Formation Automatique et Informatique Industrielle

Master 1 S2

Matière : Systèmes Embarqués et Systèmes Temps Réel SE-STR

Par : ATOUI Hamza

Plan du cours

- Positionnement du problème à travers un processus industriel.
- Les deux célèbres langages de programmation des API (LADDER et GRAFCET) en bref.
- Conversion du LADDER à un algorithme informatique.
- Conversion du GRAFCET à un algorithme informatique.
- Exercices de TD.
- Annexe : conversion du GRAFCET en LADDER

Positionnement du problème à travers un processus industriel

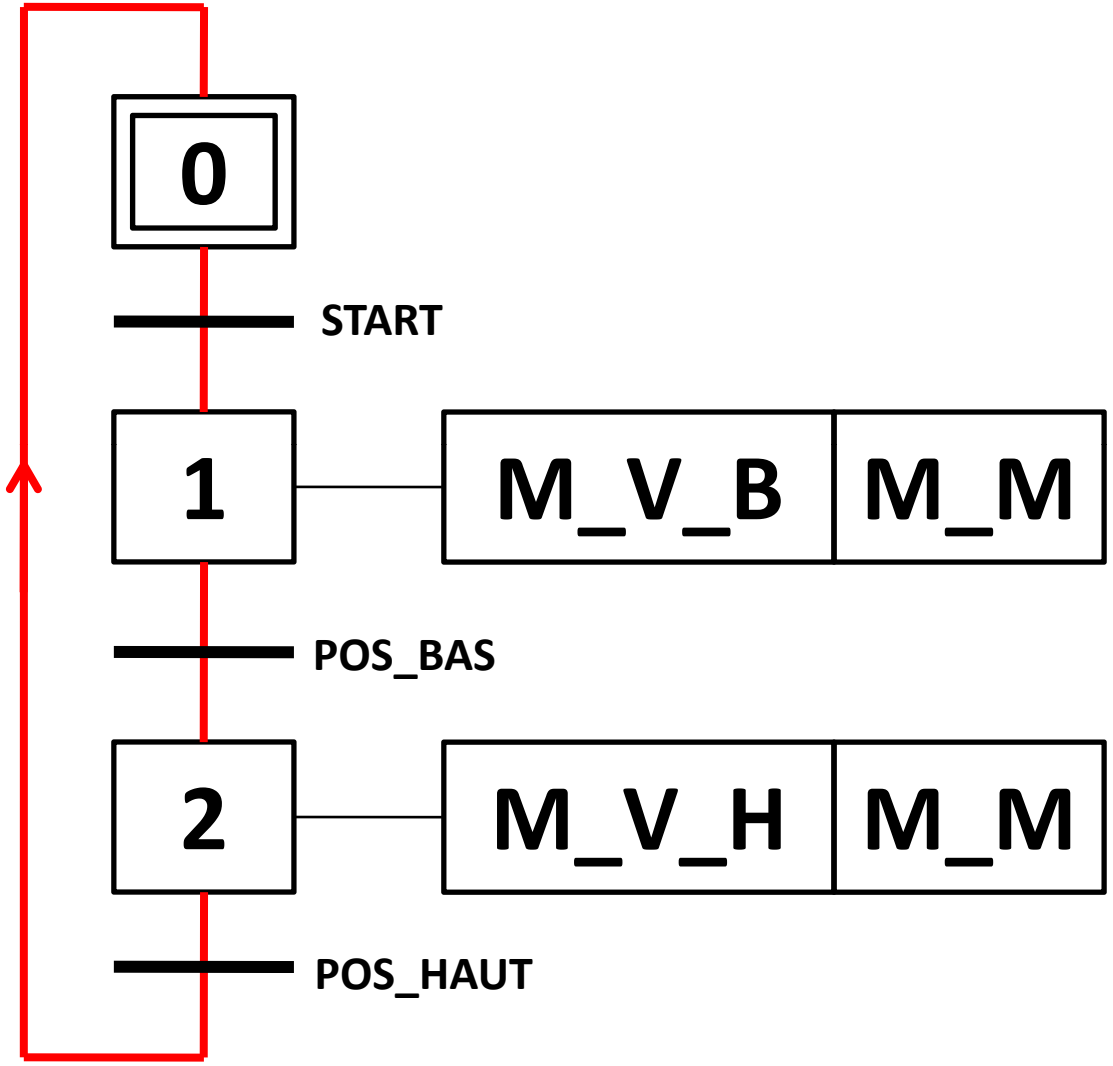
- Soit le processus suivant qui fait le perçage des pièces après fabrication:
  1. Nécessite un bouton START pour démarrer le cycle.
  2. Nécessite deux fin de course pour localiser la position de la perceuse.
  3. Nécessite un moteur pour déplacer verticalement la perceuse (du bas en haut ou du haut en bas) (deux signaux de commande) et un autre moteur pour tourner la mèche (un signal de commande).
  4. Cahier des charges : Le système attend l'appui sur le bouton START (moteur mèche à l'arrêt), si le cas, la perceuse descend vers le bas jusqu'à la fin de course niveau bas. Puis le système donne l'ordre à la perceuse pour monter vers le haut jusqu'à la fin de couse niveau haut (cycle de perçage) (moteur mèche en marche) puis le système attend une autre fois l'appui sur le bouton START pour démarrer un autre cycle.

Positionnement du problème à travers un processus industriel

- Donc, le système qu'on va réaliser est un système à 3 entrées et 3 sorties comme indique la figure suivante :

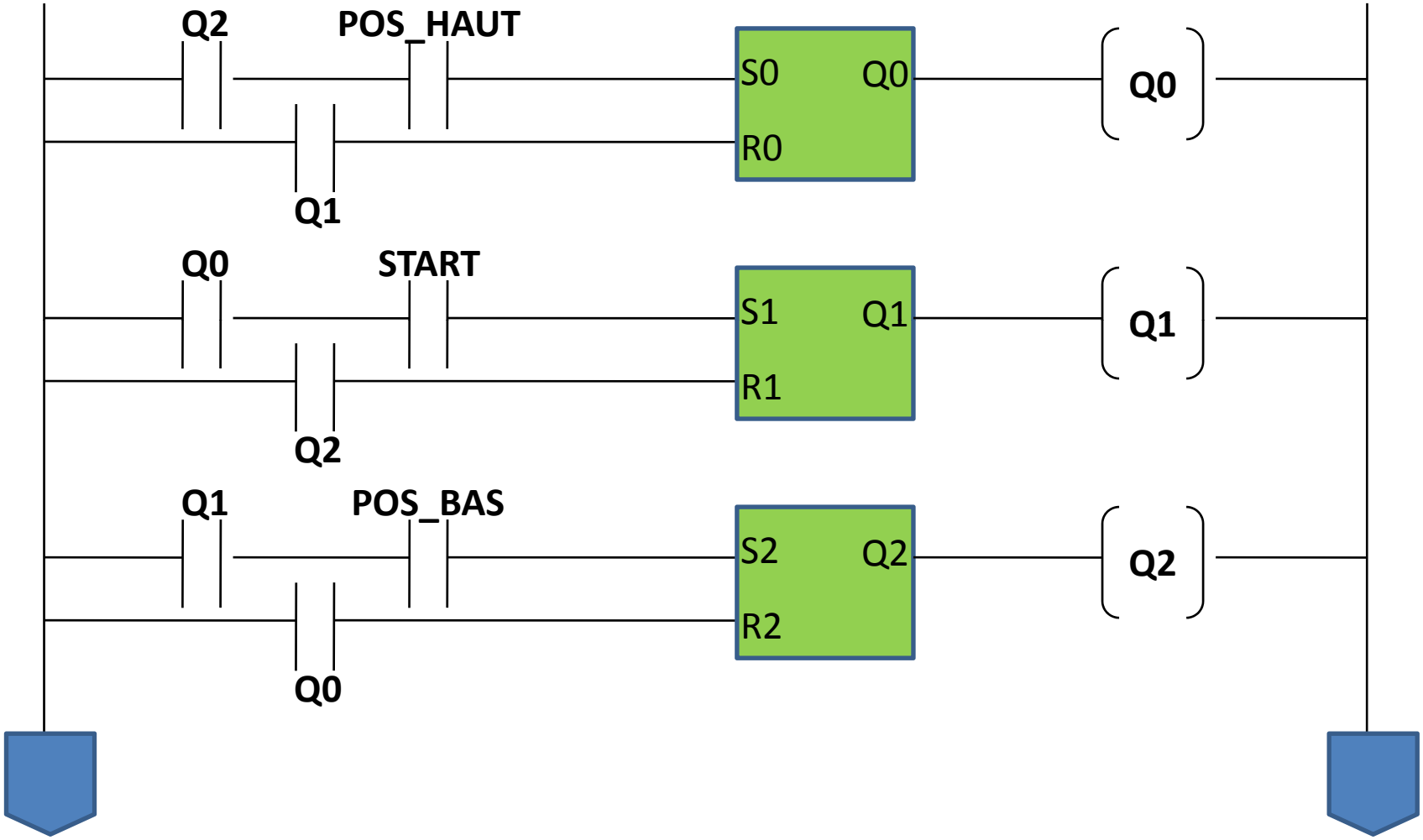
Positionnement du problème à travers un processus industriel

- D’après le cahier des charge Le **GRAFCET** du système est le suivant :

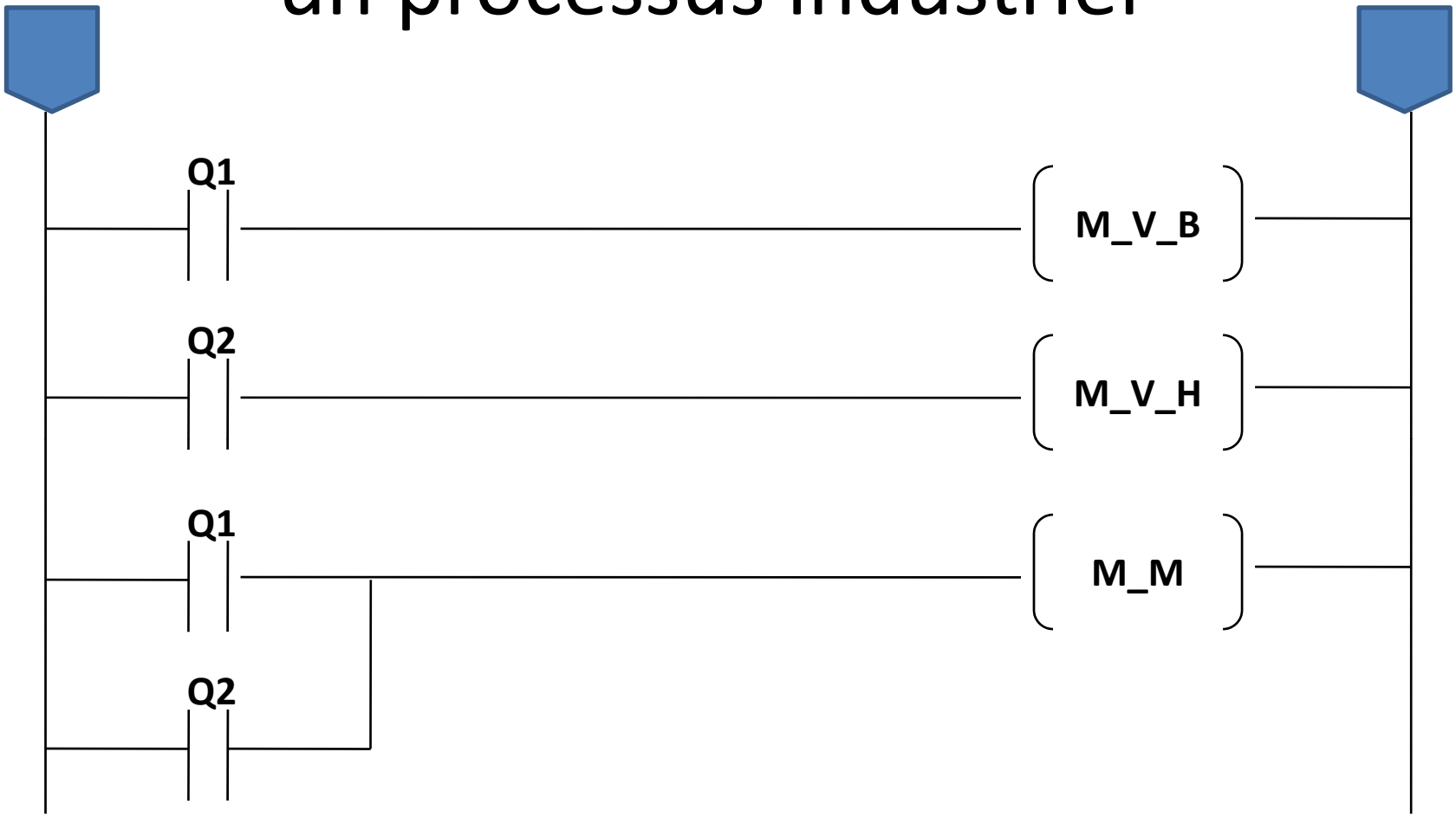


Positionnement du problème à travers un processus industriel

- Et le **LADDER** : le processus de la perceuse est complètement séquentiel, donc, on va utiliser des **bascules SR** pour réaliser les différences étapes de la séquence de perçage.



Positionnement du problème à travers un processus industriel



N.B : j'ai considéré que au moment de démarrage de l'API, le code l'initialisation fait le SET de Q0 et le RESET du reste {Q1 et Q2}

Positionnement du problème à travers un processus industriel

- ***Pour faire soit le LADDER ou le GRAFCET, est ce que on est obligé d'acheter un API avec ses outils de développement ?***
- ***Est ce que c'est pratique d'utiliser tout un API qui ça coûte plus de 50.000 DA pour automatiser notre perceuse ?***
- ***Est-ce qu'il y a un autre issu pour ne pas gaspiller ?***

Positionnement du problème à travers un processus industriel

- Absolument, oui, il y a un autre issu, ***c'est la merveille des systèmes embarqués à base de µC***.

- Une autre question: ***est-ce que on peut traduire le GRAFCET ou le LADDER vers un algorithme informatique ?***

Positionnement du problème à travers un processus industriel

- Tout simplement oui, par ce que l'API est une plateforme à base de µP comme les µC, donc faisable. Il suffit de connaitre comment traduire le LADDER/GRAFCET en algorithme.

