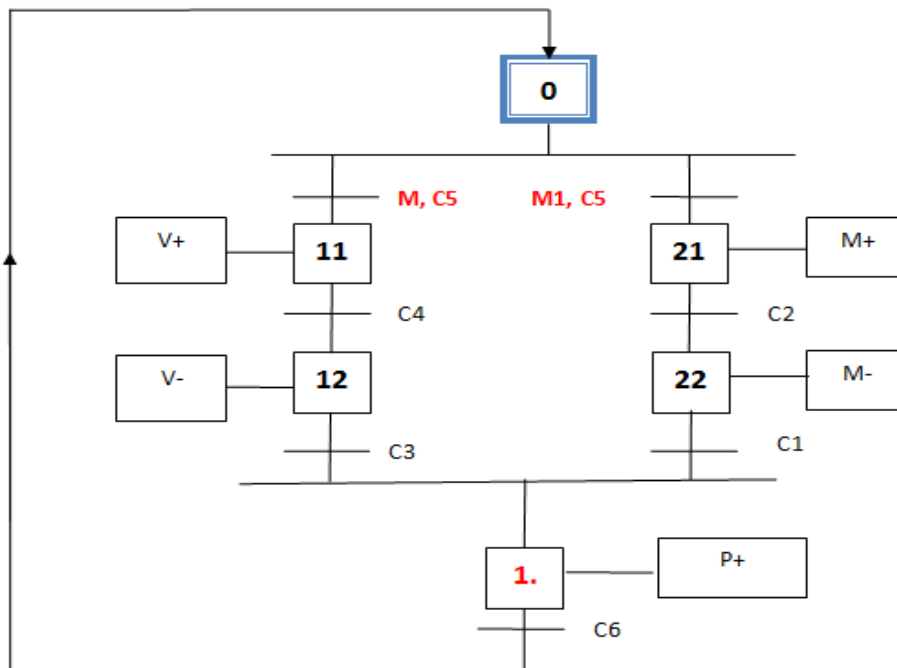


Figure 33. Grafctet de niveau-02 (aiguillage en OU)

- **Grafctet de niveau-03**

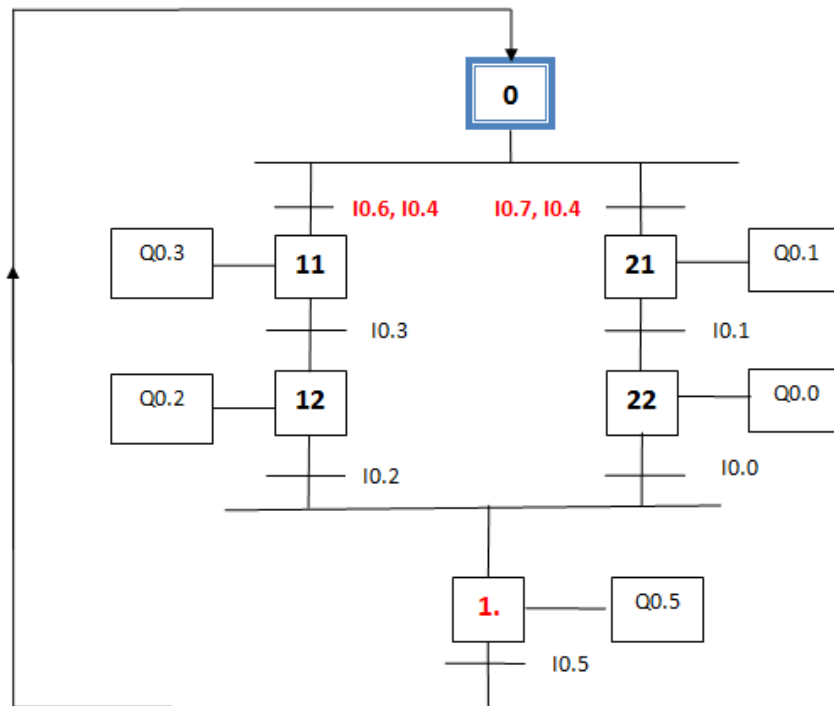


Figure 34. Grafctet de niveau-03 (aiguillage en OU)

V.6.2. Aiguillage en ET

Les séquences dans ce type du grafctet sont exécutées au même temps (simultanément).

Exemple d'application sur le grafctet ET

On prend l'exemple précédent, mais on a un seul bouton poussoir **M** qui fait démarrer le système automatisé.

Questions :

- Réaliser le grafctet de niveau-01 ;
- Réaliser le grafctet de niveau-02 ;
- Réaliser le grafctet de niveau-03.

Solution

- **Grafctet de niveau-01**

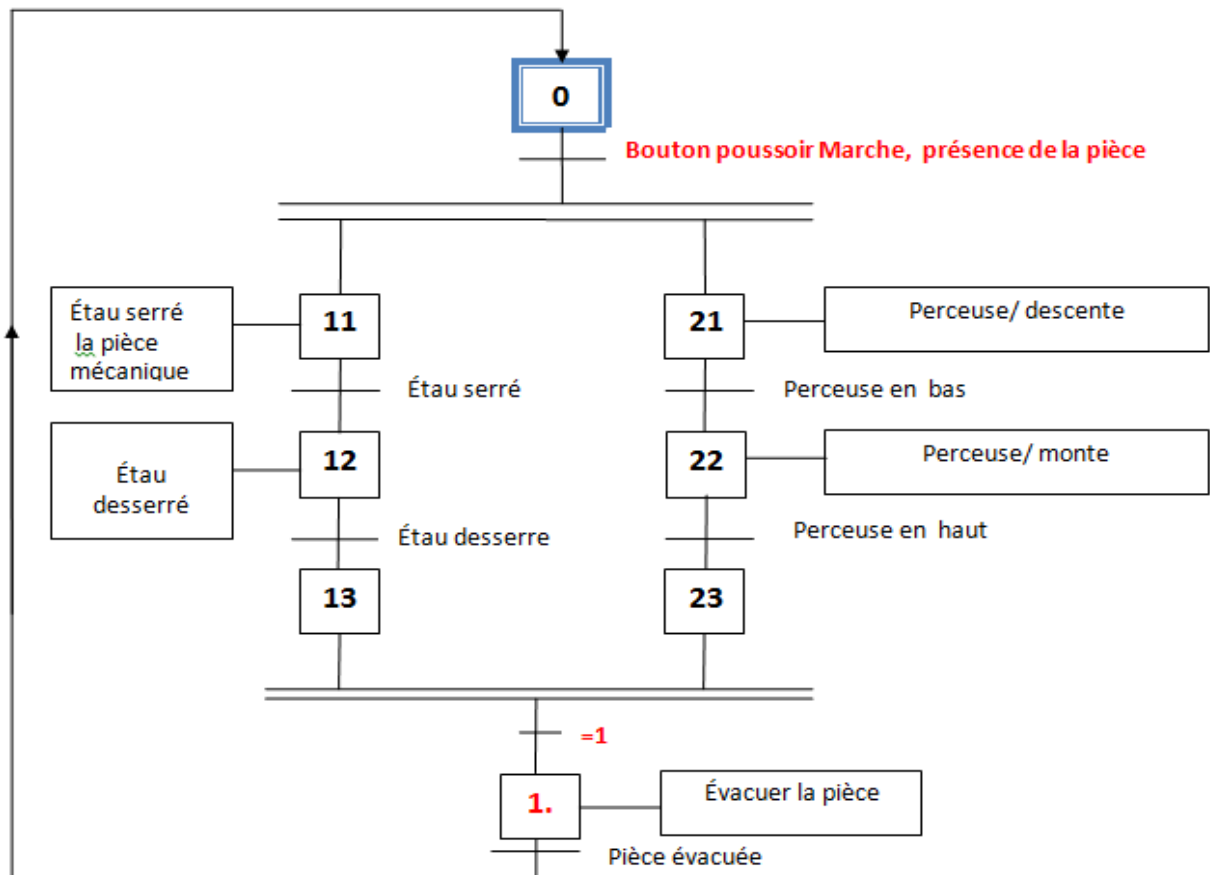


Figure 35. Grafcet de niveau-01 (aiguillage en ET)

