Chapitre V: Automates Programmables Industriels (API)

Objectif du chapitre

Découvrir le fonctionnement et la mise en œuvre des Automates Programmables Industriels (API).

Contenu du chapitre

V.1 Définition d'un API

La définition est donnée par la norme NFC 63-850 :

« Appareil électronique qui comporte une mémoire programmable par un utilisateur automaticien (et non informaticien) à l'aide d'un langage adapté, pour le stockage interne des instructions composant les fonctions d'automatisme comme par exemple :

- Logique séquentielle et combinatoire ;
- Temporisation, comptage, décomptage, comparaison ;
- Calcul arithmétique ;
- Réglage, asservissement, régulation, etc, pour commander, mesurer et contrôler au moyen d'entrées et de sorties (logiques, numériques ou analogiques) différentes sortes de machines ou de processus, en environnement industriel. »

V.2 Architecture interne d'un API

Un AP est constitué essentiellement de 5 modules :

L'unité centrale-Le module d'entrées-Le module de sorties-Le module d'alimentation-

Le module de communication

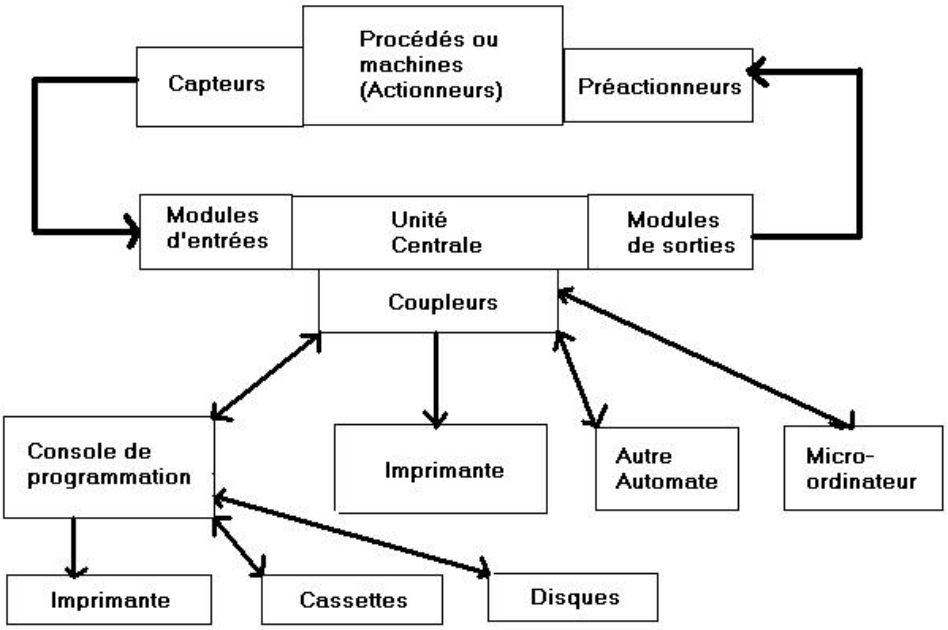


Figure V.1 : L'automate programmable et ses auxiliaires.

V.2.1 L'unité centrale

Représente le cœur de la machine, et comprend le/les processeur(s) et la mémoire.

V-2-1-1 Le processeur

Il est composé :

- d'une Unité Logique (UL) qui traite les opérations logiques ET, OU et Négation.
- d'une Unité Arithmétique et Logique (UAL) qui traite les opérations de temporisation, de comptage et de calcul.
- d'un Accumulateur qui est un registre de travail dans lequel se range une donnée ou un résultat.
- d'un Registre d'Instruction qui contient, durant le temps de traitement, l'instruction à exécuter.
- d'un Décodeur d'Instruction qui décode l'instruction à exécuter en y associant les microprogrammes de traitement.
- d'un Compteur Programme ou Compteur Ordinal qui contient l'adresse de la prochaine instruction à exécuter et gère ainsi la chronologie de l'exécution des instructions du programme.

Figure V.2 : Le processeur.

V-2-1-2 La mémoire

La mémoire centrale est l'élément fonctionnel qui peut recevoir, conserver et restituer. Elle est découpée en zones où l'on trouve :

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

Ces mémoires peuvent être :

- Durant la phase d'étude et de mise au point du programme :

- des mémoires vives RAM (Random Access Memory) volatiles
- des mémoires EAROM (Electrically Alterable Read Only Memory) non volatiles et effaçables partiellement par voie électrique.

- Durant la phase d'exploitation:

- des mémoires vives RAM qui imposent un dispositif de sauvegarde par batterie rechargeable pour éviter la volatilité de leur contenu en cas de coupure de courant
- des mémoires mortes ROM à lecture seulement ou PROM programmables à lecture seulement.
- des mémoires re-programmables EPROM (Erasable PROM) effaçables par un rayonnement ultra-violet et EEPROM (Electric Erasable PROM effaçables électriquement.

V.2.2 Le module d'entrées

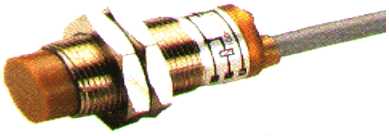
Les **cartes d'entrées tout ou rien** permettent de raccorder à l'automate les différents capteurs logiques tels que :

- boutons poussoirs
- pressostats

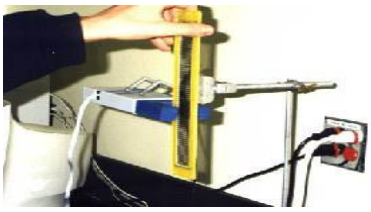
Entièrement programmable sur site par touches numériques sécurisées

- thermostats

- fins de course
- capteurs de proximité inductifs ou capacitifs

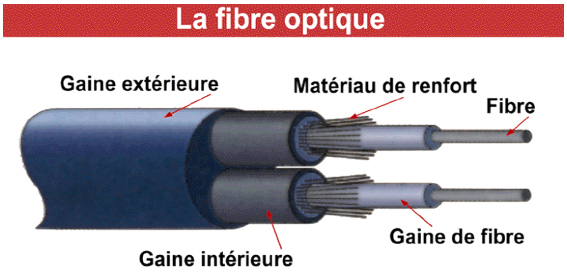


- Capteurs photo-électriques



Acquisition des données avec l'interface portable munie d'un capteur photoélectrique.

- Fibres optiques



- roues codeuses
- etc.

Elles assurent l'adaptation, l'isolement, le filtrage et la mise en forme des signaux électriques. Une diode électroluminescente située sur la carte donne l'état de chaque entrée. Le nombre d'entrées sur une carte est de : 4, 8, 16, 32.

Les tensions d'entrées sont de : 24, 48, 110, 220 volts en courant continu ou alternatif. Les **cartes d'entrées analogiques** permettent de gérer des grandeurs analogiques en faisant varier un code numérique au sein du module. Il existe 3 types d'entrées analogiques :

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

Sur le marché, il existe des modules à 2, 4, 8 voies d'entrées. Les entrées analogiques disposent d'un seul convertisseur analogique /numérique, elles sont scrutées les unes à la suite des autres par un multiplexeur à relais.