- Dans ce qui suit, on va étudier cette traduction (conversion)!

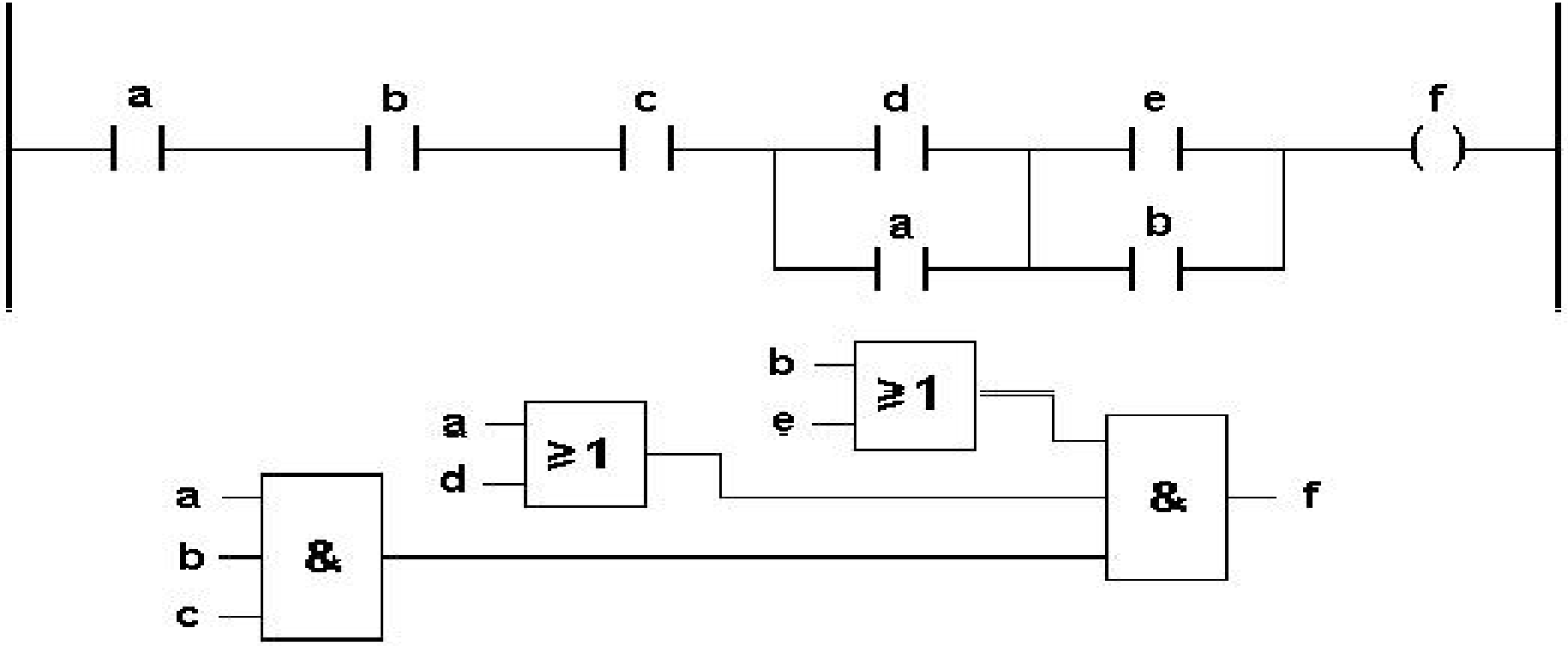
Les deux célèbres langages de programmation des API (LADDER et GRAFCET) en bref

- Les deux célèbres langages de programmation des **API** sont le **LADDER** et le **GRAFCET**. Dont, un **automate programmable industriel**, ou **API**, est un dispositif électronique programmable destiné à la commande/contrôle de processus industriels par un **traitement séquentiel** (**à cause de µP qui se trouve à l'intérieur de l'API**).

- **Ladder Diagram** (**LD**) ou **Langage Ladder** ou **schéma à contacts** est un langage graphique très populaire auprès des automaticiens pour programmer les API. Il ressemble un peu aux schémas électriques et il est facilement compréhensible.

Les deux célèbres langages de programmation des API (LADDER et GRAFCET) en bref

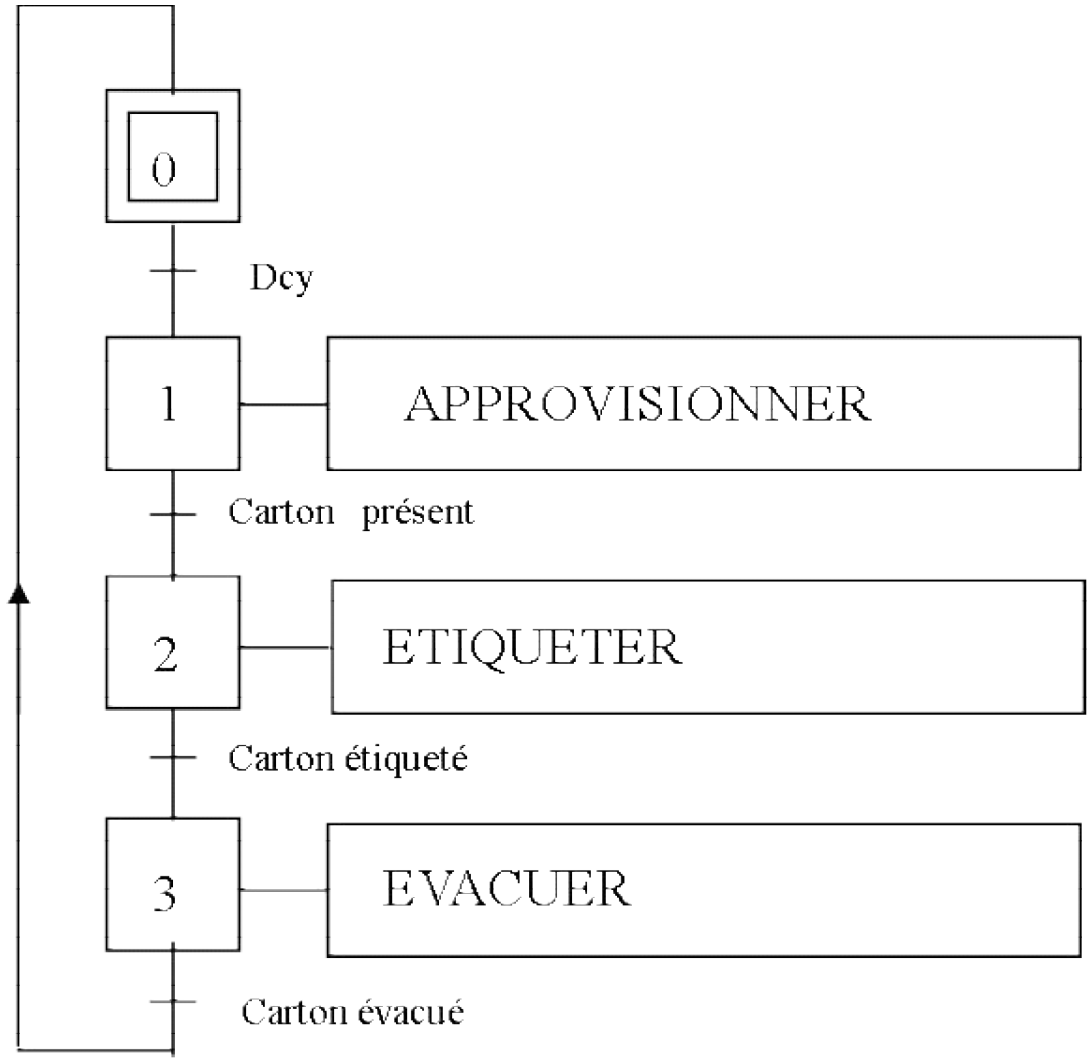
- La figure suivante présente une ligne en LADDER avec son équivalant en logique combinatoire.



Les deux célèbres langages de programmation des API (LADDER et GRAFCET) en bref

- Le **GRAFCET** (Graphe Fonctionnel de Commande des Étapes et Transitions) est un mode de représentation et d'analyse d'un automatisme, particulièrement bien adapté aux systèmes à **évolution séquentielle**, c'est-à-dire décomposable en étapes. Il est dérivé du modèle mathématique des réseaux de Petri.
- Le **GRAFCET** est donc un langage graphique représentant le fonctionnement d'un automatisme par un ensemble :
  - d'étapes auxquelles sont associées des actions ;
  - de transitions entre étapes auxquelles sont associées des conditions de transition (réceptivités) ;
  - des liaisons orientées entre les étapes et les transitions.
- La figure suivante présente un automatisme en GRAFCET.

Les deux célèbres langages de programmation des API (LADDER et GRAFCET) en bref



Conversion du LADDER à un algorithme informatique

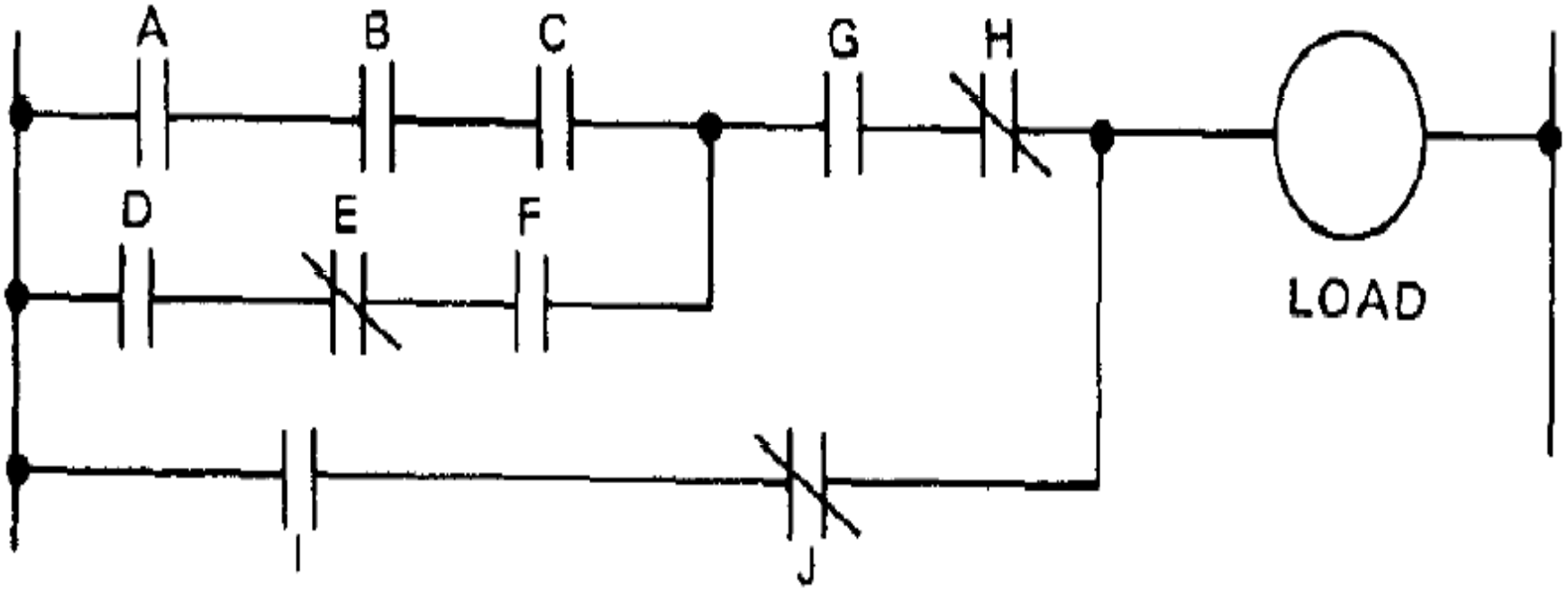
- Les étapes à suivre pour convertir un LADDER au langage informatique d'une **façon professionnelle** :
  - Faire la conception du processus par la vision d'un automaticien en LADDER.
  - Faire la conversion du LADDER vers un schéma logique.
  - Noter les signaux internes sur le schéma logique pour connaitre le nombre des variables internes (mémentos).
  - Passer vers la conversion par :

Conversion du LADDER à un algorithme informatique

1. Associer à chaque entrée une variable.
2. Associer à chaque mémento une variable.
3. Associer à chaque sortie une variable.
4. Faire l'algorithme suivant:
    1. Initialiser les variables de sortie (l'initialisation des mémentos reste optionnelle à l'exception des bascules).
    2. Boucler infiniment sur le cycle automate:
        1. La mise à jour des sorties par le contenu des variables de sortie (**mise à jour des sorties**).
        2. La mise à jour des variables d'entrée par les entrées de système (**lecture des entrées**).
        3. Réalisation des différentes équations logiques en commençant par les mémentos les plus internes vers les variables de sortie (**Traitement**).

