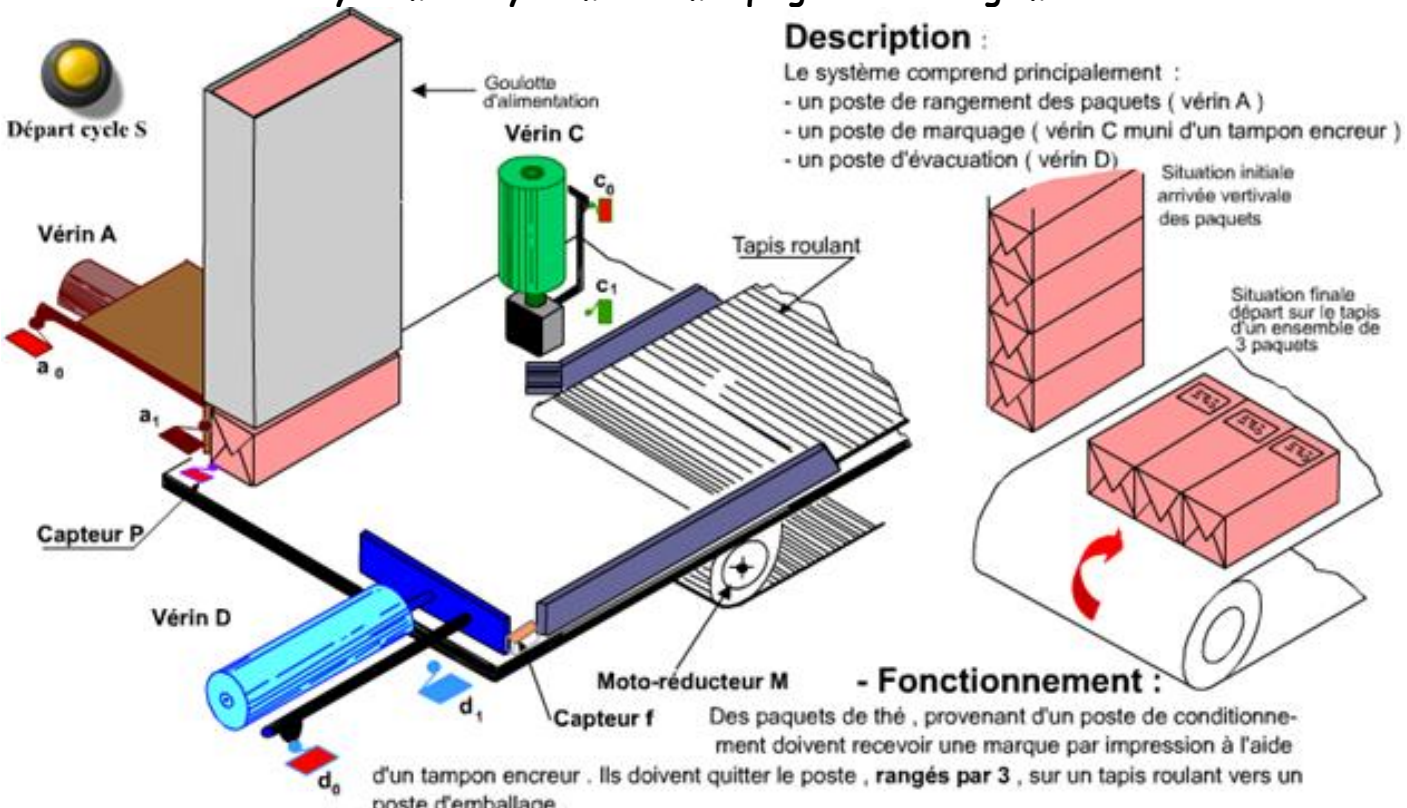
LES AUTOMATES PROGRAMMABLES INDUSTEILS : A.P.I

A – Mise en situation : (voir livre de cours page 66)

B – Rappel : GRAFCET

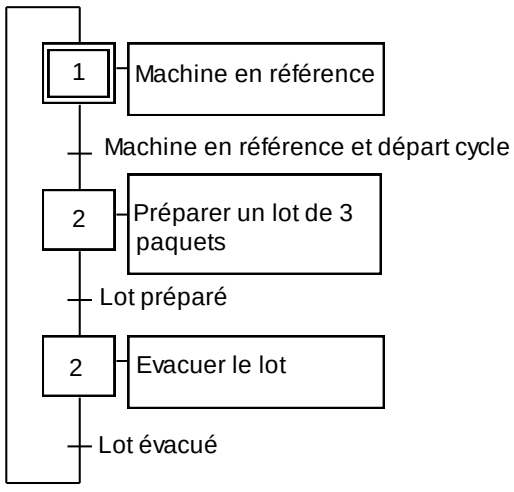
1 – Présentation du système : Système de marquage et de rangement

