3. Logiques Combinatoires et Séquentielle. Le GRAFCET

Pierre Duysinx
Université de Liège
Année académique 2021-2022

Introduction

- Trois phases à la mise en œuvre d'un automate programmable
  - La conception
    - Outils méthodologiques et modes de représentations propres à l'automaticien
  - La programmation
    - Transposer la loi de contrôle en langage propre à l'automate
  - L'exécution du programme
    - Prise en charge par un logiciel interne au PLC, l'exécution

Introduction

- Importance de la cohérence
  - Sur la facilité d'utilisation
  - Sur les performances
  - Sur la maintenance

- Deux types de problèmes
  - Combinatoire: les sorties sont déterminées en fonction des entrées uniquement
  - Séquentiel: les sorties dépendent aussi de l'histoire via des variables internes mémorisées

LOGIQUE COMBINATOIRE

Problèmes combinatoires

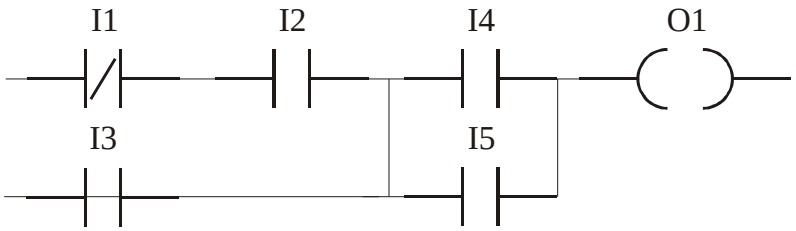
- DEFINITION
  - Problème combinatoire: tous les signaux provenant du processus sont susceptibles de provoquer à n'importe quel moment une réaction de l'automate
  - Cas typique: surveillance de condition d'alarme

- MODES DE REPRÉSENTATION
  - Formalisme à relais
  - Logigrammes
  - Equations booléennes

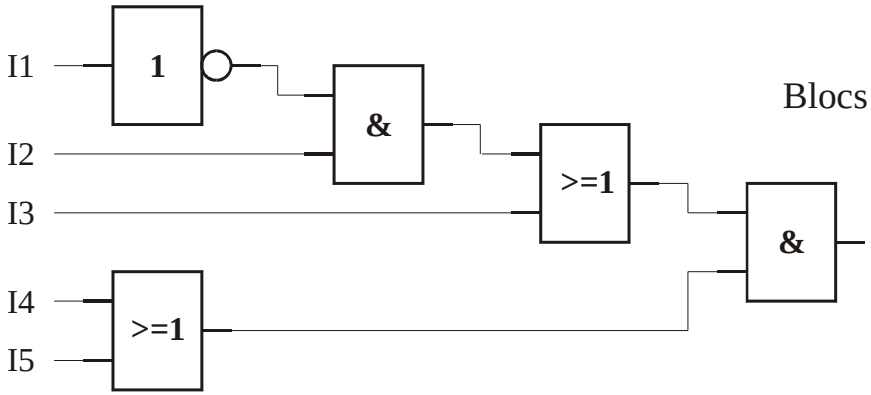
  - Hypothèse: Modules fonctionnels travaillant en parallèle et de manière instantanée

Problèmes combinatoires

- MODES DE REPRÉSENTATION



Relais



Blocs fonctionnels

$$\mathbf{O1} = (\overline{\mathbf{I1}} \,.\, \mathbf{I2} + \mathbf{I3}) \,.\, (\mathbf{I4} + \mathbf{I5})$$

Equation booléenne

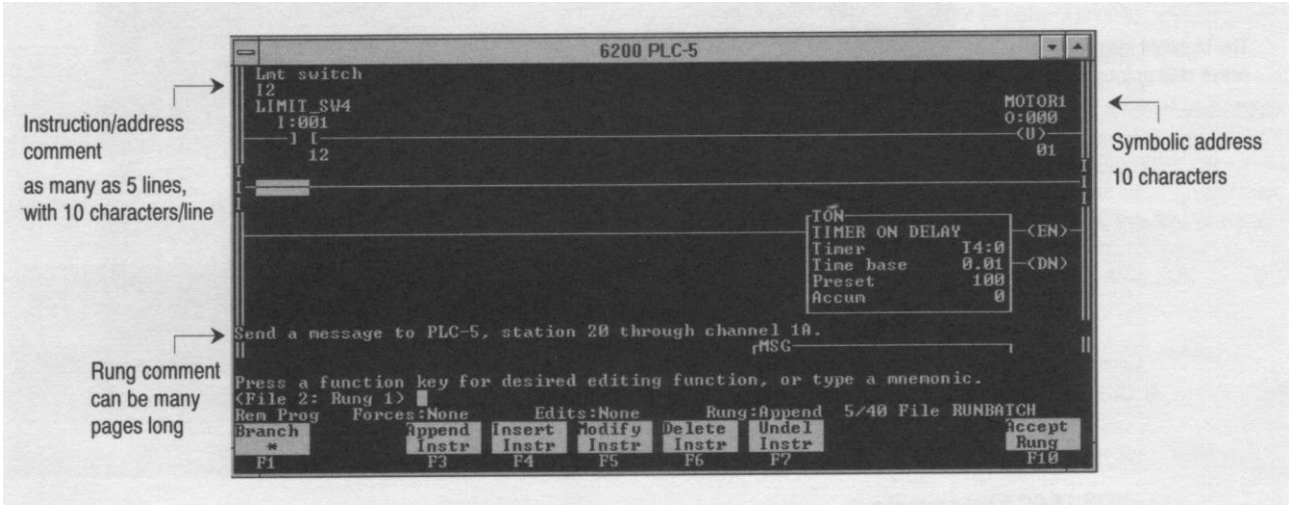
Problèmes combinatoires

- PROGRAMMATION
  - Langages à relais
  - Langages en logigrammes
  - Langages en texte structuré

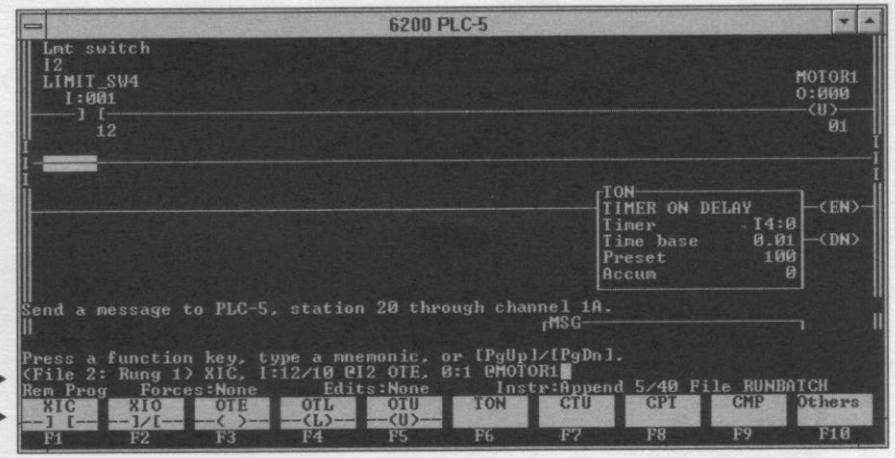
  = Prolongation naturelle des modes de représentation

    - Programmer = reproduire l'équation et les schémas de représentation
    - Traduction automatique en instructions machines
  - Langages par liste d'instructions
    - Utilisateur traduit l'équation en la décomposant et en la sérialisant
    - Moins cohérent
    - Attention à la séquence
    - Lisibilité faible
    - Documentation nécessaire

Problèmes combinatoires



Use function-key assisted or command-line entry. With function-key entry, you select each instruction and the software prompts you for the parameters. Use command-line entry to build an entire rung, including symbols.

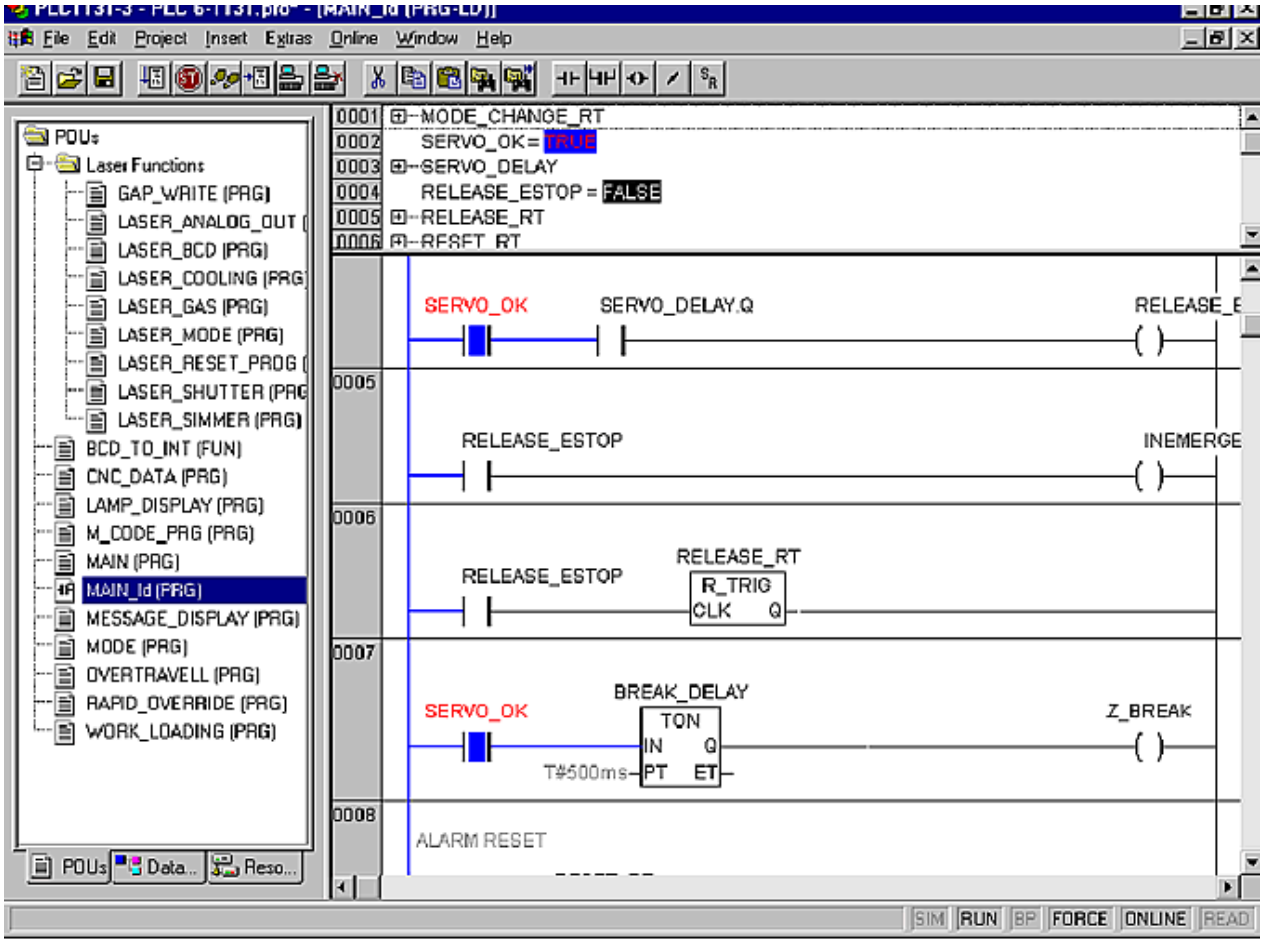


Command-line entry

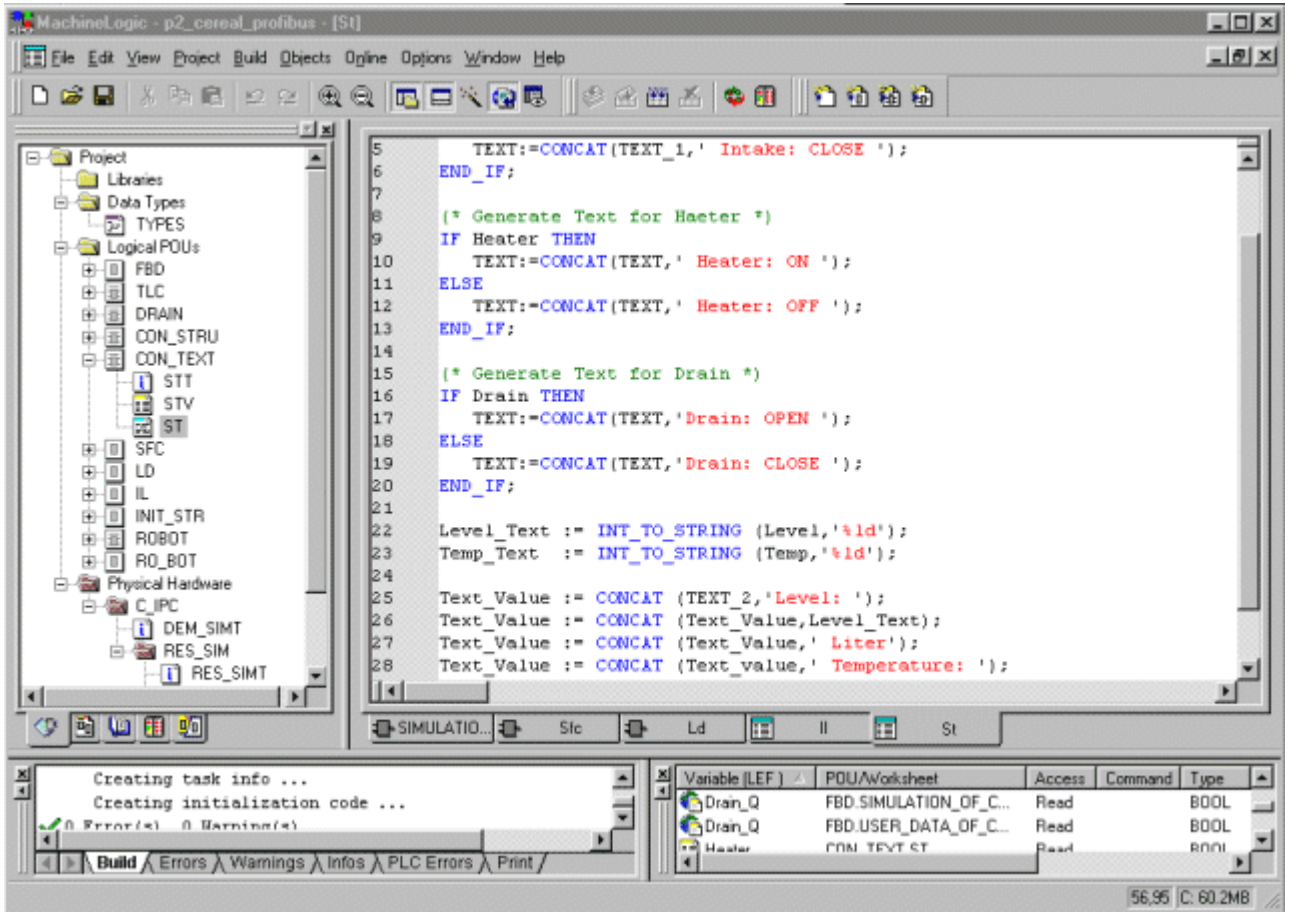
Function-key assisted entry

Ladder diagramme (Allen Bradley)