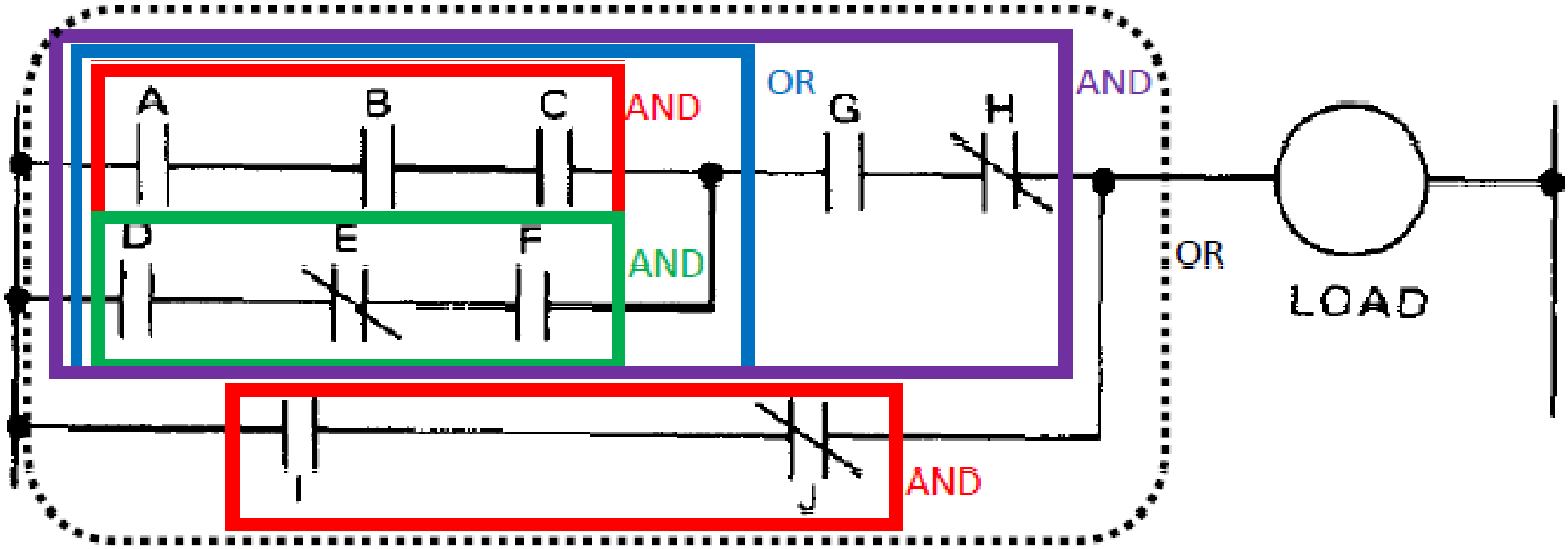
Conversion du LADDER à un algorithme informatique

- Exemple : soit le LADDER suivant qui présente l'automatisme d'un processus industriel.



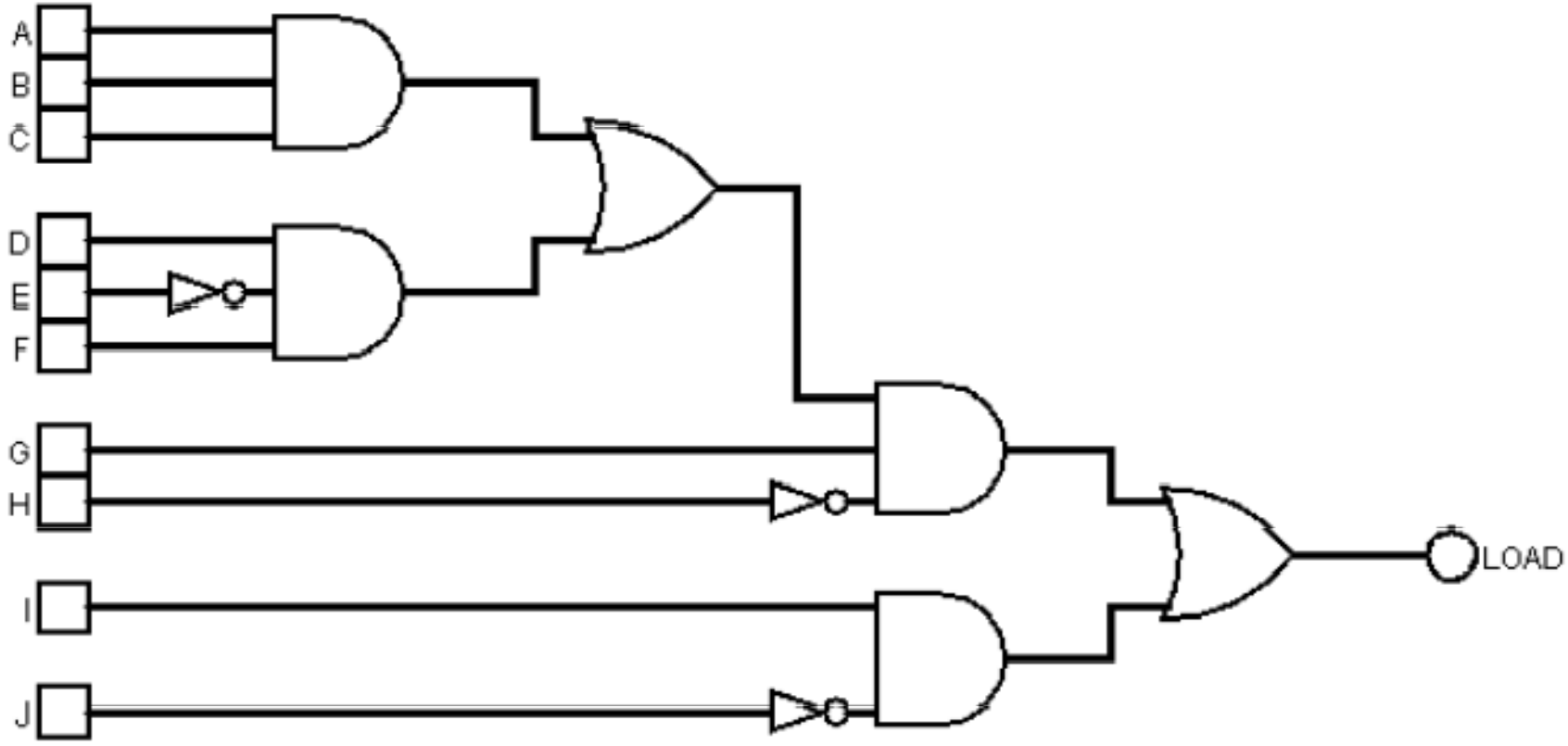
Conversion du LADDER à un algorithme informatique

- Conversion en schéma logique:



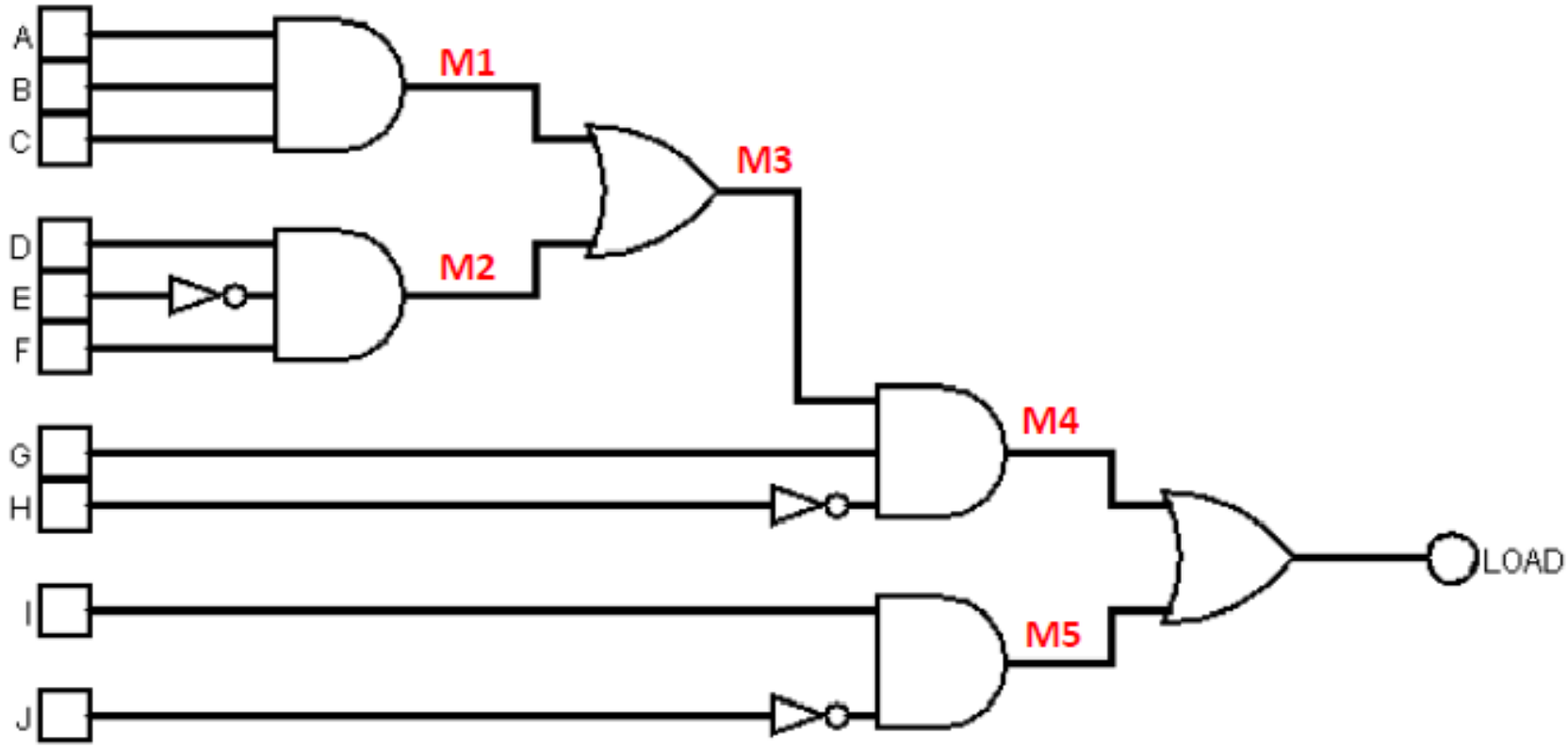
Conversion du LADDER à un algorithme informatique

- Conversion en schéma logique:



Conversion du LADDER à un algorithme informatique

- Notation des signaux internes :



Ce n'est pas la peine de noter les signaux des inverseurs.

Conversion du LADDER à un algorithme informatique

- D’une façon générale, associer à chaque signal interne et externe une variable:

- **Les variables d’entrée {varA, varB, varC, varD, varE, varF, varG, varH, varI et varJ}.**
- **Les variables internes (mémentos) {M1, M2, M3, M4, et M5}.**
- **Les variables de sortie {varLOAD}.**

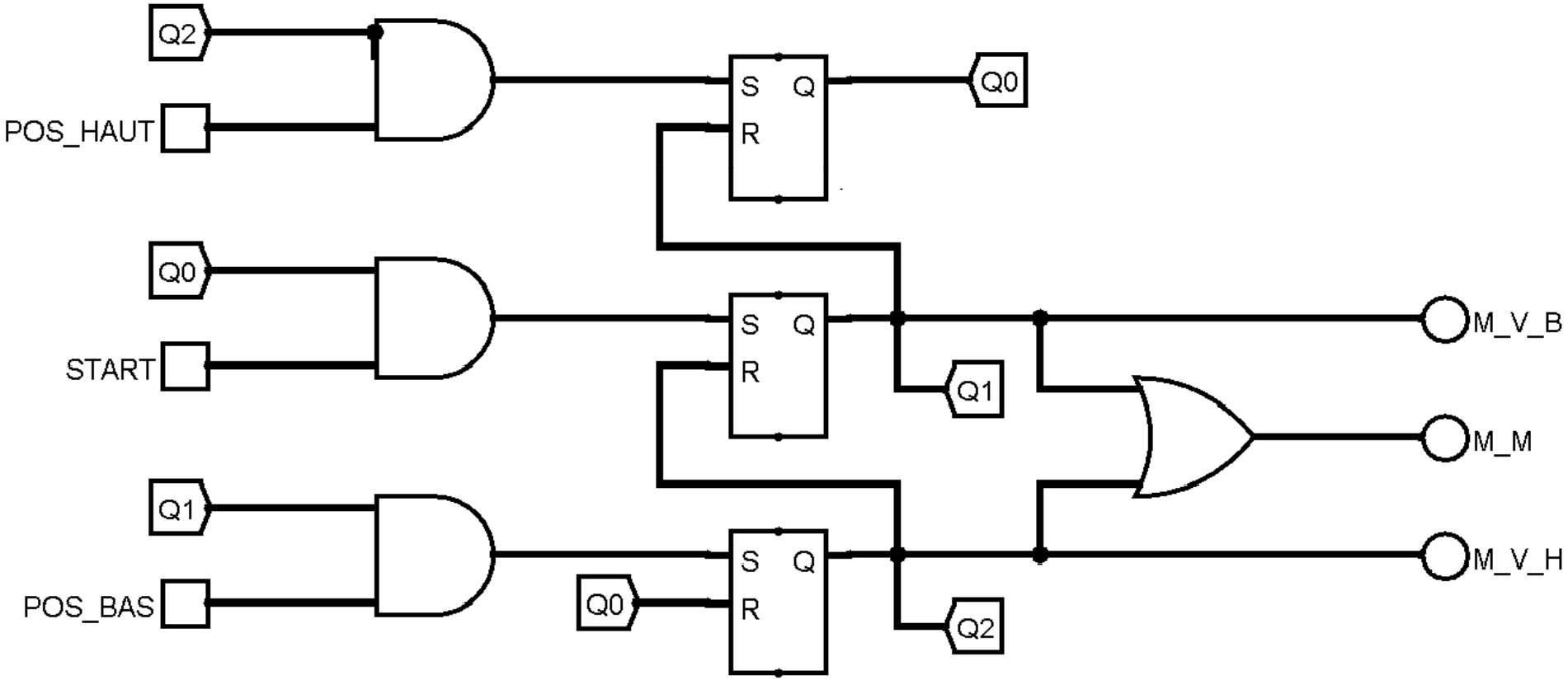
Conversion du LADDER à un algorithme informatique

- Algorithme:

```
//  Initialisation
varLOAD ← 0;
//  Cycle automate
While (True)
  //  la mise à jour des sorties
  LOAD ← varLOAD;
  // la mise à jour des variables d'entrée
  varA ← A; varB ← B; varC ← C; varD ← D; varE ← not(E);
  varF ← F; varG ← G; varH ← not(H); varI ← I; varJ ← not(J);
  // le traitement
  M1 ← varA and varB and varC; M2 ← varD and varE and varF;
  M3 ← M1 or M2; M4 ← M3 and varG and varH; M5 ← varI and varJ;
  varLOAD ← M4 or M5;
End While
```