V-2-3 Le module de sorties

Les **cartes de sortie tout ou rien** permettent de raccorder à l'automate les différents pré-actionneurs tels que :

- Vannes

Vanne Eléctomagnétique

Pour distribution d'eau à une température comprise entre 1° et 60°C, sous 10 bars maxi, On utilise une **vanne électromagnétique** de

- Débit : 36 litres/minute, sous 6 bars
- Courant d'appel : 26 VA. Courant de maintien : 14 VA
- Raccordement par taraudage 15 x 21 (1/2 " gaz cylindrique)
- Branchement électrique par connecteur
- Alimentation : 2 modèles : 12 V / 50 Hz ou 24 V / 50 Hz

- Contacteurs

- Voyants

Voyant Pneumatique

- Electrovannes

Vanne permettant d'ouvrir et de fermer un circuit d'eau par une commande électrique 24V.

- Relais de puissance

Pour le pilotage de vos éléments chauffants résistifs en monophasé ou triphasé

- Afficheurs

- etc.…

Les tensions de sorties usuelles sont de 5 volts en continu ou de 24, 48, 110, 220 volts en continu ou en alternatif.

Les courants vont de quelques milliampères à quelques ampères.

Ces cartes possèdent soit des relais, soit des triacs, soit des transistors.

L'état de chaque sortie est visualisé par une diode électroluminescente.

Les **cartes de sortie analogiques** permettent de gérer des grandeurs analogiques en faisant varier un code numérique au sein du module. Il existe deux grands types de cartes de sorties :

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

Ces modules assurent la conversion numérique/analogique.

L'intensité ou la tension est proportionnelle à la valeur numérique.

Avec les résolutions 8 bits il y a 256 valeurs numériques possibles, tandis qu'avec les résolutions de 12 bits il y en a 4096.

Les sorties analogiques peuvent posséder un convertisseur par voie. Le nombre de voies sur ces cartes est de 2 ou 4.

V-2-4 Le module d'alimentation

Composé de blocs qui permettent de fournir à l'automate l'énergie nécessaire à son fonctionnement. A partir d'une alimentation en 220 volts alternatif, ces blocs délivrent des sources de tension dont

l'automate a besoin : 24V, 12V ou 5V en continu. En règle générale, un voyant positionné sur la façade indique la mise sous tension de l'automate.

V.2.5 Le module de communication

Comprend les consoles, les boîtiers de tests et les unités de dialogue en ligne.

V.2.5.1 Les consoles

Il existe deux types de consoles. L'une permet le paramétrage et les relevés d'informations (modification des valeurs, et visualisation), l'autre permet en plus la programmation, le réglage et l'exploitation. Cette dernière dans la phase de programmation effectue :

- L'écriture
- La modification
- L'effacement
- Le transfert d'un programme dans la mémoire de l'automate ou dans une mémoire REPROM.

Dans la phase de réglage et d'exploitation elle permet :

- D'exécuter le programme pas à pas
- De le visualiser
- De forcer ou de modifier des données telles que les entrées, les sorties, les bits internes, les registres de temporisation, les compteurs, etc.
- La sortie sur une imprimante du programme si un port de sortie existe.

La console peut également afficher le résultat de l'autotest comprenant l'état des modules d'entrées et de sorties, l'état de la mémoire, de la batterie, etc. Les consoles sont équipées (pour la plupart), d'un écran à cristaux liquides. Certaines consoles ne peuvent être utilisées que connectées à un automate (l'automate fournit l'alimentation à la console), d'autres peuvent fonctionner de manière autonome grâce à la mémoire interne et à leur alimentation.

V-2-5-2 Les boîtiers de tests

Destinées aux personnels d'entretien, ils permettent de visualiser le programme ou les valeurs des paramètres. Par exemple :

- Affichage de la ligne de programme à contrôler
- Visualisation de l'instruction (code opératoire et adresse de l'opérande)
- Visualisation de l'état des entrées
- Visualisation de l'état des sorties.

V.2.5.3 Les unités de dialogue en ligne

Elles sont destinées aux personnels spécialistes du procédé et non de l'automate programmable, et leur permet d'agir sur certains paramètres :

- Modification des constantes, compteurs temporisations
- Forçage des entrées/sorties
- Exécution de parties de programme
- Chargement de programmes en mémoire à partir de cassettes.

Ces boîtiers se présentent sous la forme enfichable dans l'unité centrale ou séparés de celle-ci. Il comporte des touches de fonctions, numériques, une visualisation, un dispositif de sécurité, l'ensemble est piloté par micro-processeur.

Objectif du chapitre

Avoir les habilités de programmer des API avec le logiciel GRACET

Contenu du chapitre

Le GRAFCET (GRAphe Fonctionnel de Commande Etape Transition) est un outil méthode, descriptif du cahier des charges de tout système séquentiel. Il permet de décrire les comportements attendus de l'automatisme au niveau du traitement des informations délivrées par la partie opérative et des ordres transmis à cette même partie.

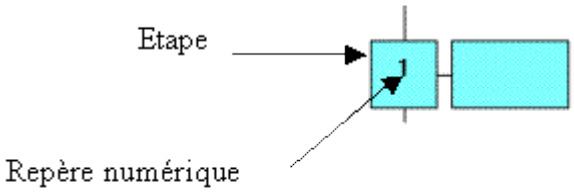
VI.2- Eléments de Base

Le GRAFCET se compose des éléments suivants :

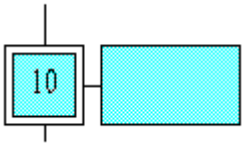
- étapes auxquelles sont associées des actions.
- transitions auxquelles sont associées des réceptivités
- liaisons orientées reliant les étapes aux transitions et les transitions aux étapes.

VI-2-1- Etape :

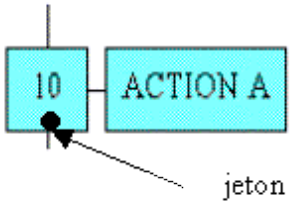
Une étape est symbolisée par un carré repéré numériquement



L'étape initiale est représentée par un double carré.

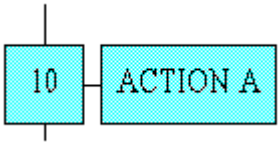


Une étape peut prendre deux valeurs soit active soit inactive. L'activité d'une étape est représentée par un jeton.



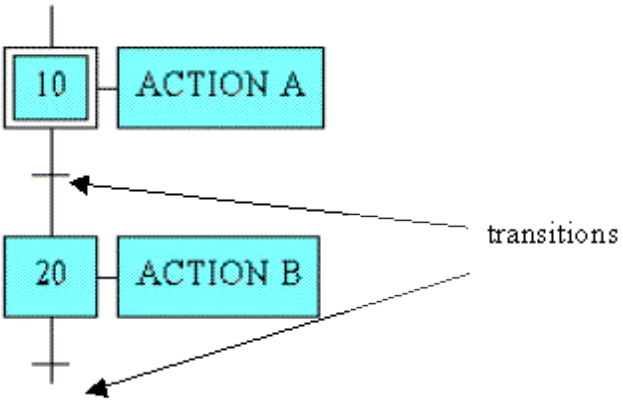
VI.2.2- Action associée à une étape :

Une action est symbolisée par un rectangle relié au symbole de l'étape associée.



IV.2.3 Transition :

Une transition est représentée par une barre perpendiculaire à la liaison orientée. Une transition indique la possibilité d'évolution entre deux étapes successives.



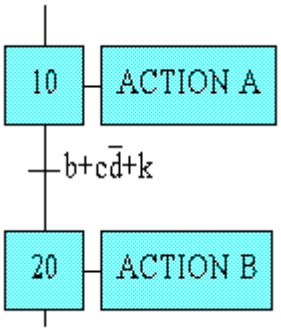
Une transition est soit validée soit non validée. Elle est dite validée lorsque toutes les étapes immédiatement précédentes reliées à cette transition sont actives.