- Grafcet de niveau-02

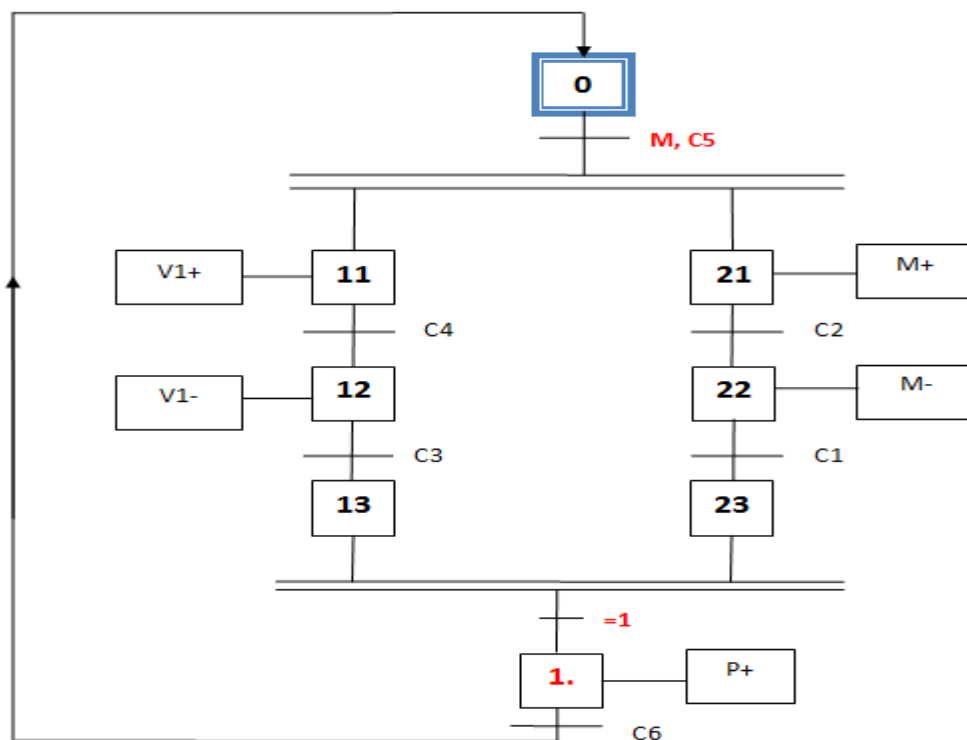


Figure 36. Grafcet de niveau-02 (aiguillage en ET)

- **Grafctet de niveau-03**

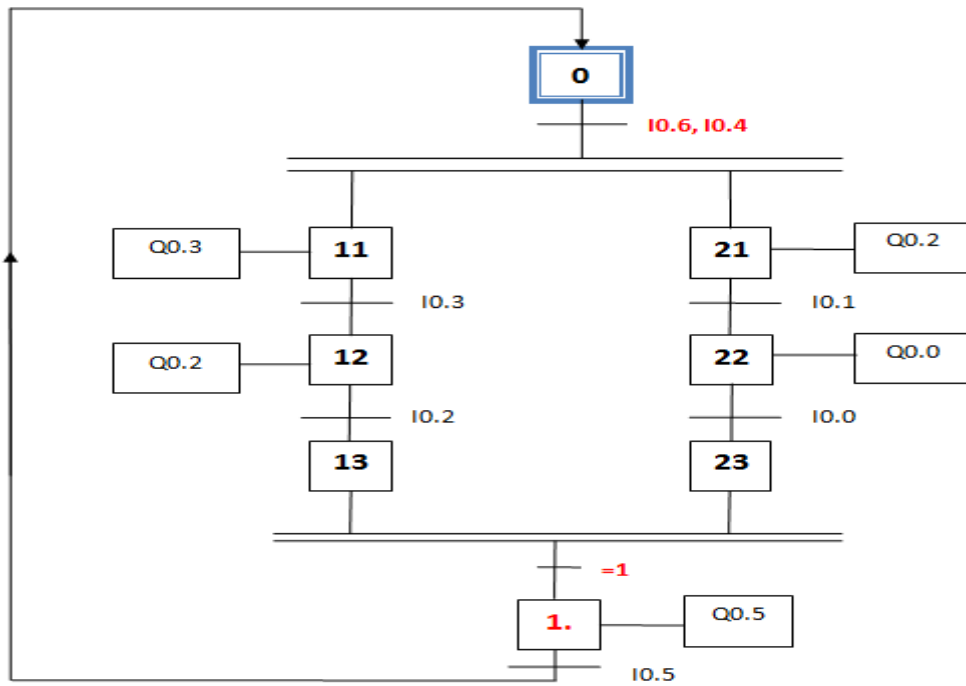


Figure 37. Grafctet de niveau-03 (aiguillage en ET)



Exercice d'application sur le grafctet où nous avons un moteur asynchrone à cage

Soit le circuit de puissance et de commande du moteur asynchrone à cage d'écureuil.

Voir le schéma ci-dessous.