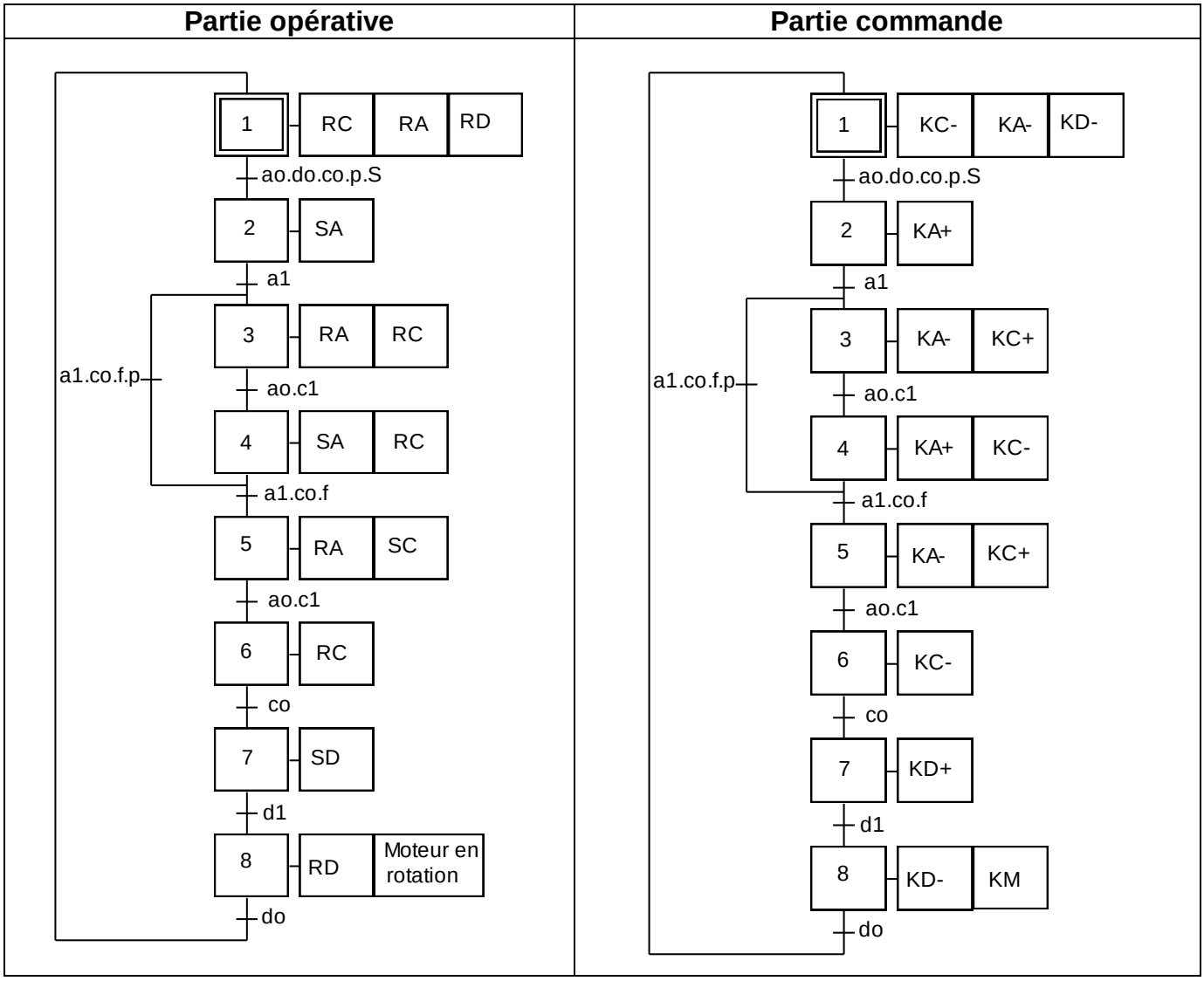


- Une fois , les 3 paquets sont rangés , un capteur f provoque leur évacuation sur le tapis roulant (Les vérins A , C et D sont de type double effet et commandés chacun par un distributeur 5/4/2 à pilotage électrique . On donne pour le vérin KX+ (X = A , B ou C) : KX+ : pilotage de la sortie de X ; KX- : pilotage du retour de X . Le moto-réducteur M est commandé par un discontacteur KM ; les capteurs de position sont de type électrique . P : capteur détectant la présence des paquets dans la goulotte .

2 – Analyse fonctionnelle : Charger le fichier < activitG7> puis le simuler. Décrire le fonctionnement du système en complétant le GRAFCET d'un point de vue :

- Système :





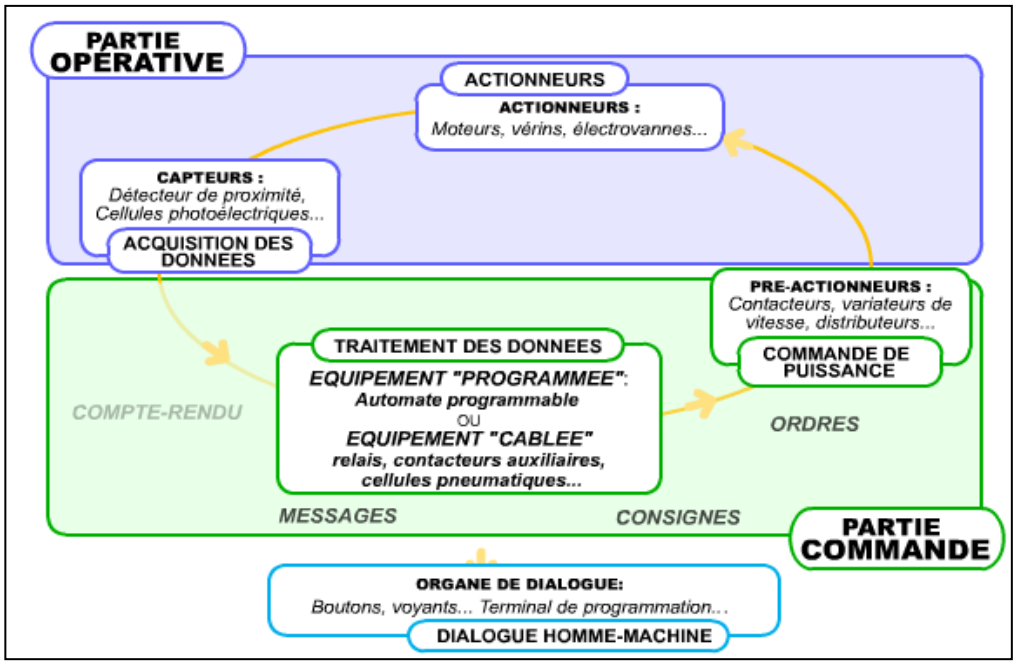
3 – Mise en équation d'une étape d'un GRAFCET : Rappelons qu'une étape s'active lorsque toutes les étapes immédiatement précédentes soient actives et la réceptivité associée à la transition immédiatement précédente soit vraie .Elle se désactive par l'activation de toutes les étapes immédiatement suivantes.