IV.2.4 Réceptivité associée à la transition :

A chaque transition est associée une proposition logique appelée réceptivité qui peut être soit vraie soit fausse. La réceptivité peut s'écrire sous plusieurs formes :

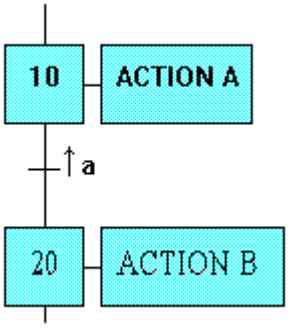
- Réceptivités sous forme de proposition logique ou sous forme d'une fonction combinatoire.

Exemple 1 : réceptivité sous forme de proposition logique



- Réceptivité sous forme d'activation ou de désactivation d'une variable d'entrée du système.

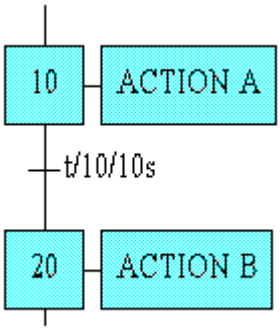
Exemple 2 : réceptivité sous forme d'activation



Où a↑ est le front montant de la variable a.

- Réceptivité sous forme de temporisation

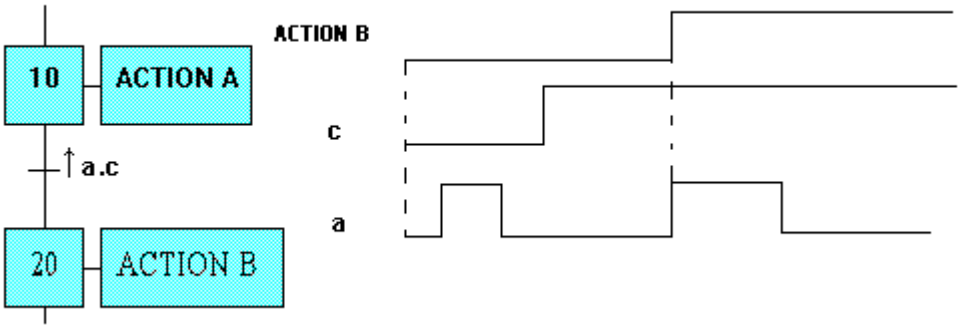
Exemple 3 : réceptivité sous forme de temporisation



t/10/10s exprime que 10s sont écoulées depuis la dernière activation de l'action de l'étape 10.

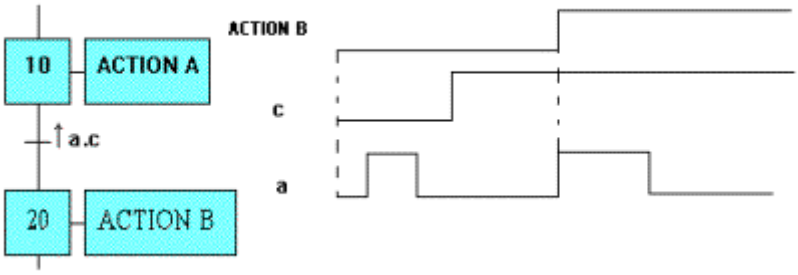
- Réceptivité sous forme de condition ET logique entre une variable logique et un événement.

Exemple 4 : réceptivité sous forme d'un ET logique entre une variable logique et un événement.



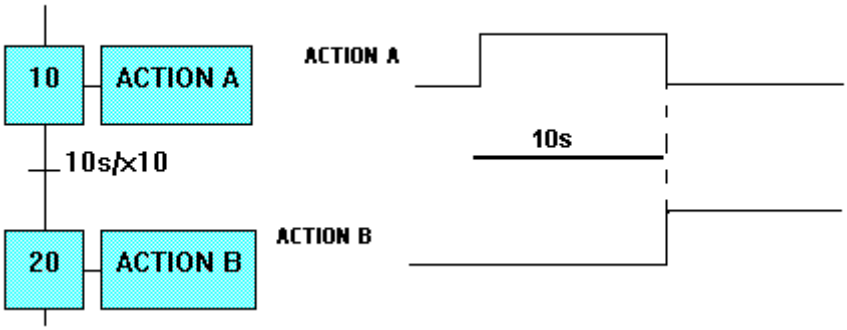
- Sous forme d'une expression temporelle : l'utilisation de la variable temps est possible à travers délai et temps-enveloppe.

Exemple 5 : réceptivité sous forme d'expression temporelle



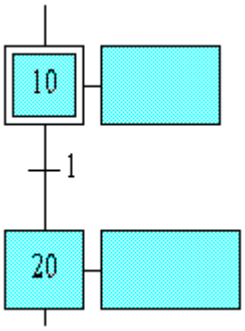
t1, t2 : retards. a : événement t1/a/t2 est vraie après t1 compté à partir de a. t1/a/t2 est fausse après t2 compté à partir de a

Exemple 6 : réceptivité sous forme d'expression temporelle



X10 est une variable booléenne représentant l'état d'une étape. Après 10s compté à partir de l'action A, la transition devient vraie.

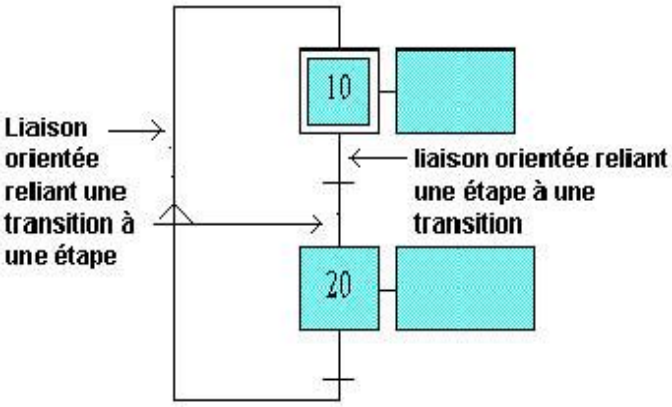
- Réceptivité toujours vraie.



Exemple 7 : réceptivité toujours vraie

IV.2.5 Liaisons orientées :

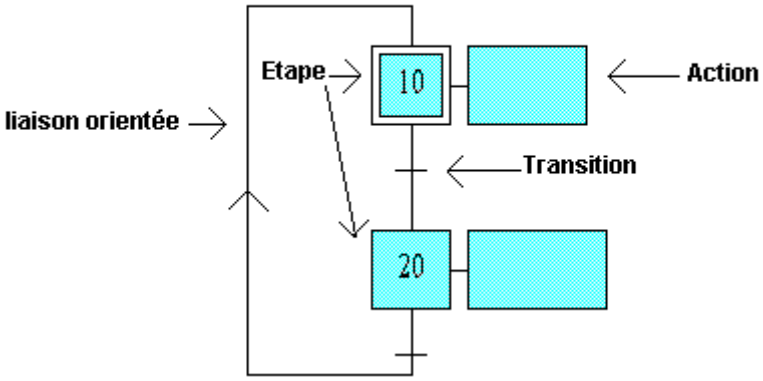
Les liaisons orientés relie une étape à une transition ou inversement.



VI.2.6 Récapitulation

Les trois éléments de base d'un GRAFCET sont :

- L'étape et ses actions,
- La transition et sa réceptivité,
- Les liaisons orientées.



