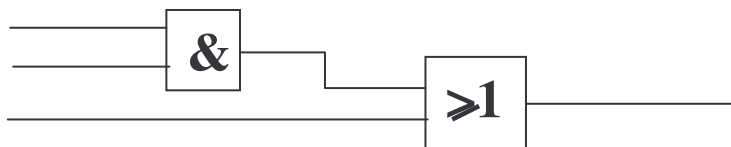


L'AUTOMATE PROGRAMMABLE A.P.I (T.S.X 17)

1°) Les A.P.I et les systèmes automatisés

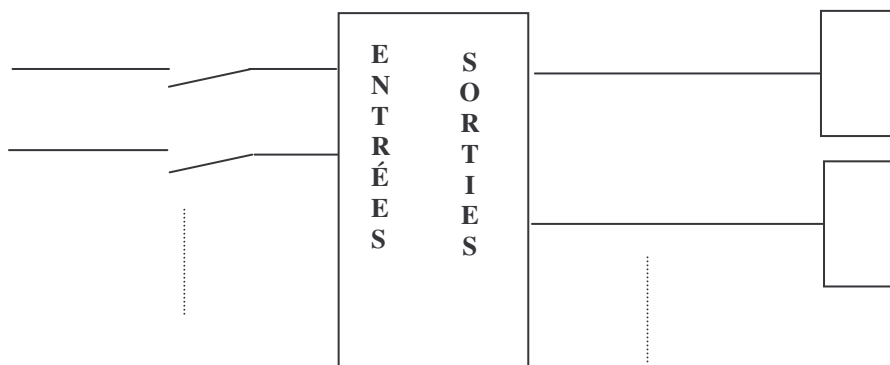
- Les premiers automatismes réalisés, l'étaient à l'aide de circuits à portes logiques (ET, OU, NAND, ...). Ces circuits étaient fragiles et non modulables, donc non adaptés à de petites modifications car il fallait tous revoir d'où du temps et une énorme perte de production



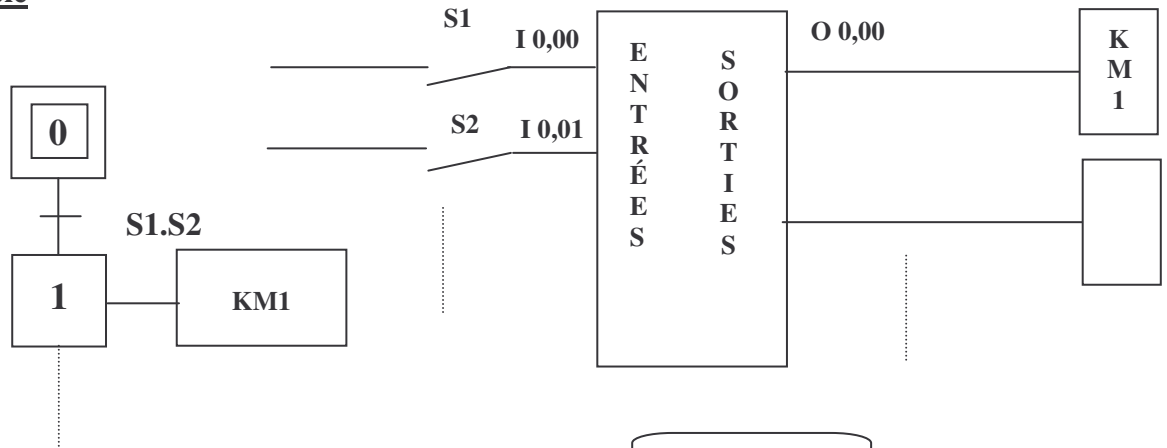
- Au fil du progrès, les automates ont vu le jour, ce qui modifia le traitement des informations, réduit les cabines de câblage et rendit les systèmes plus flexibles.
En fait les automates reprennent le fonctionnement des portes logiques mais maintenant elles sont programmables et réduites à un faibles encombrement.

2°) Comment fonctionne un A.P.I.

- Un API est comme un cerveau, il reçoit des informations de ces capteurs, les compare à son programme et active ou pas des sorties.