Exemples :

Etapes	**Equations**
1	X1 = (X8.do + m1) $\bar{X}$
3	X3 = (X2.a1 + X4.a1.co.f.p + m2) X$\bar{X}$
4	X4 = (X3.ao.c1 + m4) ($\overline{X5 + X3}$)
Sorties	**Equations**
KC-	KC- = X1 + X4 + X6)
KA-	KA- = X1 + X3 + X5)
KD+	KD+ = X7

4 – Matérialisation d'un GRAFCET : (choix d'une technologie de réalisation)

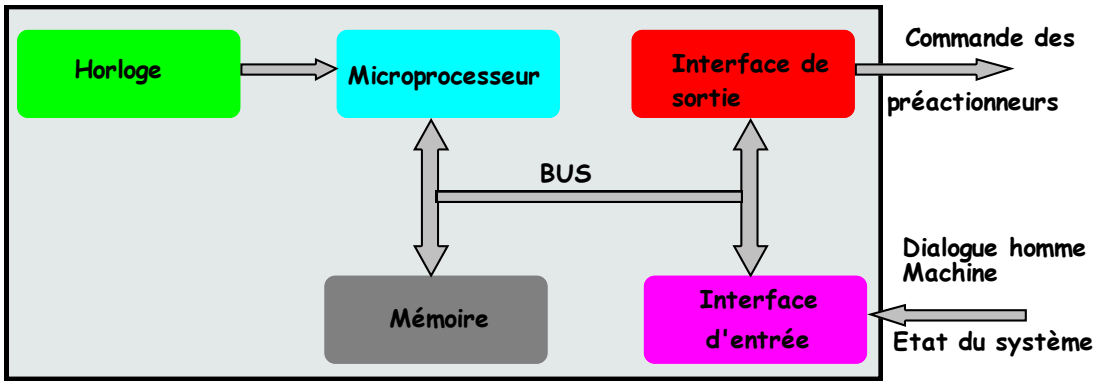
La structure générale d'une installation automatisée est la suivante :



Le traitement des données est géré par une logique **CÂBLÉE ou PROGRAMMÉE**

Logique câblée	**Logique programmée**
Le fonctionnement de l'installation de l'automatisme est définie	
Par câblage (schéma électrique , tableau de connexion …etc.)	Par un programme (instructions)
Avantages	
Technologie d'hier	-Câblage et volume réduits -Erreurs ,modifications , extensions : facile à réaliser
Inconvénients	
-Câblage encombré -Modification du fonctionnement impose une modification de câblage	Technologie d'aujourd'hui

C – Architecture interne d'un API :



L'automate programmable **reçoit** les informations relatives à l'état du système et puis **commande** les pré-actionneurs suivant le programme inscrit dans sa mémoire.
Un API se compose donc de trois grandes parties : **Le processeur** ; **La zone mémoire** ; **Les interfaces Entrées/Sorties**

1- Le microprocesseur : Le **microprocesseur** réalise toutes les fonctions logiques ET, OU, les fonctions de temporisation, de comptage, de calcul... à partir d'un programme contenu dans sa **mémoire**. Il est connecté aux autres éléments (mémoire et interface E/S) par des liaisons **parallèles** appelées '**BUS**' qui véhiculent les informations sous forme binaire..

2- La zone mémoire :

a- La Zone mémoire va permettre :

- De recevoir les informations issues des capteurs d'entrées.
- De recevoir les informations générées par le processeur et destinées à la commande des sorties (valeur des compteurs, des temporisations, …)
- De recevoir et conserver le programme du processus

b -Action possible sur une mémoire :

- ***ECRIRE*** pour modifier le contenu d'un programme
- ***EFFACER*** pour faire disparaître les informations qui ne sont plus nécessaires
- ***LIRE*** pour en lire le contenu d'un programme sans le modifier

c - Technologie des mémoires :

- **RAM** (Random Acces Memory): mémoire vive dans laquelle on peut lire, écrire et effacer (contient le programme)
- **ROM** (Read Only Memory): mémoire morte accessible uniquement en lecture.
- **EPROM** mémoires mortes reprogrammables effacement aux rayons ultra-violets.
- **EEPROM** mémoires mortes reprogrammables effacement électrique