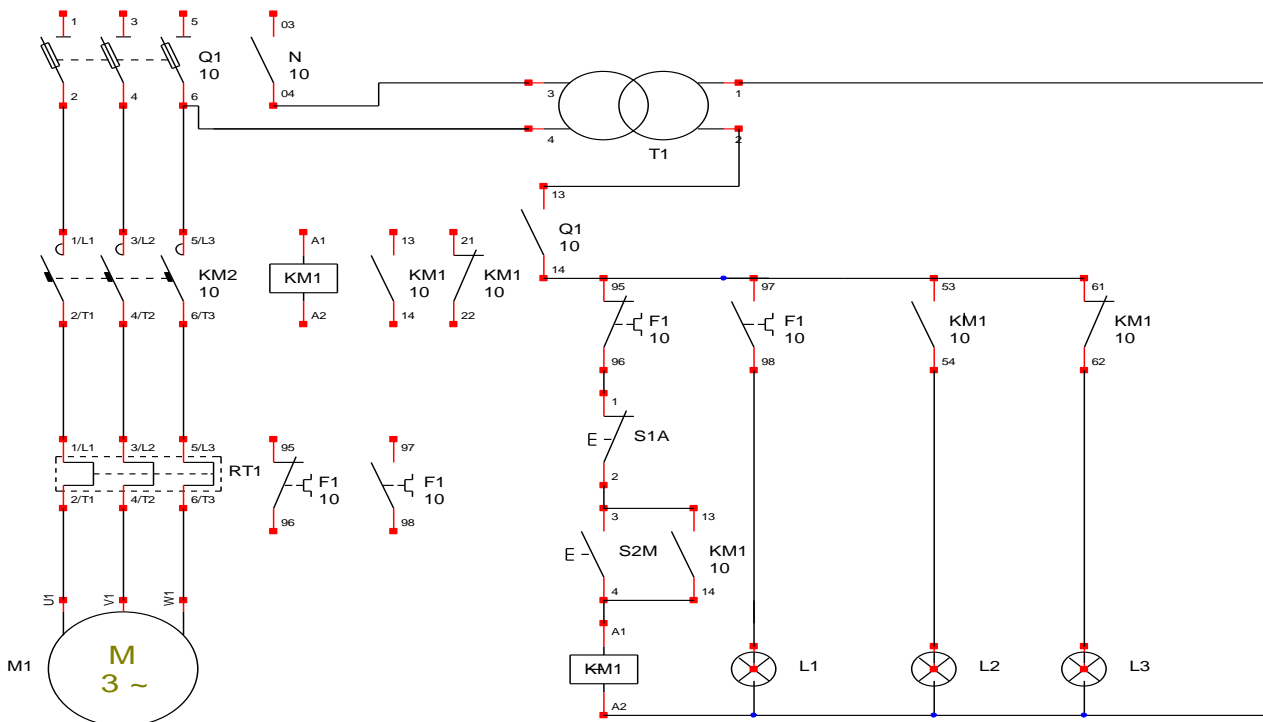


Figure 38. Démarrage direct du MAS à cage triphasé.

Questions

- 1- À partir du circuit de commande du moteur asynchrone, Réaliser le programme en langage à contact (ladder) sous **Twidosuite**.
- 2- Réaliser le grafcet de niveau-02 et niveau-03.
- 3- Réaliser le grafcet de niveau-2 si le moteur chauffe et le relais thermique disjoncte.

Remarque

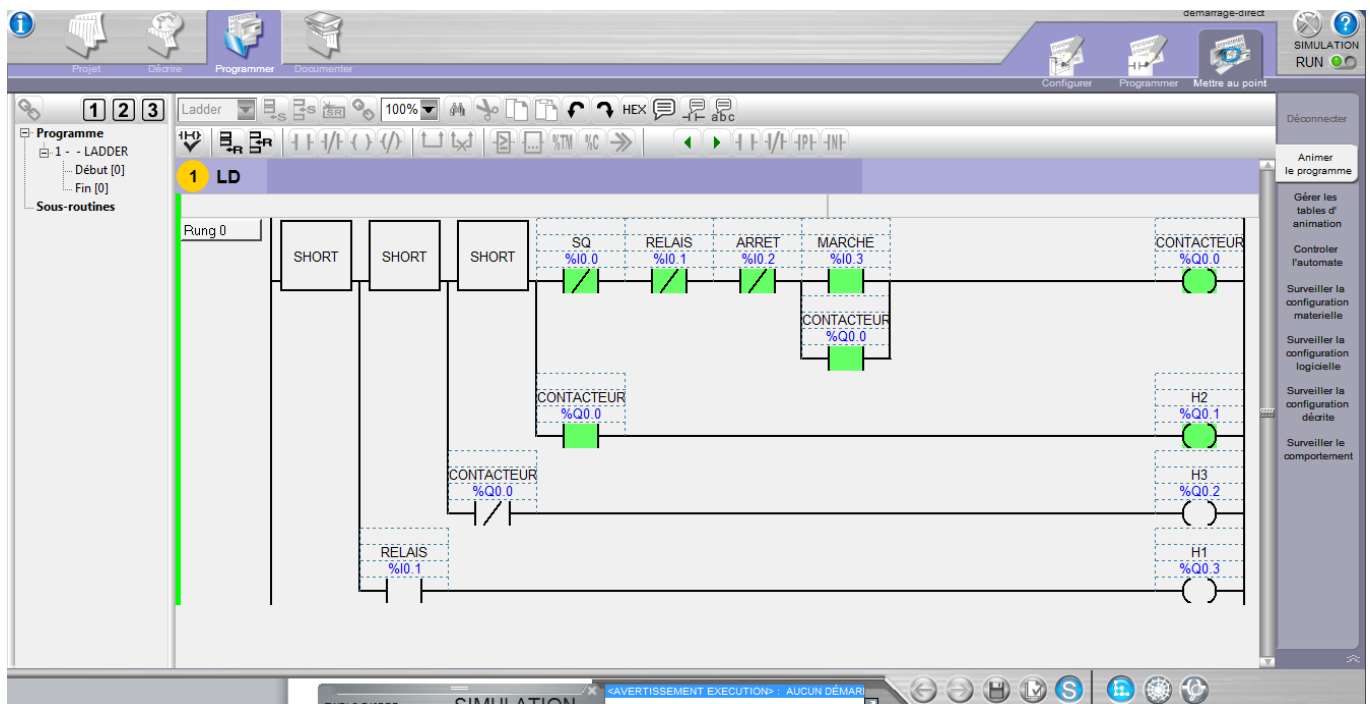
On suppose que le contact 13/14 du sectionneur est fermé, et le moteur sous tension.

1- Fiche des données techniques

Entrées automate programmable		Sorties automate programmable	
I0.0	Q1	Q0.2	L3
I0.1	F1	Q0.1	L2
I0.2	S2M+KM1	Q0.3	M+
I0.3	S1A	Q0.0	L1