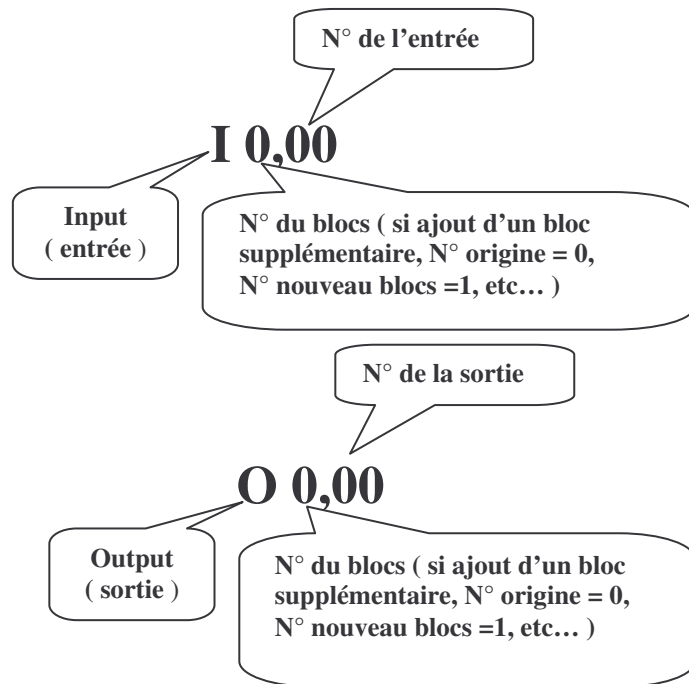


3°) Exemple

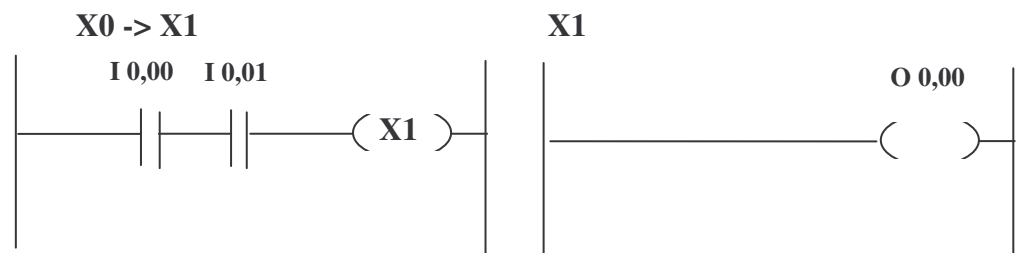


- Traduction pour l'API :

- S1 = I 0,00
 - S2 = I 0,01
 - KM1 = O 0,00



- Programmation de l'A.P.I. (ici en séquentiel)




4°) En résumé :


- On programme l'A.P.I d'après un grafcet séquentiel de fonctionnement, on rattache des capteurs à des entrées API nommées I X,XX et des relais, contacteurs à des sorties API nommées O X,XX.

5°) Le langage A.P.I (TSX 17).

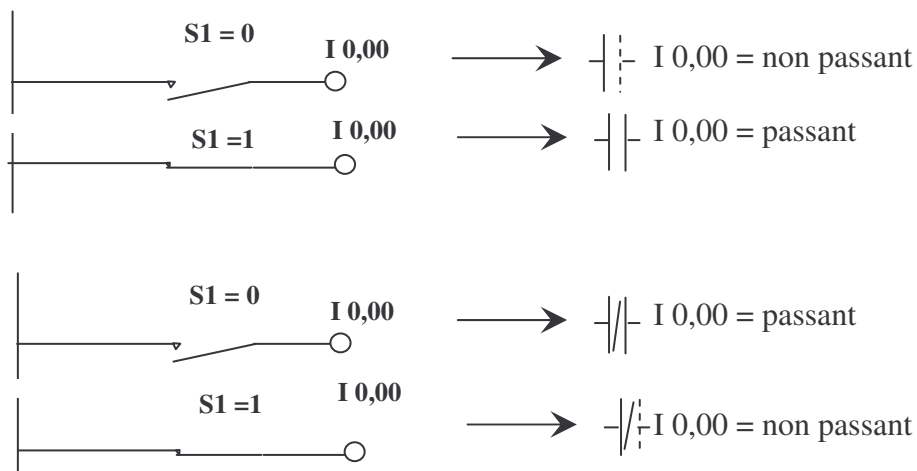
5.1°) Les entrées :

- Il existe 2 types d'entrées appelé « test » pour l'API :

les test directes :  qui s'active si l'entrée est active.

les tests indirectes :  qui s'active si l'entrée n'est pas active (idem cellule inverse).

- Exemple :



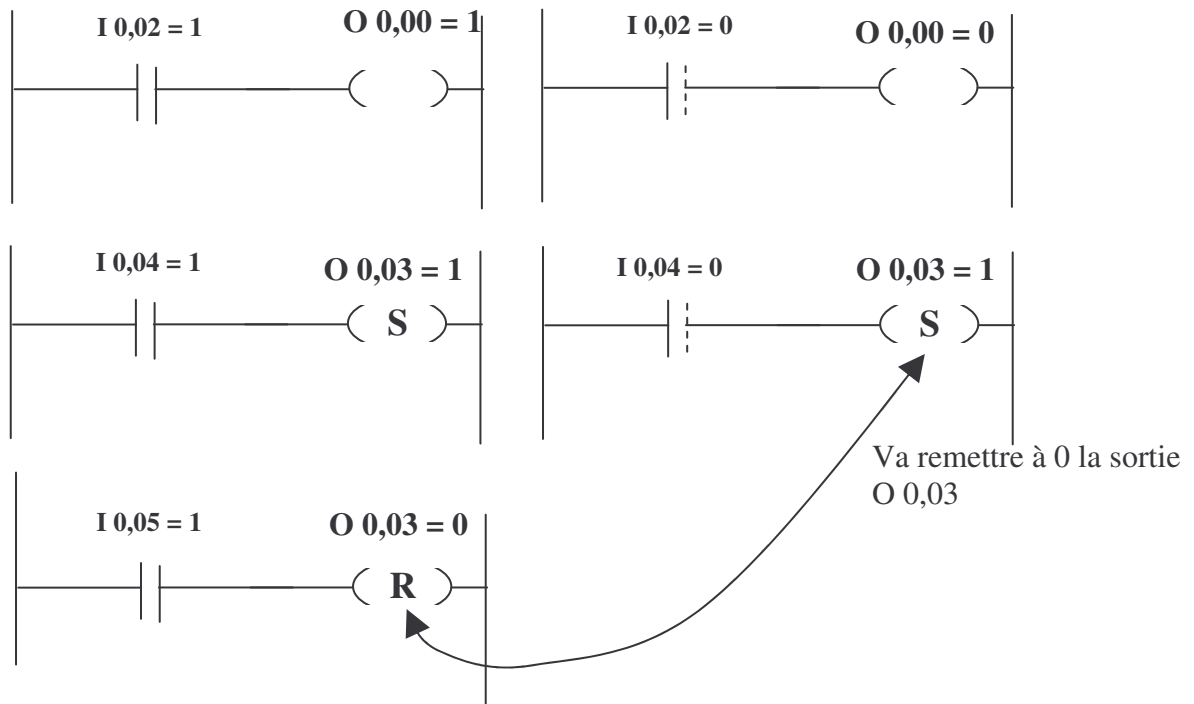
5.2°) Les sorties :