VI.3 Règles d'évolution

Aux règles d'écriture s'ajoutent les **règles d'évolution** afin de préciser les conditions pour lesquelles les étapes sont actives ou inactives.

VI.3.1 Règle.1: Initialisation

Définition : R1: INITIALISATION

La situation initiale caractérise le comportement initial de la partie commande vis - vis de la partie opérative et correspond à l'étape active au début du fonctionnement.
Elle traduit **généralement** un **comportement de repos**
Le symbole est le **double carré:**

IV.3.2 Règle 2: Franchissement d'une transition

Définition : R2: FRANCHISSEMENT d'une transition

Le franchissement d'une transition s'effectue si:

- l'étape précédente est active

- la réceptivité associée est vraie

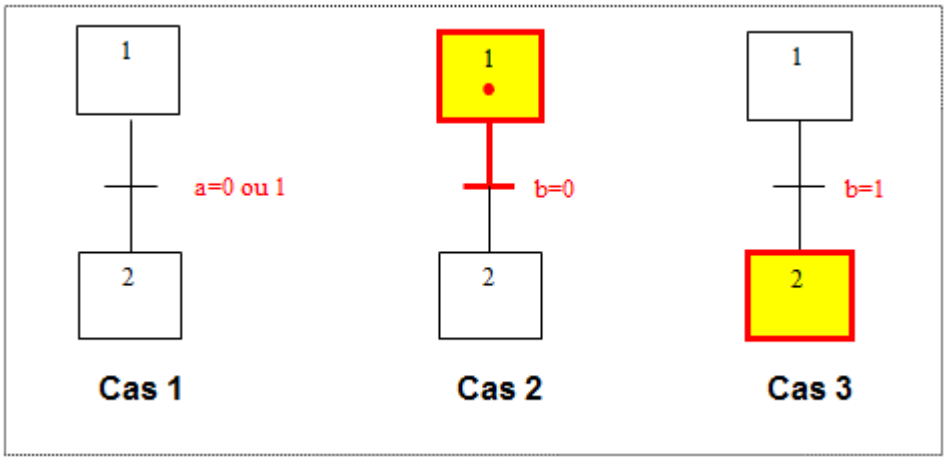
Lorsque ces **deux conditions sont réunies**, la transition devient **franchissable** et est obligatoirement franchie.

IV.3.3 Règle 3: Évolution des étapes actives

Définition : R3: EVOLUTION des étapes actives

Le franchissement d'une transition entraîne ***simultanément l'activation*** de toutes les étapes immédiatement suivantes et la ***désactivation*** de toutes les étapes immédiatement précédentes.

Exemple :



Cas 1: La transition 1-2 est non validée, l'étape 2 étant inactive.

Cas 2: L'étape 1 étant active, la transition 1-2 est validée mais ne peut être franchie car la réceptivité n'est pas vraie: b=0.

Cas 3: La transition 1-2 est franchie car la réceptivité est vraie :b=1. Dans ce cas l'étape 2 est activée et l'étape 1 est désactivée.

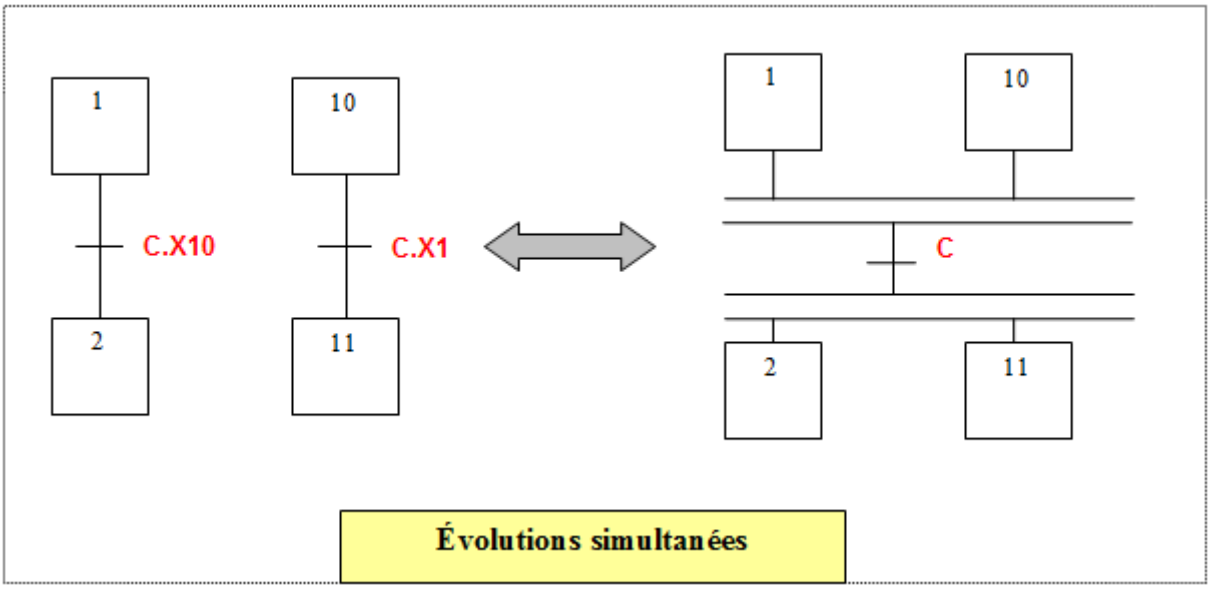
IV.3.4 Règle 4: Évolutions simultanées

Définition : R4: ÉVOLUTIONS SIMULTANEES

Plusieurs transitions simultanément franchissables sont simultanément franchies

Cette règle de franchissement **permet** notamment de **décomposer un grafcet** en plusieurs diagrammes indépendants.

Exemple :



Remarque :

X1 :**Variable Booléenne** correspondant à l'**étape 1** :

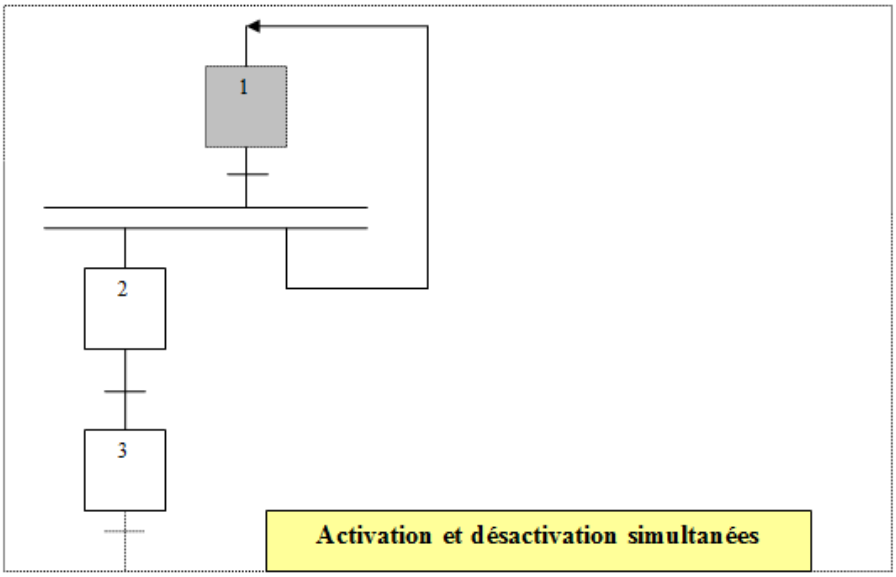
- Si l'étape 1 est **active** X1= **1**
- Si l'étape 1 est **inactive** X1=**0**

VI.3.5 Règle 5 : Activation et désactivation simultanées

Définition : R5 : ACTIVATION et DESACTIVATION simultanées

Si au cours du fonctionnement de l'automatisme une même étape doit être simultanément activée et désactivée, elle reste activée.