Problèmes combinatoires

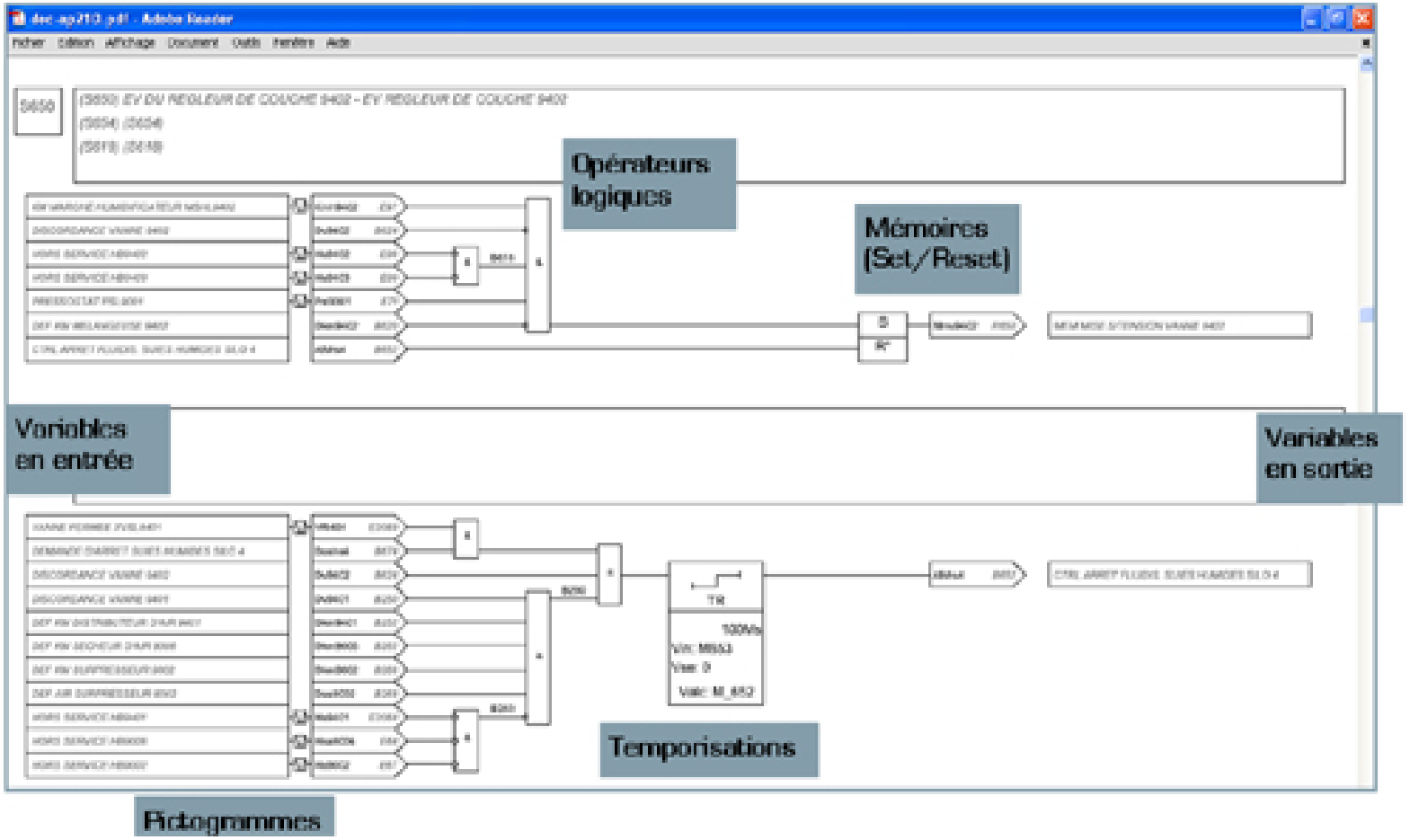


Problèmes combinatoires



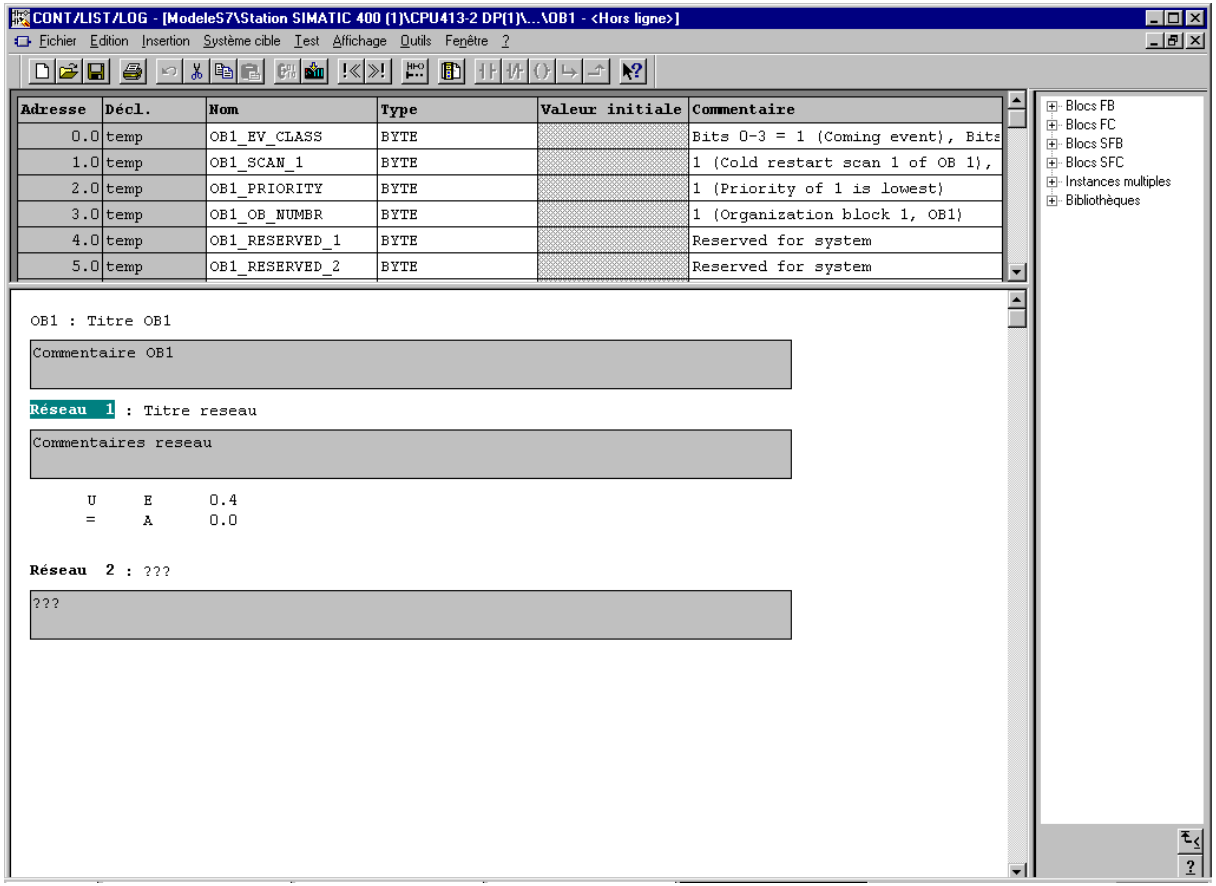
Texte structuré (Télémécanique Schneider))

Problèmes combinatoires



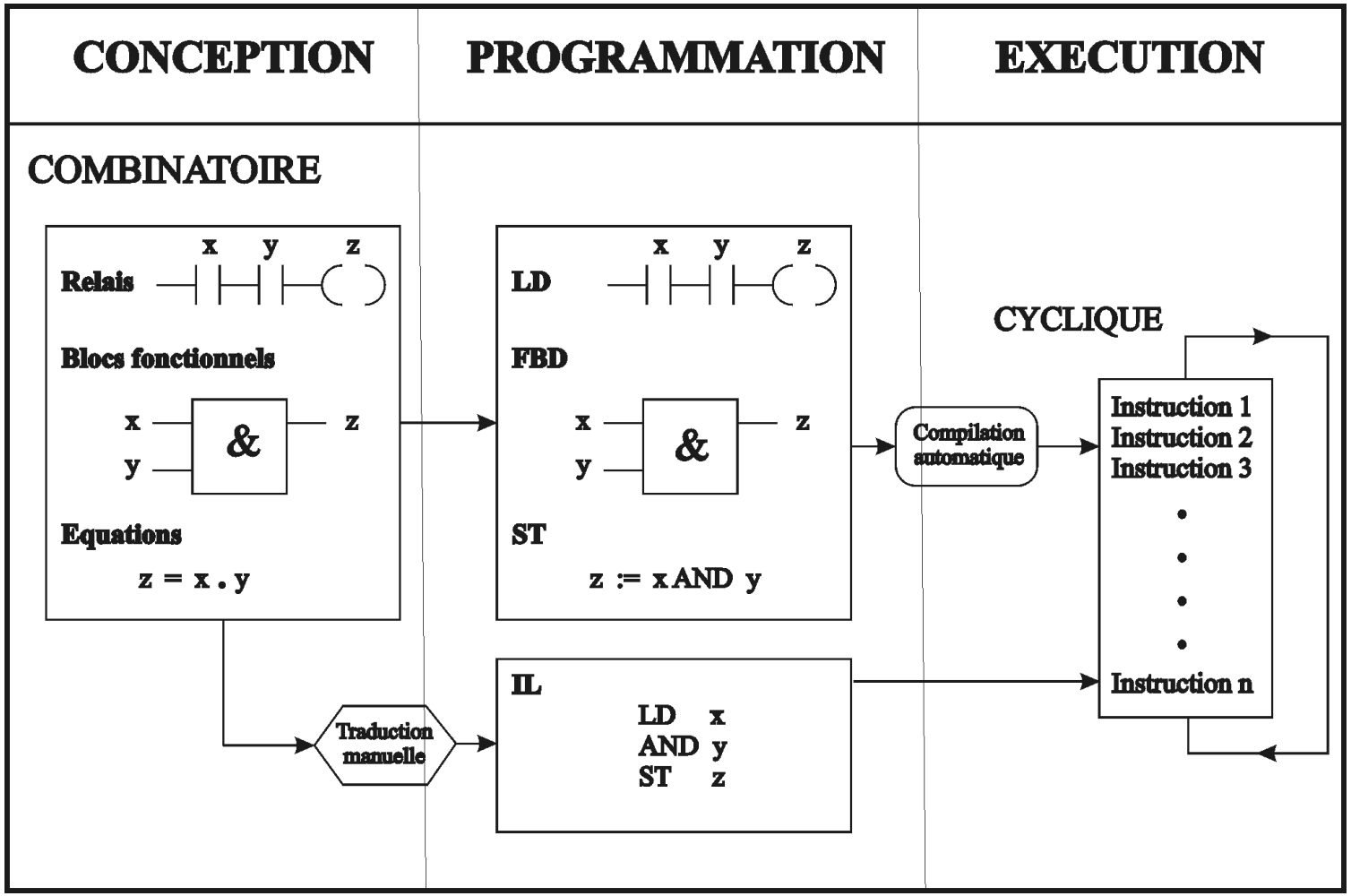
Logigrammes (Siemens S7)

Problèmes combinatoires



Liste d'instructions (Siemens S7)

Problèmes combinatoires



Problèmes combinatoires

- EXECUTION
  - Simuler au mieux le mode de fonctionnement en parallèle simultané
  - Exécuter les instructions de manière linéaire et cyclique
  - Scrutation des entrées au début du cycle
  - Copies des sorties à la fin de chaque cycle

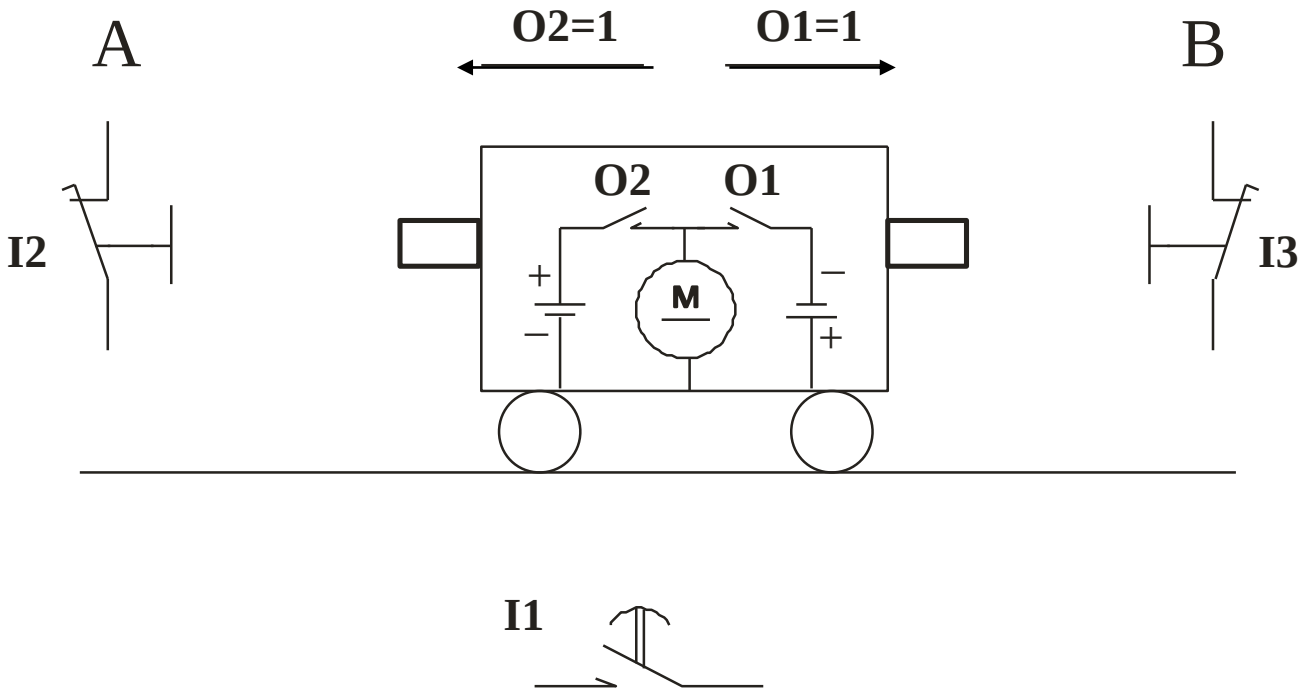
LOGIQUE SEQUENTIELLE

Problèmes séquentiels

- Souvent les problèmes industriels sont de nature séquentielle (problème avec transports d'outil, de matière…)

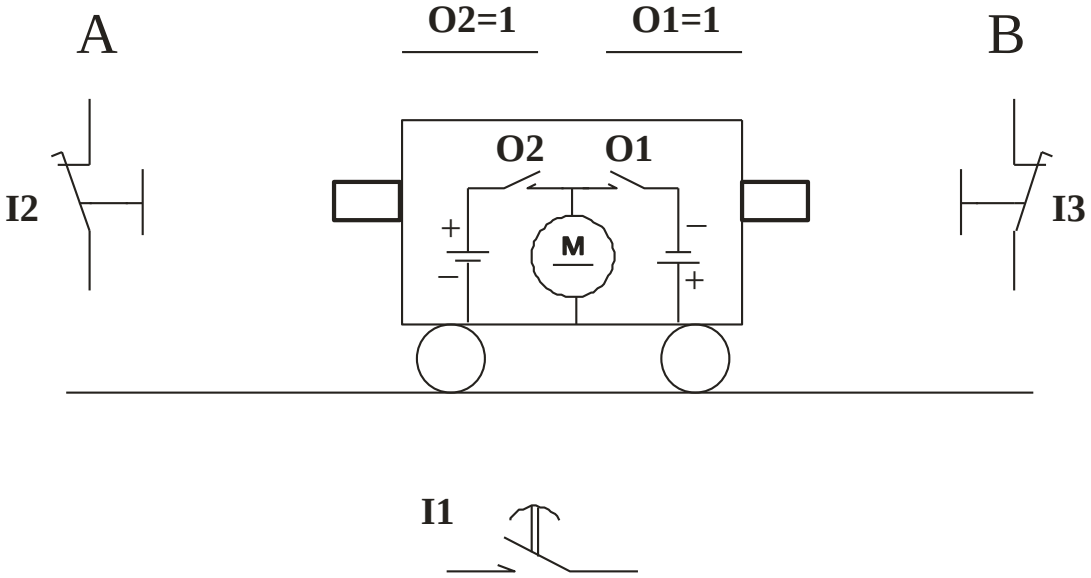
- DEFINITION
  - A un moment donné, seule une série limitée de signaux donnés doivent être pris en considération et seul un nombre limité de commandes sont susceptibles d'être appliquées au processus
  - Entre d'autres mots, les sorties dépendent aussi de l'histoire via des variables internes mémorisées