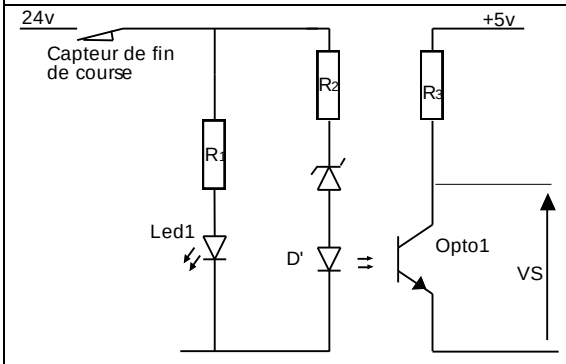
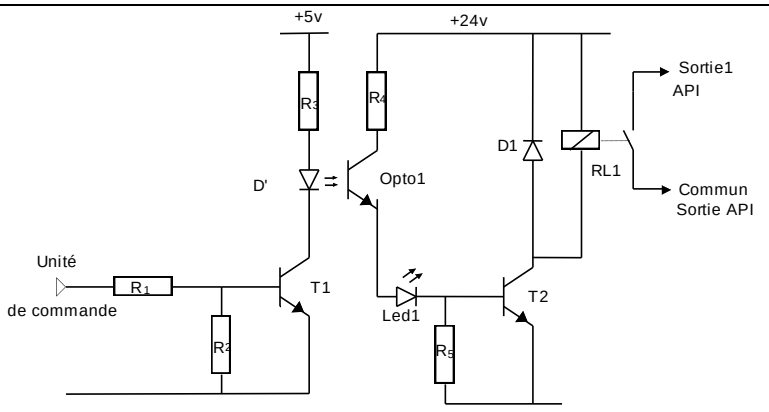
Remarque :

La capacité mémoire se donne en mots de 8 BITS (Binary Digits) ou octets.

Exemple:

Soit une mémoire de 8 Koctets = 8 x 1024 x 8 = 65 536 BITS. Cette mémoire peut contenir 65 536 informations binaires.

3 -Les interfaces d'entrées/sorties :

Interface d'entrée	**Interface de sortie**
	
Les entrées reçoivent des informations en provenance des éléments de détection (capteurs) et du pupitre opérateur (BP).	Les sorties transmettent des informations aux pré-actionneurs (relais, électrovannes …) et aux éléments de signalisation (voyants) du pupitre

D – Programmation d'un API :

Elle peut s'effectuer de trois manières différentes :

- Sur l'A.P.I. lui-même à l'aide de touches.
- Avec une console de programmation reliée par un câble spécifique à l'A.P.I.
- Avec un PC et un logiciel approprié.

I – Langages de programmation :

On cite les cinq langages de programmation couramment utilisées dans l'industrie :

• **IL(I**nstruction **L**ist → liste d'instructions) : Un programme écrit en langage liste d'instructions se compose d'une suite d'instructions exécutées séquentiellement par l'automate. Chaque instruction est composée d'un code instruction et d'un opérande

• **ST(S**tructured **T**ext →Texte structure) : permet la programmation de tout type d'algorithme plus ou moins complexe.

• **LD(L**eader **D**iagram → schéma à contacts) : Un programme écrit en langage à contacts se compose d'une suite de réseaux de contacts composés d'un ensemble d'éléments graphiques disposées sur grille organisée en lignes et colonnes.

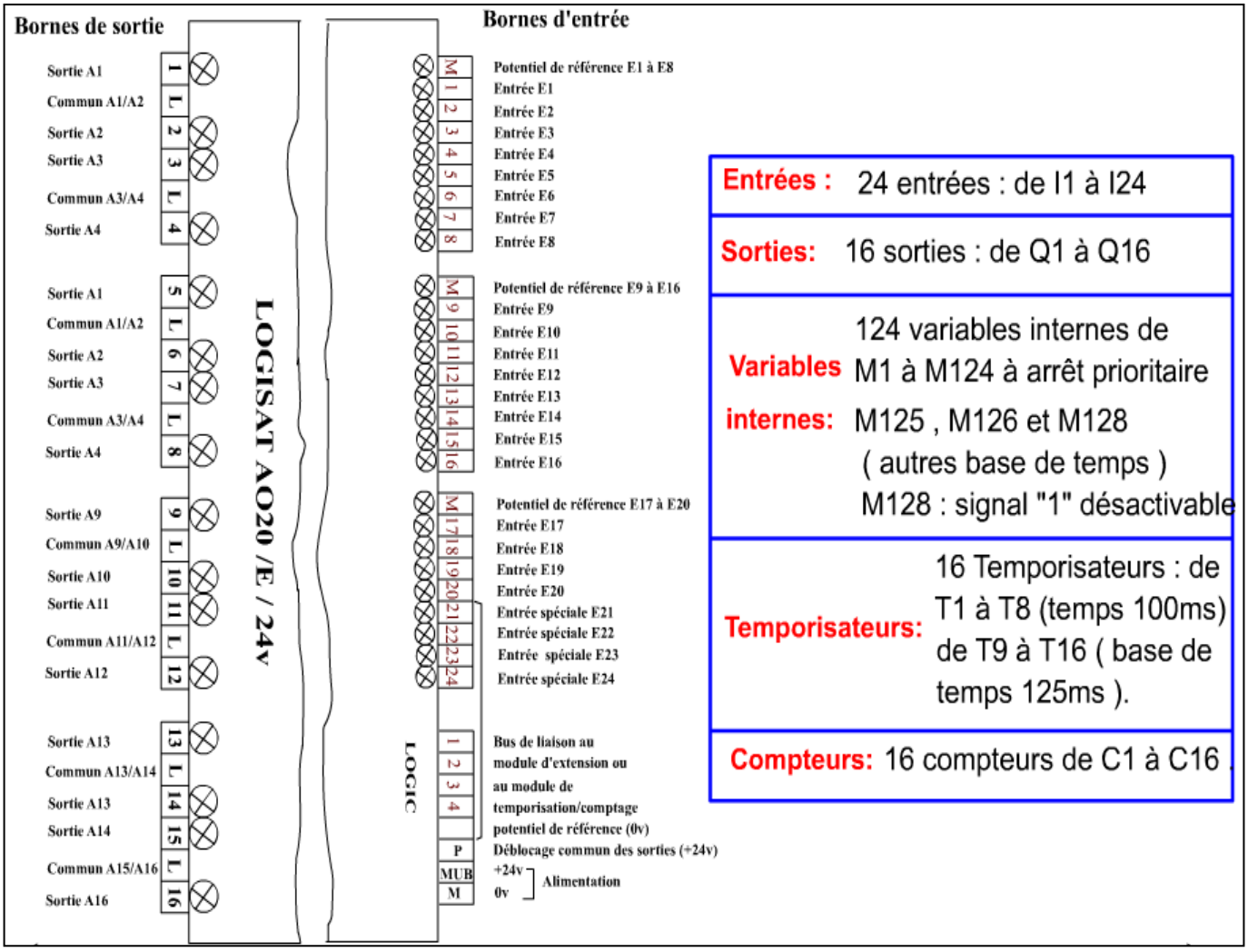
• **SFC(S**équentiel **F**unction **C**hart → langage G7) : permet la programmation de tous les procédés séquentiels .