- Il existe plusieurs types de sorties pour l'API, mais nous n'allons en étudier que 3 sortes :

- Sortie directe : $-(\quad)-$ qui fonctionne comme l'entrée direct.
- Sortie SET (S) : $-(\text{ S })-$ qui agit comme une mémoire et reste à 1.
- Sortie RESET (R) : $-(\text{ R })-$ qui remet à zéro la sortie S.

Nota : Les sorties S et R sont surtout utilisées pour les préactionneurs monostables ou des information à mémoriser (auto-maintien de contacteur à maintenir activé pour ne pas qu'il retombe à l'état repos et pour la **programmation des étapes du grafcet en mode séquentiel**.

5.3°) exemple :



6°) La programmation :

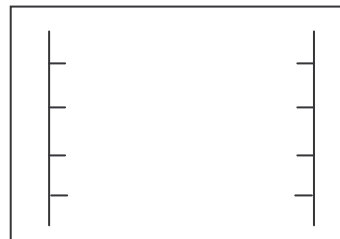
- il existe 2 modes de programmations ; le mode **LADDER** et le mode **SEQUENTIEL**

- La principale différence réside dans l'ordre d'exécution, en **LADDER**, il n'y a pas d'ordre dans les **LABELS**, le label 102 peut s'exécuter avant le label 1 ! donc il n'est pas ou mal adapté à une programmation par séquence (type grafcet) ; contrairement au séquentiel qui lui suit directement une programmation grafcet.

6.1°) Le mode de programmation LADDER.

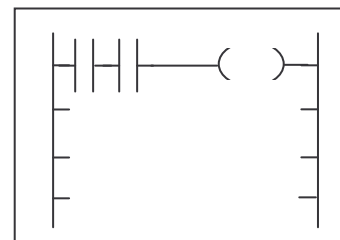
- Le **LADDER** ou *langage à contact* se programme par ligne avec les entrées (test) et les sorties sur les mêmes pages de programmations.
- Chaque page se nomme un LABEL, ils vont de 0 à 999, chaque label comprend 4 lignes de programmations, il faut le nommer pour que l'A.P.I. le reconnaisse comme ligne de programmation. Pour le nommer, il faut utiliser la touche « LAB » et le numéroter de 0 à 999.
- les labels **ne comportent pas d'ordre d'exécution**, c'est-à-dire que le label 50 peut s'exécuter avant le label 1 ; on ne peut pas programmer suivant un ordre séquentiel (suivant un grafcet) car *une entrée peut enclencher plusieurs sorties si elle est utilisée dans plusieurs labels.*

- L'écran se compose de 4 lignes de programmation :




- Sur ces lignes on peut programmer des entrées et des sorties pour former une ligne de programme

- Les sorties peuvent être des bobines ou des compteurs, des temporisations , ...



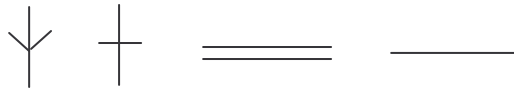
6.2°) Le mode de programmation SEQUENTIEL .:

- Dans le mode SEQUENTIEL, on trouve **3** parties :
 - Le préliminaire ou « **PRE** » qui s'occupe de la *gestion des mises en marche* après arrêts désirés ou non (arrêts d'urgences, ...)
 - Le séquentiel ou « **SEQ** » dans lequel on va écrire la structure ou « squelette » du grafcet et où l'on peut aussi *programmer ce grafcet* (à condition qu'il ne soit pas trop important : 4 lignes de programme par étape et uniquement en sortie **SET** et **RESET**). .
 - Le postérieure ou « **POS** » dans lequel on programme comme en langage *LADDER* mais où l'on peut utiliser le grafcet pour ordonner les *LABELS* (en incluant des tests directs appartenants aux bits d'étapes, exemple :  X11); chose impossible en mode *LADDER*.