Exemple



- Problème:
  - Par pression sur un bouton poussoir (I1), un chariot initialement en A (capteur I2 ouvert) est mis en mouvement vers la position B (sortie O1 vraie)
  - Arrivé en B (capteur I3 ouvert) , le chariot repart en sens inverse (sortie O2 vraie) jusque A
  - Recommencer si I1 vrai

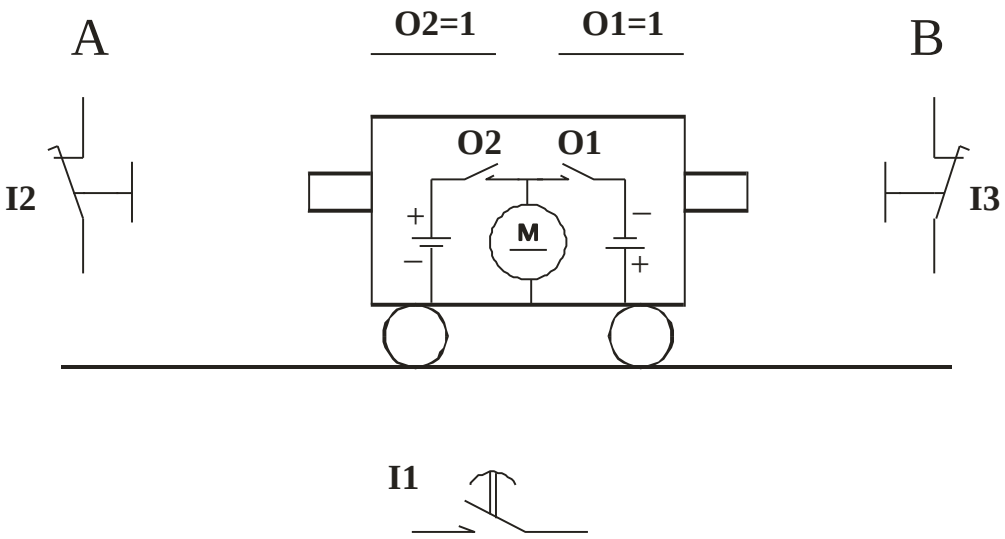
Exemple

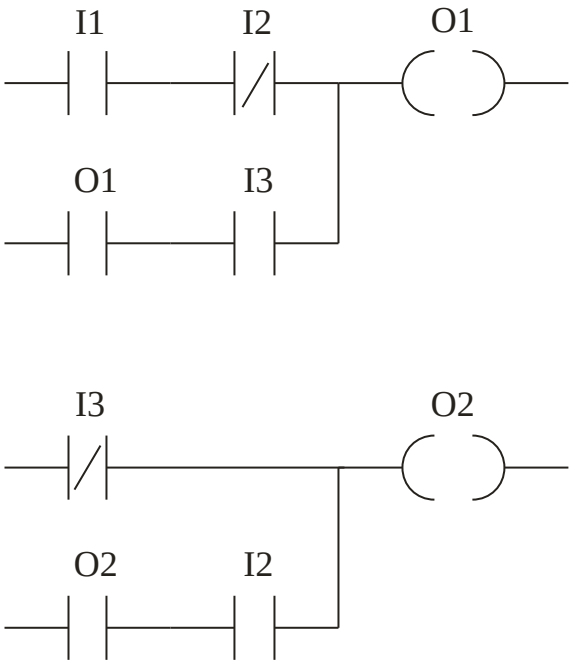


- Etape 1:
  - chariot en position de départ (I2=0) , moteur arrêté (out1=out2=0)
  - Seul l'entrée In1 peut faire évoluer l'état du système
  - Seule la sortie out1 est activable
- Etape 2:
  - Après pression de I1, le chariot est en mouvement vers B (out1=1)
  - Seule l'entrée I3=0 peut faire évoluer la situation
- Etape 3:
  - Suite à l'arrivée du chariot en B (I3=0), retour du chariot vers A (out1=0, out2=1)
  - Seule I2 peut faire évoluer la situation

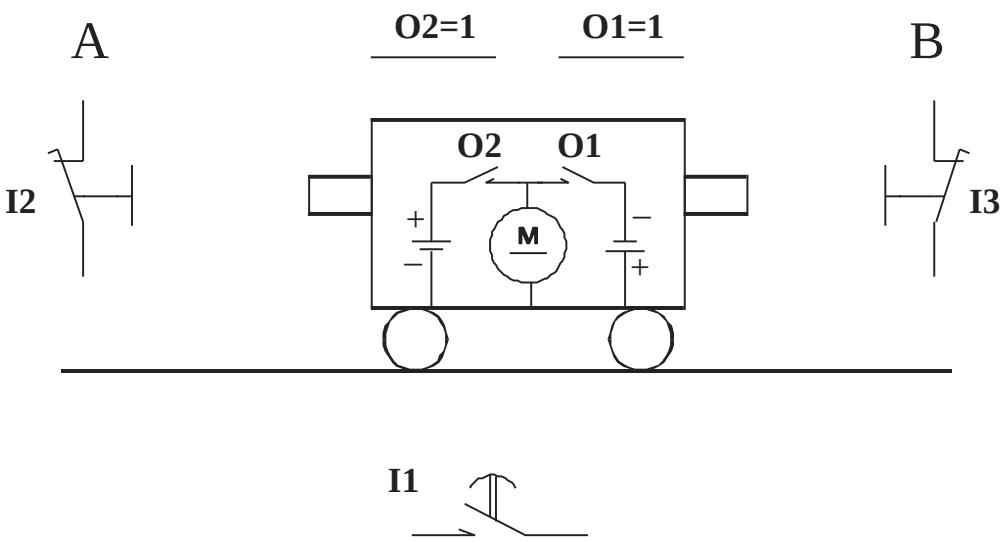
Conception de contrôles séquentiels

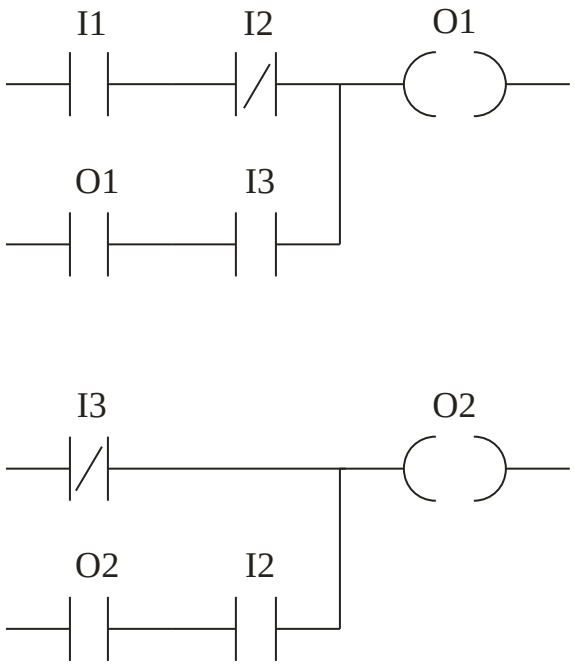
- FORMALISME PARALLELE
  - Dans le cas simple de l'exemple, on peut encore envisager de concevoir le système de contrôle par un mécanisme parallèle c'est-à-dire utiliser les outils combinatoires.
  - Exemple





Exemple (suite)





Conception parallèle

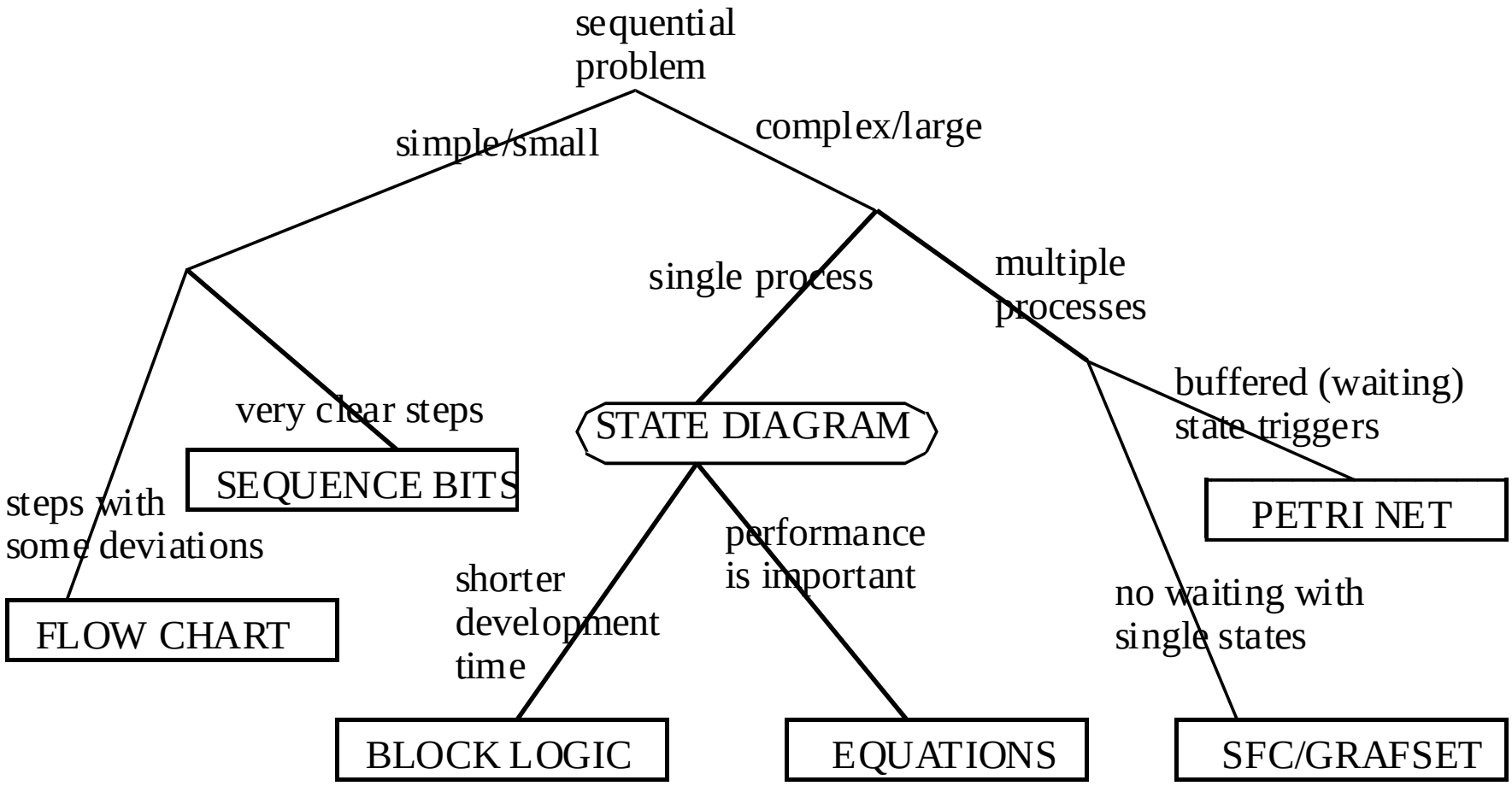
Conception de contrôles séquentiels

- FORMALISME PARALLELE
  - Difficultés du contrôle parallèle
    - Disparition de la séquence caractéristique dans le mode de programmation
    - La scrutation cyclique systématique des entrées qui conduit au mode parallèle est inutile,
      - Il n'y a jamais qu'une seule entrée qui peut faire évoluer le système
    - La scrutation cyclique systématique des sorties est en principe superflue
      - Les entrées ne peuvent avoir une influence que lorsqu'on passe d'une étape à l'autre
    - Extension à des problèmes complexes extrêmement difficiles

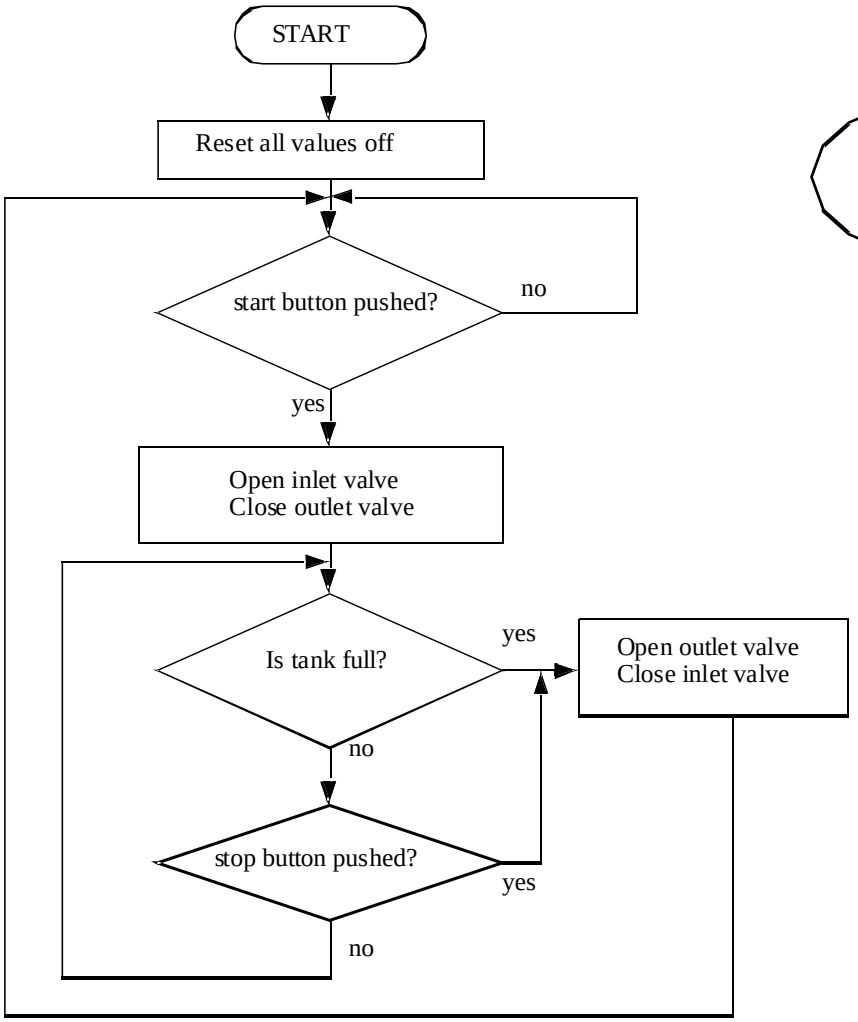
Conception de contrôles séquentiels

- FORMALISMES SEQUENTIELS
  - Pour prendre en charge la conception de systèmes séquentiels, on a mis au point différents formalismes
    - Flow chart
    - Diagrammes d'état: (state flow)
    - GRAFCET: GRAF = graphe – C = commande – E = étape - T= transition
    - Réseaux de Petri