• **FBD(F**unction **B**lock **D**iagram → Schéma par Bloc) : permet de programmer graphiquement à l'aide des blocs, représentant des variables , des opérateurs ou des fonctions .

NB : Chaque type d'API a ses propres instructions (voir dossiers techniques pages 88---96 livre de cours)

II – Programmation d'un grafcet en utilisant un API de type AEG020:

1 – L'automate AEG020 :



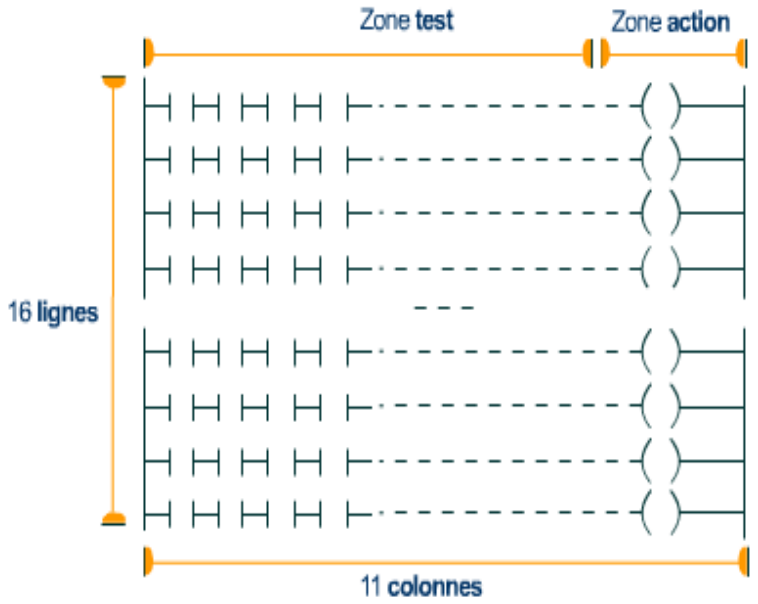
2 – Liste d'instructions (IL):

Type d'opération	Opéra-teur	Action	Opérandes utilisables
Opérations logiques	**A**	Opération logique ET , signal positif	**Ixx , Qxx , Mxxx**
	AN	Opération logique ET , signal négatif	**Txx , Cxx**
	O	Opération logique OU , signal positif	**Ixx , Qxx , Mxxx**
	ON	Opération logique OU , signal négatif	**Txx , Cxx**
	A(	Opération logique ET , parenthèse ouverte	**Ixx , Qxx , Mxxx,Txx , Cxx**
	O(	Opération logique OU , parenthèse ouverte	**Ixx , Qxx , Mxxx**
	)	parenthèse fermée positive	**Txx , Cxx**
	)N	parenthèse fermée négative	
Opérations de sorties	**=**	Sortie positive	**Qxx , Mxxx**
	=N	Sortie négative	**Qxx , Mxxx**
	SL	Activation mémoire	**Qxx , Mxxx**
	RL	Désactivation mémoire	**Qxx , Mxxx**
Opérations de comptage / temporisation	**= T**	Entrée temporisation (sortie tempo .)	**Ixx , Qxx**
	= Z	Transfert consigne compteur (effacement)	
	= P	Entrée compteur (C .. sortie compteur	
Opérations d'organisation de programme	**JI**	Saut si "1"(conditionnel positif)	
	LS	Chargement immédiat(en mémoire de signaux)	
	NO	Sans effet , opération nulle	
	PE	Fin de programme	

3 – Eléments graphiques du langage à contacts (LD) :

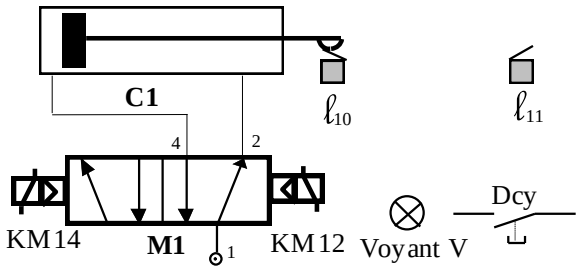
Eléments graphiques du langage à contacts LD	Structure d'un réseau de contacts

Désignation		Symboles
Eléments de test	Contact à fermeture	⊣ ⊢
	Contact à ouverture	⊣/⊢
	Contact détection de changement d'état	⊣P⊢ ⊣N⊢
Eléments de liaison	Connexion horizontale	——
	Connexion verticale	|
Eléments d'action	Bobine directe	—( )—
	Bobine inverse	—(/)—
	Bobine d'enclenchement	—(S)—
	Bobine de déclenchement	—(R)—
	Sauf conditionnel à autre réseau (JUMP)	->>%Li
	Bobine dièse	—(#)—
	Bobine appel à un sous-grogramme (CALL)	—(C)—
	Retour de sous-grogramme	<return>
	Arrêt programme	<halt>