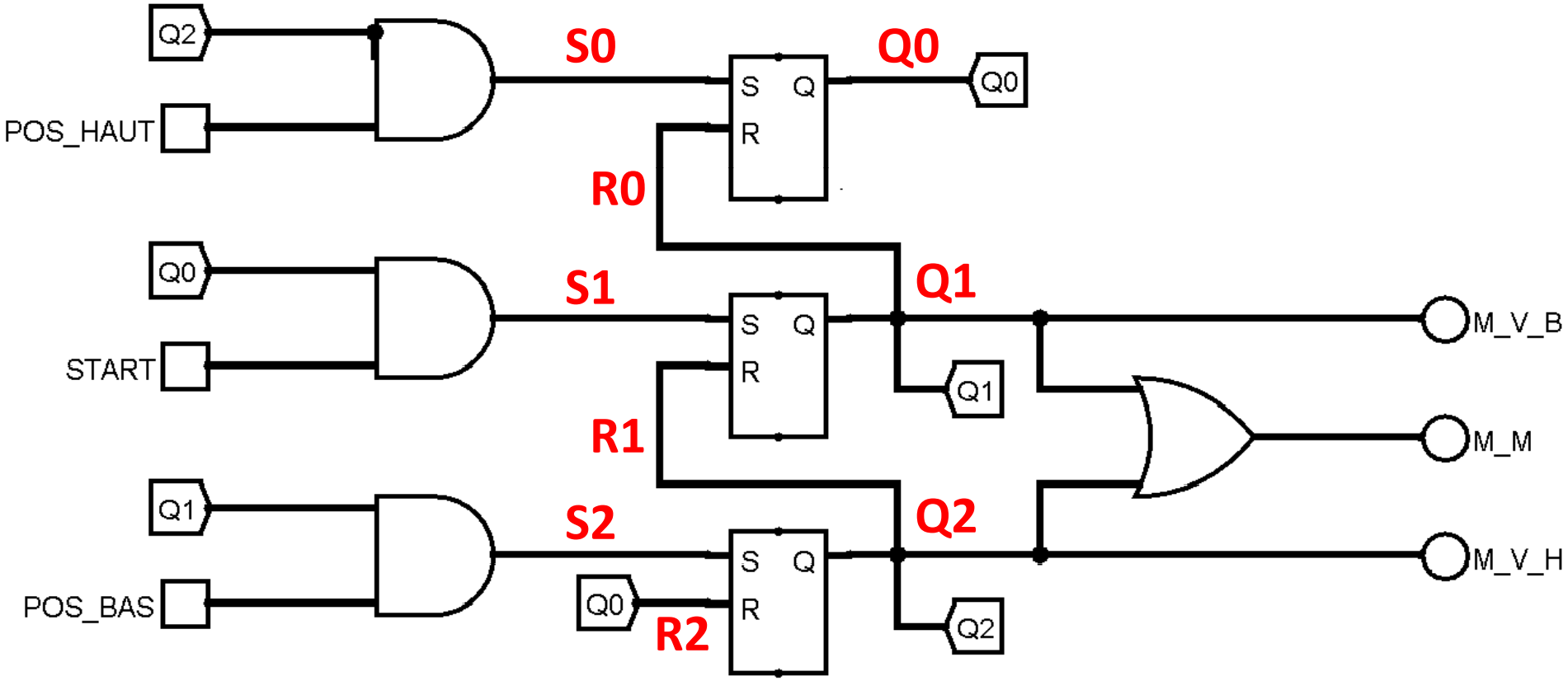
Conversion du LADDER à un algorithme informatique

- L'exemple de la perceuse (**Soyez prudent à cause de présence des bascules**):
- Conversion en schéma logique:



Conversion du LADDER à un algorithme informatique

- Notation des signaux internes :



Conversion du LADDER à un algorithme informatique

- D’une façon générale, associer à chaque signal interne et externe une variable:

- **Les variables d’entrée {varPH, varPB et varSTART}.**
- **Les variables internes (mémentos) {S0, S1, S2, R0, R1, R2, Q0, Q1 et Q2}.**
- **Les variables de sortie {varMVB, varMVH et varMM}.**

Conversion du LADDER à un algorithme informatique

- Algorithme:

```
//  Initialisation
varMVB ← 0; varMVH ← 0; varMM ← 0;
// faites attention à les bascules, il faut initialiser
Q0 ← 1;
Q1 ← 0;
Q2 ← 0;
//  Cycle automate
While (True)
  //  la mise à jour des sorties
  M_V_B ← varMVB;
  M_V_H ← varMVH;
  M_M ← varMM;
```

Conversion du LADDER à un algorithme informatique

- Algorithme (la suite):

```
// la mise à jour des variables d'entrée
varPH ← POS_HAUT; varPB ← POS_BAS; varSTART ← START;
//  le traitement (attention à les bascules, les SET puis les RESET)
//  les SET
S0 ← Q2 and varPH;
S1 ← Q0 and varSTART;
S2 ← Q1 and varPB;
If (S0=1)  then Q0 ← 1; end if;
If (S1=1)  then Q1 ← 1; end if;
If (S2=1)  then Q2 ← 1; end if;
```

Conversion du LADDER à un algorithme informatique

- Algorithme (la suite):

```
  // les RESET
  R0 ← Q1;
  R1 ← Q2;
  R2 ← Q0;
  If (R0=1) then Q0 ← 0; end if;
  If (R1=1) then Q1 ← 0; end if;
  If (R2=1) then Q2 ← 0; end if;
  // les variables de sortie
  varMVB ← Q1; varMVH ← Q2; varMM ← Q1 or Q2;
End while
```

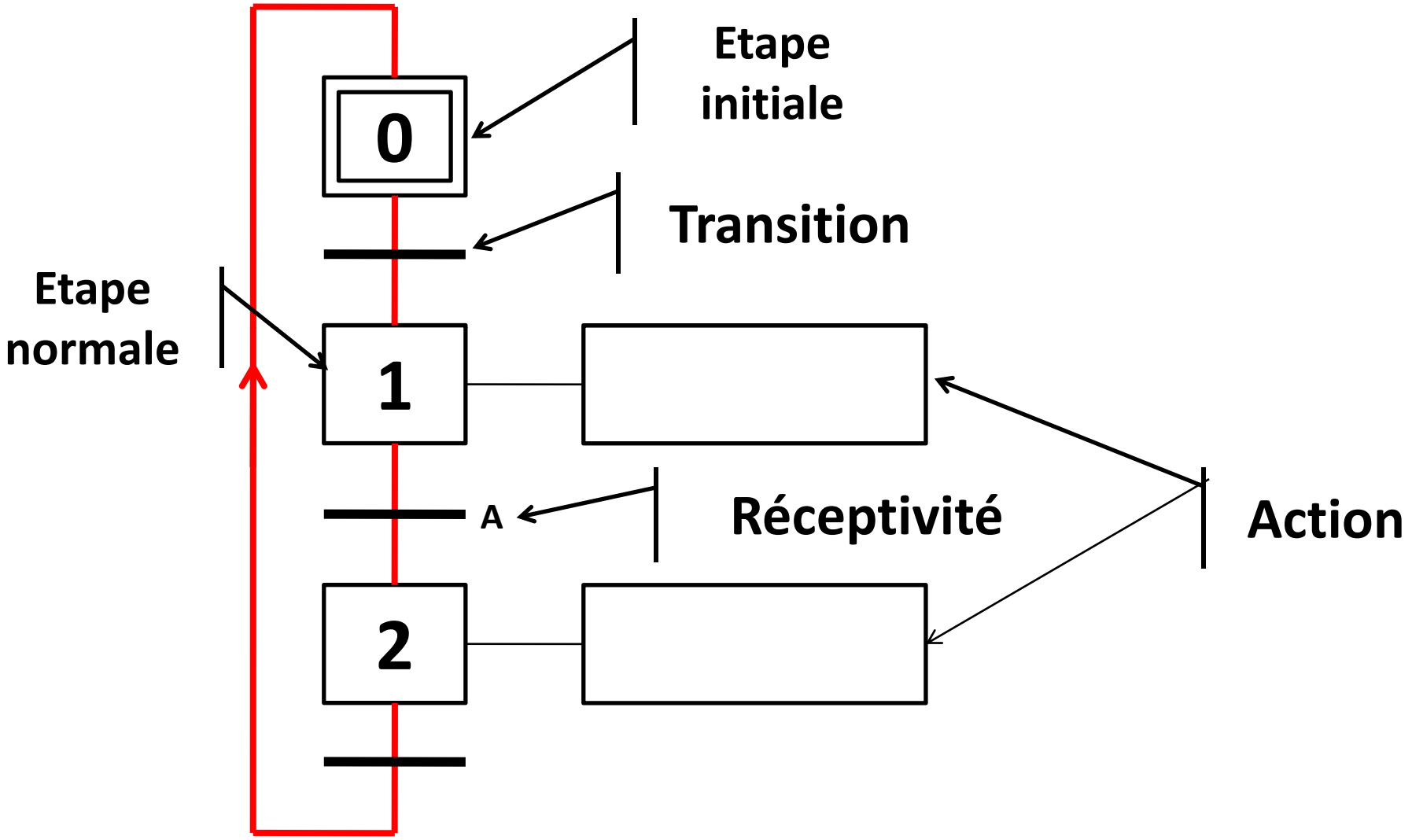
Conversion du LADDER à un algorithme informatique

- Pour faire un travail élégant sur l'axe du programme et rendre le, plus lisible et la mise au point facile à faire !

- Tout simplement, il faut utiliser la programmation modulaire, c'est d'assembler les lignes de code qui présente une tâche bien précise comme:
  - Une fonction pour initialiser,
  - Une fonction pour la mise à jour des sorties,
  - Une fonction pour lire les entrées du processus,
  - Une fonction pour assurer le traitement.

Conversion du GRAFCET à un algorithme informatique

- Présentation d'un GRAFCET :



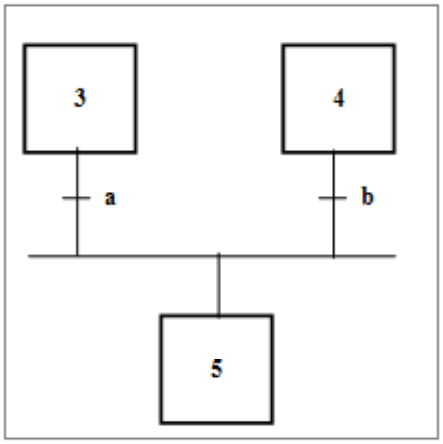
Conversion du GRAFCET à un algorithme informatique

- **Eléments d'un GRAFCET :**
  - **Etapes initiales,**
  - **Etapes normales,**
  - **Transitions,**
  - **Réceptivités,**
  - **Actions,**
  - **Règles d'évolution:**
    - **Convergence/Divergence en OR**
    - **Convergence/Divergence en AND**

Conversion du GRAFCET à un algorithme informatique

- Convergence/Divergence en OR et Convergence/Divergence en AND

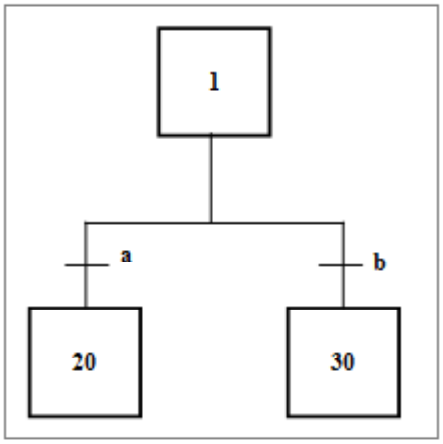
OR



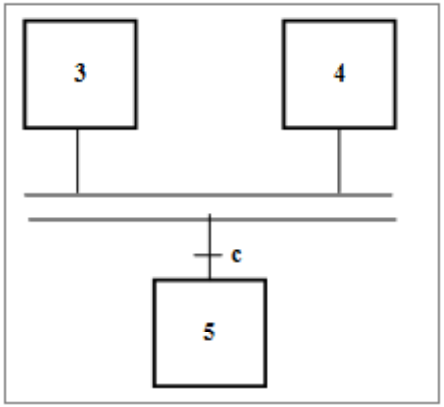
Convergence

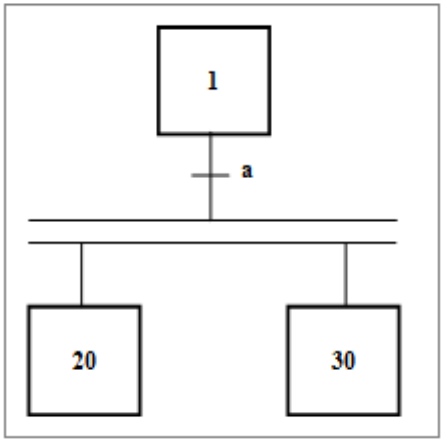
???

Divergence



AND





Conversion du GRAFCET à un algorithme informatique

- Les étapes à suivre pour convertir un GRAFCET au langage informatique d'une **façon professionnelle** :
  - Faire la conception du processus par la vision d'un automaticien en GRAFCET.
  - Passer vers la conversion par :

Conversion du GRAFCET à un algorithme informatique

1. Associer à chaque entrée une variable.
2. Associer à chaque étape une variable.
3. Associer à chaque transition une variable.
4. Faire l'algorithme suivant:
    1. Initialiser les étapes par marquage des étapes initiales et démarquage des étapes normales.
    2. Boucler infiniment sur le cycle automate:
        1. La mise à jour des sorties par le contenu des étapes (**mise à jour des sorties**).
        2. La mise à jour des variables d'entrée par les entrées de système (**lecture des entrées**).
        3. Calcul des transitions par la réalisation des différentes équations logiques. (**calcul des transition**).
        4. La mise à jour des étapes sur la lumière de contenu des transitions (**mise à jour des étapes**).

Conversion du GRAFCET à un algorithme informatique

- L'exemple de la perceuse

Bilan:

==========

Les entrées {START, POS_BAS, POS_HAUT}.