Exemple :



VI.4 Mise en équation du grafcet

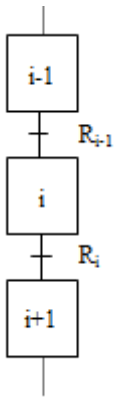
Soit le grafcet simple suivant :

A chaque étape i est associée une variable **Xi** :

- **Xi=1** si l'étape i est **active**
- **Xi=0** si l'étape i est **inactive**

La réceptivité Ri a pour valeur :

- **Ri=0** si la réceptivité est **fausse**
- **Ri=1** si la réceptivité est **vraie**



Le **but** est de déterminer les variables qui interviennent dans l'activité de l'étape i : **Xi=f(?)**

D'après ***la règle 2*** du grafcet, la **C**ondition d'**A**ctivation de l'étape i donne :

$$\mathbf{CAX_i = X_{i-1}\ R_{i-1}}$$

- D'après ***la règle 3*** du grafcet, , la **C**ondition de **D**ésactivation de l'étape i donne :

$$\mathbf{CDXi = X_i\ R_i = X_{i+1}}$$

- Si la **CA** et la **CD** de l'étape i sont fausses, l'étape i reste dans son état (effet mémoire). L'état de **Xi** à l'instant **t + δt** dépend de l'état précédent de **Xi** à l'instant **t.**

On peut alors écrire la **table de vérité de l'étape i** : X_i

Xi(t)	CAXi	CDXi	Xi(t+δt)	Remarque
0	0	0	0	L'étape reste inactive (effet mémoire)
0	0	1	0	L'étape reste inactive
0	1	0	1	Activation de l'étape
0	1	1	1	Activation ET désactivation = Activation
1	0	0	1	L'étape reste active (effet mémoire)
1	0	1	0	Désactivation de l'étape
1	1	0	1	L'étape reste active
1	1	1	1	Activation ET désactivation = Activation

Tableau de Karnaugh associé :

Xi \ CAXi .CDXi	00	01	11	10
0	0	0	1	1
1	1	0	1	1

L'équation de Xi est : $X_i = CAXi + \overline{CDX_i} \cdot X_i$

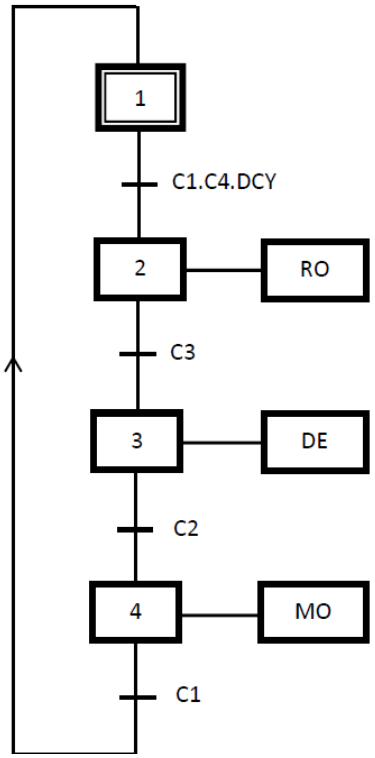
OU

$$X_i = X_{i-1} R_{i-1} + \overline{X_{i+1}} \cdot X_i$$

Ce ne sont pas tous les automates qui se programment en GRAFCET directement. Mais, généralement ils peuvent être programmés en « diagramme échelle » (ou LADDER). Il faut donc pouvoir transformer le GRAFCET qui est la meilleure approche qui existe pour traiter les systèmes séquentiels en « diagramme échelle » qui est le langage le plus utilisé par les automates.

VI.5 Exemple

Soit le grafcet suivant :



X1 { Encl : X4.C1 ; RAZ : X2 }

$$X1 = X4.C1 + \overline{X2}\,.\,X1$$

X2 { Encl : X1.C1.C4.DCY ; RAZ : X3 }

$$X2 = X1.C1.C4.DCY + \overline{X3}\,.X2$$

X3 { Encl : X2.C3 ; RAZ : X4 }

$$X3 = X2.C3 + \overline{X4}\,.\,X3$$

X4 { Encl : X3.C2 ; RAZ : X1 }

$$X4 = X3.C2 + \overline{X1}\,.\,X4$$

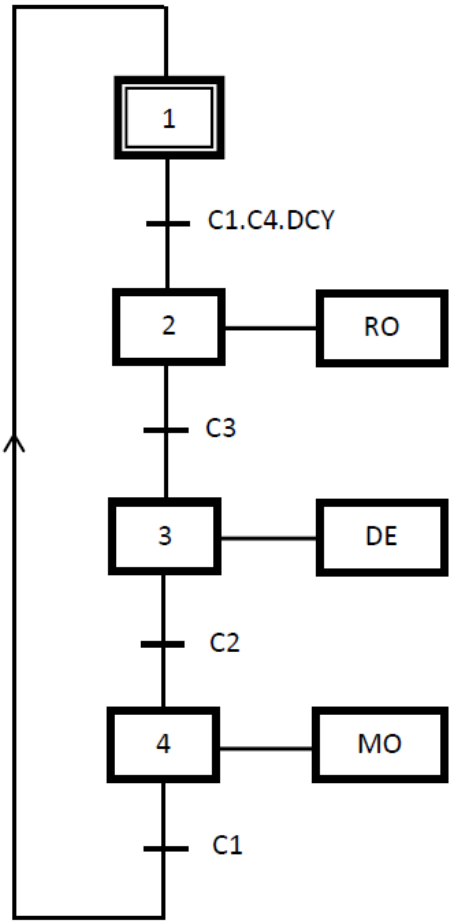
Afin de respecter les règles d'évolution du GRAFCET, chaque étape peut être matérialisée par une mémoire du type marche prioritaire possédant une structure de la forme :

$$X = Encl + \overline{RAZ}.X$$

Les termes d'enclenchement et de remise à zéro sont définis de la manière suivante :

ETAPE X

- Encl : Etat logique de l'Etape(s) précédente(s). Réceptivité
- RAZ : Etat logique de l'Etape(s) suivante(s)