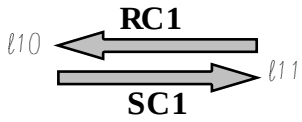


4 –Applications :

a -GRAFCET à séquence unique (Cycle pendulaire)

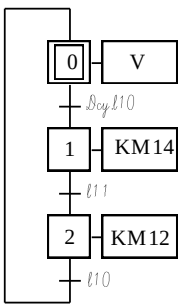


L'action sur un bouton départ cycle **Dcy** déclenche le cycle suivant :

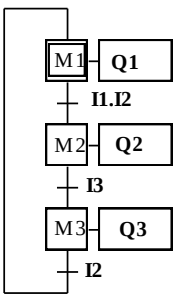


Le voyant **V** signale le repos du cycle (tige rentrante).

GRAFCET PC	Tableau d'affectations	GRAFCET codé automate AEG020



Entrées	Sorties
Dcy → I1	V → Q1
ℓ10 → I2	KM14 → Q2
ℓ11 → I3	KM12 → Q3



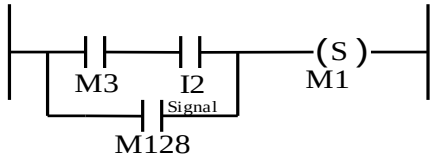
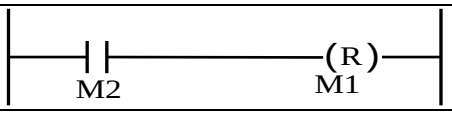
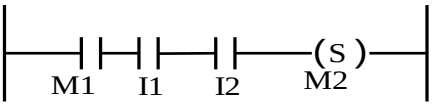
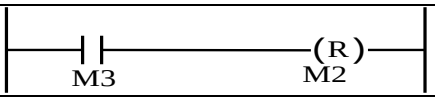
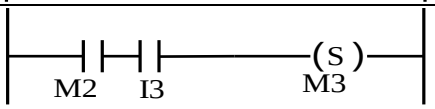
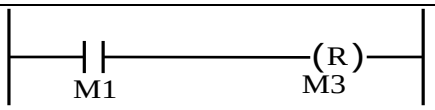
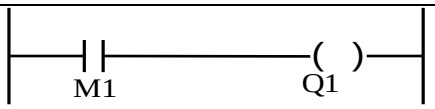
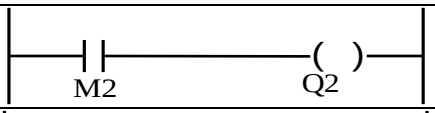
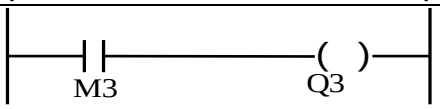
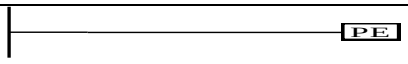
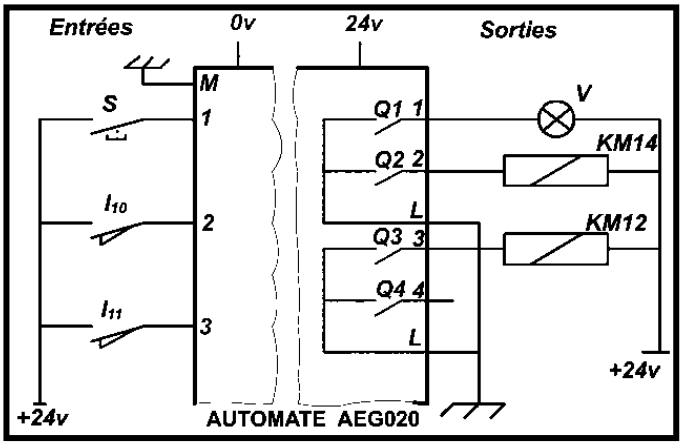
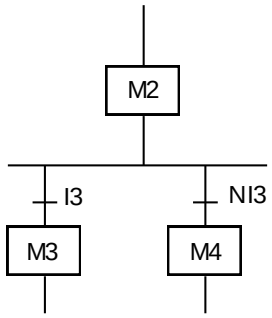
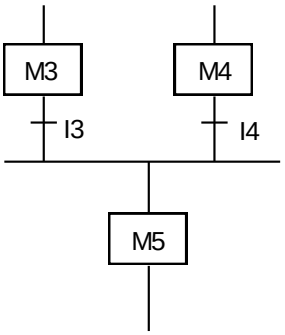
Langage IL			Langage LD
ADR	**INSTRUC**	**Commentaire**	
1 :	A M3	si l'étape 3 est active	
2 :	A I 2	et réceptivité 3 vraie	
3 :	O M128	variable interne mise à 1 à la mise sous tension	
4 :	SL M1	Activation de l'étape 0	
5 : 6 :	A M2 RL M1	si l'étape 1 est active alors désactiver l'étape 0	
7 : 8 : 9 : 10 :	A M1 A I1 A I2 SL M2	activation de l'étape 1	
11 : 12 :	A M3 RL M2	désactivation de M2	
13 : 14 : 15 :	A M2 A I3 SL M3	Activation de l'étape 2	
16 : 17 :	A M1 RL M3	désactivation de l'étape 2	
18 :	RL M128	mise à 0 de la variable interne (boucle)	
19 : 20 :	A M1 = Q1	sortie 1	
21 : 22 :	A M2 = Q2	sortie 2	
23 : 24 :	A M3 = Q3	sortie 3	
25 :	PE	fin de programme	