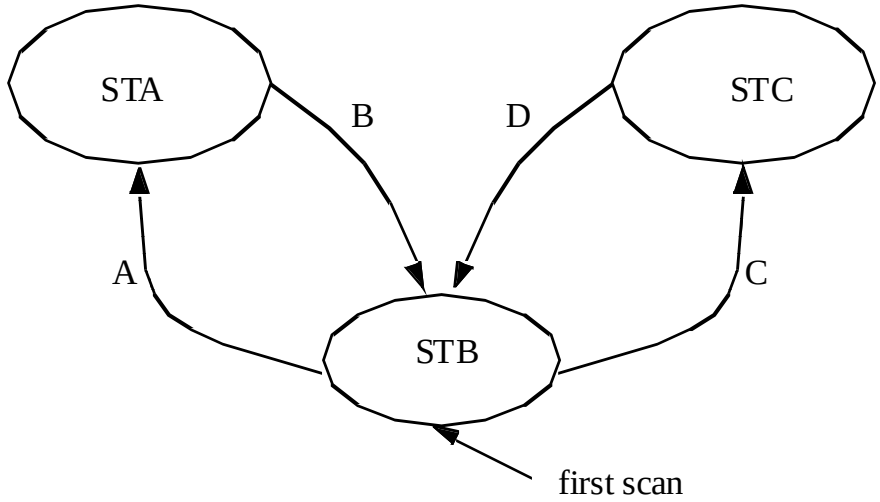
Conception de contrôles séquentiels



Conception de contrôles séquentiels



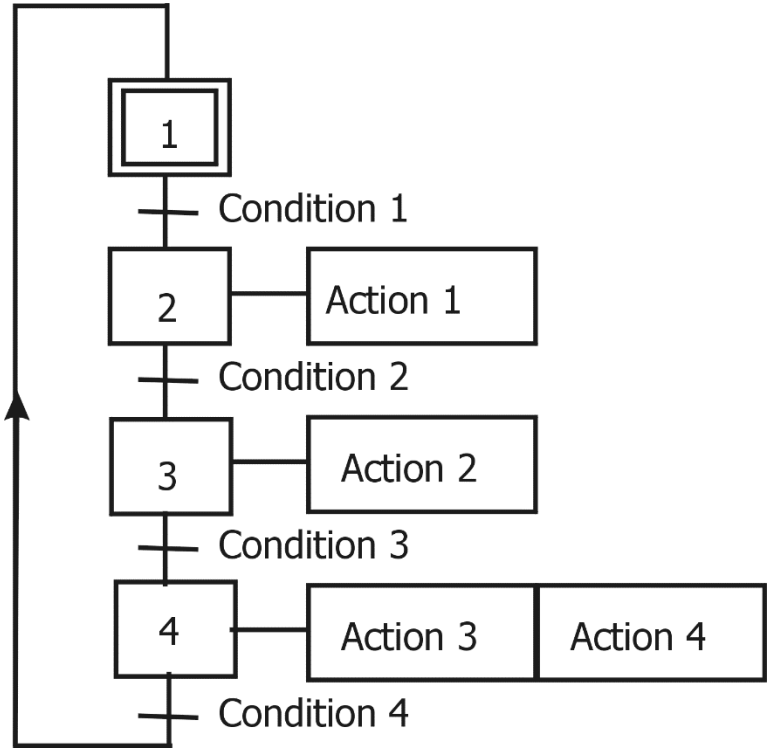
Flow chart



State diagram

Conception de contrôles séquentiels

- FORMALISMES SEQUENTIELS
  - Le GRAFCET aussi connu sous le nom SFC (Sequential Function Chart)
    - Normalisation CEI IEC 848
    - Les carrés = étapes de fonctionnement
    - Les rectangles associés = les actions à prendre si étape active
    - Le passage d'une étape à l'autre est verrouillé par une condition logique appelée réceptivité
    - Lorsqu'une étape est active et que la réceptivité est vraie, on passe automatiquement à l'étape suivante tandis que l'étape précédente est désactivée



GRAFCET: historique

- GRAFCET: GRAF = graphe – C = commande – E = étape - T= transition
- Au départ (1975), outil méthodologique de description du cahier des charges de la partie commande des systèmes automatisés
- Normalisation en France (1982), puis en, Europe par la CEI sous le nom de Sequential Function Chart
  - Outil méthodologique unanimement adopté
- L'outil méthodologique de description GRAFCET est devenu un langage de programmation sous la norme CEI 1131-3
  - Langage de programmation au succès mitigé
    - Problème d'interface graphique
    - Problème d'absence de normalisation au niveau sémantique
  - La majorité des programmeurs recourent à une traduction manuelle

LES REGLES DU GRAFCET

Eléments de base du GRAFCET

- Le GRAFCET est un organigramme qui décrit dans une suite logique et organisée le cycle et le déroulement des différentes opérations appelées « étapes » d'un système automatisé.

- Une ÉTAPE correspond à une situation dans laquelle le comportement de tout ou une partie du système par rapport à ses entrées et ses sorties est invariant.

- Les TRANSITIONS indiquent les possibilités d'évolution entre étapes.

Eléments de base du GRAFCET

- On associe à chaque transition une condition logique appelée RECEPTIVITE qui permet de distinguer parmi toutes les informations disponibles uniquement celles qui sont susceptibles de faire évoluer la partie commande à un instant donné.

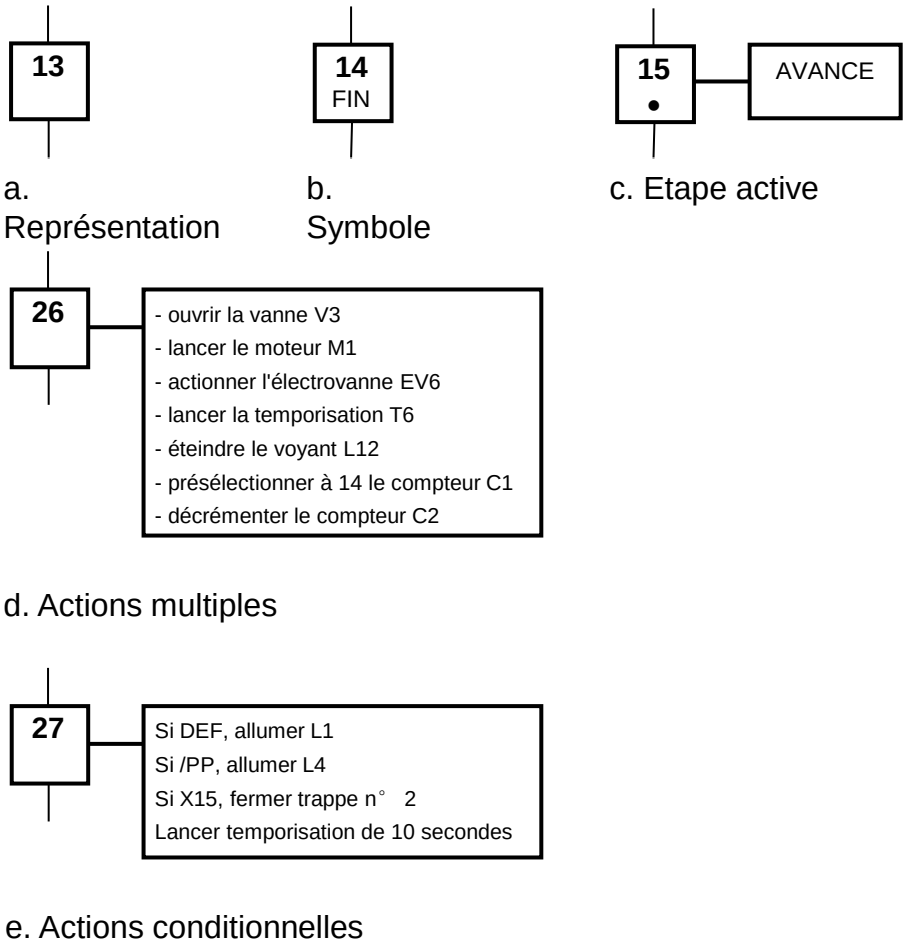
- La RECEPTIVITE écrite sous forme de proposition logique est une fonction combinatoire des informations extérieures (directives opérateur, capteurs, compteurs, tempos, changement de variables, etc.) et de variables auxiliaires ou de l'état actif ou inactifs d'autres étapes.

- Les LIAISONS indiquent les voies d'évolution de l'état du GRAFCET

Eléments de base du GRAFCET

- L'étape se représente par un carré repéré par un numéro

- On peut également adjoindre un nom symbolique
- Une étape peut être active ou inactive.
- La situation du système est entièrement définie par l'ensemble de ses étapes actives
- On peut marquer l'étape active par un point



a. Représentation

b. Symbole

c. Etape active

d. Actions multiples

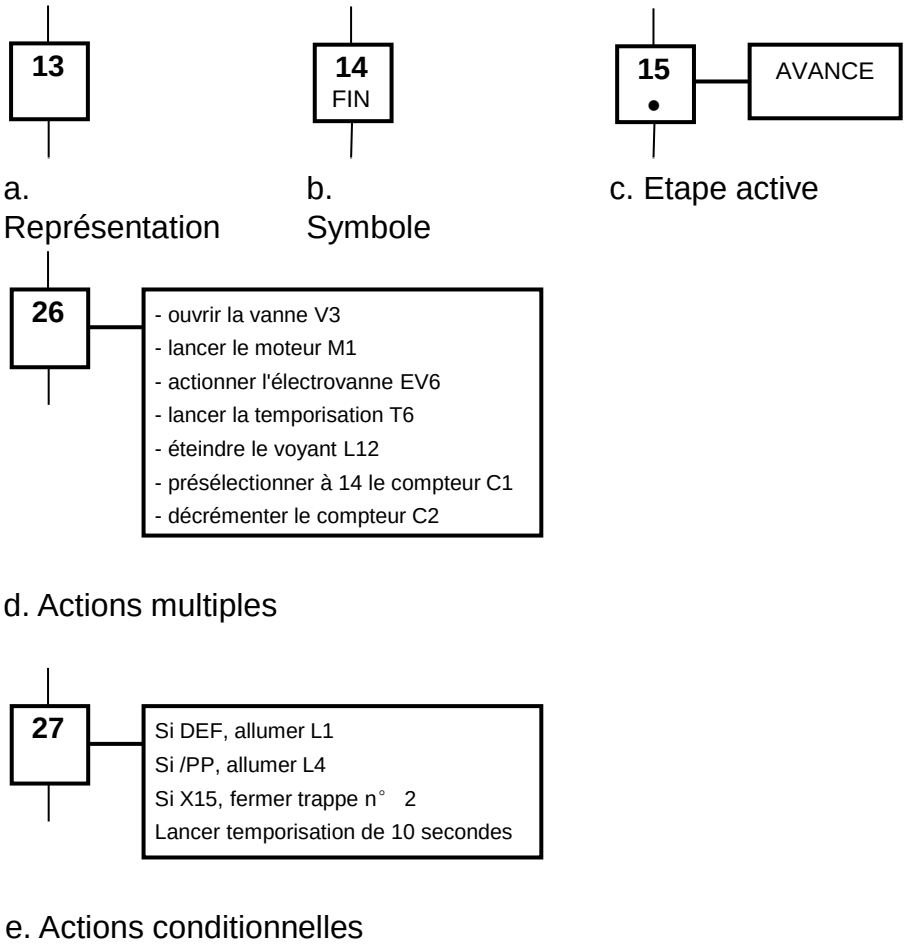
e. Actions conditionnelles

Etapes du GRAFCET

Eléments de base du GRAFCET

- On précise pour chaque étape les actions caractéristiques à effectuer

- Les actions sont effectives lorsque l'étape est active
- L'exécution des actions peut encore être soumise à des conditions



a. Représentation

b. Symbole

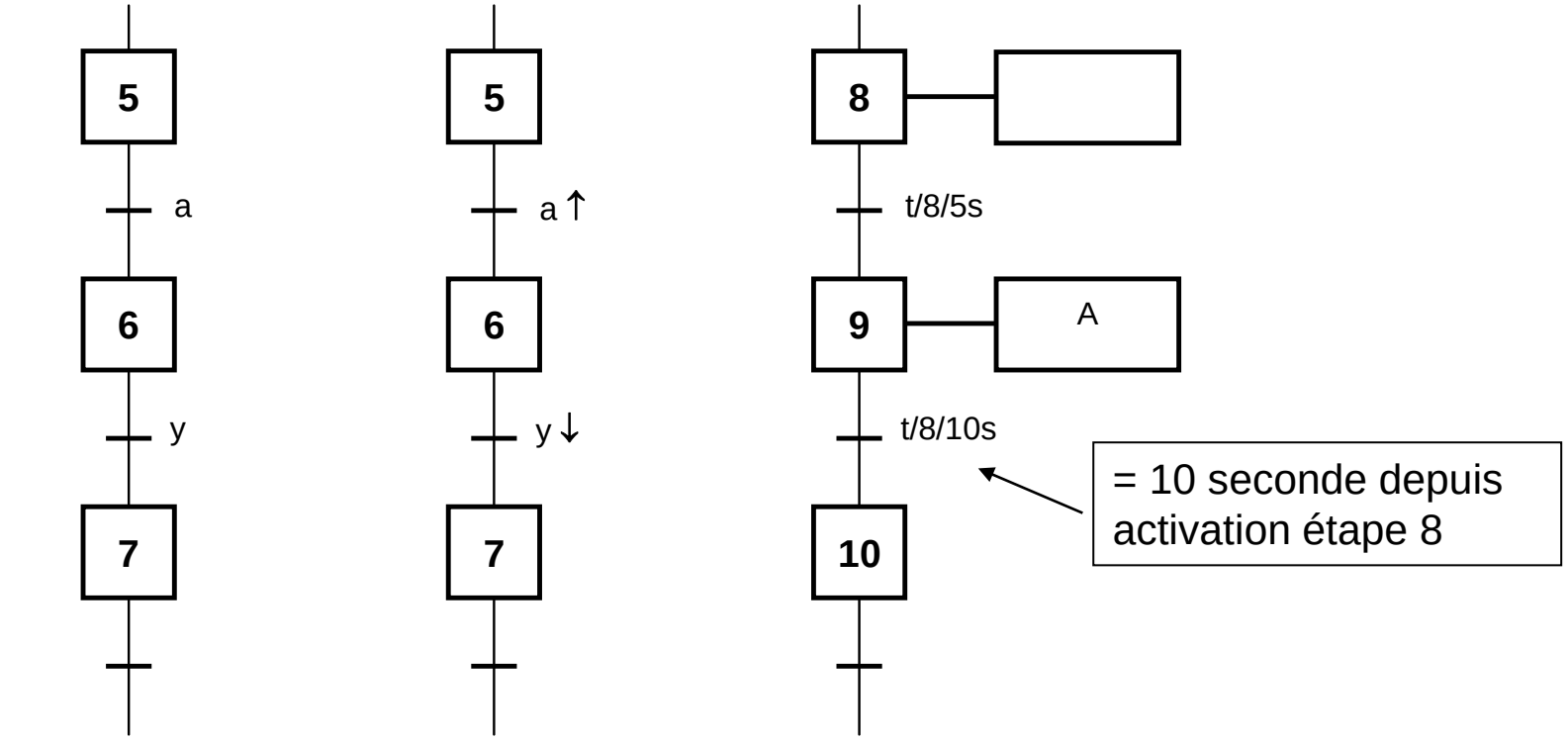
c. Etape active

d. Actions multiples

e. Actions conditionnelles

Etapes du GRAFCET

Eléments de base du GRAFCET



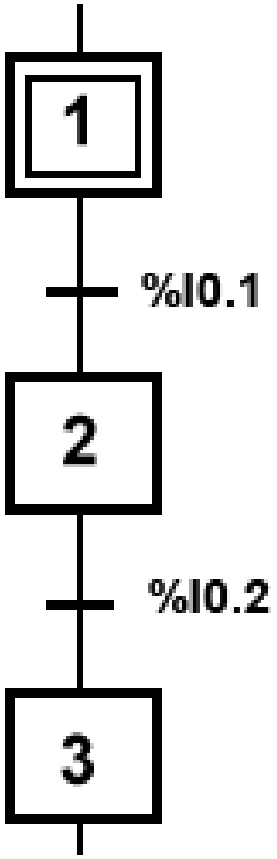
a. Etats logiques b. Changements d'état c. Dépendant du temps

Transitions et réceptivités

Eléments de base du GRAFCET

- La transition d'une étape à l'autre est verrouillée par une condition logique appelée réceptivité

- Les liaisons sont horizontales ou verticales
  - Adopter une représentation qui contribue au mieux à la clarté du fonctionnement
  - Utiliser les flèches pour améliorer la compréhension ou que l'orientation n'est pas respectée
  - Eviter les croisements pour éviter les ambiguïtés
- Sens général de parcours = de haut en bas
  - L'arrivée à une étape est en haut et le départ en bas



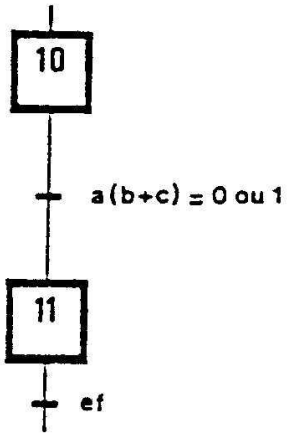
Règles d'évolution du GRAFCET

- Règle 1: Etape initiale
  - Elle est activée inconditionnellement au départ du système
  - Correspond à un comportement au repos de la PO

Règle 1: étape initiale

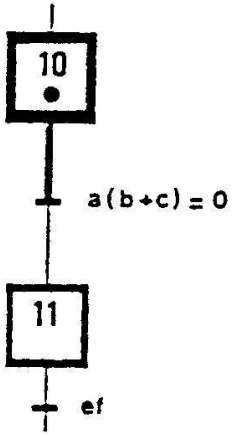
- Règle 2: Franchissement d'une transition
  - Le franchissement de la transition ne peut se produire que si:
    - La transition est validée (étapes immédiatement précédentes toutes actives)
    - ET si la réceptivité associée est vraie
  - La transition est obligatoirement franchie

Règles d'évolution du GRAFCET



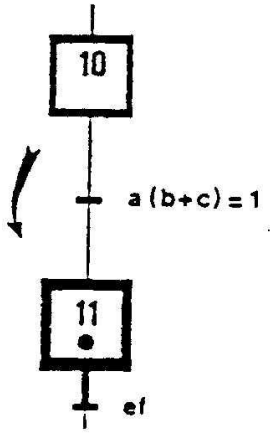
Transition non validée

La transition 10–11 est non validée, l'étape 10 étant inactive.