==========

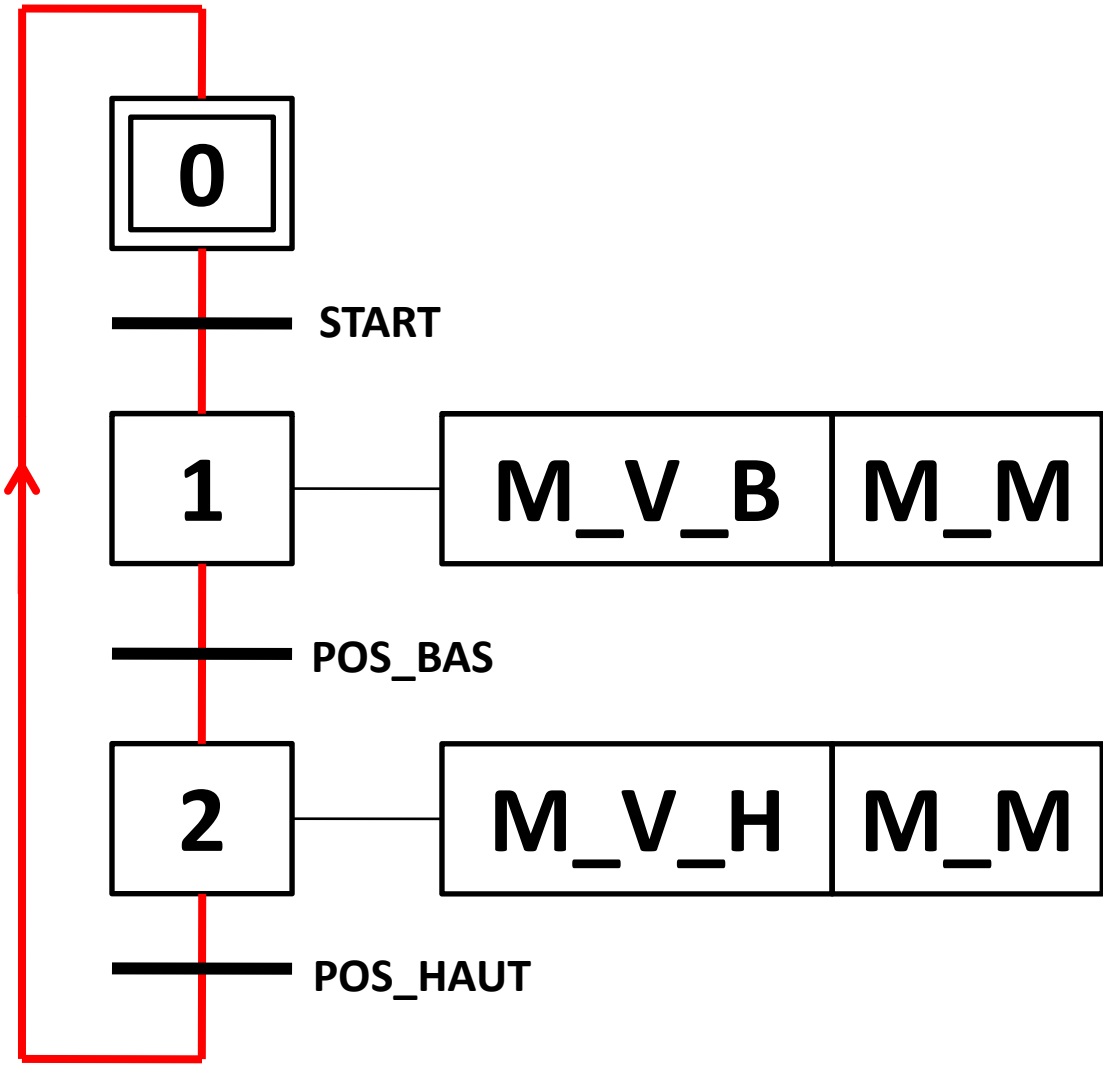
Les sorties {M_V_B, M_V_H, M_M}.

==========

Nombre de transitions :3

==========

Nombre d'étapes :3



Conversion du GRAFCET à un algorithme informatique

- Les variables:
- Les variables d’entrée {varSTART, varPB et varPH}.
- Les variables des étapes {varE0, varE1 et varE2}.
- Les variables des transitions {varT0, varT1 et varT2}.

Conversion du GRAFCET à un algorithme informatique

- Algorithme:

```
// Initialisation des étapes
varE0 ← 1;
varE1 ← 0;
varE2 ← 0;
// Cycle automate
While (True)
        call updateOutputs; // la mise à jour des sorties
        call readInputs;       // la lecture des entrées
        call computeTrans;  // le calcul des transitions
        call updateEtapes;   // la mise à jour des étapes
End While
```

Conversion du GRAFCET à un algorithme informatique

- Le sous-programme ‘updateOutputs’ traitement sur sortie:

M_V_H ← varE1 ;

M_V_B ← varE2;

M_M ← varE1 or varE2;

Conversion du GRAFCET à un algorithme informatique

- Le sous-programme ‘updateOutputs’ traitement sur étape :

```
If (varE0 = 1) then
        M_V_B ← 0; M_V_H ← 0; M_M ← 0;
End If

If (varE1 = 1) then
        M_V_B ← 1; M_V_H ← 0; M_M ← 1;
End If

If (varE2 = 1) then
        M_V_B ← 0; M_V_H ← 1; M_M ← 1;
End If
```

Conversion du GRAFCET à un algorithme informatique

- Le sous-programme ‘readInputs’:

```
varSTART ← START;
varPB ← POS_BAS;
varPH ← POS_HAUT;
```

- Le sous-programme ‘computeTrans’:

```
varT0 ← varE0 and varSTART;
varT1 ← varE1 and varPB;
varT2 ← varE2 and varPH;
```

Conversion du GRAFCET à un algorithme informatique

- Le sous-programme ‘updateEtapes’:

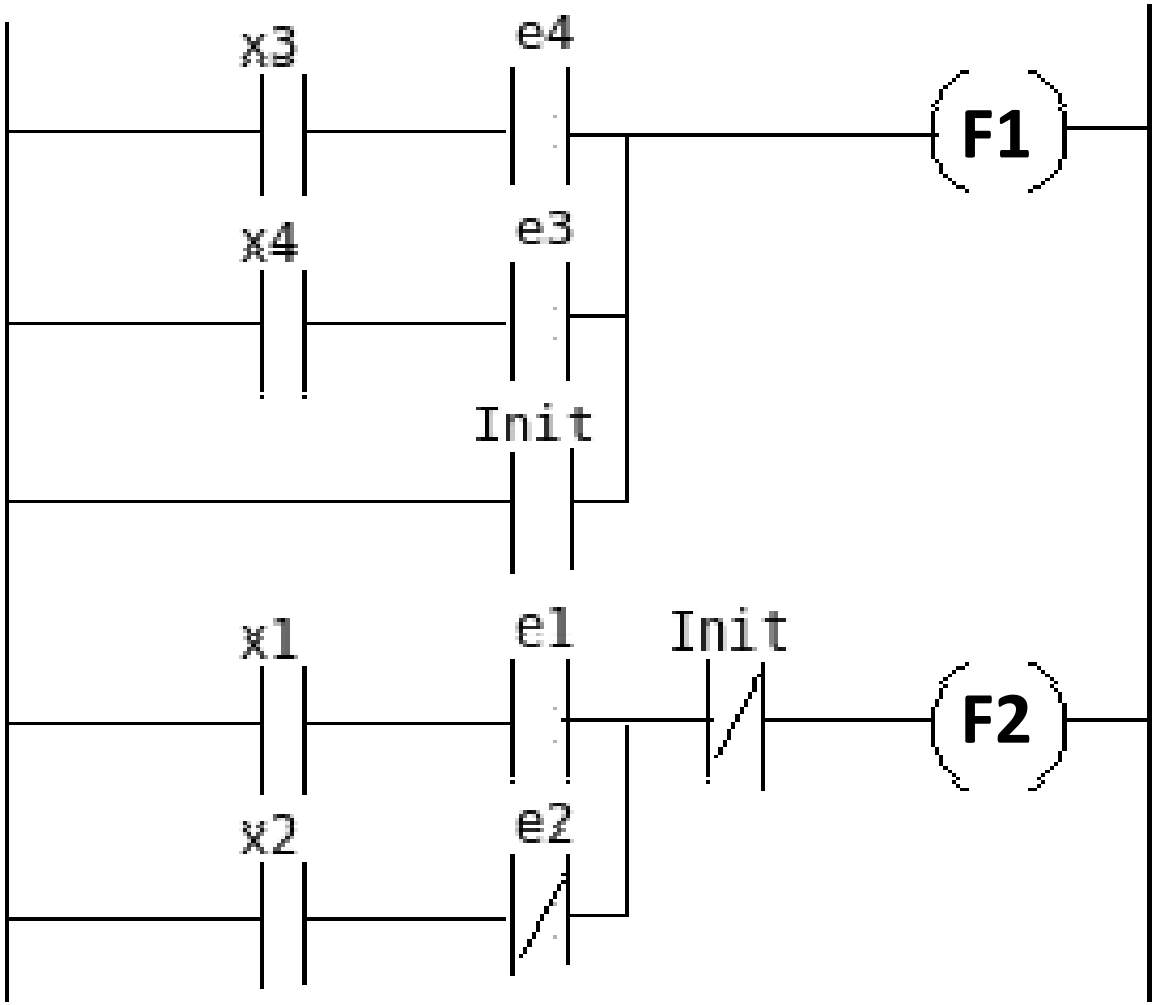
```
If (varT0 = 1) then
        varE1 ← 1; varE0 ← 0;
End If

If (varT1 = 1) then
        varE2 ← 1; varE1 ← 0;
End If

If (varT2 = 1) then
        varE0 ← 1; varE2 ← 0;
End If
```

Exercices de TD

- Exercice 1: convertir le LADDER suivant en algorithme:

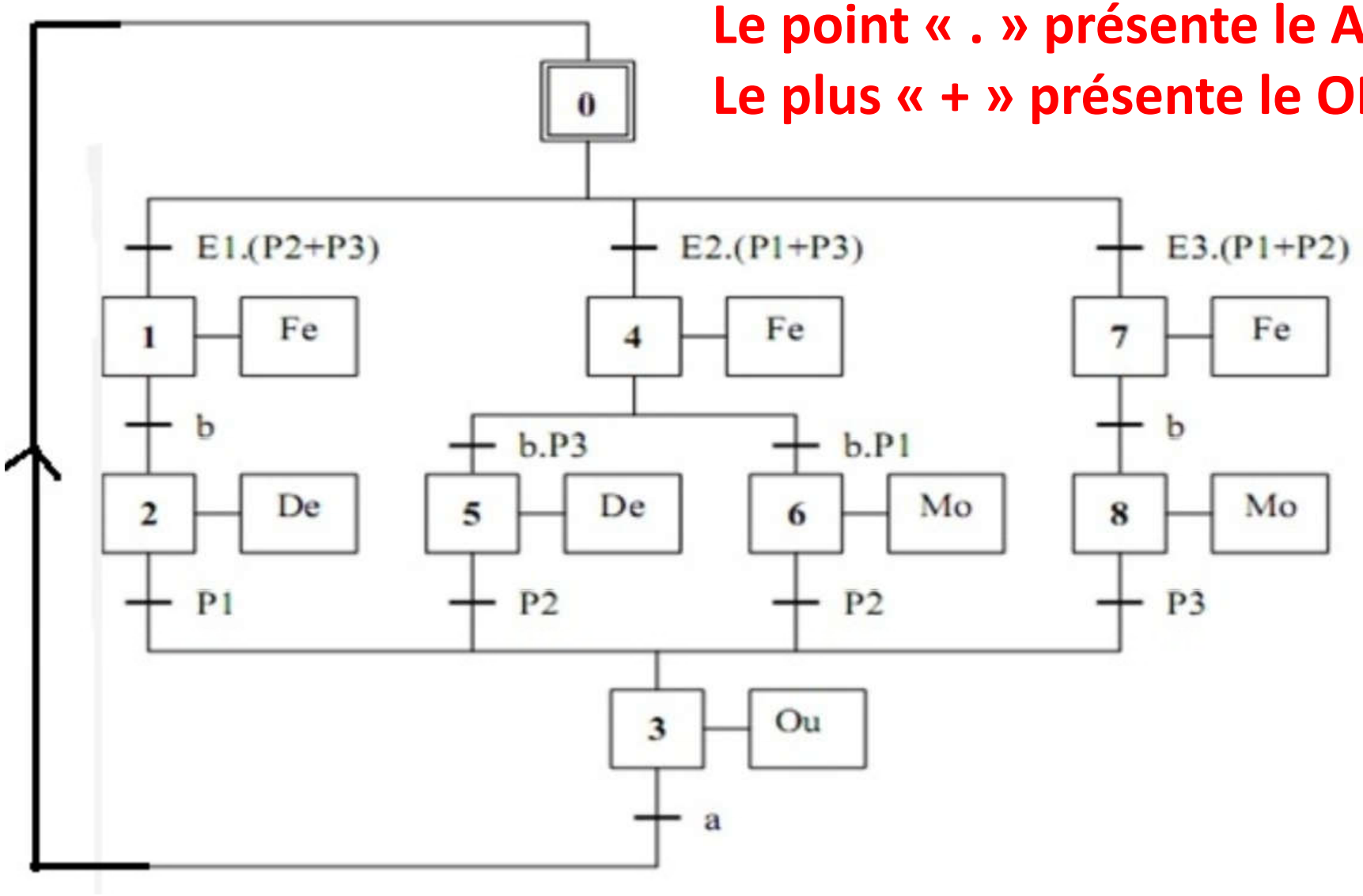


Exercices de TD

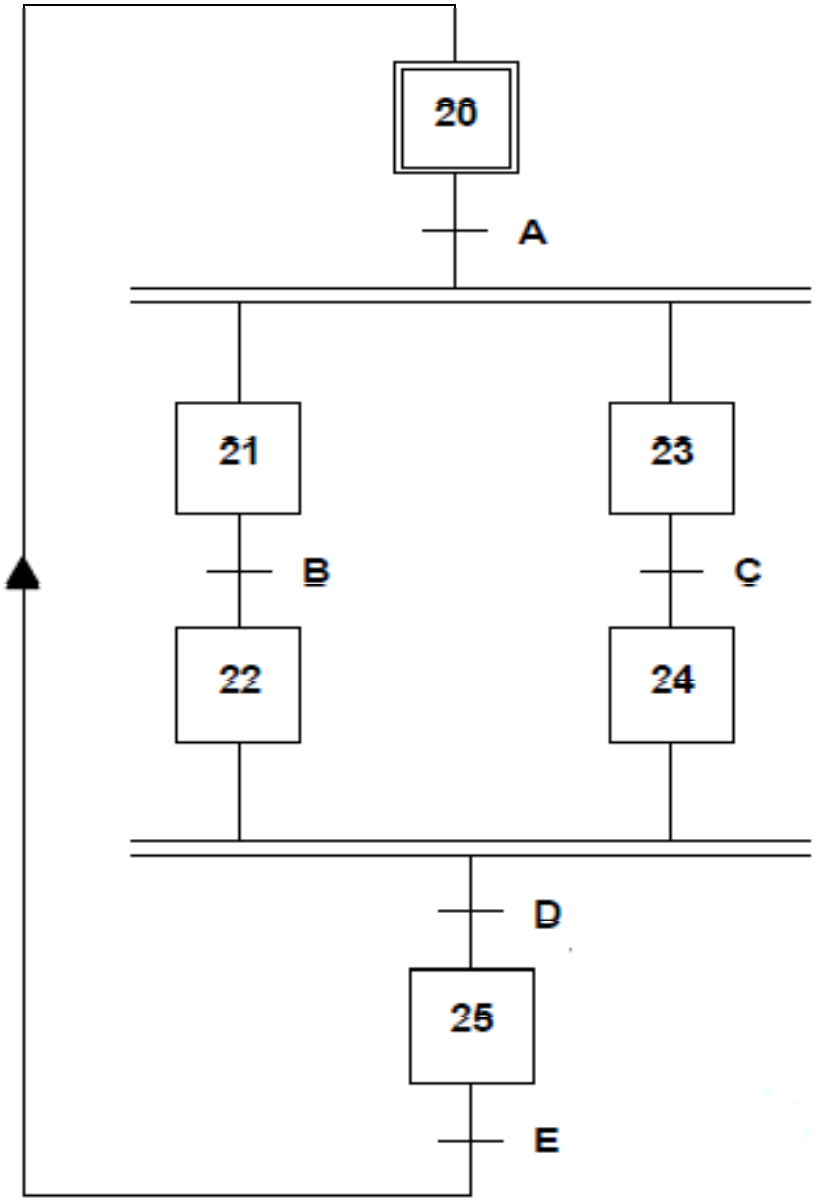
- Exercice 2: convertir les GRAFCET suivants en algorithme:

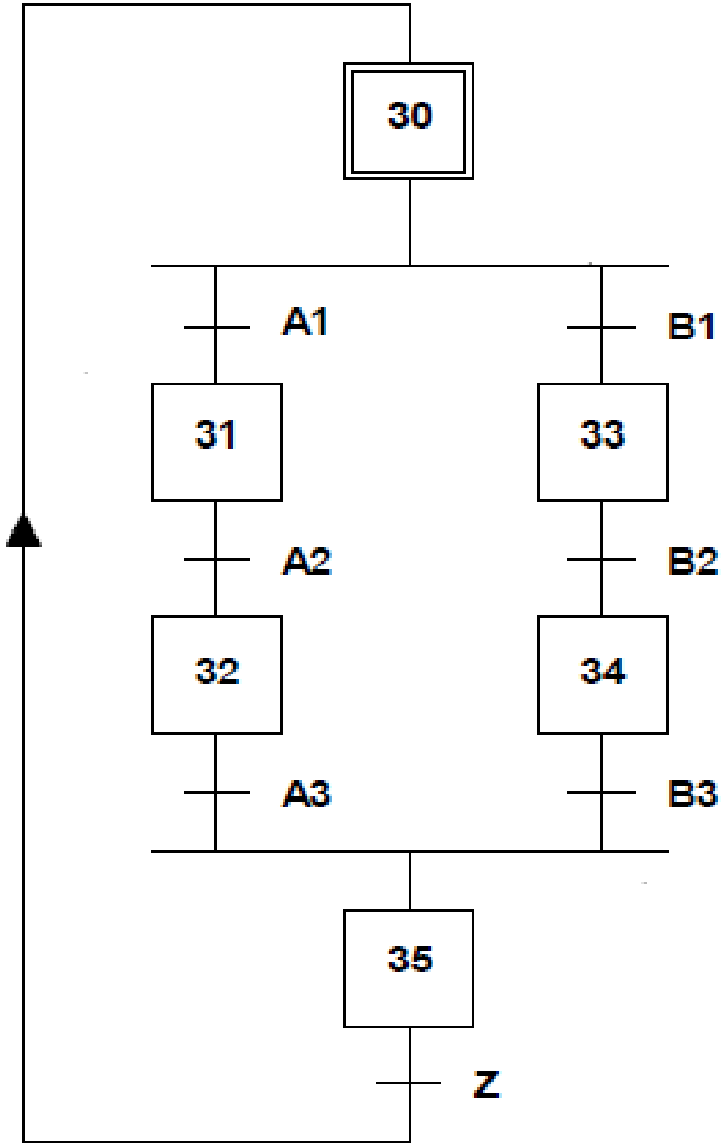
Le point « . » présente le AND

Le plus « + » présente le OR

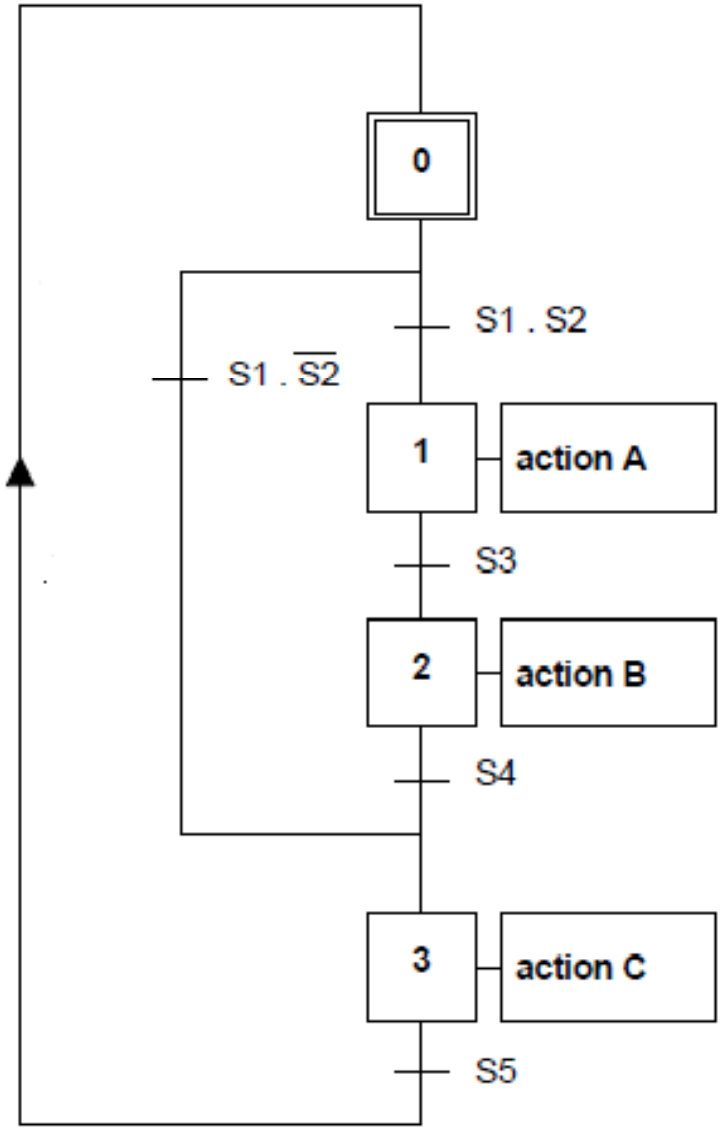


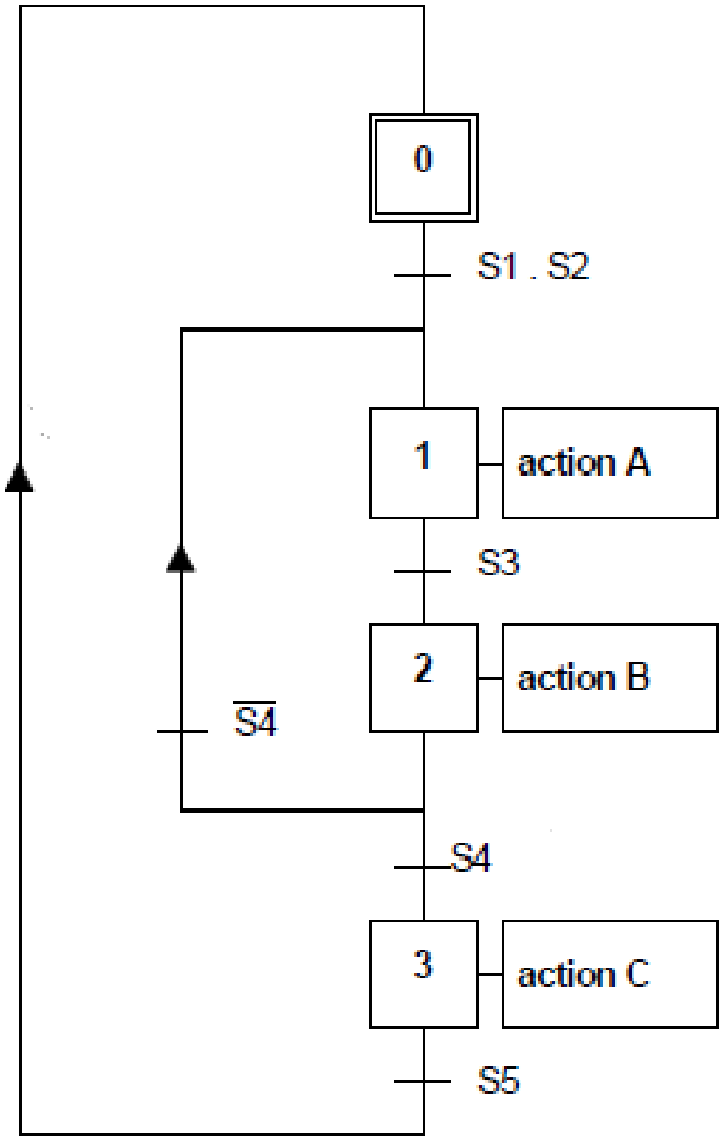
Exercices de TD





Exercices de TD





Annexe

CONVERSION DU GRAFCET EN LADDER

Conversion du GRAFCET en LADDER

- Il existe des automates programmables avec des outils de développement ne supporte pas le développement en GRAFCET, mais seulement le LADDER. Donc c’est très intéressant de connaitre comment faire la conversion du GRAFCET en LADDER surtout pour les processus purement séquentiel (un ensemble d’étapes).

Conversion du GRAFCET en LADDER

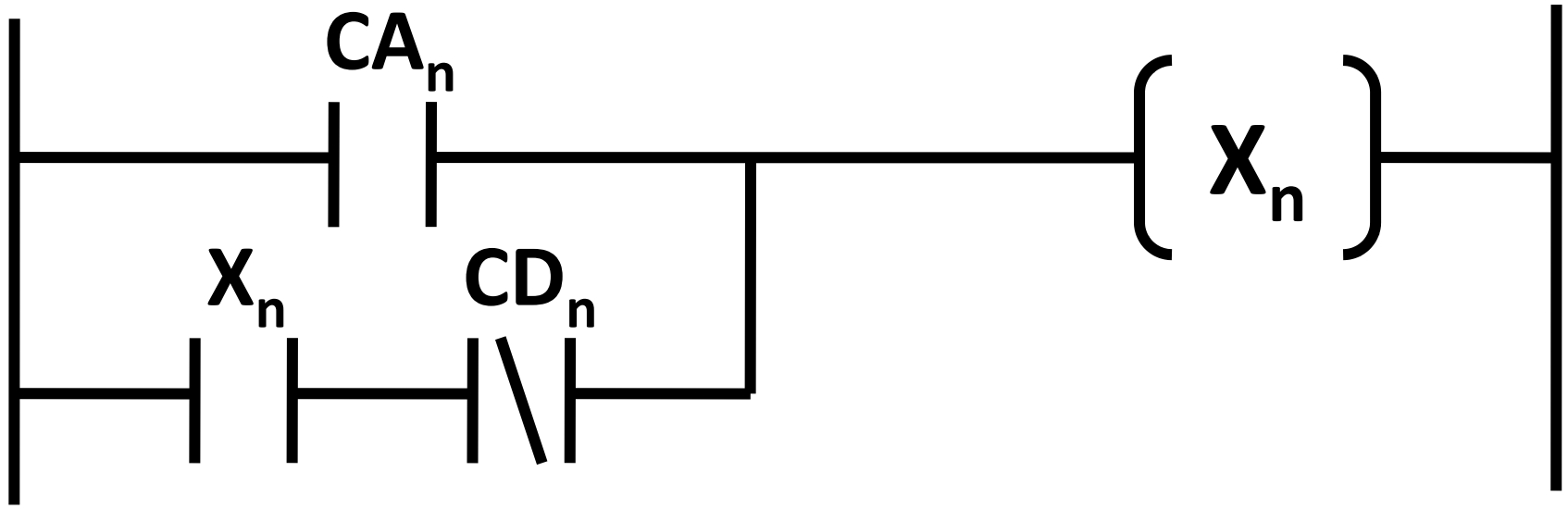
- Les **étapes** de GRAFCET peuvent être considérées comme des **fonctions mémoires**. Elles disposent d'une **condition d'activation** (**CA**) et d'une **condition de désactivation** (**CD**).
- Condition d'activation : une étape est activée si l'étape immédiatement précédente est active ET que la transition associée est vérifiée.
- Condition de désactivation : une étape sera désactivée si la condition d'activation de l'étape suivante est validée.