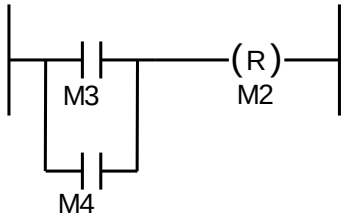
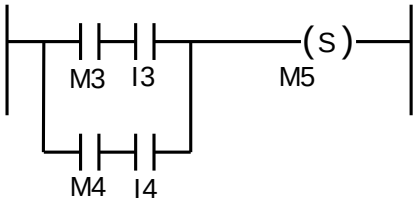
Schéma de câblage :



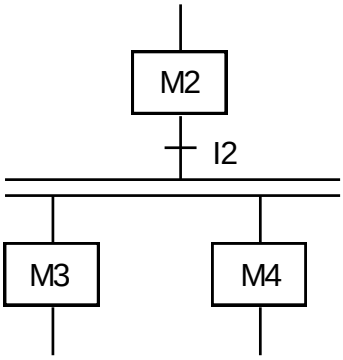
b-Cas de divergence /convergence en OU :

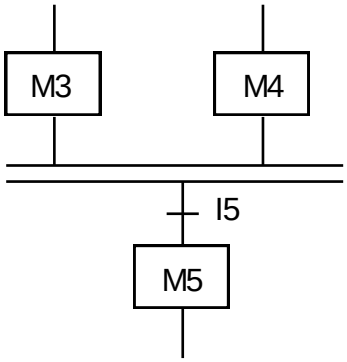


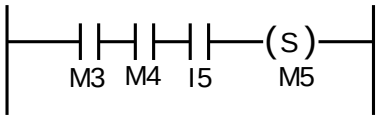


Langage IL	Langage LD	Langage IL	Langage LD
Désactivation de M2: AM3 OM4 RLM2		Activation de M5: AM3 AI3 O(AM4 AI4) SLM5	

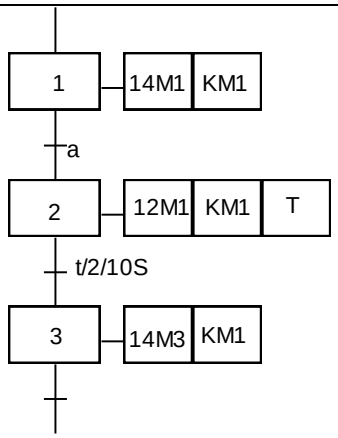
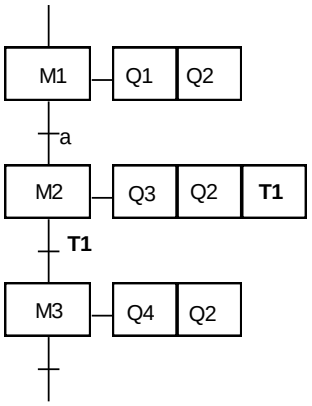
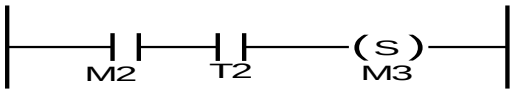
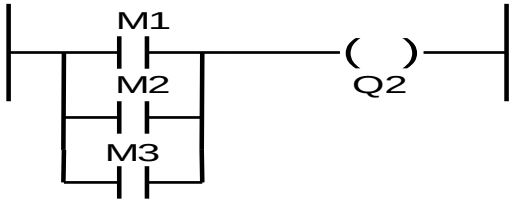
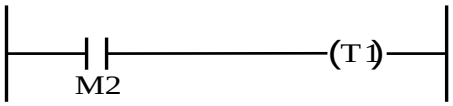
c-Cas de divergence /convergence en ET :