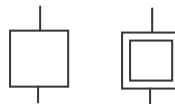
- **6.2.1°) Le séquentiel :**

- La première chose à réaliser est la création de la structure du grafcet, pour cela vous devez vous *positionner dans le mode séquentiel*. Tout d'abord vous devez *initialiser l'automate* en mode « **SEQUENTIEL** ».
- Vous arrivez ensuite dans l'écran de création du grafcet, cet écran est en fait une partie de page composer de 14 lignes horizontales et 8 colonnes. Chaque lignes comprend différents symboles et chaque ligne à ses symboles :

- Lignes paires : renvoies, transitions, lignes, et, ou,

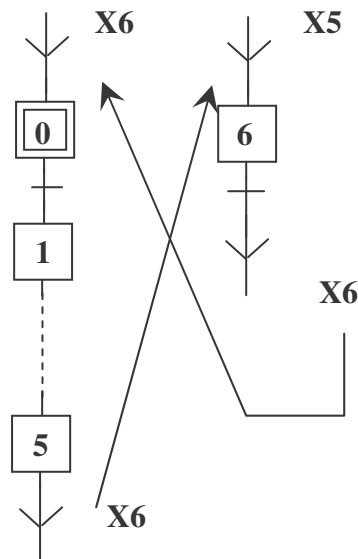


- Lignes impaires : étapes,



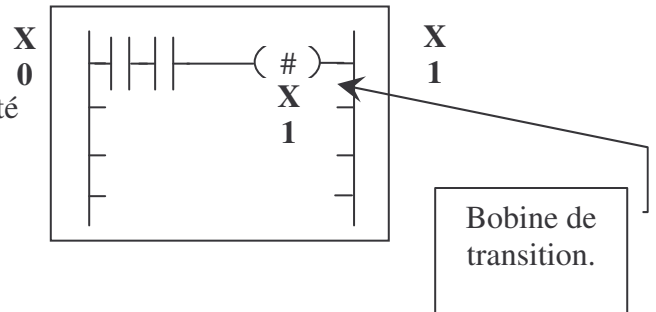
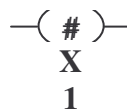
- Vous rentrer ensuite le grafcet en utilisant les renvoies s'il dépasse les 5 étapes (voir ci-dessous).

« d'où je viens, ou je vais »



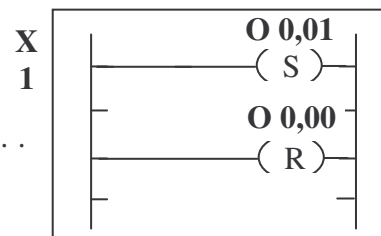
- Une fois le grafcet entré dans l'A.P.I., vous pouvez le programmer à condition que le programme ne soit pas très conséquent (pas plus que **4 lignes**). Pour cela vous devez positionner le curseur devant l'étape à programmer à l'aide des *touches de direction* et « **ZOOMER** » dans l'étape ou la transition choisie à l'aide de la touche « **ZOOM** ».
- Vous pouvez ensuite rentrer votre programme mais il faut savoir que les seules « **sorties** » disponibles sont de la forme « **SET** » et « **RESET** ». *Il faut donc penser à toujours « riseter » une étape après l'avoir « seter » car sinon le programme se bloque : deux bobines de distributeur ne peuvent et ne doivent pas êtres enclenchées en même temps.*
- La programmation s'effectue donc sur des étapes et sur des transitions mais les écrans n'ont pas les mêmes fonctions :
- Les transitions : représentation des entrées.

Elles ne contiennent pas de sorties, car elle enclenche l'étape 1, représenté par :



- Les étapes : représentations des sorties

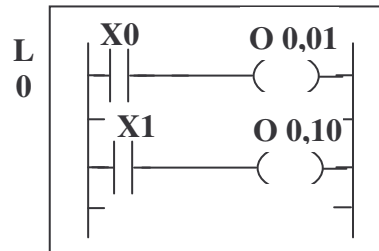
Elles ne contiennent pas, en règle générale d'entrées sauf sécurité ou bit de temporisation., de compteur,



Il ne faut pas oublier de « riseter » une bobine que l'on utilise plus

6.2.2°) Le postérieur :

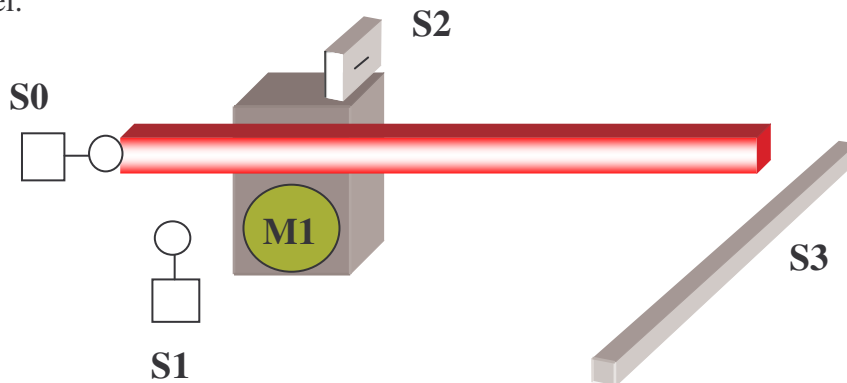
- Le postérieure se programme comme le **LADDER** mais on utilise les *bits interne d'étape* de l'automate pour pouvoir programmer d'après le grafcet. Chaque étape activée active un bit interne du même nom que l'étape, (étape X1 = bit X1). Ce bit est utilisé ensuite comme entrées permettant d'activé des sorties (voir exemple ci-dessous)



- Dans le postérieure comme dans le LADDER, toutes les sorties sont disponibles (set ,reste, direct, jump,...) .