Solution

1- Langage à contact sous **TWIDOSUITE**



2- Grafcet de niveau-2 et de niveau-3

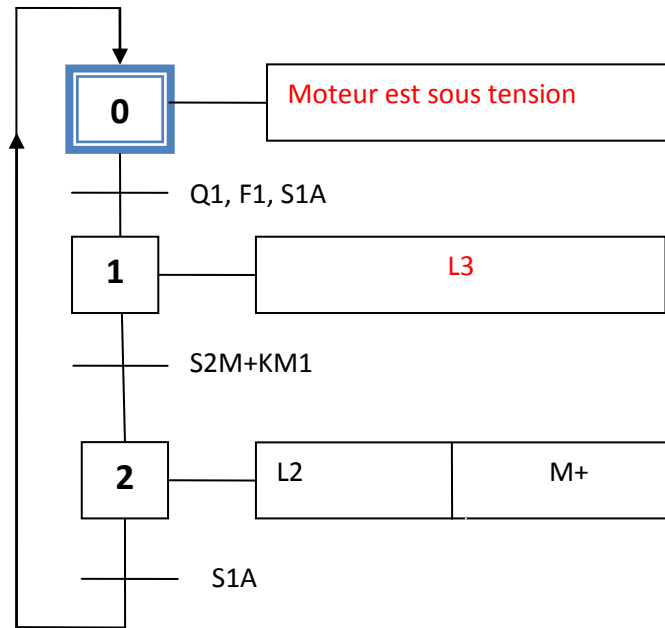


Figure.39. Grafcet de niveau-2 du MAS

- Grafcet de niveau-03

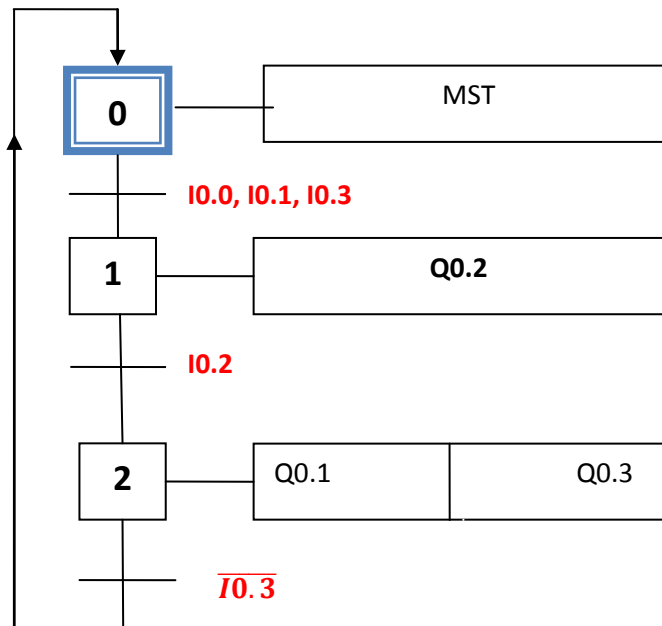


Figure 40. Grafcet de niveau-3 du MAS

3- Grafcet de niveau-2 si le moteur chauffe

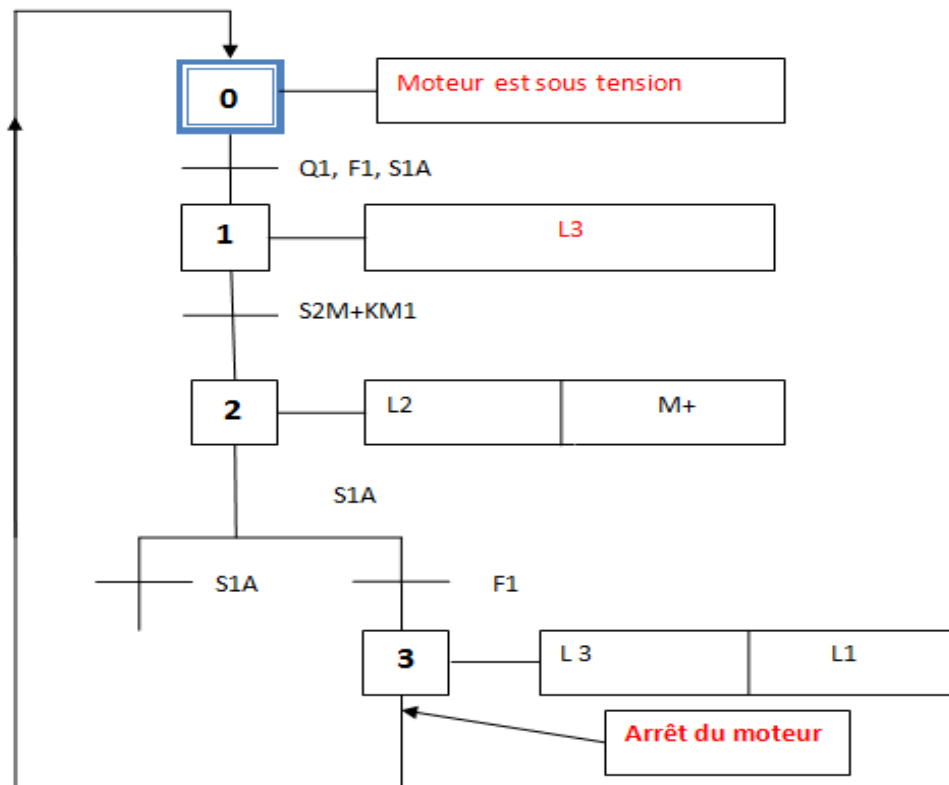


Figure 41. Grafcet de niveau-2 si le moteur chauffe

VI. Guide d'étude des modes de marches et d'arrêts (GEMMA)

VI.1. Définition

Pour mettre un système automatisé dans un fonctionnement dynamique élevé, il faut passer par l'étude d'un GEMMA ce dernier nous a permis de donner une aperçus claire et très important dans tous les systèmes automatisés séquentiels pendant la production automatique. À partir du GEMMA on connait le comportement pour chaque élément d'une chaine de production pendant le fonctionnement par exemple : la tension d'alimentation d'un moteur électrique est insuffisante, température d'un four électrique très basse, un arrêt d'urgence d'une installation ne fonctionne pas correctement, le GEMMA affiche tous ces défauts et permet d'intervenir pour régler les anomalies avant qu'il ne soit trop tard.

VI.2. Représentation graphique du GEMMA

Le GEMMA est un guide graphique structuré qui propose des modes de fonctionnement types. Selon les besoins du système automatisé à étudier on choisit d'utiliser certains modes de fonctionnement.

Le guide graphique GEMMA est divisé en "rectangle d'état". Chaque rectangle d'état a une position précise sur le guide graphique. Chaque rectangle d'état est relié à un ou plusieurs autres rectangles d'états par des flèches orientées.

Le passage d'un rectangle d'état à un autre s'effectue un peu à la manière du franchissement d'une transition de grafcet. Le guide graphique GEMMA n'est pas un outil figé, il est modulable à volonté suivant les spécifications à obtenir.

VI.3. Structuration du GEMMA

Le GEMMA est constitué par deux zones qui sont :

VI.3.1. Partie commande hors énergie (PZ) : Dans cet état la partie opérative n'est pas

sous le contrôle de la partie commande. La partie opérative peut être en énergie ou hors énergie. La sécurité est garantie par les choix technologiques et la procédure de mise en énergie de la partie opérative. Cette zone du GEMMA, située à l'extrême gauche, correspond à l'état inopérant de la partie commande.

VI.3.2. Partie commande sous énergie : C'est la partie qui va nous permettre de définir

les différents modes démarche et d'arrêt de notre machine ainsi que les conditions de passage d'un mode à l'autre. Cette partie est subdivisée en trois zones ou en trois familles de procédures.

- ✚ Les procédures de fonctionnement (**F**);
- ✚ Les procédures en défaillances (**D**);
- ✚ Les procédures d'arrêts (**A**).

La figure ci-dessous représente les deux zones de la partie commande, hors énergie et en énergie avec ces procédures.