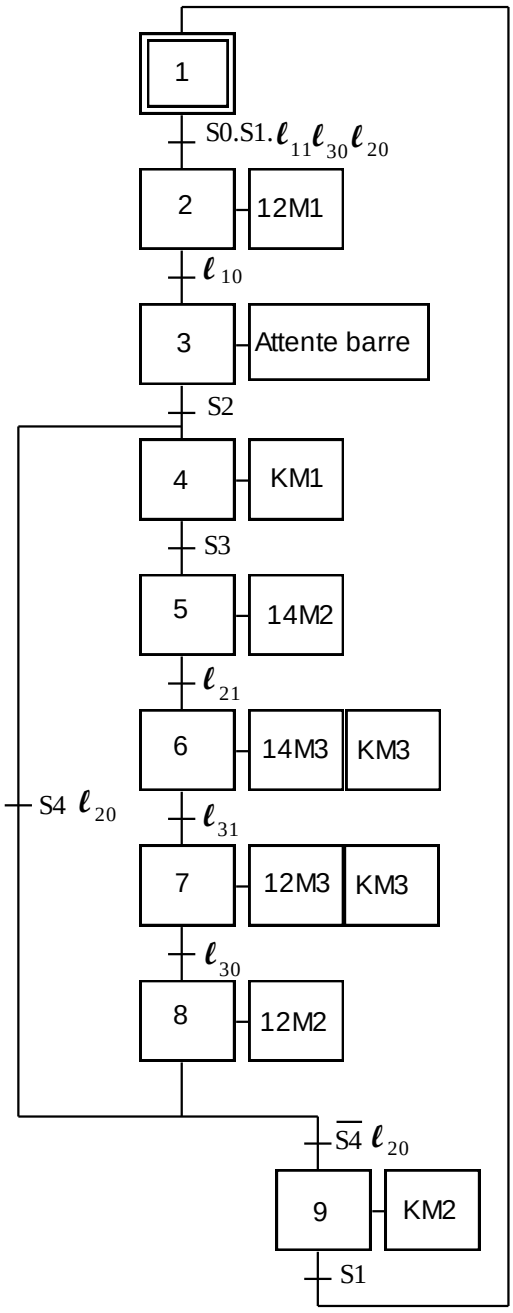
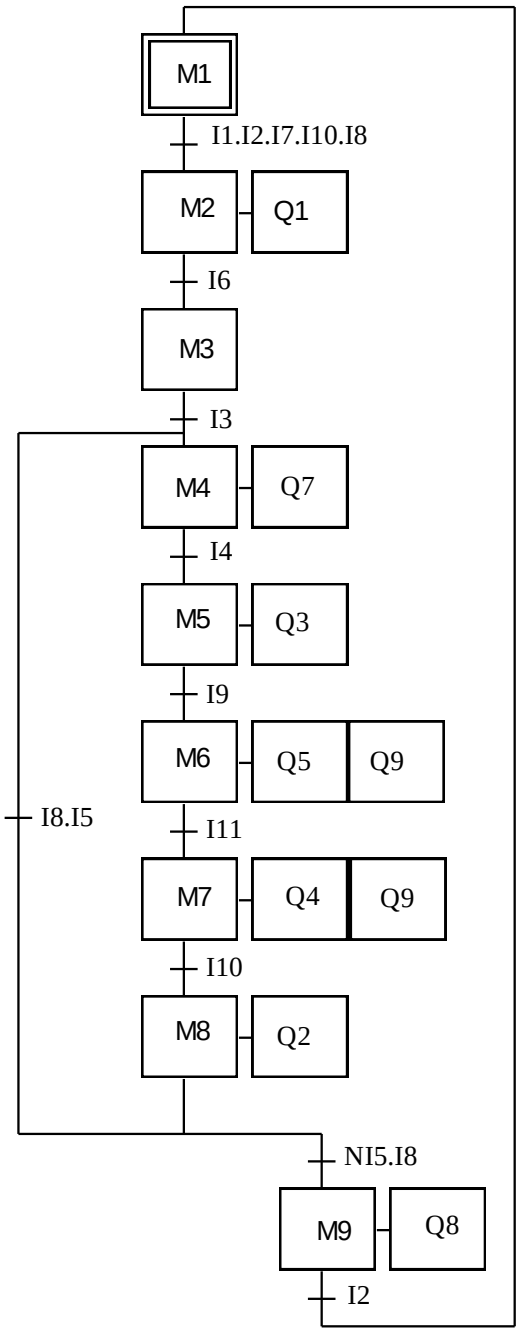


Langage IL	Langage LD	Langage IL	Langage LD
Désactivation de M2: AM3 AM4 RLM2		Activation de M5: AM3 AM4 AI5 SLM5	

d-Cas d'une temporisation et d'une étape à plusieurs sorties :

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Activation de M3 : AM2
AT2
SLM3</td>
<td> 
</td>
</tr>
<tr>
<td>Sortie Q2 : AM1
OM2
OM3
=Q2</td>
<td> 
</td>
</tr>
<tr>
<td>Sortie T1 : AM2
= T1 (100)</td>
<td> 
</td>
</tr>
</table>

Exercice N°4 : Unité de tronçonnage automatique

- GRAFCET PC	- GRAFCET codé automate AEG020
	

Equations d'activation (A) et de désactivation (D) des étapes 1 , 4 et 8 :

Etape	Activation	Désactivation
1	$A_0 = X_9.S1$	$D_0 = X_2$
4	$A_4 = X_8.S4.\ell_{20} + X_3.S2$	$D_4 = X_5$
8	$A_8 = X_7.\ell_{30}$	$D_8 = X_9 + X_4$

- Programme en langage IL et LD dédié à l'automate AEG020 /

Etape	Adr	Ins
1	1:	AM9
	2:	AI2
	3:	OM128
	4:	SLM1
	5:	AM2
	6:	RLM1
2	7:	AM1
	8:	AI1
	9:	AI2
	10:	AI7
	11:	AI10
	12:	AI8
	13:	SLM2
	14:	AM3
	15:	RLM2
3	16:	AM2
	17:	AI6
	18:	SLM3
	19:	AM4
	20:	RLM3

Etape	Adr	Ins
4	21:	AM3
	22:	AI3
	23:	O(
	24:	AM8
	25:	AI8
	26:	AI5
	27:	)
	28:	SLM4
	29:	AM5
	30:	RLM4
5	31:	AM4
	32:	AI4
	33:	SLM5
	34:	AM6
	35:	RLM5
6	36:	AM5
	37:	AI9
	38:	SLM6
	39:	AM7
	40:	RLM6

Etape	Adr	Ins
7	41:	AM6
	42:	AI11
	43:	SLM7
	44:	AM8
	45:	RLM7
8	46:	AM7
	47:	AI10
	48:	SLM8
	49:	AM9
	50:	OM4
	51:	RLM8
9	52:	AM8
	53:	ANI5
	54:	AI8
	55:	SLM9
	56:	AM1
	57:	RLM9
	58:	RLM128
12M1	59:	AM2
	60:	=Q1

Sorties	Adr	Ins
KM1	61:	AM4
	62:	=Q7
14M2	63:	AM5
	64:	=Q3
14M3	65:	AM6
	66:	=Q5
KM3	67:	AM6
	68:	OM7
	69:	=Q9
12M3	70:	AM7
	71:	=Q4
12M2	72:	AM8
	73:	=Q2
KM2	74:	AM9
	75:	=Q8
FIN	76:	PE

Activation étapes	Désactivation étapes	Sorties