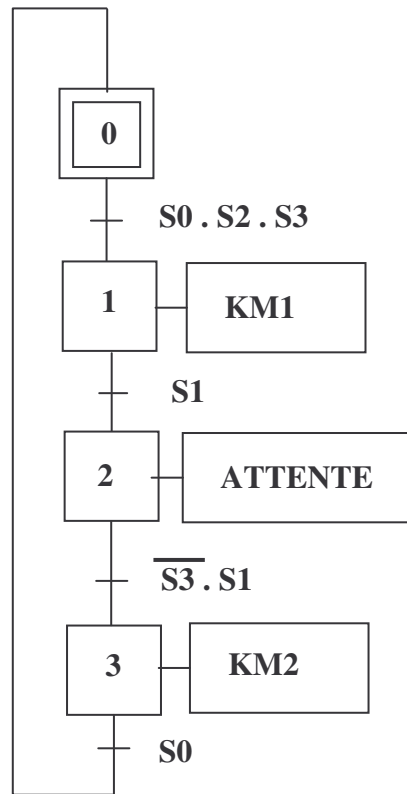
7°) Exercice :

- Nous allons réaliser une programmation étape par étape d'après un grafcet de fonctionnement séquentiel.



- Pour entrer dans un parking il faut une carte et une présence de véhicule ; puis la barrière se lève et lorsque la cellule s3 indique que le véhicule est passé, la barrière se baisse.

- GRAFCET :



7.1°) Identification du nombre d'entrées et de sorties

- Entrées = _____

- Sorties = _____

7.2 °) Traduction des entrées, sorties en langage API :

- S0 = I __ , __ __

- KM1 = O __ , __ __

- S1 = I __ , __ __

- KM2 = 0 __, __ __

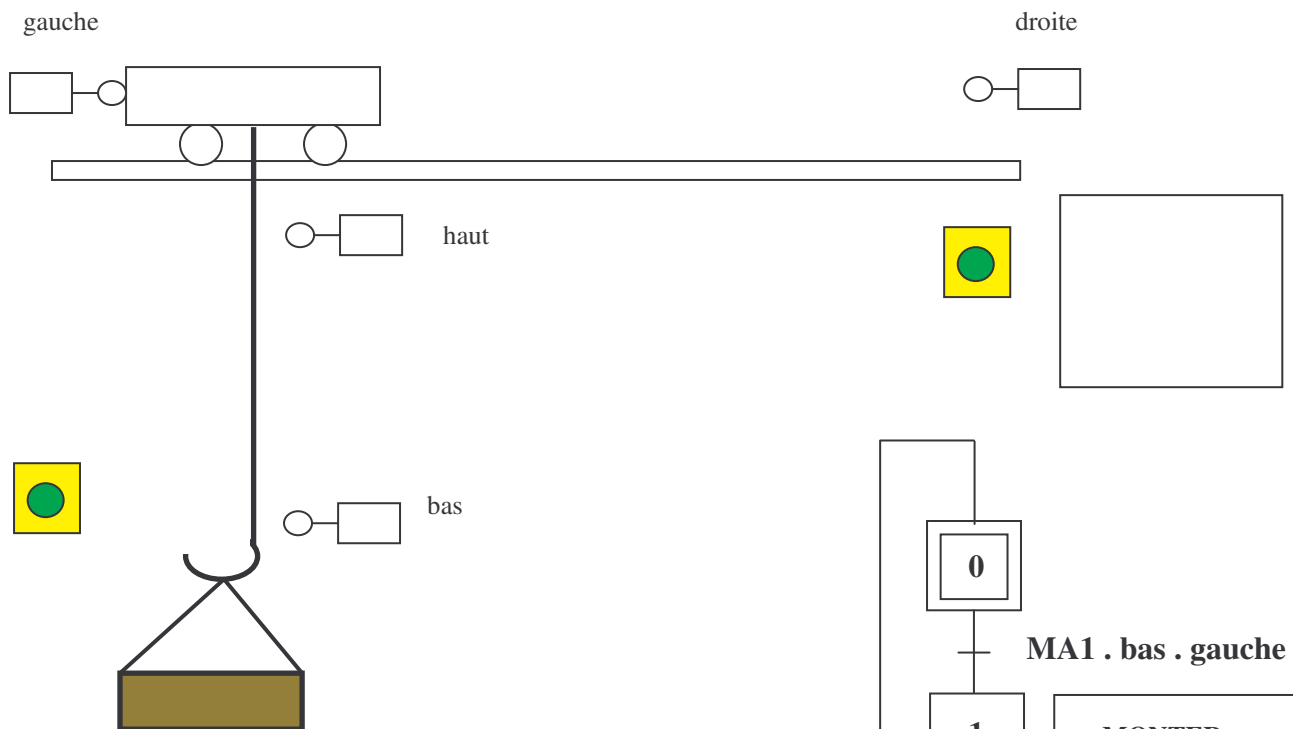
- S2 = I __ , __ __

- S3 = I __ , __ __

7.3°) Réaliser le programme en **LADDER** et indiquer vos constatations, puis réaliser ce même programme en **SEQUENTIEL** et indiquer vos constatations :

Nota : Pour cet exercices, nous allons utiliser des « **bits internes** » qui serviront de relais (ou de mémoire) pour différentier certaines parties du cycle et éviter des « nœuds » (répétition de deux actions au même moment).

- Le bit interne s'utilise comme une « sortie », mais elle n'occupe pas de place extérieure, c'est à dire qu'elle ne peut pas être câblée.