X1
- Encl : X4.C1
- RAZ : X2

$$X1 = X4.C1 + \overline{X2}\,.\,X1$$

X2
- Encl : X1.C1.C4.DCY
- RAZ : X3

$$X2 = X1.C1.C4.DCY + \overline{X3}\,.X2$$

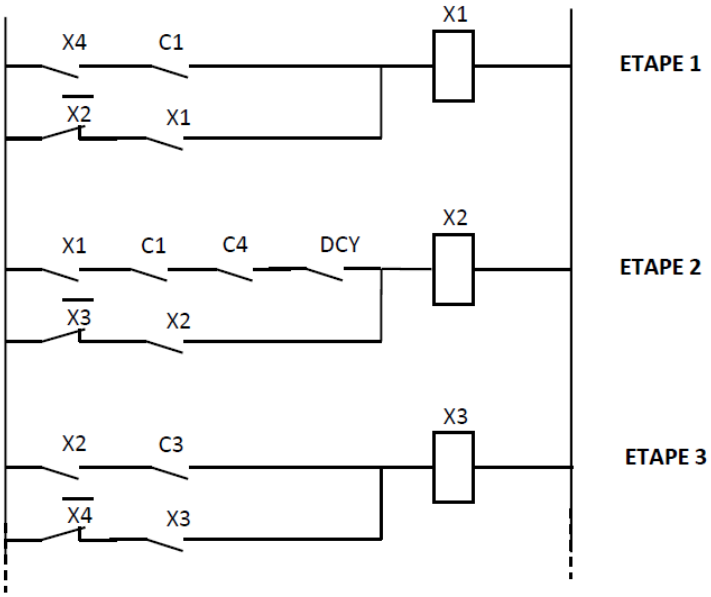
X3
- Encl : X2.C3
- RAZ : X4

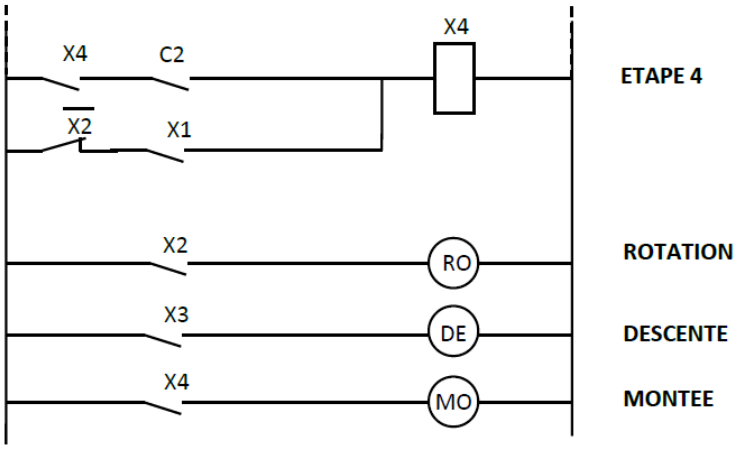
$$X3 = X2.C3 + \overline{X4}\,.\,X3$$

X4
- Encl : X3.C2
- RAZ : X1

$$X4 = X3.C2 + \overline{X1}\,.\,X4$$

Les équations des mémoires étape déterminée précédemment nous donnent le schéma de câblage électrique suivant :





Pour établir la commande de chaque sortie, il suffit de considérer la ou les étapes durant lesquelles la sortie doit être enclenchée. Ainsi :

La sortie RO a lieu durant l'ETAPE 2 d'où RO = X2

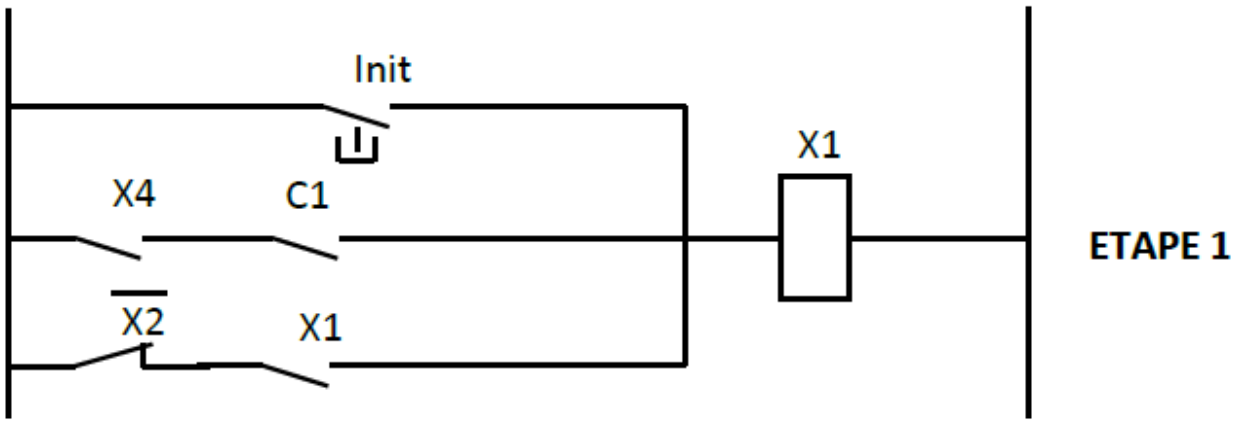
La sortie DE a lieu durant l'ETAPE 3 d'où DE = X3

La sortie MO a lieu durant l'ETAPE 4 d'où MO = X4

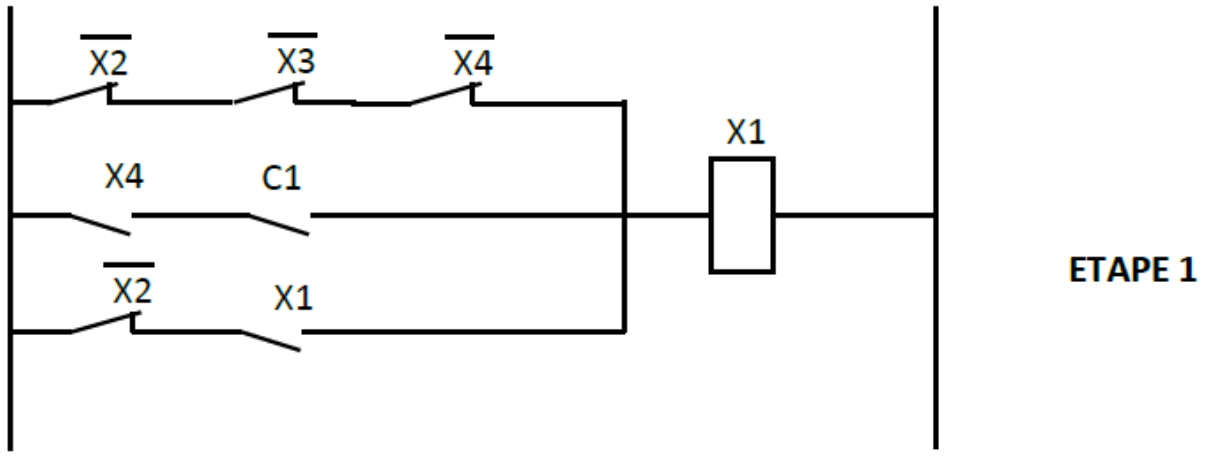
VI.5.1 Initialisation de la séquence :

Nous remarquons sur le schéma précédent qu'à la mise sous tension, toutes les mémoires se trouvant ici à l'état repos, aucune évolution n'est possible. Il est donc impératif d'initialiser la séquence en venant enclencher la mémoire X1 matérialisant l'étape initiale de notre GRAFCET. Ceci est obtenu :

- ✓ Soit en utilisant un contact d'initialisation ou un contact de passage commandé lors de la mise sous tension de l'automatisme, comme le montre le schéma suivant :



- ✓ Soit en testant l'état repos de toutes les mémoires d'étape suivantes, pour venir alors systématiquement enclencher la mémoire X1 , comme le montre le schéma suivant :



IV.5.2 Le langage LADDER (LD) :

Le langage des API d'origine américaine utilise le symbolisme classique des schémas à relais accompagné de blocs graphiques préprogrammés pour réaliser des fonctions d'automatisme (calculs, temporisation, compteur,…..). C'est une suite de réseaux qui seront parcourus séquentiellement. Les entrées sont représentées par des interrupteurs -| |- ou -|/|- si entrée inversée, les sorties par des bobines **-( )-** ou des bascules **-(S)- -(R)-.**Il y a également d'autre opérations :

l'inverseur **-|NOT|-**,

l'attente d'un front montant **-(P)-** ou descendant **-(N)-.**

Les sorties sont obligatoirement à droite du réseau On doit évidemment identifier nos **E/S**, soit directement par leur code (**Ia.b** / **Qa.b**), ou avec leur libellé en clair défini dans la table des mnémoniques. On relie les éléments en série pour la fonction **ET**, en parallèle pour le **OU**. On peut utiliser des bits internes (peuvent servir en bobines et interrupteurs), comme on utilise dans une calculatrice une mémoire pour stocker un résultat intermédiaire (**Ma.b**). On peut aussi introduire des éléments plus complexes, en particulier les opérations sur **bits** comme par exemple une bascule **SR** (priorité déclenchement), **RS** (priorité enclenchement), **POS** et **NEG** pour la détection de fronts... on trouvera d'autres fonctions utiles, les compteurs, les temporisateurs et le registre à décalage. On peut également utiliser des fonctions plus complexes (calculs sur mots par exemple).