Transition validée

La transition 10–11 est validée, l'étape 10 étant active, mais ne peut être franchie car la réceptivité a(b+c) = 0



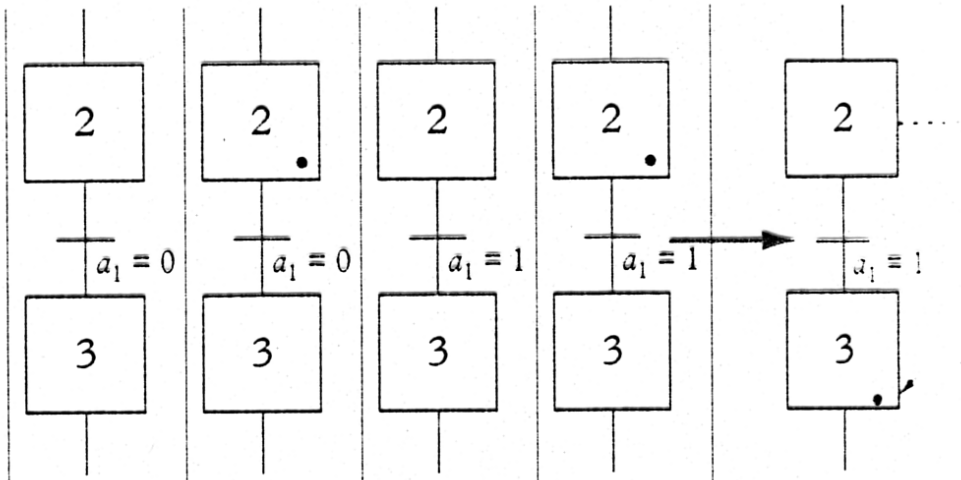
Transition franchie

La transition 10–11 est franchie car la réceptivité a(b+c) = 1

Règle 2: franchissement d'une transition

Règles d'évolution du GRAFCET

Fig. 1



Étapes actives	—	2	—	2	3
Transition validée	non	oui	non	oui	Le franchissement est autorisé et obligatoirement franchi
Réceptivité vraie	non	non	oui	oui	
Transition franchissable	non	non	non	oui	

Règle 2: franchissement d'une transition

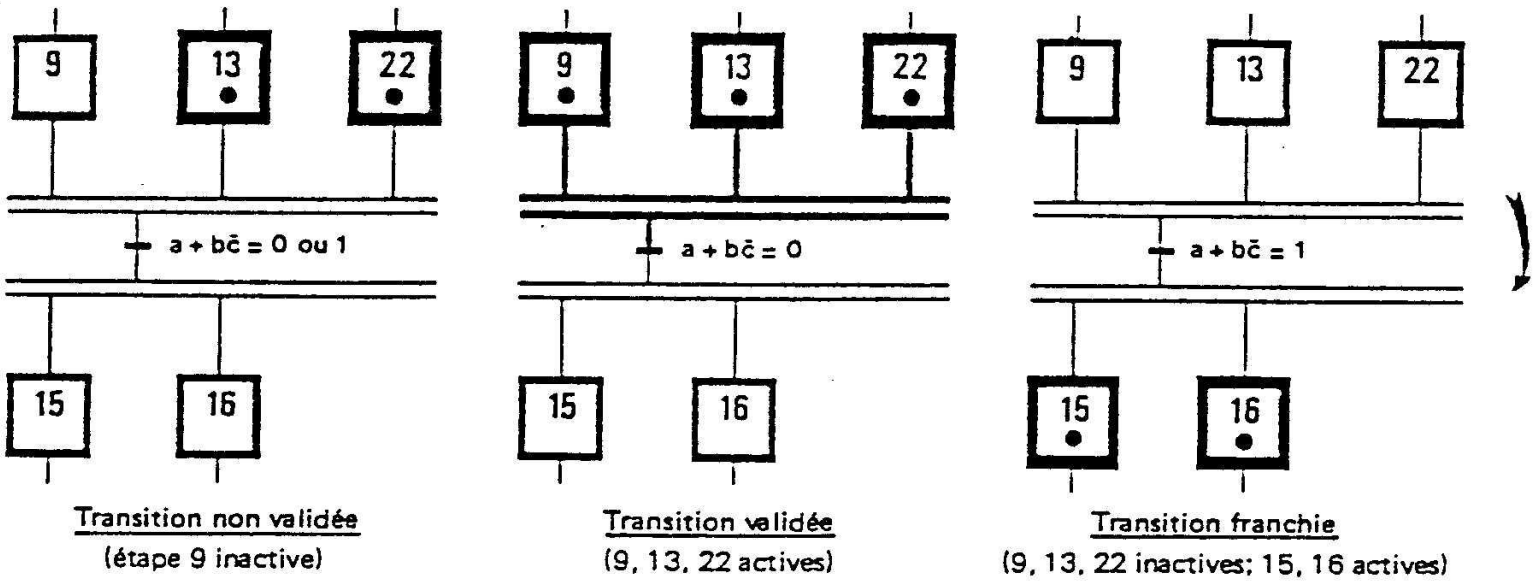
Règles d'évolution du GRAFCET

- Règle 3: Evolution des étapes actives
  - Le franchissement d'une transition entraîne l'activation de *toutes* les étapes immédiatement suivantes reliées à cette transition et la désactivation de *toutes* les étapes immédiatement précédentes reliées à la transition.

- Règle 4: Evolutions simultanées
  - Plusieurs transitions simultanément franchissables (franchissables en même temps) sont simultanément franchies (franchies en même temps).

- Règle 5: Activation – désactivation simultanée
  - Si au cours du fonctionnement une même étape doit être à la fois activée et désactivée, elle reste active.

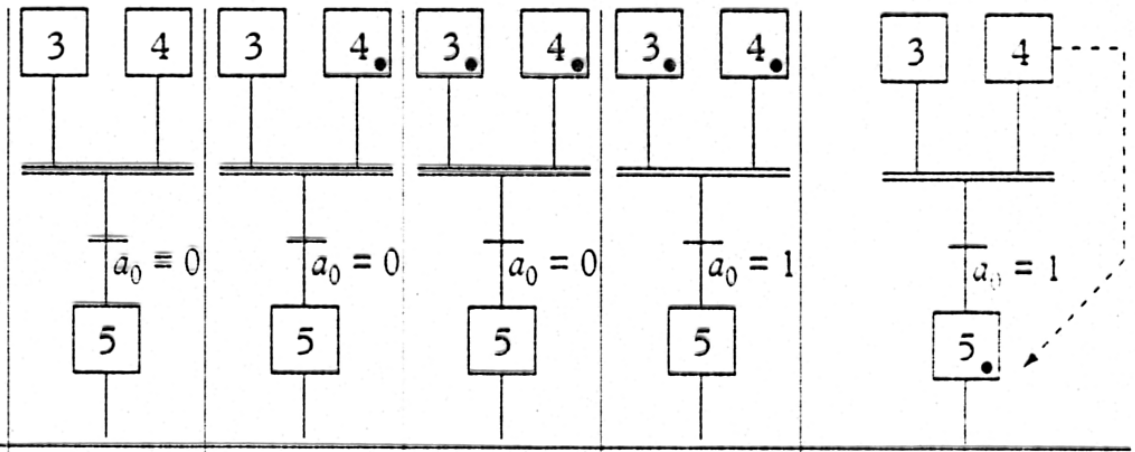
Règles d'évolution du GRAFCET



Règle 3: Transitions entre plusieurs étapes

Règles d'évolution du GRAFCET

Fig. 2

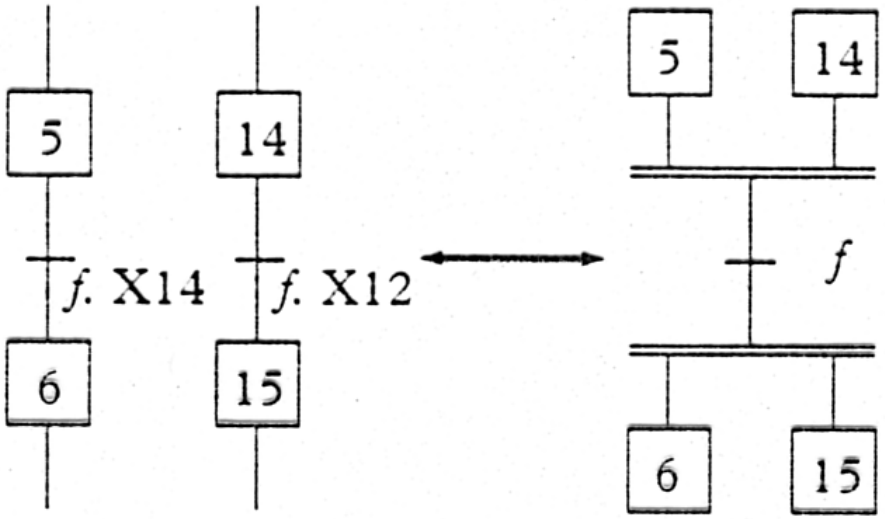


Étapes actives	—	4	3 et 4	3 et 4	5
Transition validée	non	non	oui	oui	oui
Réceptivité vraie	non	non	non	oui	oui
Transition franchissable	non	non	non	oui	est franchie

Règle 3: Evolution des étapes actives

Règles d'évolution du GRAFCET

Fig. 3



Si 5 active, X12 = 1
Si 14 active, X14 = 1
Si 14 inactive, X14 = 0

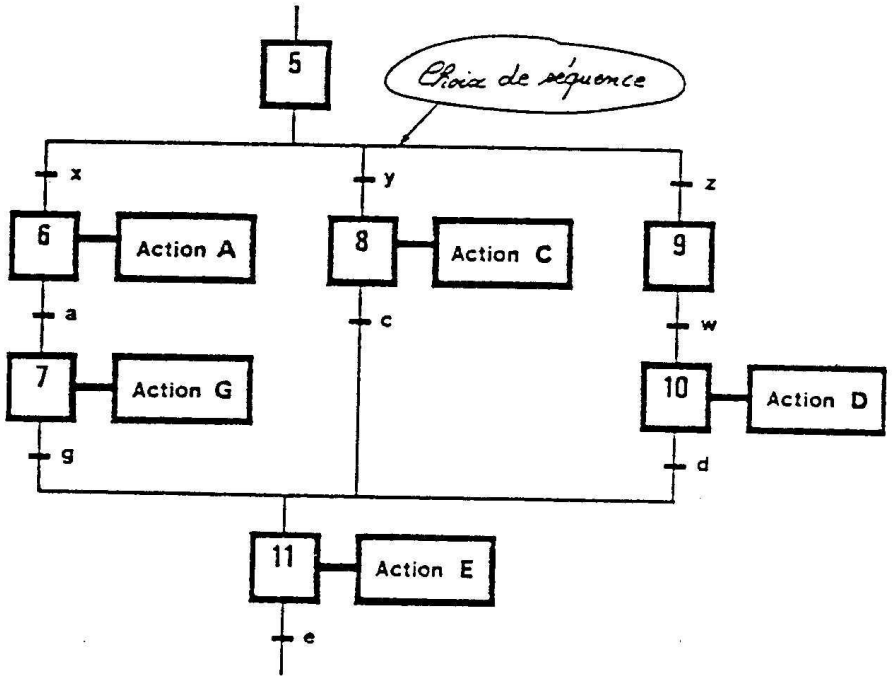
Règle 4: Evolution simultanée

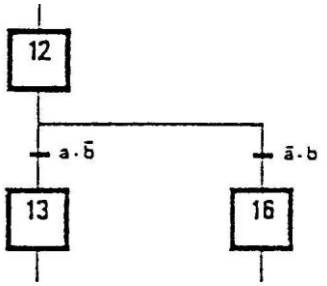
GRAFCET: séquences multiples

CHOIX CONDITIONNEL OU

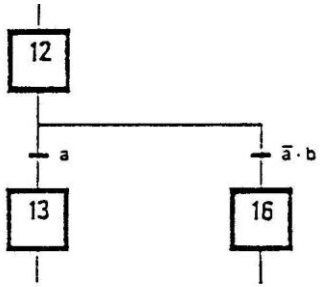
- Configuration OU apparaît quand à la sortie d'une étape le chemin que peut prendre le cycle a la possibilité de se faire sur différentes séquences suivant l'état des réceptivités

- En pratique, prendre des réceptivités mutuellement exclusives





Réceptivités a.$\bar{b}$ et $\bar{a}$.b exclusives
Si a et b sont présents à la fois, aucune transition ne pourra être franchie à partir de l'étape 12.

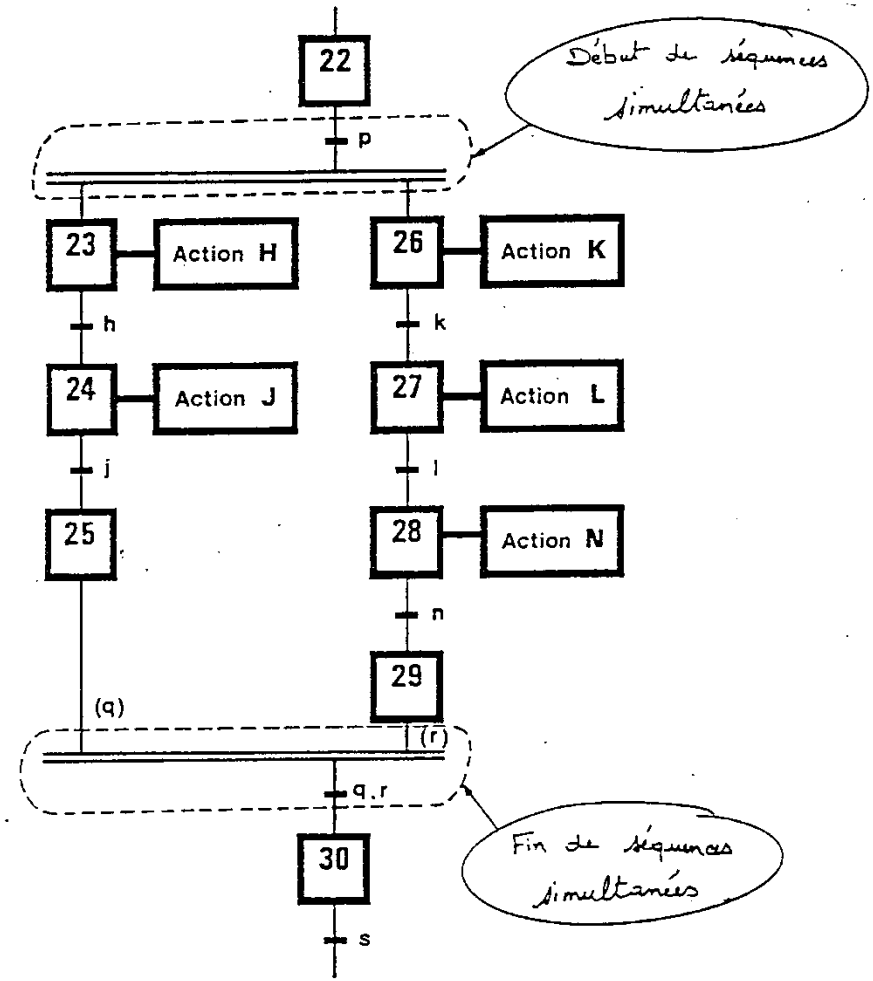


Priorité à la réceptivité a. La priorité donnée à la transition 12-13 permet à celle-ci d'être franchie lorsque a et b sont présents en même temps.