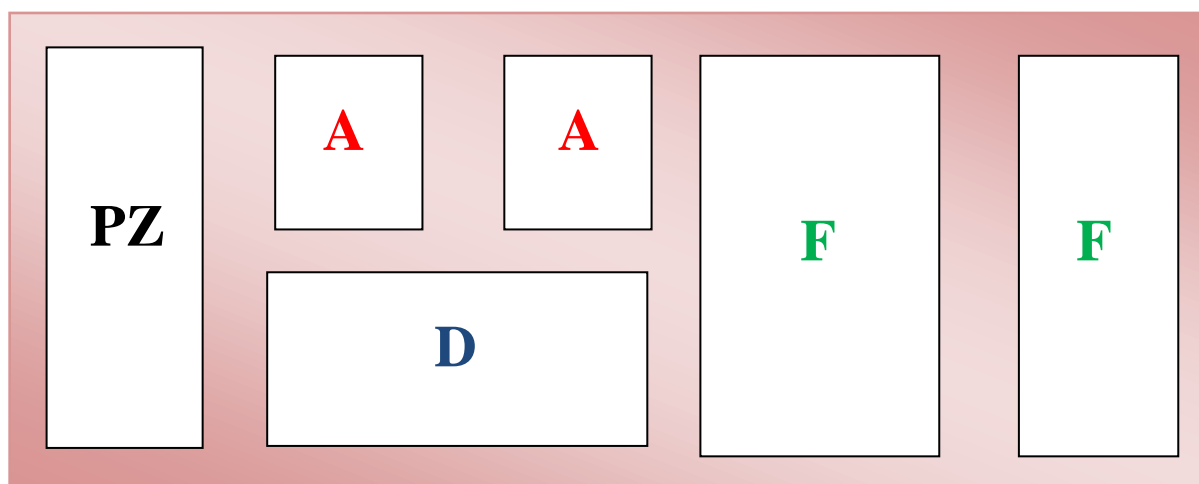


Figure 42. Les zones de la partie commande du GEMMA.

VI.3.2.1. Procédures de fonctionnement (F)

L'ensemble des modes ou états sans lesquels la valeur ajoutée ne peut être obtenue est regroupée dans une zone F représentative de la famille "procédure de fonctionnement". Note : On ne produit pas systématiquement dans chacun des modes de cette famille, il peut s'agir de :

- Procédures préparatoire à la production de la valeur ajoutée
- réglages, tests ...

Ils sont néanmoins indispensables à la production de la valeur ajoutée.

➤ **F1 ‘Production normale’**

Dans cet état, la machine produit normalement : c'est l'état pour lequel elle a été conçue. C'est à ce titre que le "rectangle-état" a un cadre particulièrement renforcé. On peut souvent faire correspondre à cet état un GRAFCET que l'on appelle GRAFCET de base.

➤ **F2 ‘Marche de préparation’**

Cet état est utilisé pour les machines nécessitant une préparation préalable à la production normale : préchauffage de l'outillage, remplissage de la machine, mises en routes diverses, etc.

➤ **F3 ‘Marche de clôture’**

C'est l'état nécessaire pour certaines machines devant être vidées, nettoyées, etc., en fin de journée ou en fin de série.

➤ **F4 ‘Marche de vérification dans le désordre’**

Cet état permet de vérifier certaines fonctions ou certains mouvements sur la machine, sans respecter l'ordre du cycle.

➤ **F5 ‘Marche de vérification dans l'ordre’**

Dans cet état, le cycle de production peut être exploré au rythme voulu par la personne effectuant la vérification, la machine pouvant produire ou ne pas produire.

➤ **F6 ‘Marche de test’**

Les machines de contrôle, de mesure, de tri..., comportent des capteurs qui doivent être réglés ou étalonnés périodiquement : la ‘ Marche de test ‘ F6 permet ces opérations de réglage ou d'étalonnage.

V.3.2.2. Procédures en défaillances (D)

Les procédures en défaillance définissent les états que devra avoir la partie opérative en cas de défaillance. Ils sont au nombre de trois. Ces rectangles d'état permettent de gérer les défaillances du système tel que par exemple l'arrêt d'urgence. Les états de défaillance sont les états de défaillance de la **PO**.

- **D1 "Arrêt d'urgence"** : Cet état permet de gérer le système lors d'un arrêt d'urgence. On y prévoit non seulement les arrêts, mais aussi les cycles de dégagements, les procédures et précautions nécessaires pour éviter ou limiter les conséquences dues à la défaillance.
- **D2 "Diagnostic et ou traitement de défaillance"** : Cet état permet à la maintenance de diagnostiquer l'origine de la défaillance et d'envisager le traitement approprié qui permettra le redémarrage du système après traitement de la défaillance.
- **D3 " Production tout de même"** : Il est parfois nécessaire de continuer la production après défaillance du système : on aura alors (un production dégradée), ou une (production forcée), ou une production aidée par des opérateurs non prévus en production normale.

V.3.2.3. Les procédures d'arrêts (A).

- **A1 "Arrêt dans l'état initial "**

C'est l'état "repos" de la machine. Il correspond en général à la situation initiale du GRAFCET : c'est pourquoi, comme une étape initiale, ce "rectangle-état" est entouré d'un double cadre. Pour une étude plus facile de l'automatisme, il est recommandé de représenter la machine dans cet état initial.

- **A2 "Arrêt demandé en fin de cycle "**

Lorsque l'arrêt est demandé, la machine continue de produire jusqu'à la fin du cycle. A2 est donc un état transitoire vers l'état A1.

- **A3 " Arrêt demandé dans état déterminé "**

La machine continue de produire jusqu'à un arrêt en une position autre que la fin de cycle : c'est un état transitoire vers A4.

- **A4 " Arrêt obtenu "**

La machine est alors arrêtée en une autre position que la fin de cycle.

- **A5 " Préparation pour remise en route après défaillance "**

C'est dans cet état que l'on procède à toutes les opérations (dégagements, nettoyages,..) nécessaires à une remise en route après défaillance.

- **A6 " Mise P.O. dans état initial"**

La machine étant en A6, on remet manuellement ou automatiquement la Partie Opérative en position pour un redémarrage dans l'état initial.

- **A7 " Mise P.O. dans état déterminé "**

La machine étant en A7, on remet la P.O. en position pour un redémarrage dans une position autre que l'état initial.

VI.4. Les rectangles états

Le GEMMA regroupe seize rectangles état comme les représentés sur la figure ci-dessous.