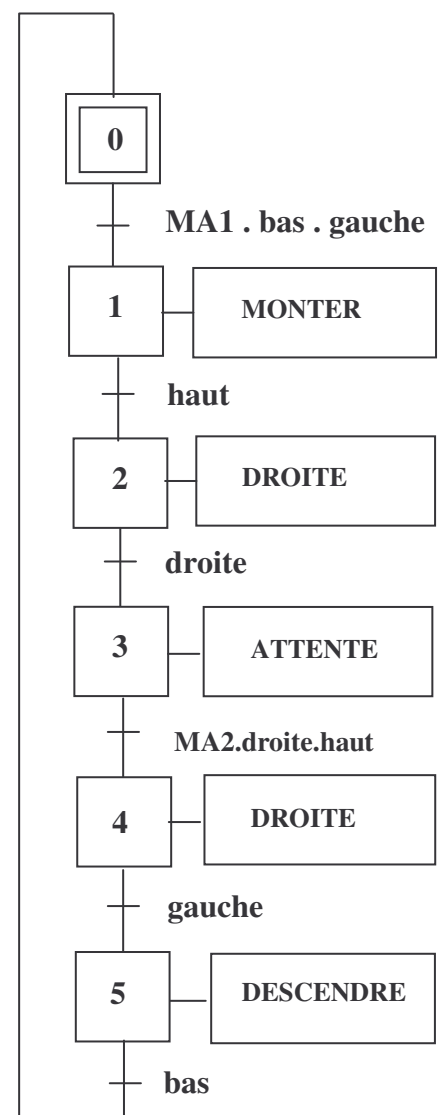
8°) Application**8.1°) LE PALAN :**

Ce palan fonctionne comme indiquer sur le grafcet ci-contre, une 1^{ère} impulsion sur le bouton MA1 fait monter la charge qui une fois en haut se dirige vers la droite, arrivée à droite tout s'arrête pour laisser un opérateur décharger la palette.

Une fois la palette déchargée, l'opérateur appuie sur bouton MA2 et refais partir la palette vers la gauche. Une fois à gauche, la palette redescend et s'immobilise en bas pour être de nouveau chargée et le cycle recommence.

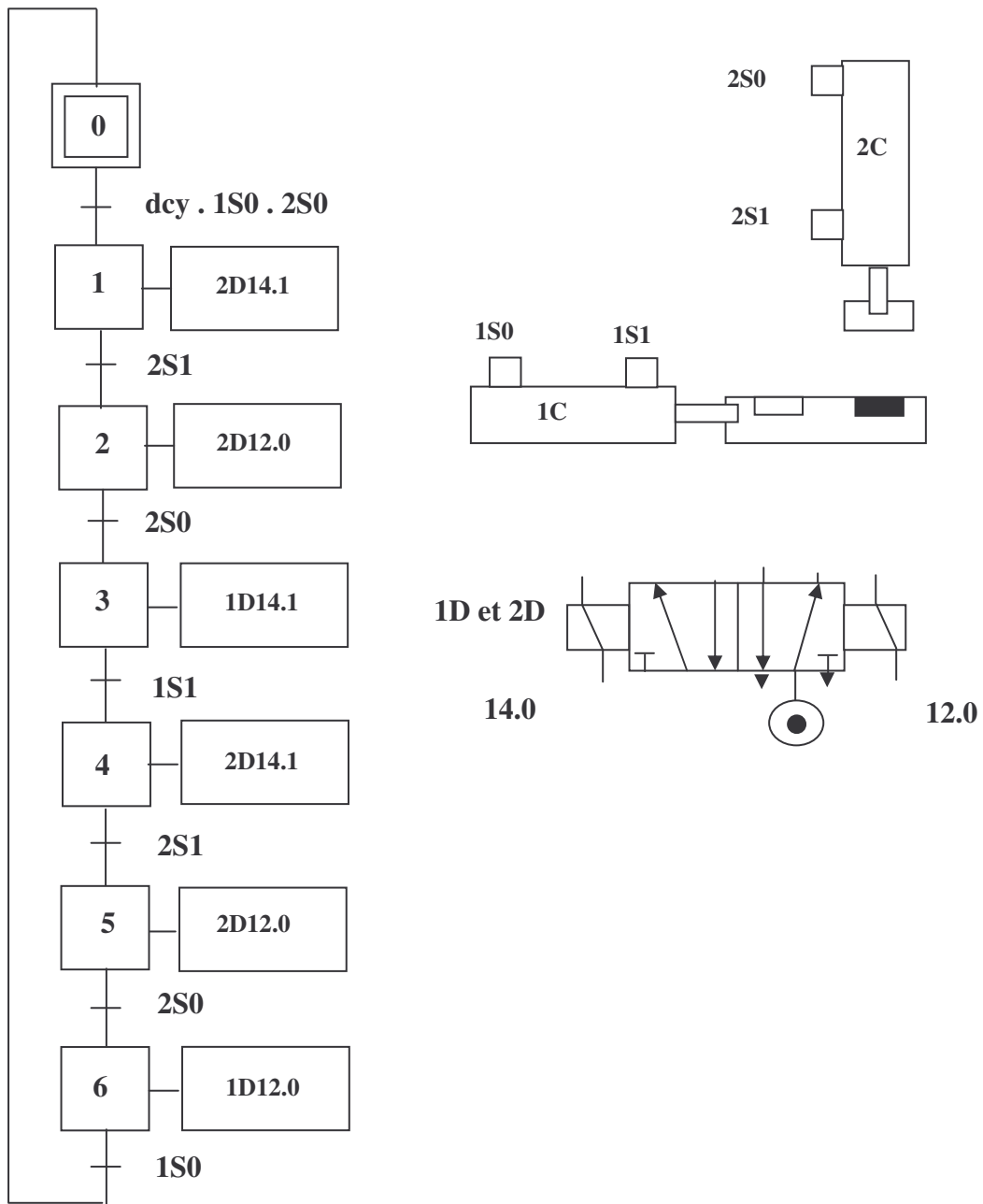
Travail demandé :

- 1°) Nommer toutes les ENTREES / SORTIES
- 2°) Ecrire le programme en **LADDER**.
- 3°) Programmer l'A.P.I
- 4°) Rajouter une temporisation.
- 5°) Rajouter un compteur.
- 6°) Réaliser la programmation en **SEQUENTIEL**.