GRAFCET: séquences multiples

SEQUENCES SIMULTANEES «ET»

- Chaque fois que le franchissement d'une transition active plusieurs chemins fonctionnant en parallèle
- L'exécution des différentes branches est indépendante, mais se déroule en même temps
- Représentation: une transition UNIQUE et deux traits parallèles indiquent le début et la fin des séquences parallèles
- A la fin attente réciproque (synchronisation)



GRAFCET: séquences multiples

SAUT D'ETAPES ET REPRISE DE SEQUENCE

- Permet de sauter une ou plusieurs étapes lorsque les actions à réaliser deviennent inutiles

REPRISE DE SEQUENCE

- Permet de reprendre plusieurs fois la même séquence tant que la condition fixée n'est pas obtenue

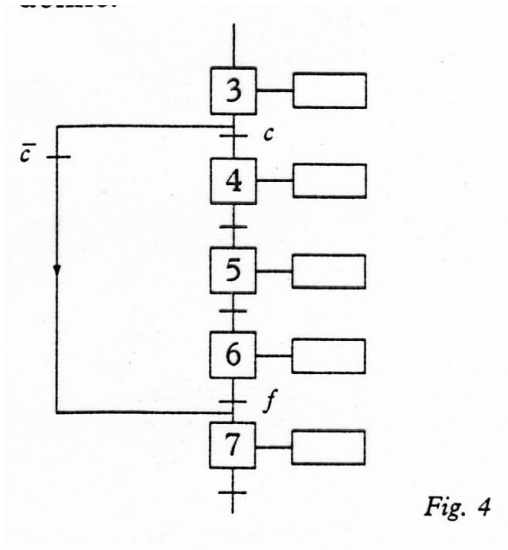


Fig. 4

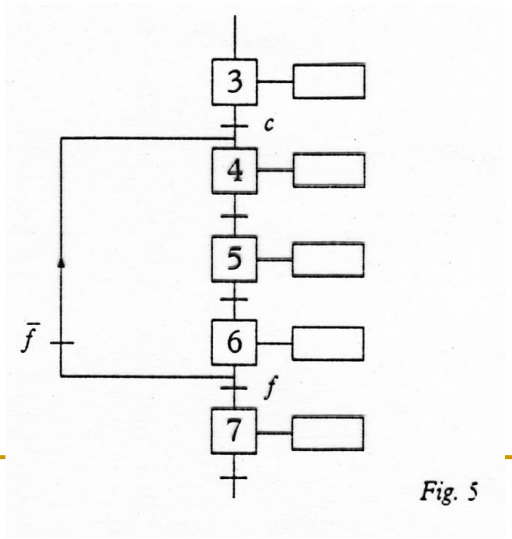
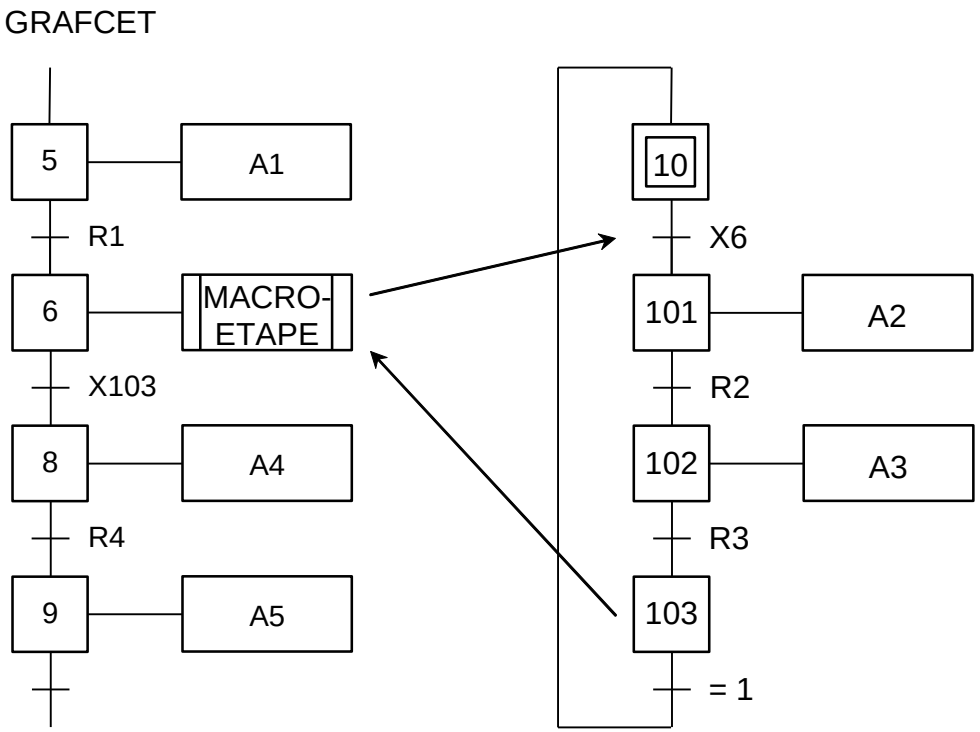


Fig. 5

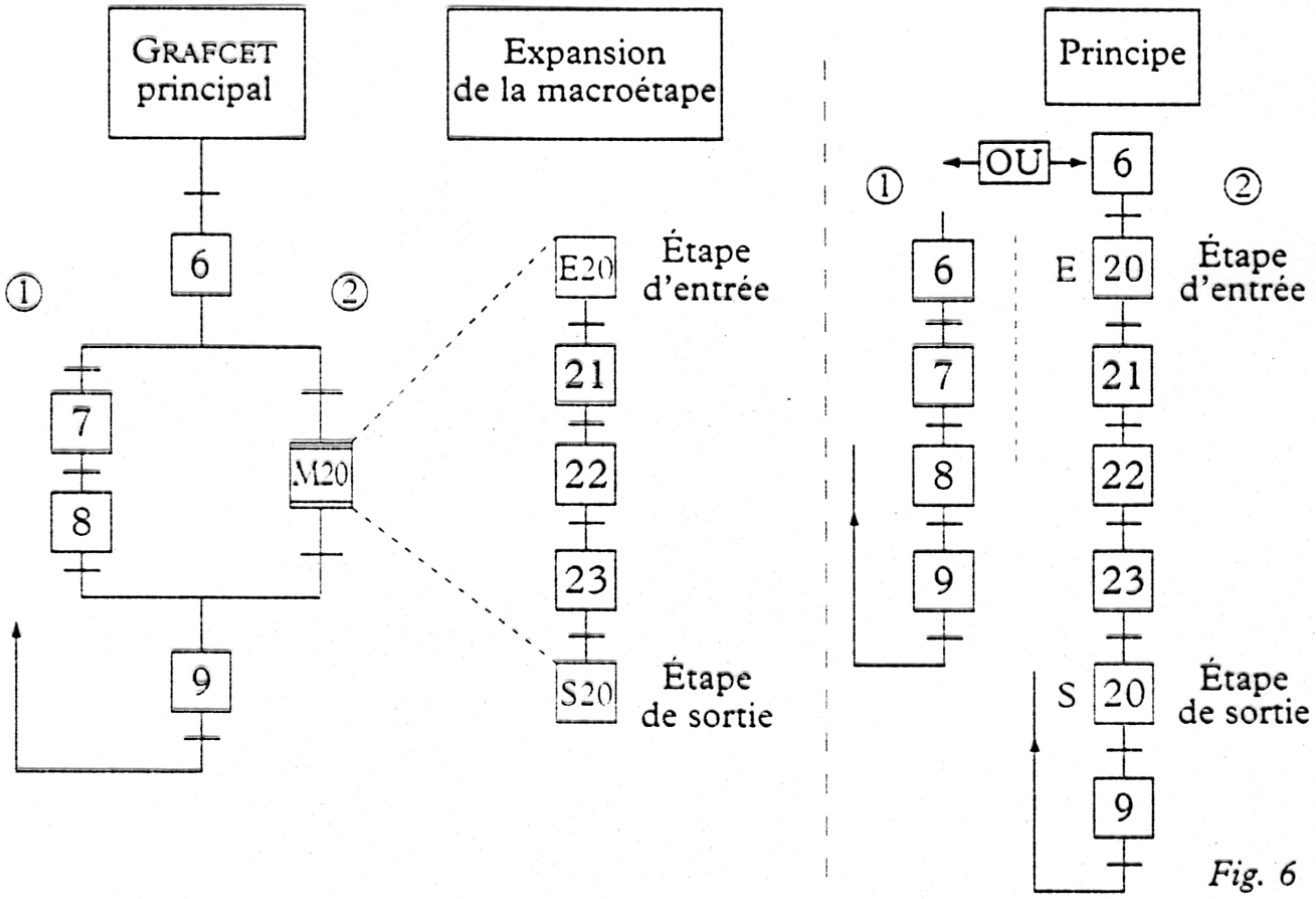
GRAFCET: organisation

LA MACRO-ÉTAPE

- Eviter la surcharge du GRAFCET principal par des détails de fonctionnement à l'intérieur d'une étape.
- Les détails de réalisation sont décrits par un autre diagramme appelé « expansion macro étape »
- La macro étape est repérée par deux barres verticales dans le rectangle des actions



GRAFCET: organisation

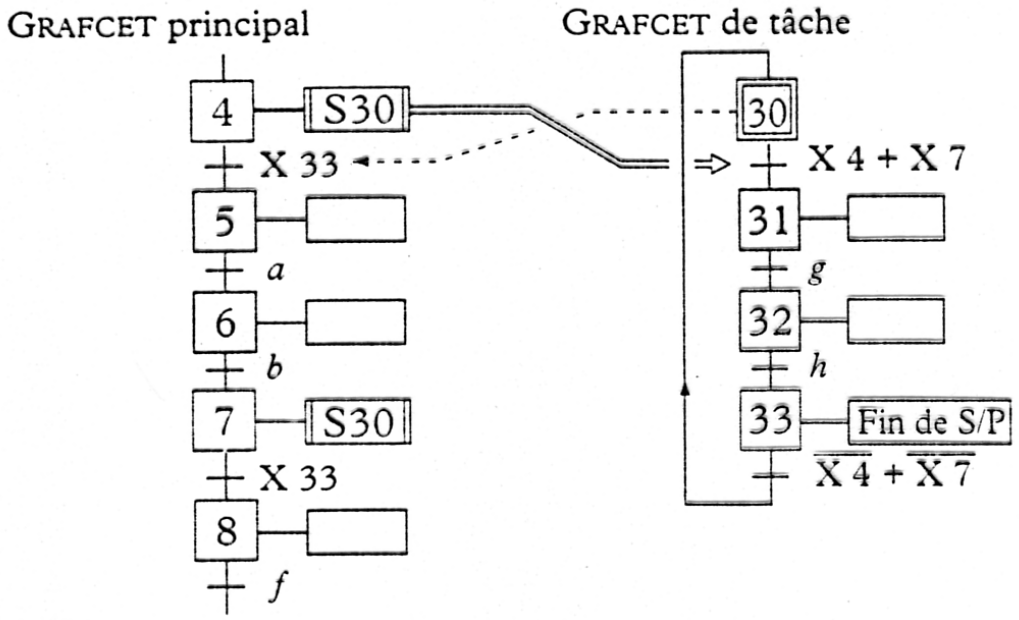


Principe de la macro étape

GRAFCET: organisation

SEQUENCE REPETE

- Certaines séquences peuvent revenir de façon répétitive dans un même cycle.
- Pour éviter la répétitions, on utilise un sous-programme écrit sous forme de GRAFCET indépendant, connecté au GRAFCET principal



GRAFCET: organisation

ETAPE DE VERROUILLAGE

- Cette étape considérée comme une étape initiale complémentaire permet le déroulement d'une seule séquence à la fois parmi plusieurs possibles

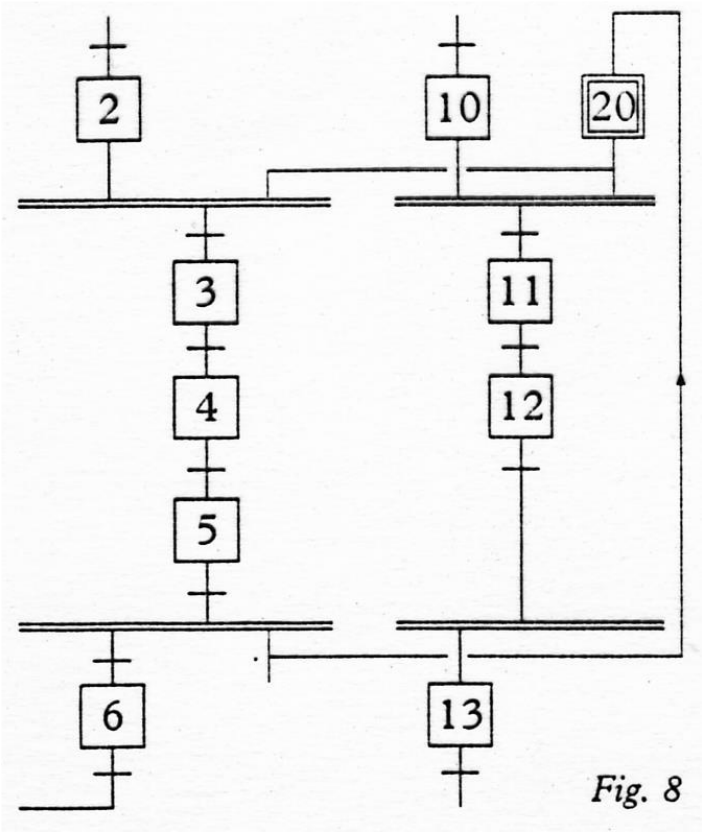


Fig. 8

GRAFCET: organisation

ETAPE DE SYNCHRONISATION

- Cette particularité se compose d'une étape initiale complémentaire et d'une étape normale.
- Elles synchronisent deux séquences dans des GRAFCETs différents.
- L'une des séquences doit obligatoirement se dérouler après l'autre.