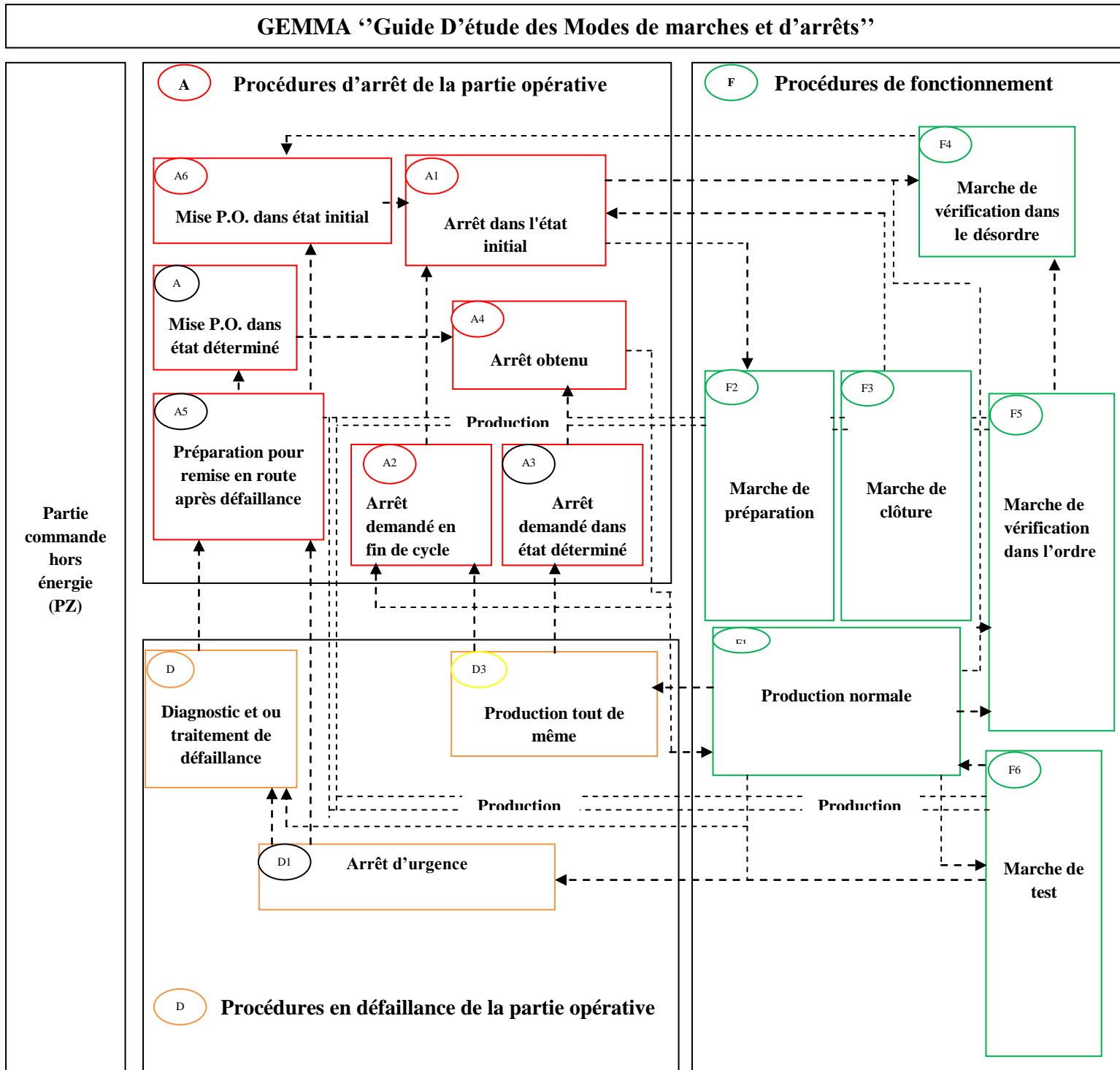


Figure 43. Les rectangles états du GEMMA.

✚ Exercice d'application sur le GEMMA :

On considère l'exemple simplifié de perçage semi-automatique illustré par la figure ci-dessous. Les pièces à percer sont montées et démontées manuellement. L'opérateur doit aussi fermer et ouvrir le capot de protection. La description des modes de marches qui tient compte des besoins de la production et de sécurité, prévoit deux modes principaux: le mode automatique (états 1 et 2) et le mode défaillance (états 3 et 4).

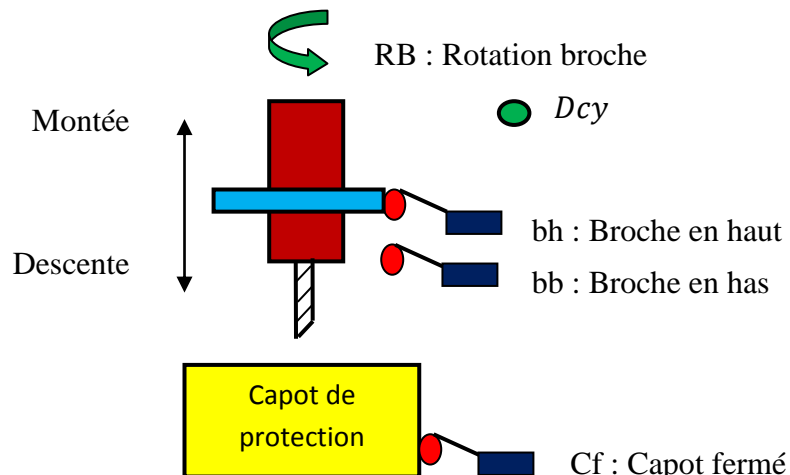


Figure 44. Perçage semi-automatique.

Description des différents états:

État 1 : La mise en place de la pièce est possible, la partie commande devra assurer la sécurité de descente de la broche tant que le capot est ouvert.

État 2 : Le bouton « départ cycle » (*Dcy*) permet le passage à l'état 2 dans lequel s'effectue le perçage automatique. La fin du cycle provoque le retour à l'état 1

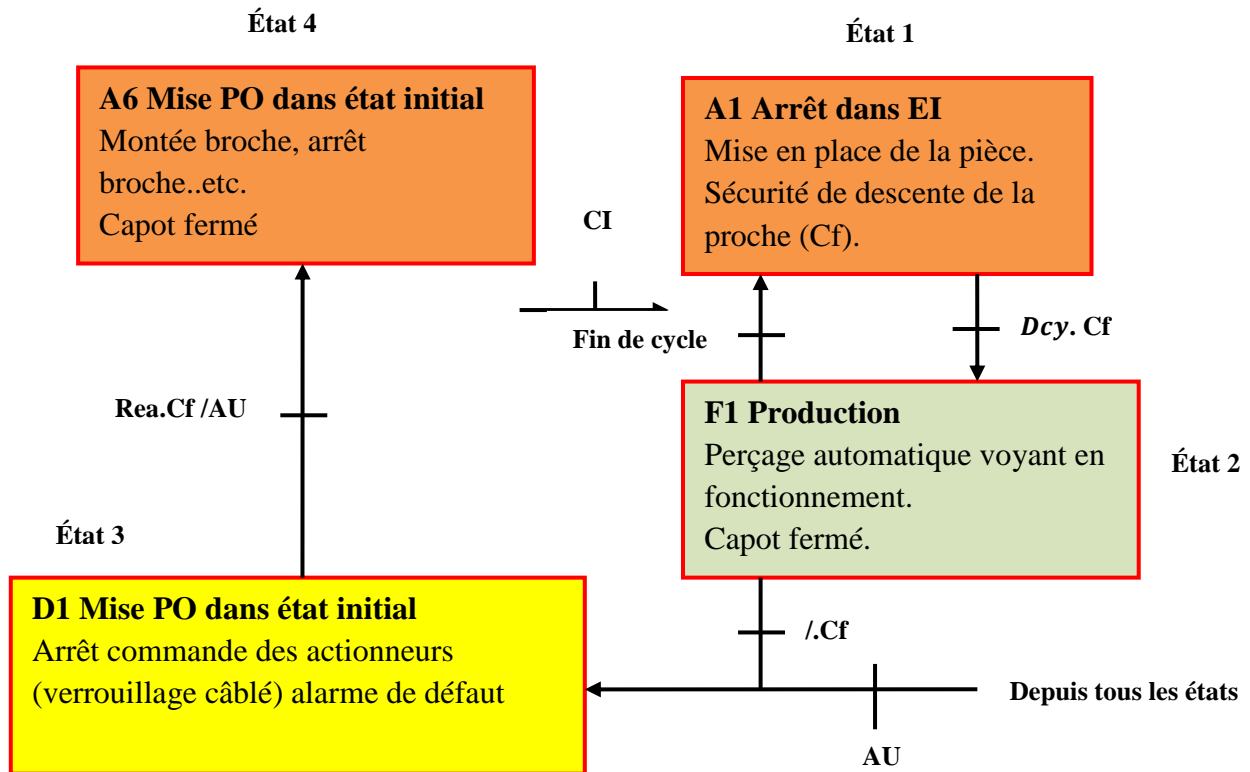


Figure 45. Modes de marches d'un système de perçage semi-automatique.