- 8.2°) LA TABLE D'IMPRIMERIE :

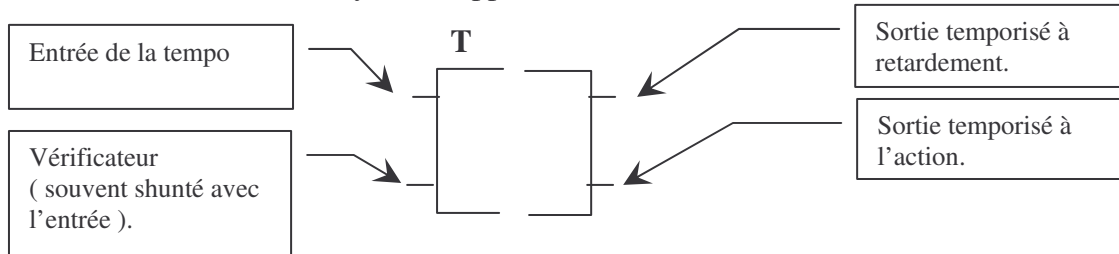
Une table d'imprimerie permet de réaliser des petites cartes de visites à la demande, il vous est demandé de réaliser le programme suivant le grafcet ci-dessous :



- 1°) En premier lieu il convient de dénommer chaque entrées et chaque sorties.
- 2°) Ecrire le programme sous la forme de « LABEL » sur le format ci-joint.
- 3°) Une fois validé par le formateur, vous pouvez essayer sur l'A.P.I en utilisant le manuel « programmation TSX 17 ».

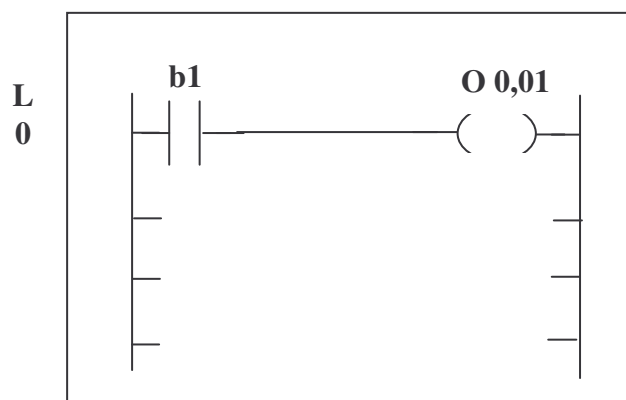
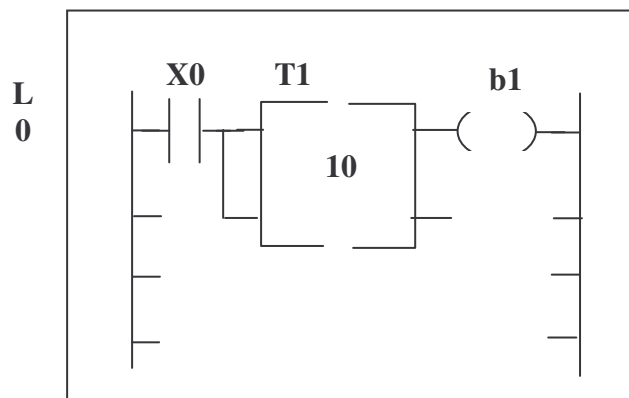
9°) Les temporisations :

- La temporisation est utilisée pour temporiser une action ou sortie ; elle peut être à l'action ou au retardement. Pour trouver le symbole d'une tempo, il faut appuyer sur l'icône [?], cet icône englobe les tempos mais aussi les compteurs, les comparateurs, Pour sélectionner la tempo, choisir l'icône [T] et alors le symbole apparaît.



- Pour programmer la tempo, il faut « zoomer » dans la tempo, et programmer le temps de base « TB » de 10 ms, 100 ms, 1 s, 1 mn et la durée « PRE » de 1 à 9999.

- Ensuite il faut choisir le contact de sortie suivant l'effet temporisé désiré, et le raccorder à la sortie ou au bit à activer.