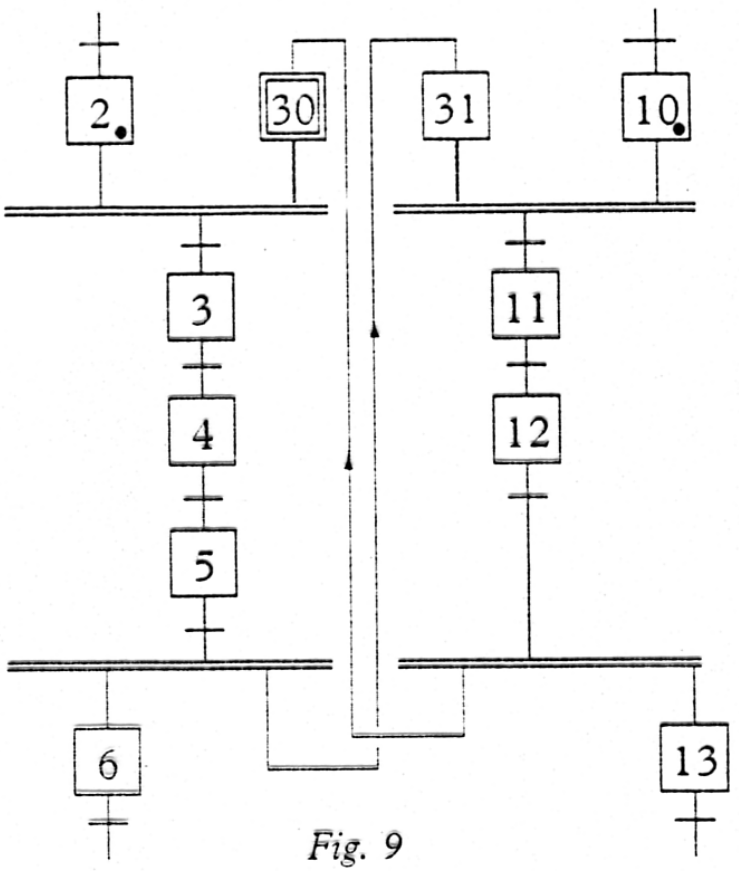
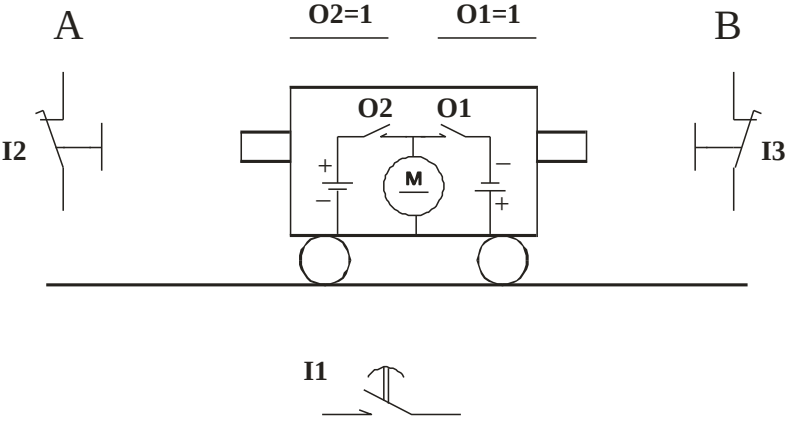


Fig. 9

Exemple (chariot)



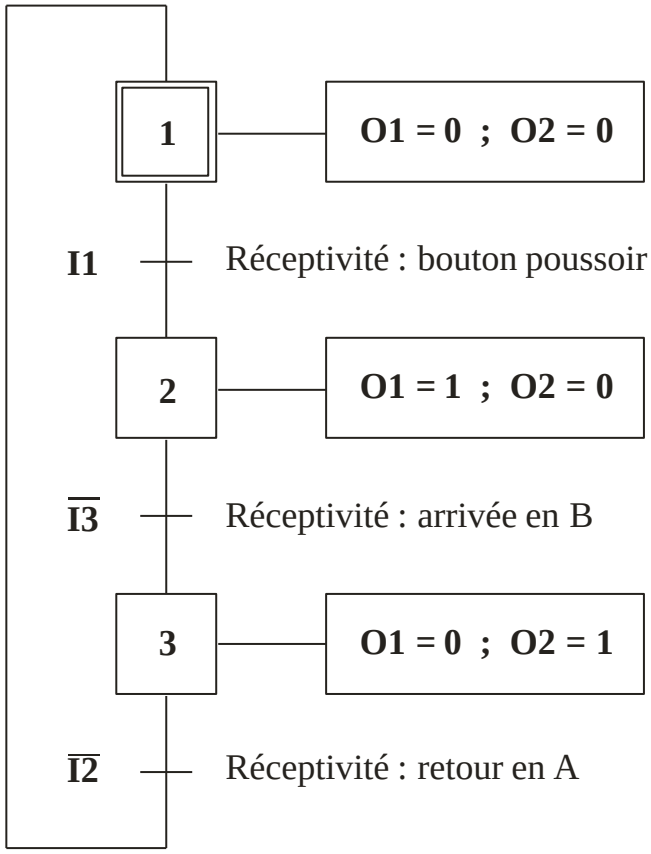
GRAFCET

Graphe de Commande Etape-Transition

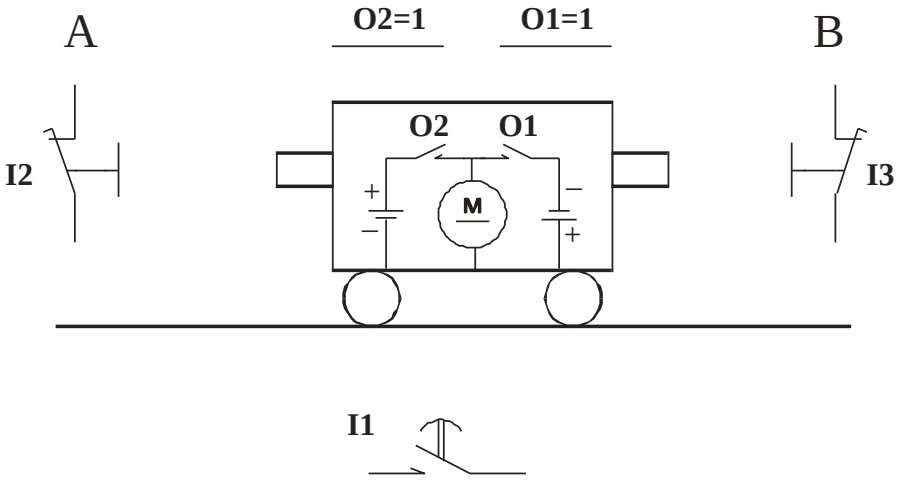
Etape 1 : Arrêt en A

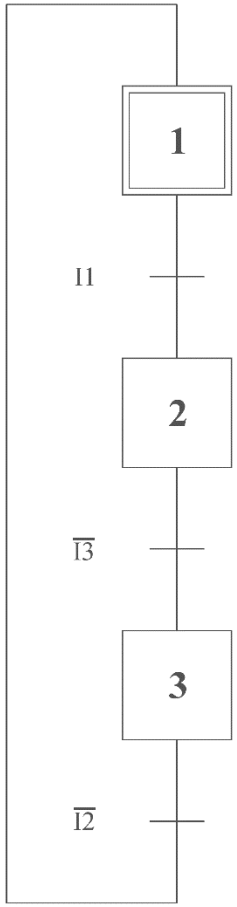
Etape 2 : Mouvement de A vers B

Etape 3 : Mouvement de B vers A

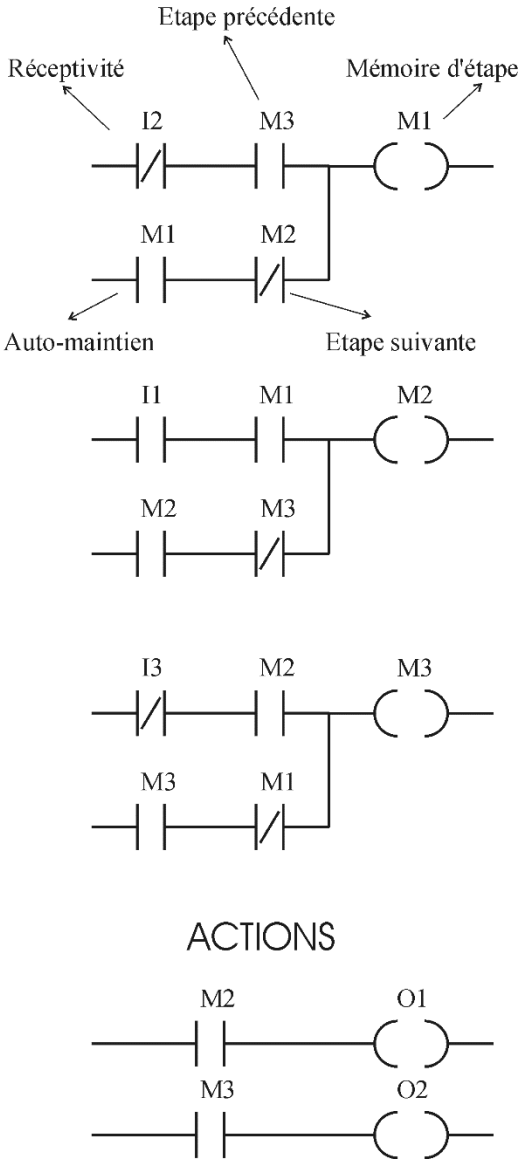


Exemple (chariot)



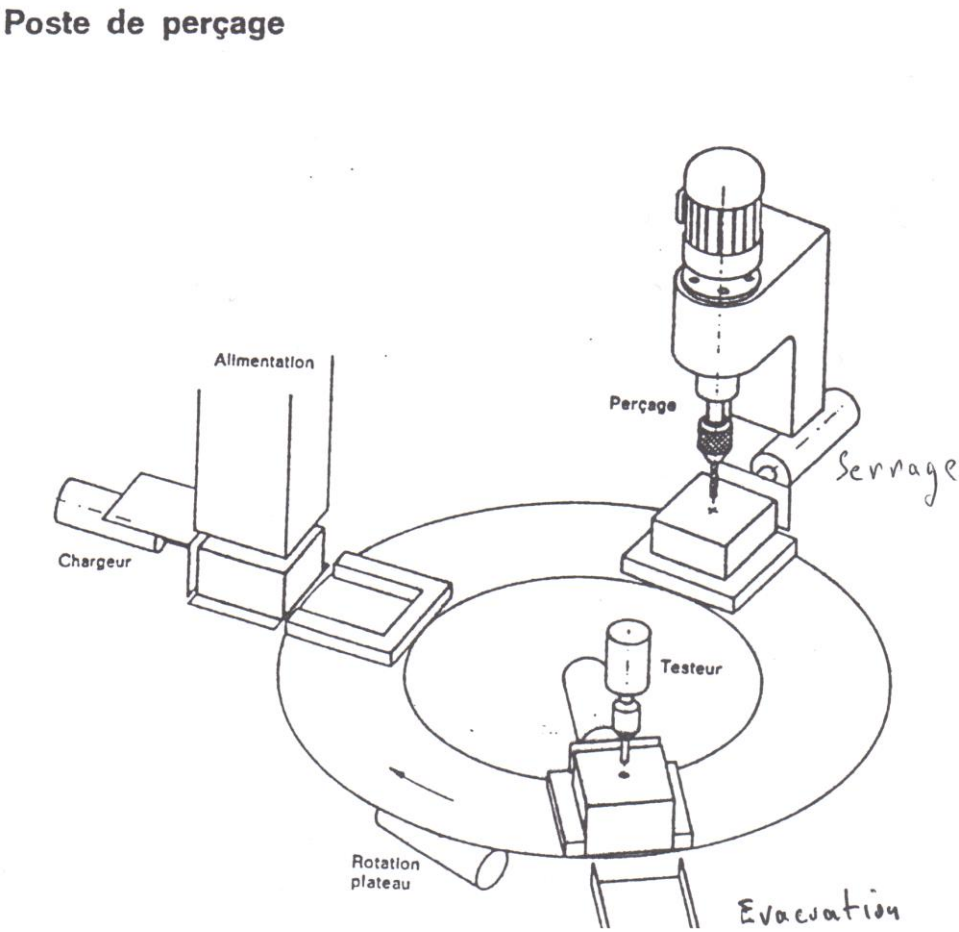


EVOLUTION DU GRAFCET



Exemple: Poste de perçage

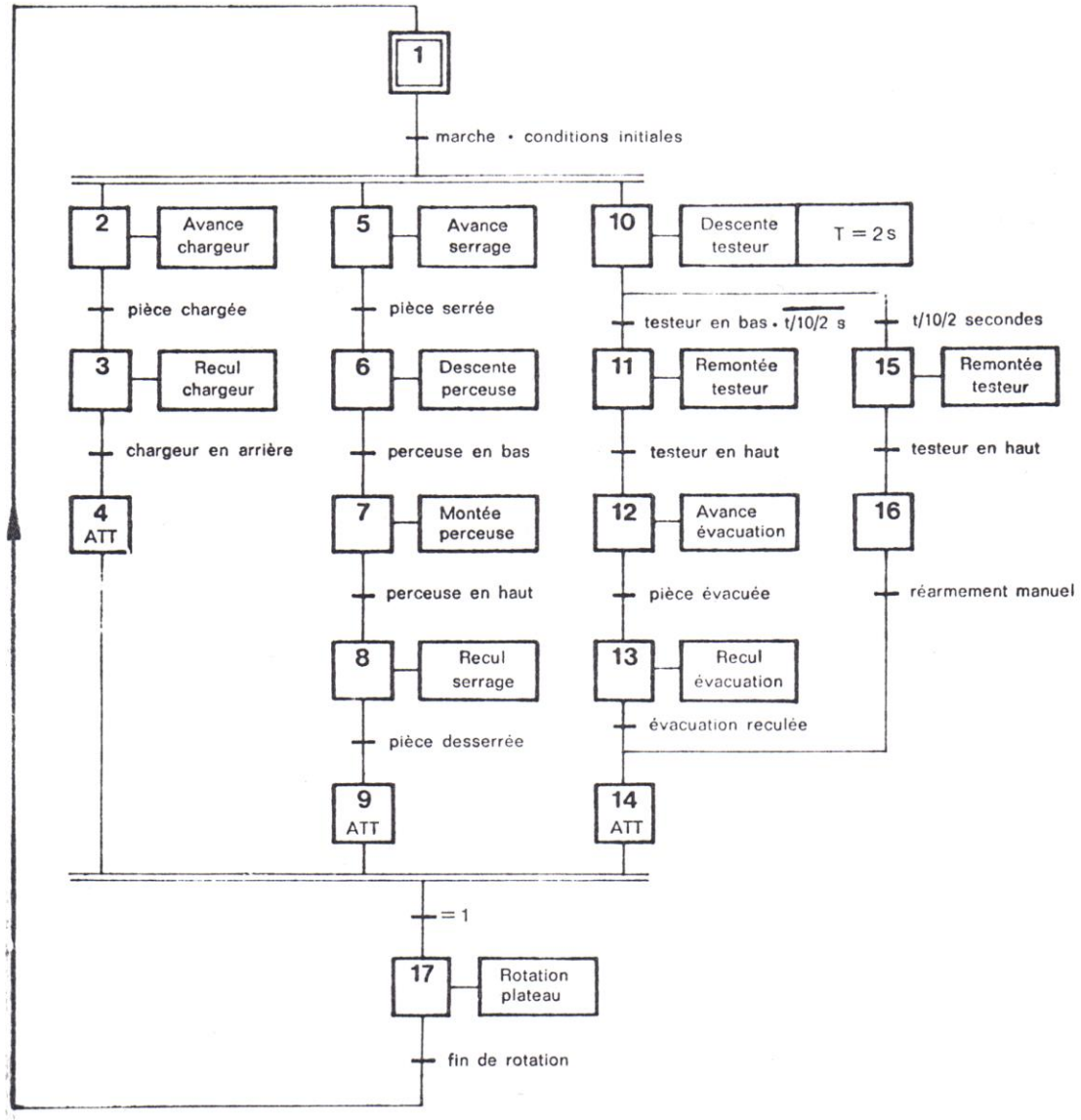
- On considère un poste de perçage comportant un système d'alimentation, une perceuse, un testeur et un plateau tournant
- On demande d'établir le GRAFCET de l'automatisme
  - Les différentes parties du poste de perçage doivent fonctionner en parallèle
  - Si le testeur n'est pas arrivé en position basse après 2 s, cela signifie que le trou a été mal percé. On prévoit alors un système permettant à l'opérateur de prendre la pièce à la main. Il relance le système avec un bouton poussoir
  - On suppose le système en régime. Il y a une pièce à chaque emplacement du plateau



Exemple: Poste de perçage

	Entrées	Actions
Alimentation	Pièce chargée	Avancer chargeur
	Chargeur en arrière	Reculer chargeur
Serrage	Pièce serrée	Avance serrage
	Pièce desserrée	Recul serrage
Perçage	Perceuse en bas	Descente perceuse
	Perceuse en haut	Montée perceuse
Testeur	Testeur en bas	Descente testeur
	Testeur en haut	Montée testeur
Evacuation	Pièce évacuée	Avance évacuation
	Evacuation en arrière	Recul évacuation
Plateau	Fin de rotation	Rotation plateau
Opérateur	Marche	Alarme
	Réarmement	