La figure ci-dessous présente le Grafcet complété (utilisant des ordres de forçage) de cet automatisme. Le suivi des modes de marches est une fonction de commande facile à décrire en Grafcet. En effet, il est possible de réaliser un Grafcet référant les modes de marches prévu par l'équipement dans lequel chaque étape représente un état défini et pour lequel chaque réceptivité associée à une transition concrétise les conditions d'évolution d'un état à un autre. Ainsi, Les étapes 1,2,3,4 correspondent respectivement aux états 1,2,3, et 4.

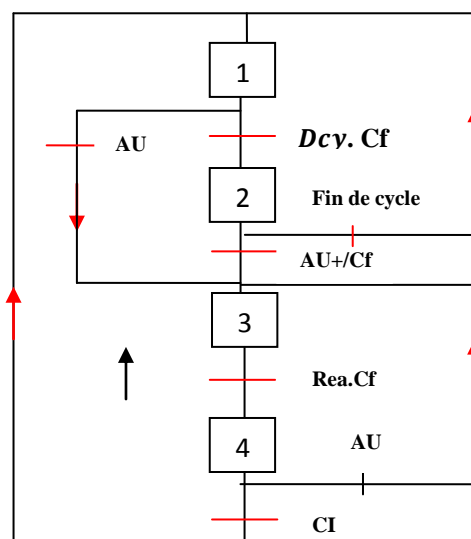
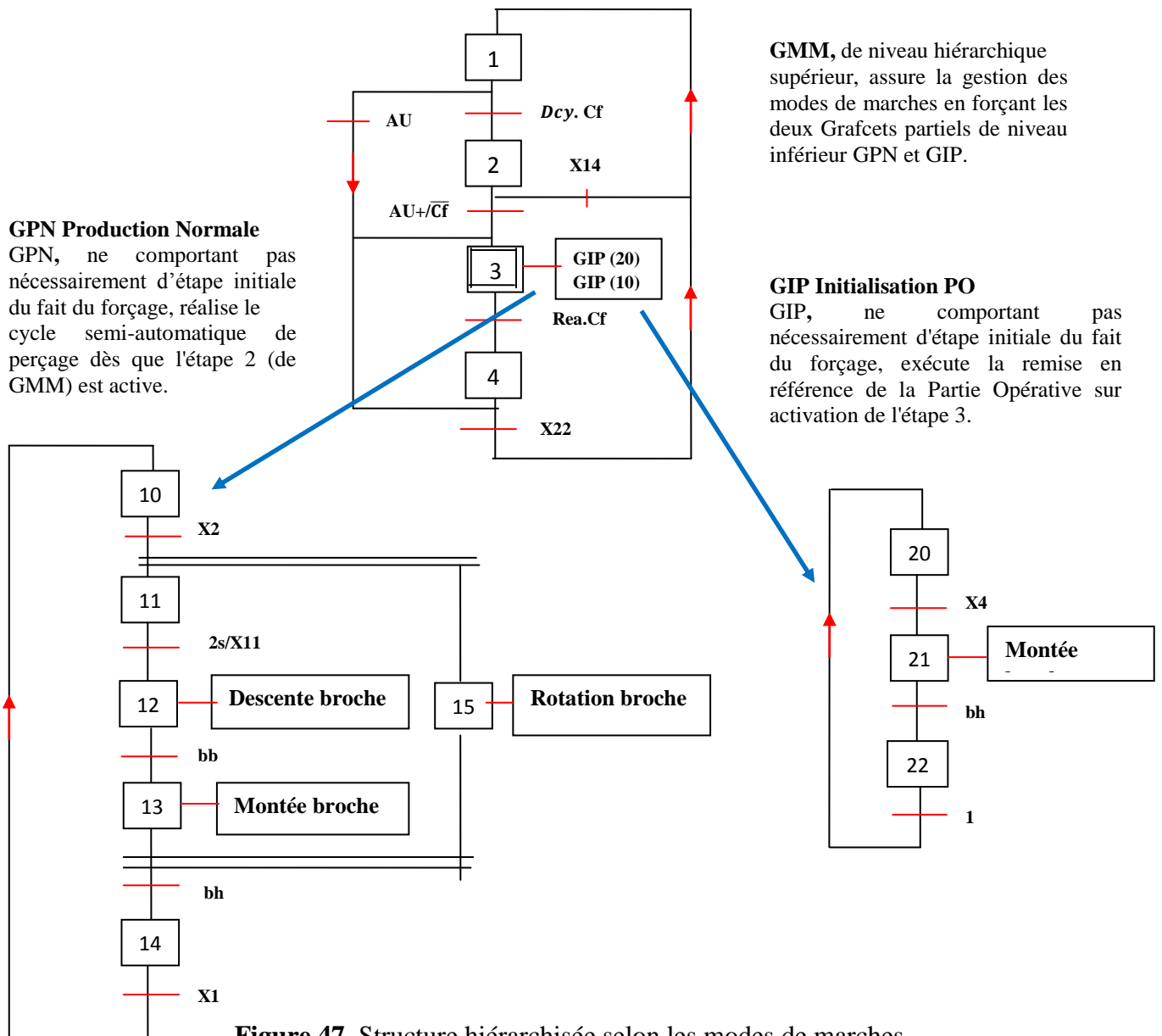


Figure 46. Grafcet complétée en utilisant les ordres de forçage.

Cette première approche dite d'enrichissement du Grafcet doit être complétée pour réaliser la gestion des modes de marches pour la partie commande.

La deuxième approche comme le montre la figure ci-dessous consiste à la structuration hiérarchisée. On note que dans cet exemple on a utilisé la notion de forçage d'un Grafcet partiel (Étape 3). En effet, l'ordre de forçage de situation émis par un Grafcet hiérarchiquement supérieur permet de modifier la situation courante d'un Grafcet hiérarchiquement inférieur, sans qu'il y ait franchissement de transition. L'ordre de forçage est un ordre interne prioritaire sur toutes les conditions d'évolution et a pour effet d'**activer la ou les étapes** correspondant à la **situation forcée** et de **désactiver les autres étapes** du Grafcet forcé. L'ordre de forçage est représenté dans un double rectangle associé à l'étape pour le différencier d'une action.