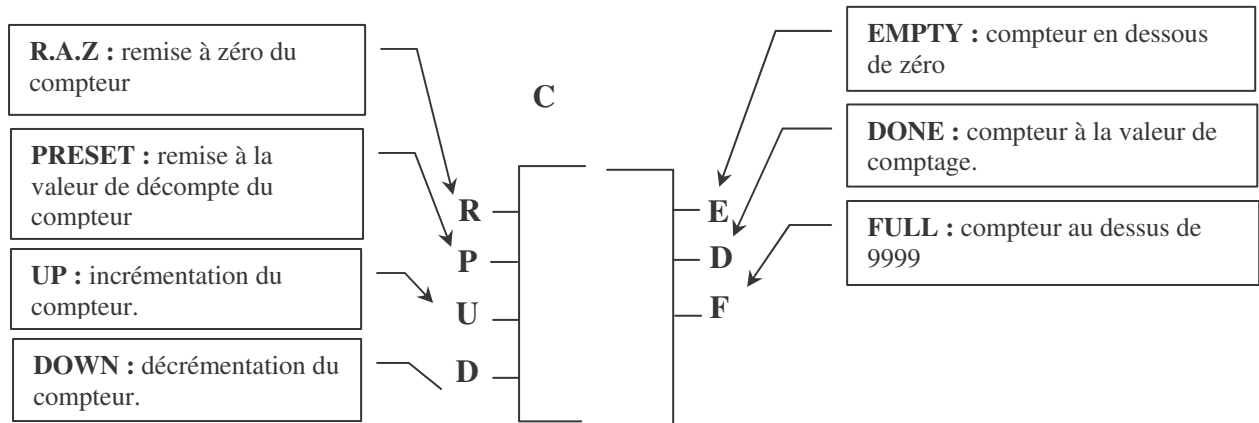
Nota : La temporisation peut être programmée dans le grafcet, si le **programme** n'est pas trop **conséquent** sinon il faudra le faire dans le postérieure.

10°) Exercices :

- Pour réalisé cet exercice, positionnez vous dans le postérieur.
- Créer un nouveau LABEL à la suite de ceux de l'exercice précédent, et utilisé comme test direct une entrée non utilisée ; puis allé sélectionner une tempo. L'API vous demandera de lui donner un numéro de 0 à 20, choisissez en un et valider.
- Raccordez à chacune des deux sorties de la tempo, une bobine de sortie direct non utilisée.
- Programmer la tempo (deux fois « ZOOM ») pour qu'elle compte en seconde (TB) jusqu'à 5 (PRESET).
- Que constatez-vous ?

11°) Le compteur :

- Le processus pour implanter un compteur est exactement le même que pour la temporisation ; seul la fonction et le câblage diffèrent.

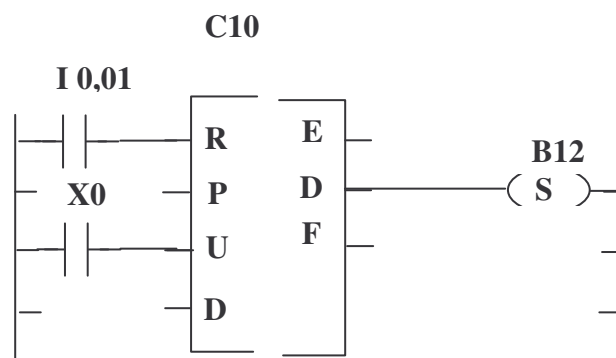


- Le compteur permet de compter des impulsions (ou des mises à 1 de tests d'entrées) et d'incrémenter ou de décrémentation de 1 la valeur de comptage du compteur.
- Si la valeur de comptage est atteinte, la sortie « DONE » se met à un.
- Si la valeur de comptage dépasse 9999, la sortie « FULL » se met à 1.
- Si la valeur de comptage décroît en dessous de 0, la sortie « EMPTY » se met à 1.
- Pour remettre le compteur à zéro en cas de comptage par incrémentation, il faut activer l'entrée « R.A.Z ».
- Pour remettre le compteur à la valeur programmée en cas de décrémentation, il faut activer l'entrée « PRESET ».

Nota :

1. La sortie « **DONE** » se met à un lorsque le compteur a atteint sa valeur programmée, si le comptage continu, **la sortie se remet à zéro**. Si l'on veut garder l'information plus longtemps, il faudra activer une sortie « **SET** » et la remettre à zéro (**RESET**) par l'intermédiaire de l'entrée « **R.A.Z** » ou « **PRESET** ».
2. Le compteur se programme toujours dans le postérieur dans un label entier (4 lignes).

- Exemple de câblage :



12°) Exercices :

12.1°) Réaliser le comptage jusqu'à 5 de la mise à un du test d'une entrée de votre choix dans le postérieure (sans toucher au programmes précédents), la fin du comptage activera une sortie non utilisée. Une fois la programmation du label effectué mettez vous en mode « run » et effectué vos essais.

12.2°) Insérez une tempo dans le cycle de la platine d'imprimerie pour chaque descente du tampon :
première tempo de 5 secondes (à l'encrage) pour bien encrer le tampon,
deuxième tempo de 2 secondes (au marquage) pour bien encrer la carte de visite.

12.3°) Insérer un compteur pour compter 5 cycles et enclencher une sortie, raccorder à un voyant, pour prévenir l'opérateur qu'il faut nettoyer les lettres du tampon. Essayer ensuite de mettre en série avec la sortie, un test d'entrée nommer « SY6 » et noter ce qu'il se passe.

12.4°) Insérer un compteur pour compter 5 cycles et enclencher une sortie, raccorder à un voyant, pour prévenir l'opérateur qu'il faut nettoyer les lettres du tampon. Essayer ensuite de mettre en série avec la sortie, un test d'entrée nommer « SY5 » et noter ce qu'il se passe.

12.5°) Même énoncé, mais le cycle doit se bloquer et n'être déverrouillable que par une entrée non utilisée précédemment .