Adressage des entrées/sorties

La déclaration d'une entrée ou sortie donnée à l'intérieur d'un programme s'appelle l'adressage. Les entrées et sorties des API sont la plupart du temps regroupées en groupes de huit sur des modules d'entrées ou de sorties numériques. Cette unité de huit est appelée **octet**. Chaque groupe reçoit un numéro que l'on appelle l'**adresse d'octet**. Afin de permettre l'adressage d'une entrée ou sortie à l'intérieur d'un octet, chaque octet est divisé en huit **bits**. Ces derniers sont numérotés de 0 à 7. On obtient ainsi l'**adresse du bit.**.L'API représenté ici a les octets d'entrée 0 et 1 ainsi que les octets de sortie 0 et 1.

Nom	Type de données	Adresse
ETAPE 1	Bool	M0.1
ETAPE 2	Bool	M0.2
ETAPE 3	Bool	M0.3
ETAPE 4	Bool	M0.4
C1	Bool	I0.0
C2	Bool	I0.1
C3	Bool	I0.2
C4	Bool	I0.3
DCY	Bool	I0.4
RO	Bool	Q0.0
DE	Bool	Q0.1
MO	Bool	Q0.2

Etape 1 de type logique (Bool) affecté à la mémoire M0.1

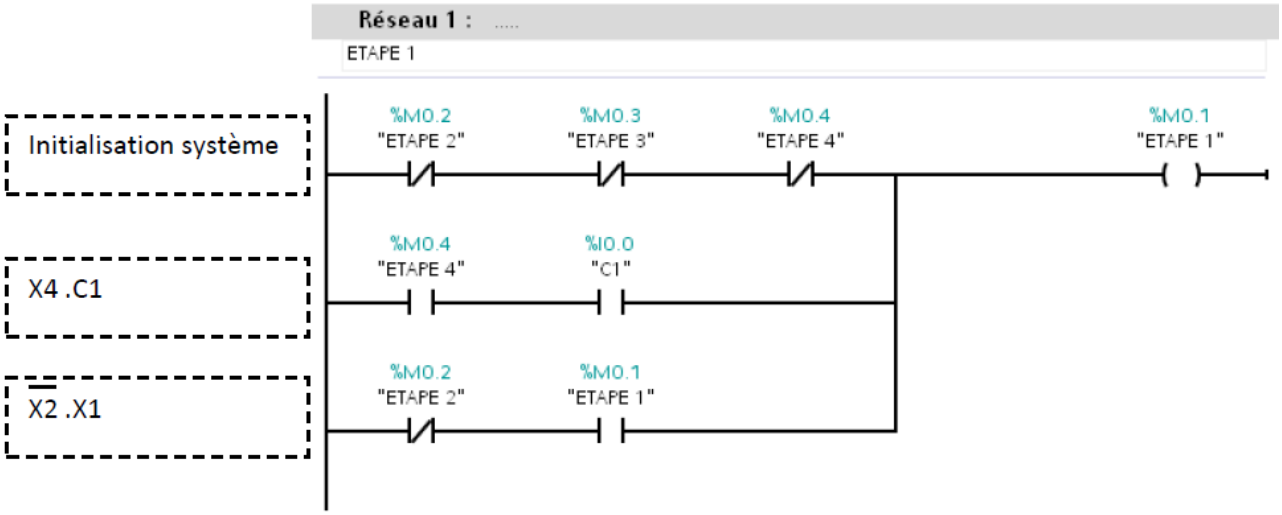
Le capteur C1 est de type logique et affecté à l'adresse I0.0

La sortie DE est de type logique et affecté à l'adresse Q0.0

Par exemple, pour adresser la 5ème entrée du **DCY** en partant de la gauche, on définit l'adresse suivante : $\mathbf{I_{0.4}}$ **I** indique une adresse de type entrée, **0**, l'adresse d'octet et **4**, l'adresse de bit. Les adresses d'octet et de bit sont toujours séparées par un point. Pour adresser la 3ème sortie, par exemple, on définit l'adresse suivante : $\mathbf{Q_{0.2}}$ **Q** indique une adresse de type Sortie, **0**, l'adresse d'octet et **2**, l'adresse de bit. Les adresses d'octet et de bit sont toujours séparées par un point. ***Remarque :*** *L'adresse du bit de la dixième sortie est un **1** car la numérotation commence à zéro.*

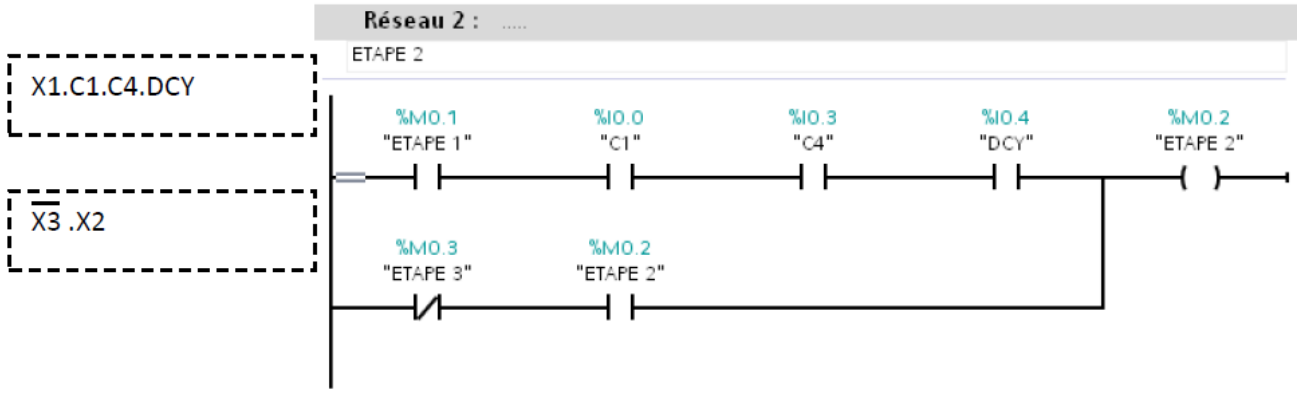
Dans l'exemple précédent et suivant la table mnémonique d'affectation le programme en LADER de la première étape est :