Conversion du GRAFCET en LADDER

- Si on note que X_n est la cellule mémoire de l'étape E_n, l'équation de chaque étape est la suivante:

$$X_n = CA_n + X_n \times \overline{CD_n}$$

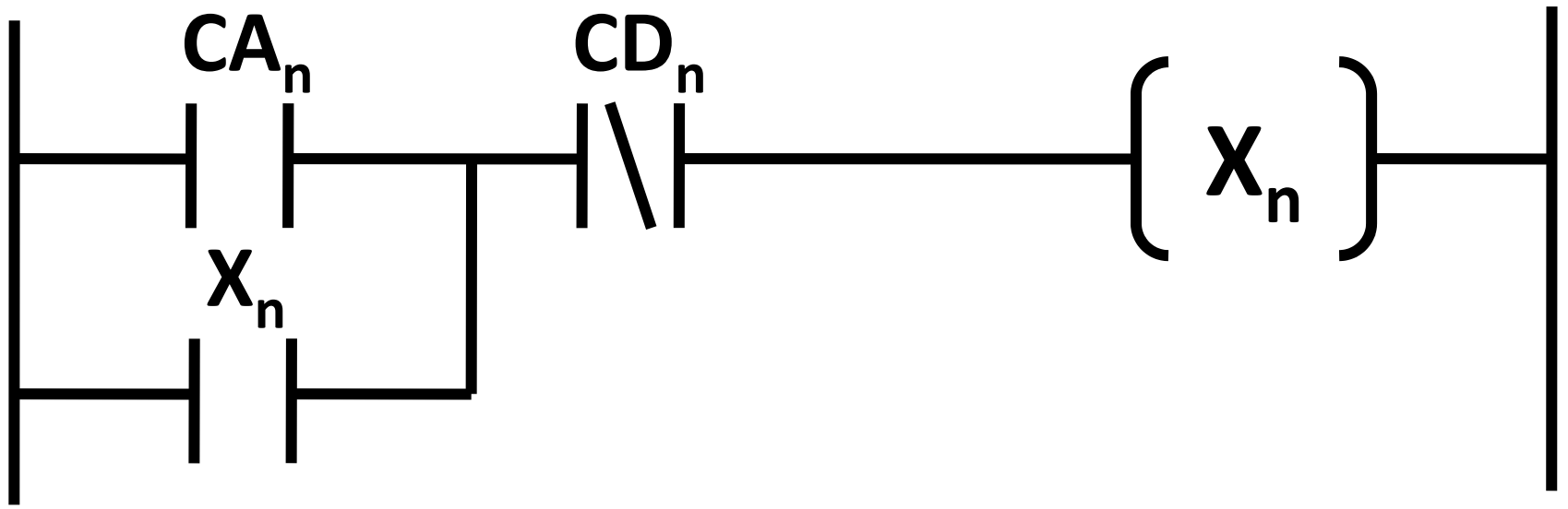
$$CD_n = X_{n+1}$$



Conversion du GRAFCET en LADDER

- Ou: $$X_n = (CA_n + X_n) \times \overline{CD_n}$$

$$CD_n = CA_{n+1}$$



Conversion du GRAFCET en LADDER

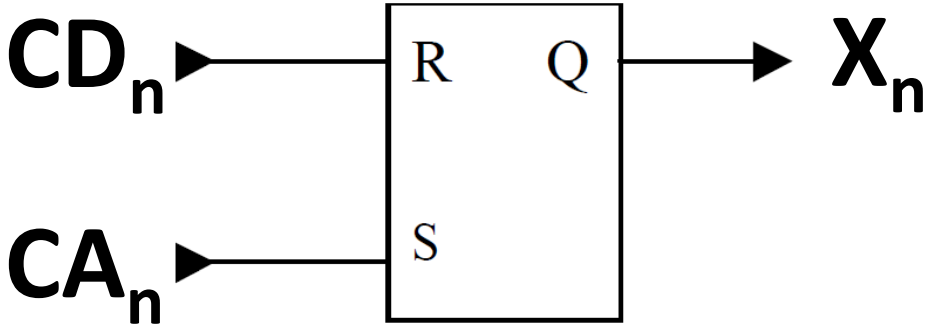
- **Quelle est la différence entre l'équation 1 et 2 ?**

$$X_n = CA_n + X_n \times \overline{CD_n}$$

$$X_n = (CA_n + X_n) \times \overline{CD_n}$$

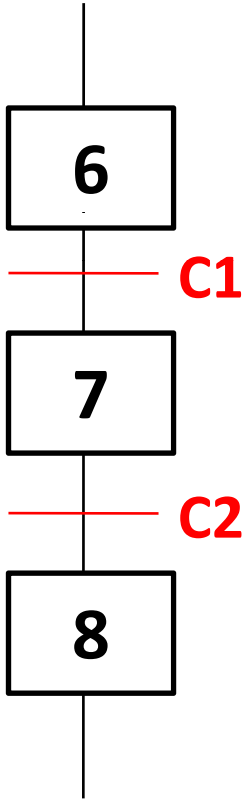
Conversion du GRAFCET en LADDER

- Pour simplifier l'opération de la conversion, on préfère d'utiliser les bascules SR par ce qu'elle présente l'élément mémoire de base dans la logique séquentiel, avec la condition d'activation (CA) est alors câblée à l'entrée S de la bascule et la condition de désactivation (CD) est câblée à l'entrée R comme indique la figure suivante:



Conversion du GRAFCET en LADDER

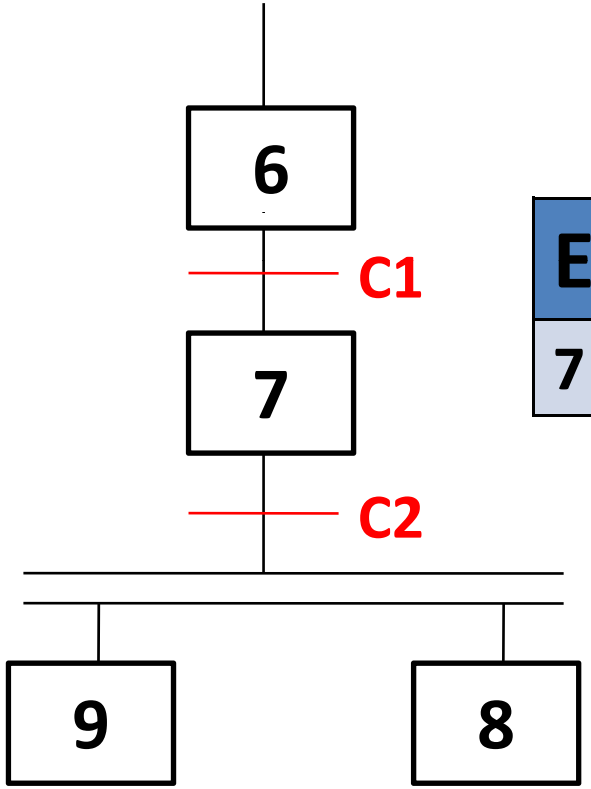
- Exemple 1 :



Etape	CA	CD
7	X6 and C1	X8

Conversion du GRAFCET en LADDER

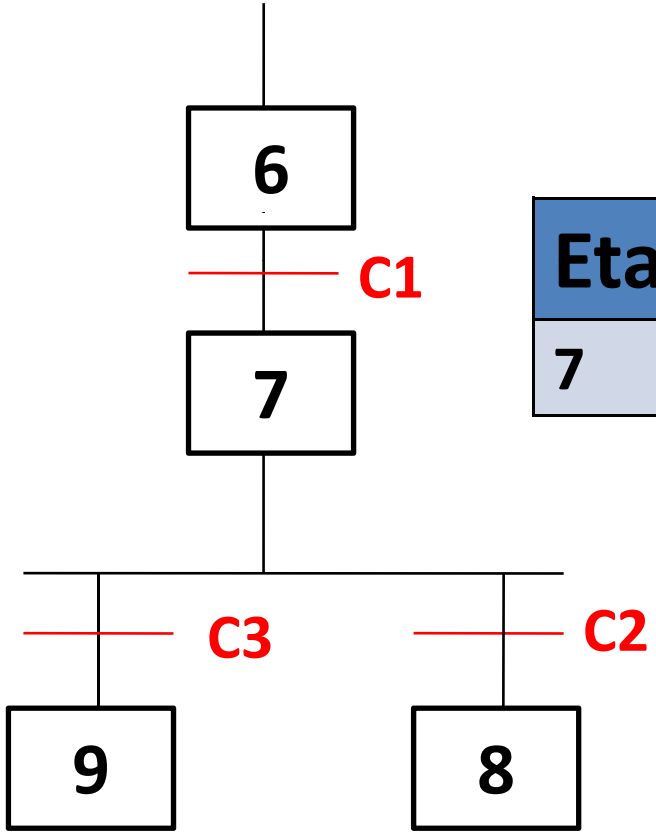
- Exemple 2 :



Etape	CA	CD
7	X6 and C1	X8 and X9

Conversion du GRAFCET en LADDER

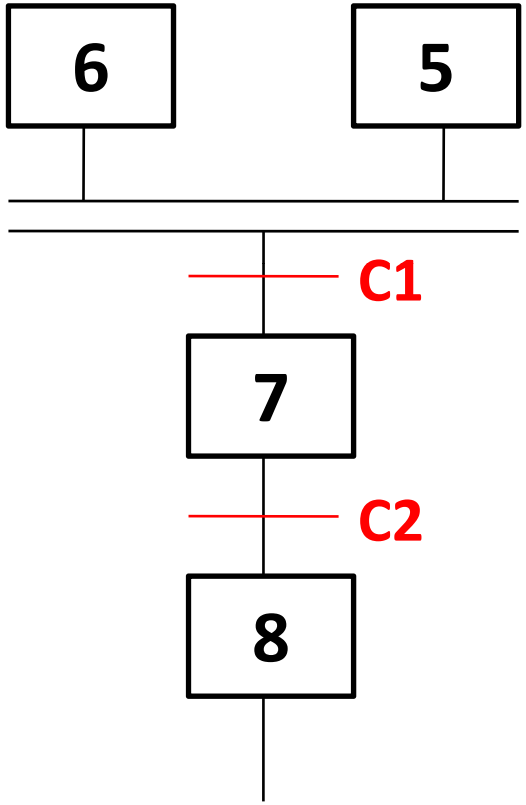
- Exemple 3 :



Etape	CA	CD
7	X6 and C1	X8 or X9

Conversion du GRAFCET en LADDER

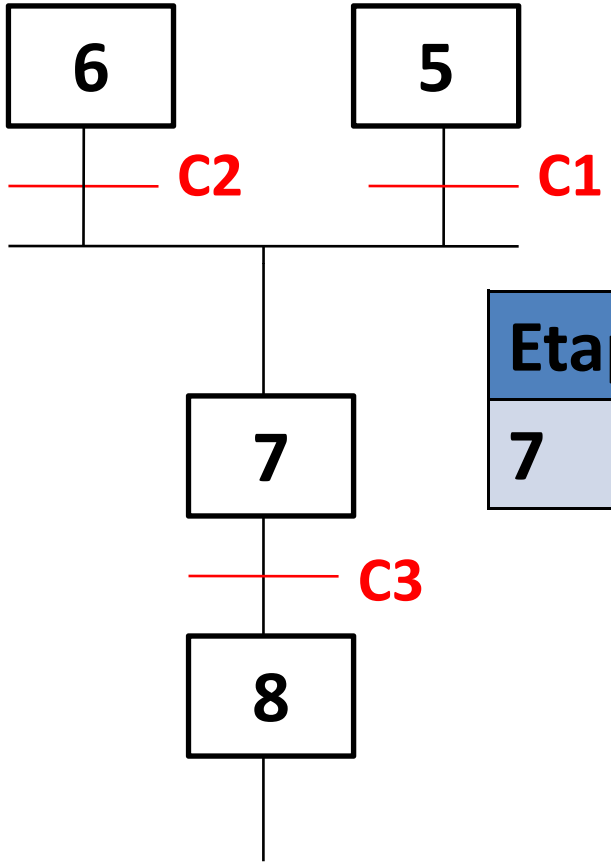
- Exemple 4 :



Etape	CA	CD
7	X5 and X6 and C1	X8

Conversion du GRAFCET en LADDER

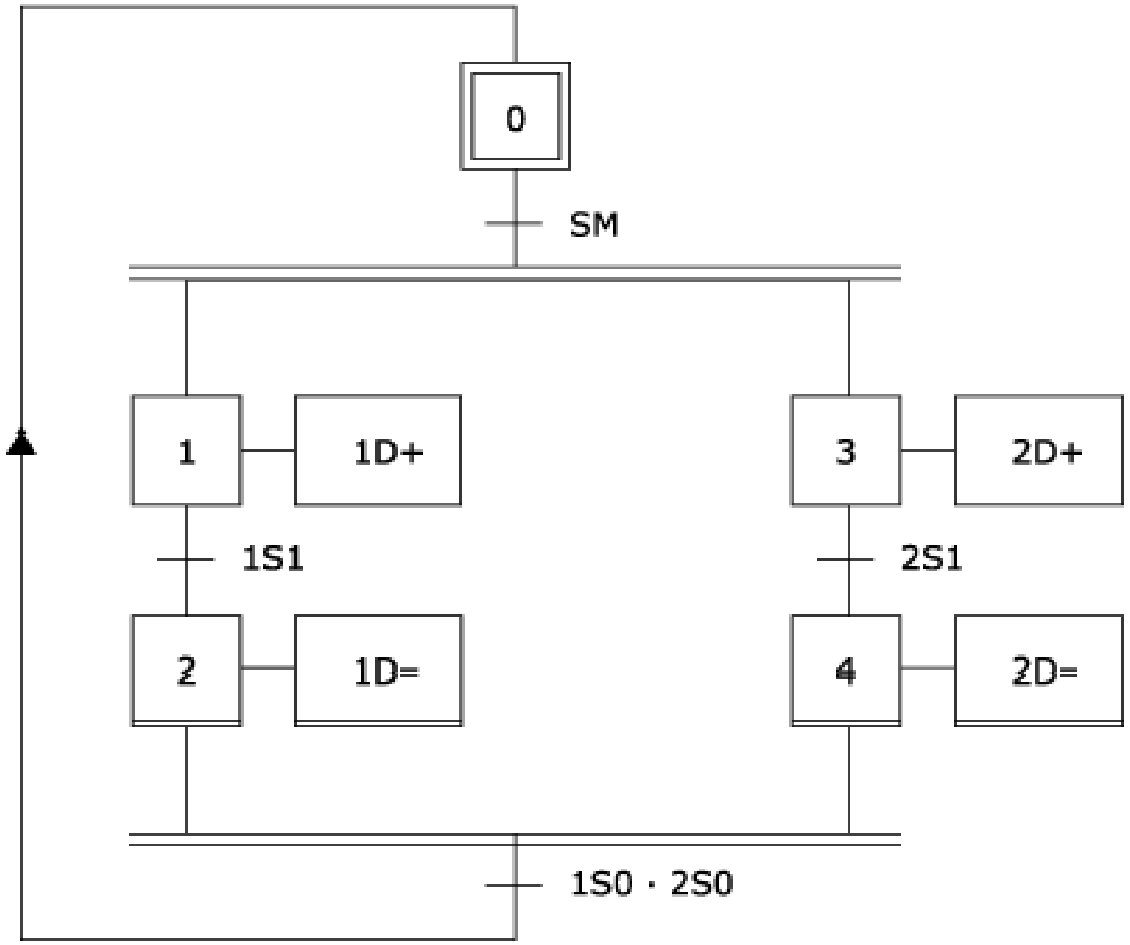
- Exemple 5 :