Bibliographie

- [1] KHATORY, "Initiation Informatique et Système de Numération" Université Sidi Mohammed ben Abdellah, École Supérieure de Technologie de Fès Filière Génie Industriel et Maintenance.
- [2] Nadia SOUAG "logique Combinatoire Cours et Exercices Corrigés", Alger 2004
- [3] Menacer Said, Menacer Mohamed, Menacer abderahmene, "Électronique Digitale Tome1 Analyse Combinatoires et Séquentielles aout 1990.
- [4] Eric Cariou, "Algèbre de Boole" Université de Pau et des Pays de l'Adour UFR Science
- [5] Michel Riquart "Logique Combinatoire"
- [6] L.Djeffal, "Application de L'algèbre de Boole aux Circuits Combinatoires et Séquentiels Cours et Exercices avec Solution", Presses de L'université de Batna 1996.
- [7] M.c Belaid, "Les circuits Logiques Combinatoires et Séquentiels", Cours et Exercices corrigés Alger 2004.
- [08] Peter YK cheung, "Lecture 11 Digital Logic, Boolean Algebra", Flip-Flops, 29, may 2018.
- [09] W.Bolton, "Chapter 11 Ladder and Functional Block Programming"
- [10] Paulo Jorge Oliveira, José Gaspar, "Industrial Automation PLC Programming Languages Instruction list", 2010/2011.
- [11] W. Bolton, "Programmable Logic Controllers", Fourth Edition, Elsevier E ELSEVIE
- [12] Mourad KCHAOU, "Introduction à l'Automatisme GRAFCET & GEMMA", Université de Sousse
- [13] Moez AYADI, "Cours Automatisme Industriels", ISET — Nabeul A.U. : 2016 / 2017

Sites Web

https://www.academia.edu/29304339/formation_modulaire_compagnons_%c3%89lectriciens_du_devoir

<https://docplayer.fr/amp/20682866-les-automatismes-gjc-lycee-l-rascol-10-rue-de-la-republique-bp-218-81012-albi-cesdex.html>

<http://bts.crsa.rascol.free.fr/automatismes/cours/le%20gemma.pdf>

<http://docplayer.net/36888564-chapter-9-discrete-control-using-programmable-logic-controllers.html>

http://www.just.edu.jo/~haalshraideh/courses/ie431/lecture_slides/ch9_plc.pdf

http://www.gecif.net/articles/genie_electrique/cours/terminale/cours/les_differeents_codes.pdf

<http://www.pdf4free.com 'cours 4 circuits combinatoires'>

http://meidoyen.openelement.fr/files/other/structure%20generale_prof.pdf

<https://sb47a502462c9ef47.jimcontent.com/download/version/1491229657/m.pdf>

http://lyceeduruy.fr/si/files/2010/09/cours_systemes.pdf.....8

<https://www.google.com/url?sa=i&url=https%3a%2f%2fwww.cdiseout.com%2fhigh-.html>

https://www.technic-achat.com/boutique/images_produits/1272xxxxxx-z.png

https://img.directindustry.fr/images_di/photo-g/9226-2463163.jpg

https://lh3.googleusercontent.com/proxy/qztzizbgan3wttvlvvhg_tfkxdreqc18ed2ze

<https://www.automation-sense.com/medias/images/automate-siemens-prix.jpg>

<https://www.axesindustries.com/images/np/9416fr.jpg>

https://img.directindustry.fr/images_di/photo-g/64974-9397959.jpg

<https://www.academia.edu>

<https://www.futura-sciences.com/tech/definitions/informatique-automate-programmable-10525/>

https://fr.wikipedia.org/wiki/automate_programmable_industriel

<http://pamelard.electro.pagesperso-orange.fr/fichier%20pdf/automatisme/programmation.pdf>

http://www.lpmei.com/cd_bac_mei/ressources/10%20ressource%20automate.pdf

<http://www.mccours.net/cours/pdf/info/gemma.pdf>

https://isetna.files.wordpress.com/2012/05/chapitre-7presentation_gemma.pdf

<http://updates@academia-mail.com>

<https://docplayer.fr/27186278-chapitre-7-gemma-pierre-duysinx-universite-de-liege.html>

https://www.academia.edu/36001045/guide_detude_des_modes_de_marche_et_darret

<https://www.technologuepro.com/cours-automate-programmable-industriel/gemma.pdf>

Liste des tableaux

Tableau 01 : Conversion binaire naturel en code binaire réfléchi sur 3bits.....	12
Tableau 02 : Conversion binaire naturel en code binaire réfléchi sur 4bits.....	12
Tableau 03 : Conversion décimal en code excès de 03.....	14
Tableau 04 : Conversion code décimal en code Aiken.....	15
Tableau 05 : Conversion code décimal en code ‘’ 2 parmi 5’’.....	15

Liste des figures

Figure 1.	Les parties d'un système automatisé.....	2
Figure 2.	Les pré-actionneurs.....	2
Figure 3.	Les actionneurs.....	3
Figure 4.	Les capteurs.....	3
Figure 5.	Automate programmable associe avec ordinateur.....	4
Figure 6.	Pupitre de commande.....	4
Figure 7.	Automate programmable.....	20
Figure 8.	Chariot de marchandise automatisé.....	24
Figure 9.	Commande d'un vérin double effet par automate programmable.....	25
Figure 10.	Porte logique NAND.....	30
Figure 11.	Porte logique NOR.....	31
Figure 12.	Porte logique XOR.....	32
Figure 13.	Remplissage d'un réservoir d'eau.....	35
Figure 14.	Représentation du grafcet à séquence unique.....	37
Figure 15.	Poste de perçage automatisé.....	39
Figure 16.	Grafcet de niveau -01 du poste de perçage automatisé.....	40
Figure 17.	Grafcet de niveau -02 du poste de perçage automatisé.....	41
Figure 18.	Grafcet de niveau -03 du poste de perçage automatisé.....	42
Figure 19.	Chariot de marchandise automatisé.....	43
Figure 20.	Grafcet de niveau -01 du chariot de marchandise automatisé	43
Figure 21.	Grafcet de niveau -02 du chariot de marchandise automatisé	44
Figure 22.	Grafcet de niveau -03 du chariot de marchandise automatisé	44
Figure 23.	Grafcet de niveau -01 du chariot de marchandise automatisé avec temporisation.....	45
Figure 24.	Grafcet de niveau -02 du chariot de marchandise automatisé avec temporisation.....	45
Figure 25.	Grafcet de niveau -03 du chariot de marchandise automatisé avec temporisation.....	46
Figure 26.	Pont roulant automatisé.....	47
Figure 27.	Grafcet de niveau -02 du pont roulant automatisé.....	48
Figure 28.	Grafcet de niveau -03 du pont roulant automatisé.....	49
Figure 29.	Grafcet de niveau -02 (reprise d'une séquence).....	50
Figure 30.	Grafcet de niveau -02 (saut d'étape).....	51

Figure 31.	Poste de perçage automatisé (aiguillage en OU).....	52
Figure 32.	Grafcet de niveau -01 (aiguillage en OU).....	53
Figure 33.	Grafcet de niveau -02 (aiguillage en OU).....	53
Figure 34.	Grafcet de niveau -03 (aiguillage en OU).....	54
Figure 35.	Grafcet de niveau -01 (aiguillage en ET).....	55
Figure 36.	Grafcet de niveau -02 (aiguillage en ET).....	55
Figure 37.	Grafcet de niveau -03 (aiguillage en ET).....	56
Figure 38.	Démarrage direct du MAS à cage triphasé.....	56
Figure 39.	Grafcet de niveau -02 du MAS.....	58
Figure 40.	Grafcet de niveau -03 du MAS.....	58
Figure 41.	Grafcet de niveau -02 si le moteur chauffe.....	59
Figure 42.	Les zones de la partie commande du GEMMA.....	60
Figure 43.	Les rectangles états du GEMMA.....	63
Figure 44.	Perçage semi-automatique.....	64
Figure 45.	Modes de marches d'un système de parcage semi-automatique.....	65
Figure 46.	Grafcet complet en utilisant les ordres de forçage.....	65
Figure 47.	Structure hiérarchisée selon les modes de marches.....	66