$$X_1 = X_4.C_1 + \overline{X_2}.C_1$$



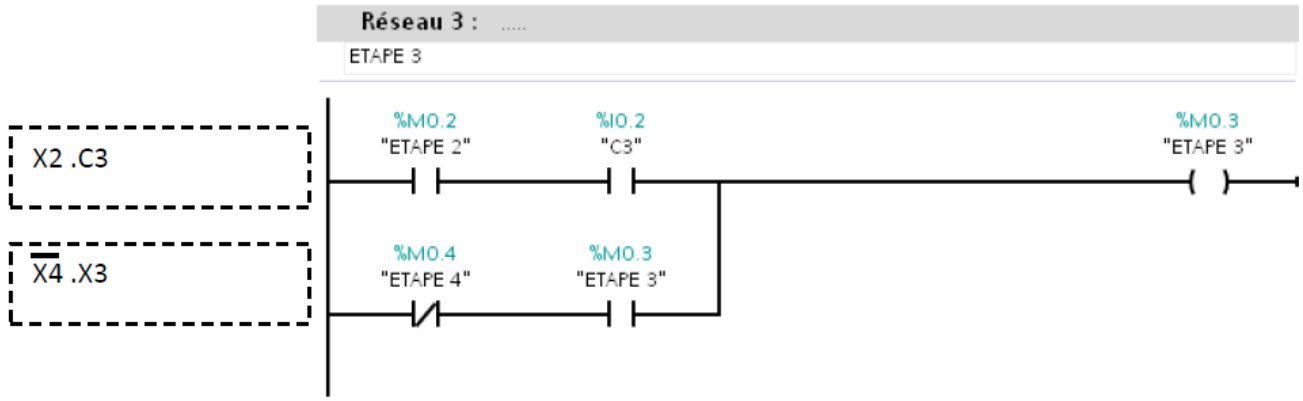
L'étape 2

$$X_2 = X_1.C_1.C_4.DCY + \overline{X_3}.X_2$$

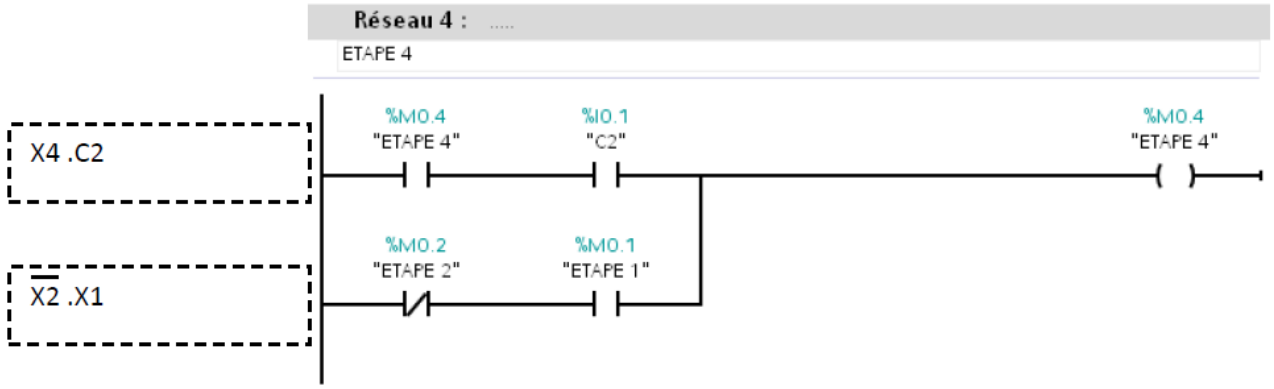


L'étape 3

$$X_3 = X_2.C_3 + \overline{X_4}.X_3$$

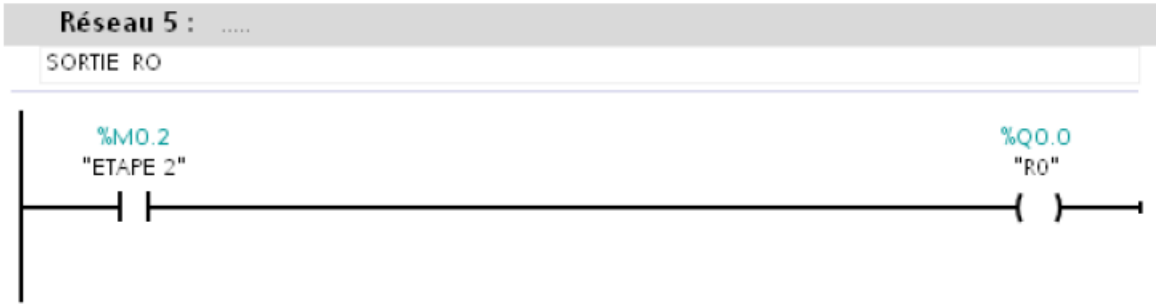


L'étape 4

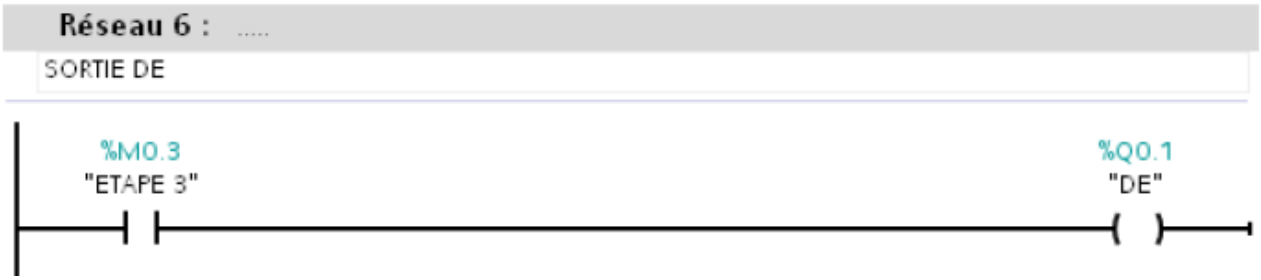


Pour la programmation des sorties

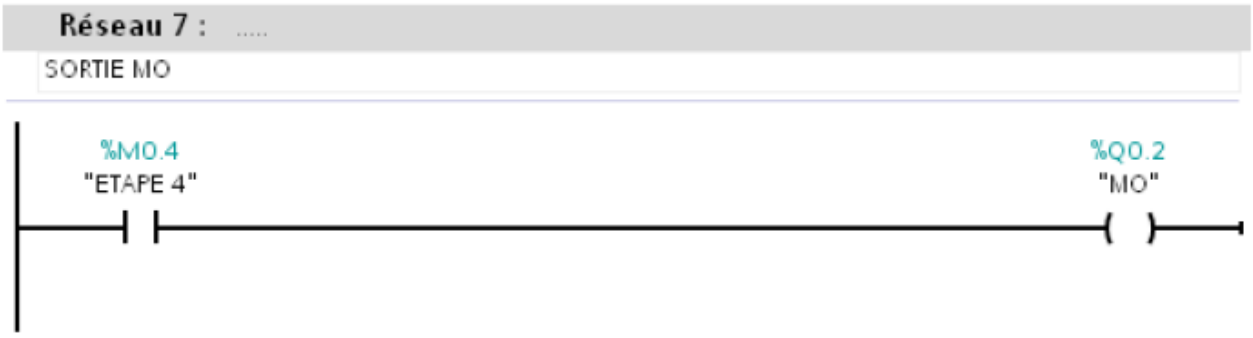
R0 : est actionné uniquement à l'étape 2



DE : est actionné uniquement à l'étape 3



MO : est actionné uniquement à l'étape 4



Le programme peut être simplifier si en utilisant les bobines **Set/ Reset** ou les bascules **SR** ou **RS** et en tenant compte des cinq règles du GRAFCET.