Etape	CA	CD
7	(X5 and C1) or (X6 and C2)	X8

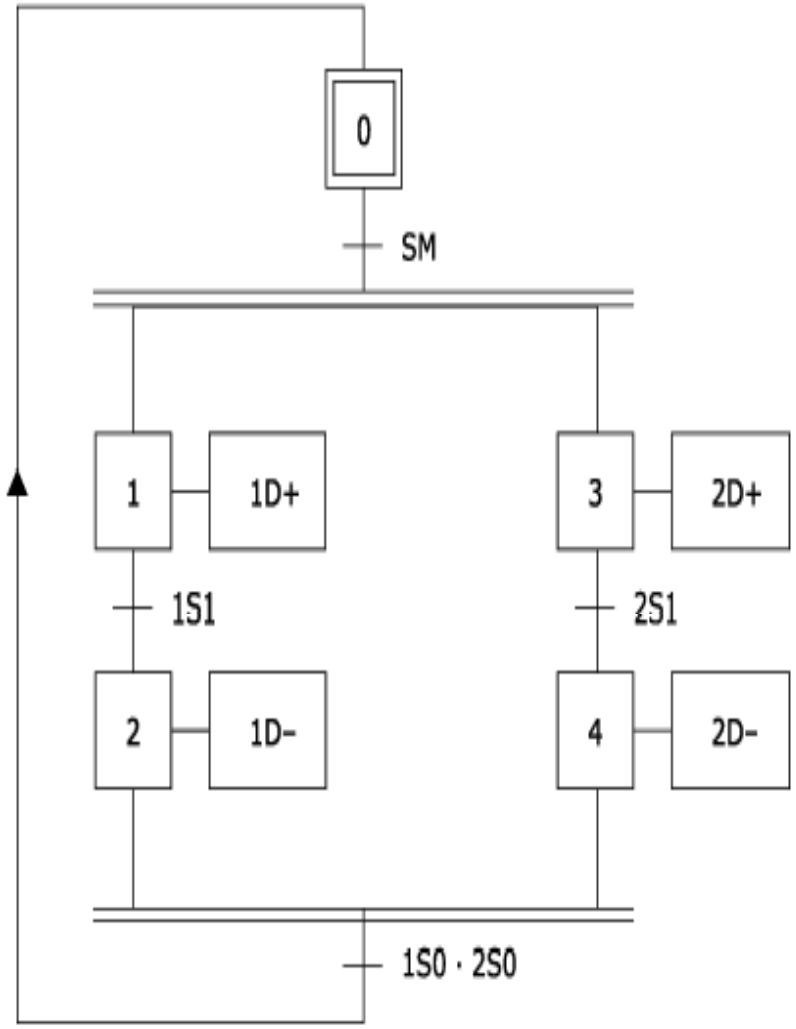
Conversion du GRAFCET en LADDER

- Exemple : Convertir le GRAFCET suivant en LADDER.



Conversion du GRAFCET en LADDER

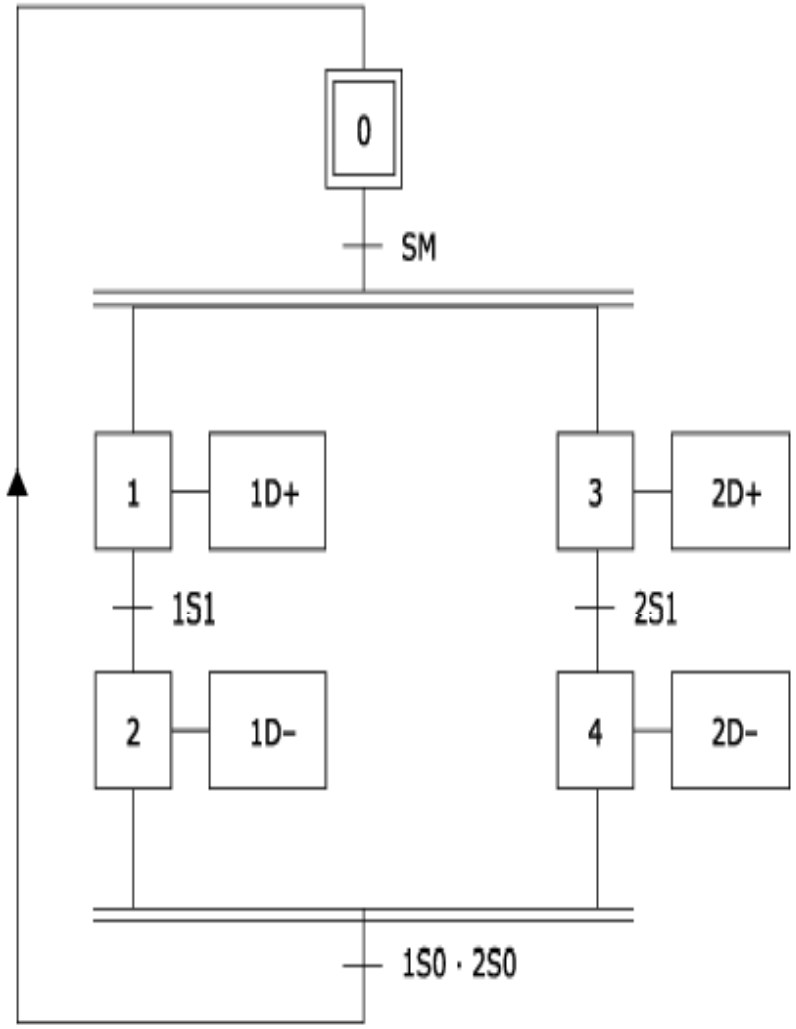
- La première étape : extraire les différentes équations des conditions d'activation et de désactivation pour chaque étape



Etape	CA	CD
X0	X2 and X4 and 1S0 and 2S0	X1 and X3
X1	X0 and SM	X2
X2	X1 and 1S1	X0
X3	X0 and SM	X4
X4	X3 and 2S1	X0

Conversion du GRAFCET en LADDER

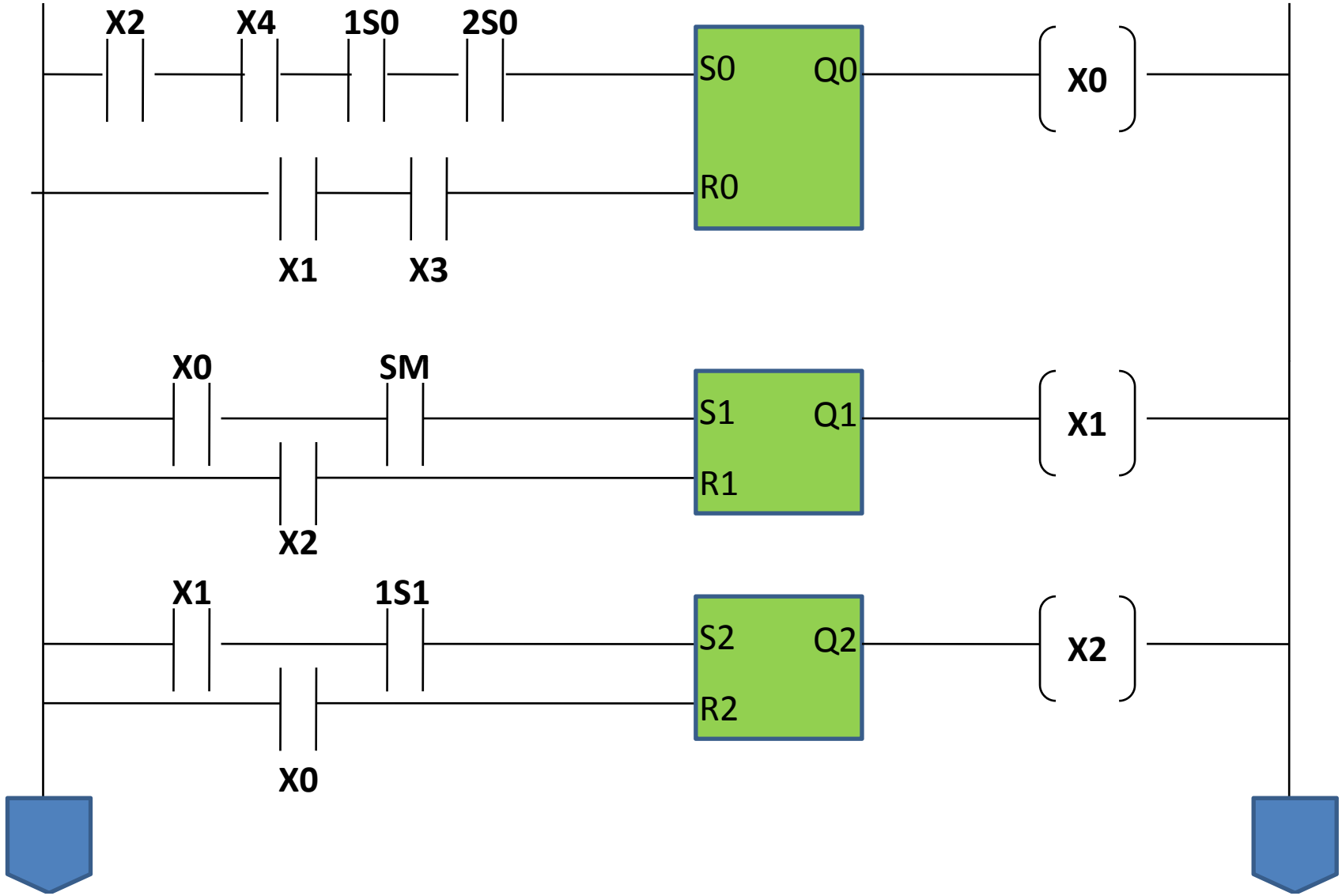
- La deuxième étape : extraire les différentes équations des sorties en fonction des X_n



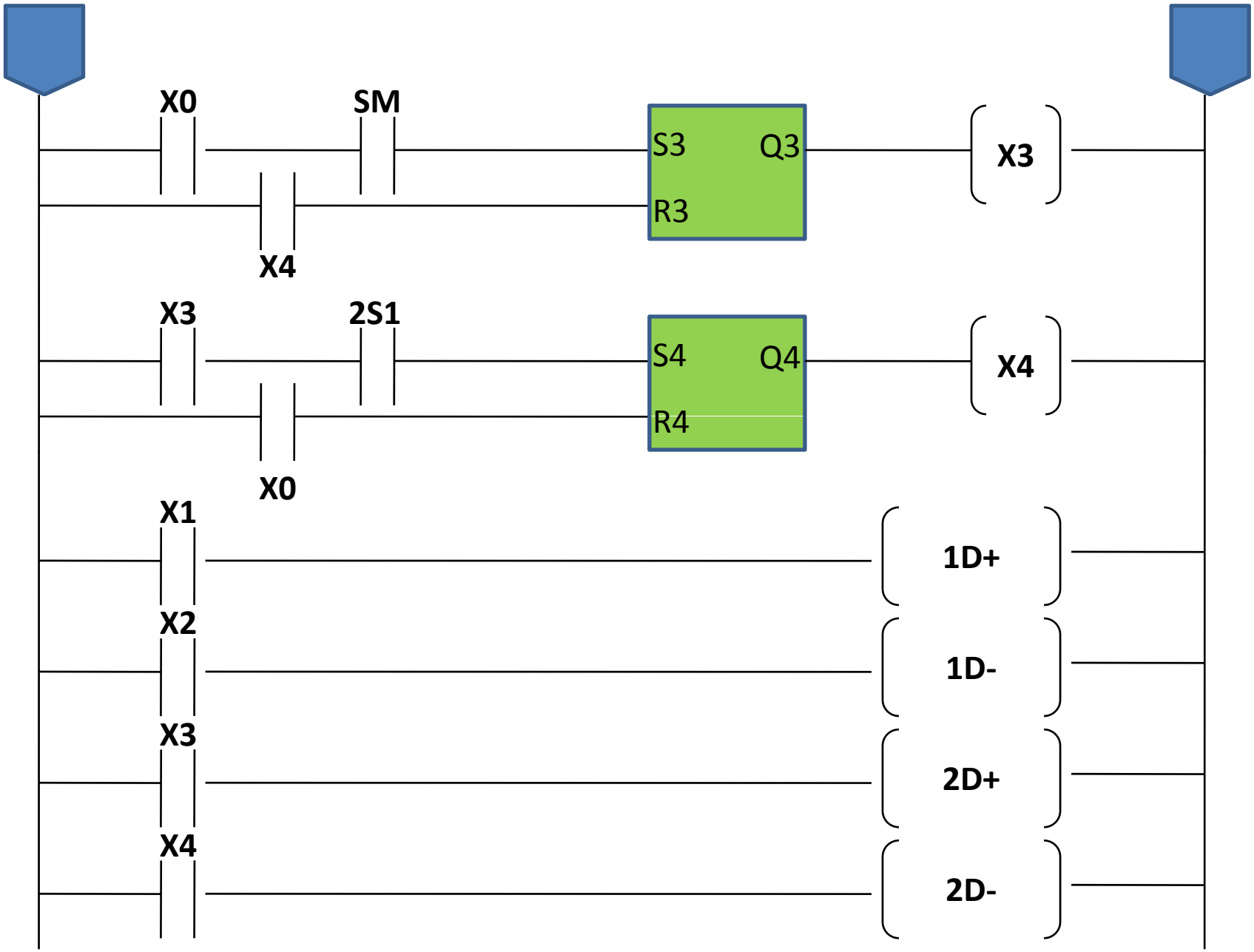
Sortie	Équation
1D+	X1
1D-	X2
2D+	X3
2D-	X4

Conversion du GRAFCET en LADDER

- Troisième étape : réaliser le LADDER.



Conversion du GRAFCET en LADDER



Exercices

- Convertir le premier GRAFCET de l'exercice 2:

Le point « . » présente le AND
Le plus « + » présente le OR