Exemple: Poste de perçage

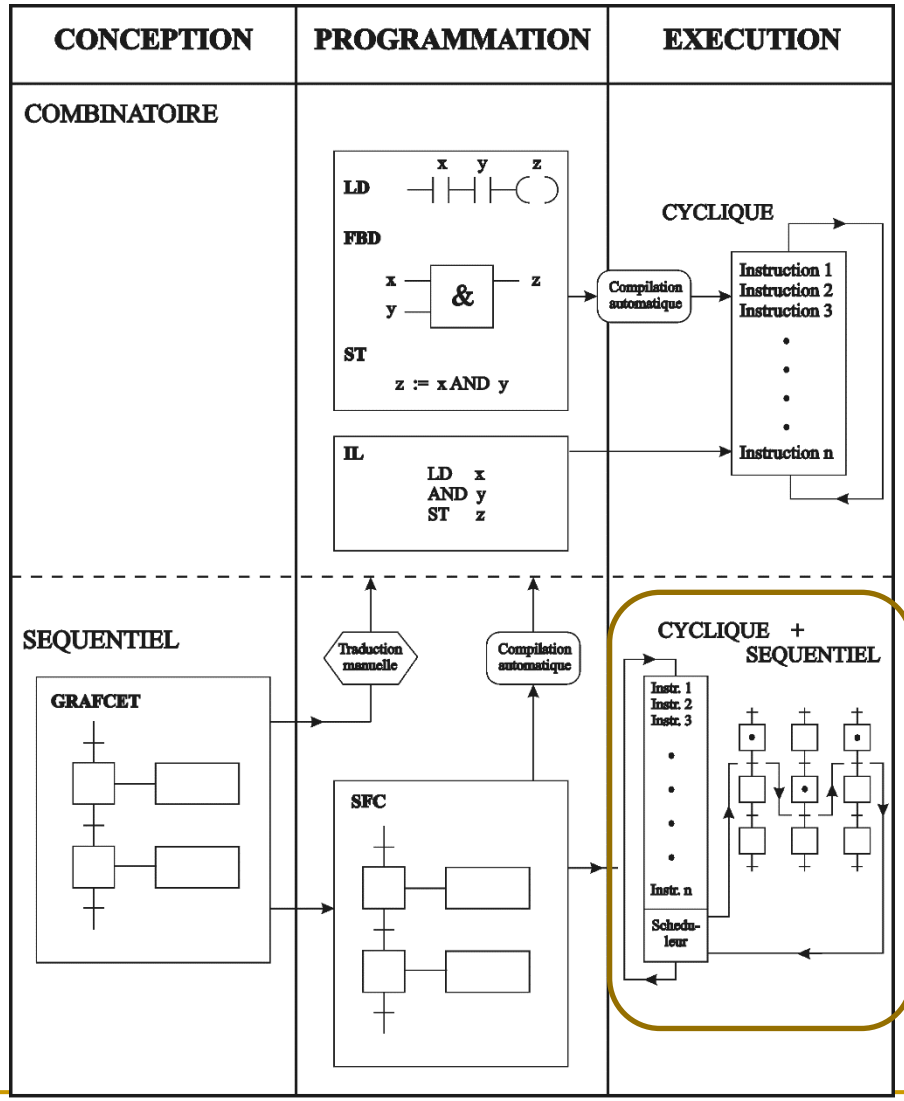


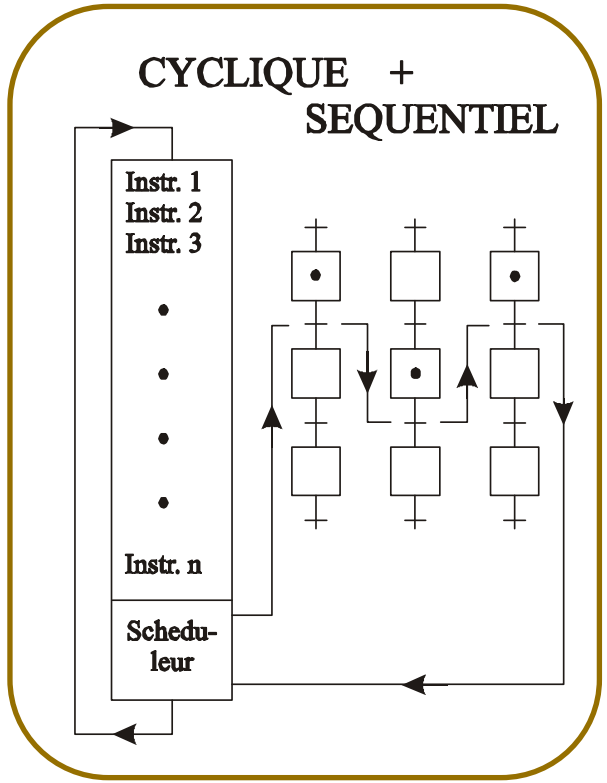
PROGRAMMATION DE CONTROLES SEQUENTIELS

Conception de contrôles séquentiels

- PROGRAMMATION
  - Il est parfaitement possible de transposer un GRAFCET en programme automate à partir d'un grand nombre de langages.
  - La programmation peut sembler assez lourde à première vue, mais l'intérêt réside principalement dans la résolution de problèmes assez complexes
  - Montrer clairement la séquence réalisée
  - Systématiser la conception des systèmes: conditions d'activation et de désactivation se construisent de manière systématique
  - Faciliter la mise au point, les tests, la maintenance

Conception de contrôles séquentiels

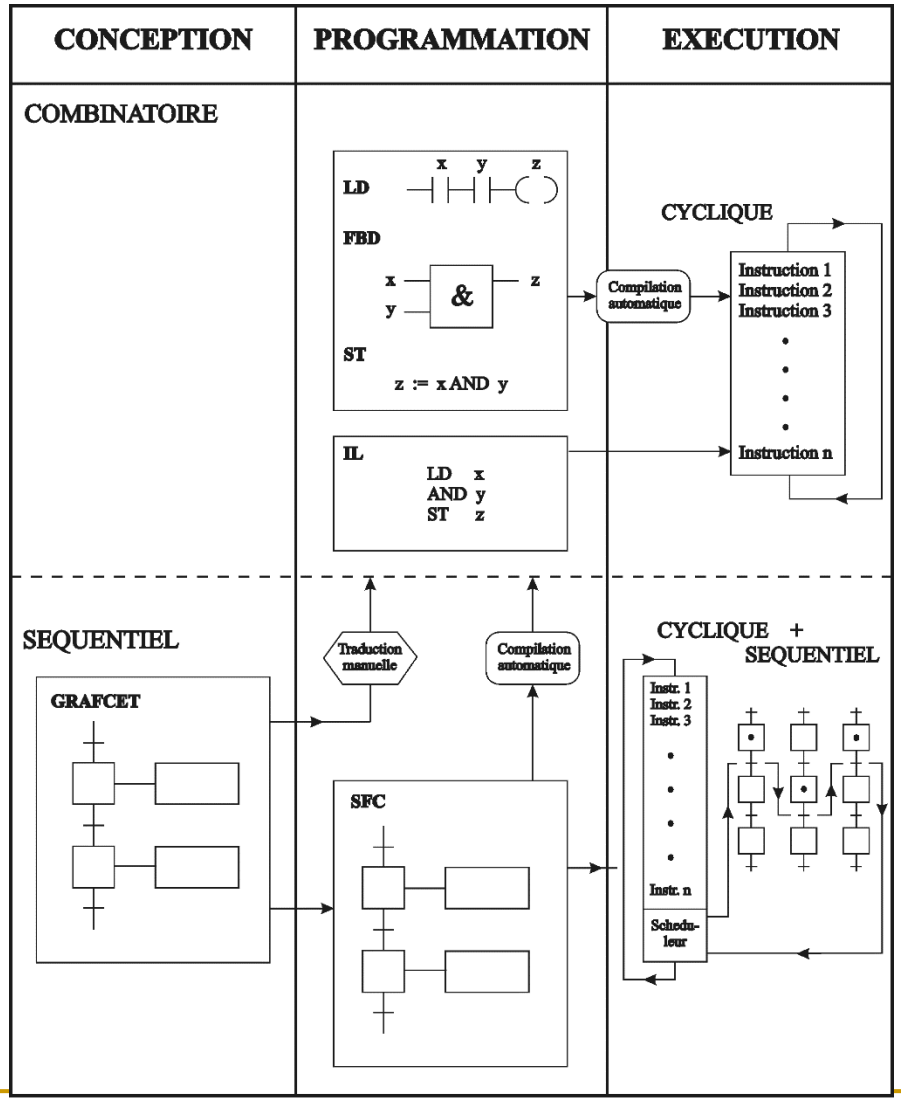




LD : Ladder Diagram **FBD** : Function Block Diagram **ST** : Structured Text

IL : Instruction List **SFC** : Sequential Function Chart

Conception de contrôles séquentiels



LD : Ladder Diagram **FBD** : Function Block Diagram **ST** : Structured Text

IL : Instruction List **SFC** : Sequential Function Chart

Conception de contrôles séquentiels

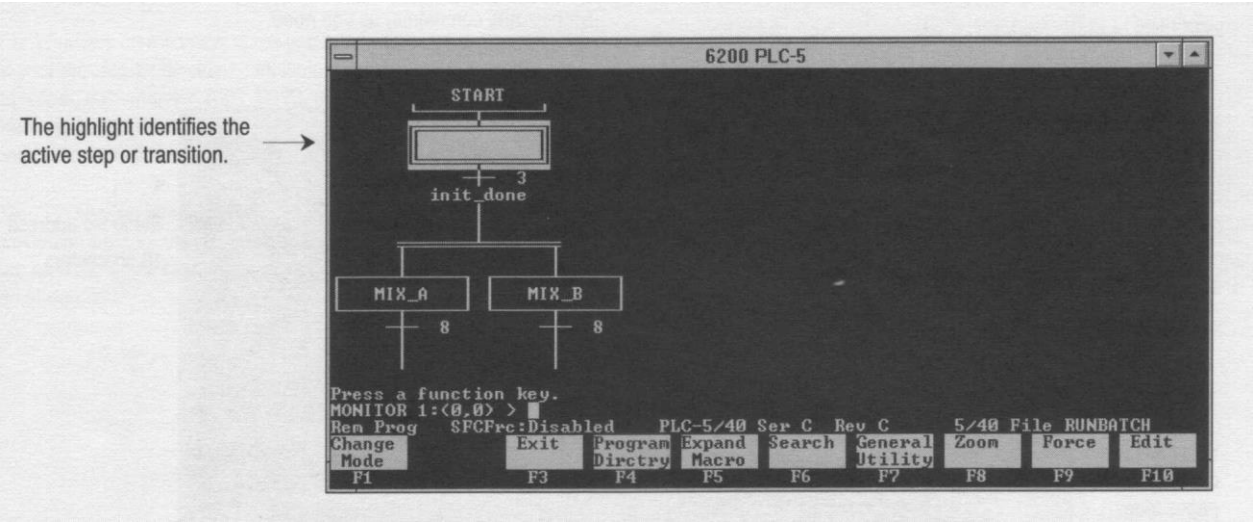
- PROGRAMMATION
  - Caractère détourné de l'approche:
    - Le mode de représentation séquentiel du problème de départ est transposé en un mode de représentation parallèle afin de pouvoir être programmé
    - La programmation donne lieu à une exécution purement séquentielle
  - La programmation de GRAFCETs est loin d'être simple et peut donner lieu à des biais et des pièges qui seront expliqués au chapitre 6.

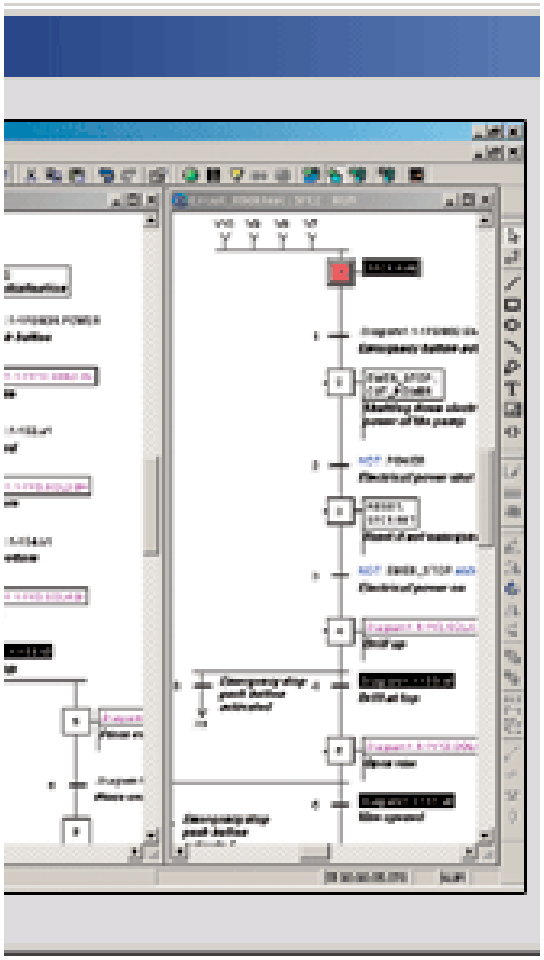
Langages séquentiels

- Etant donné l'importance des problèmes séquentiels en pratique, les automates sont pourvus d'outils de programmation séquentielle pour faciliter la tâche du programmateur

- Langage GRAFCET
  - De plus en plus de constructeurs offrent un mode de programmation graphique basé sur le GRAFCET
  - A l'exécution, les étapes actives apparaissent en surbrillance

Langages séquentiels





Exemple de langage de programmation en GRAFCET
Source : ALLEN-BRADLEY

Exécution séquentielle

- Deux prolongements possibles à l'outil d'analyse GRAFCET
  - 1/ Opérer la traduction du GRAFCET dans un langage parallèle traditionnel
  - 2/ Adapter l'exécution à la nature séquentielle du problème

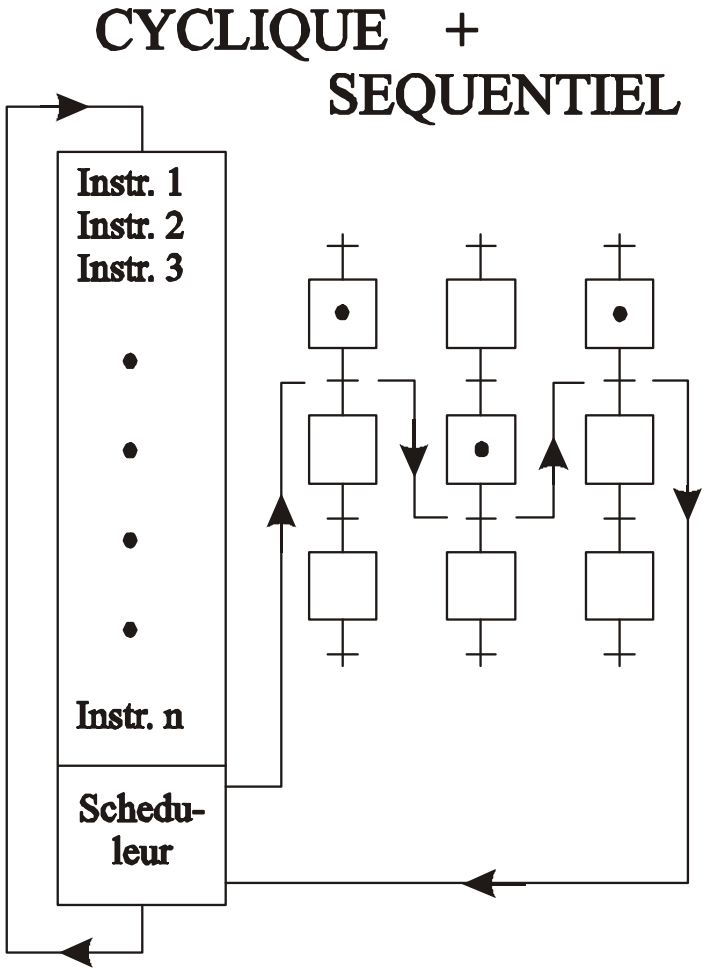
- 1/ ➔ Opérer la traduction du GRAFCET en langage parallèle
  - Exécution cyclique aveugle avec les inconvénients
    - Balayage de tout le programme
    - Scrutation de toutes les entrées
    - Recalcul de toutes les sorties
  - Travail bidirectionnel: au niveau du GRAFCET pour la conception et au niveau du programme pour les tests

Exécution séquentielle

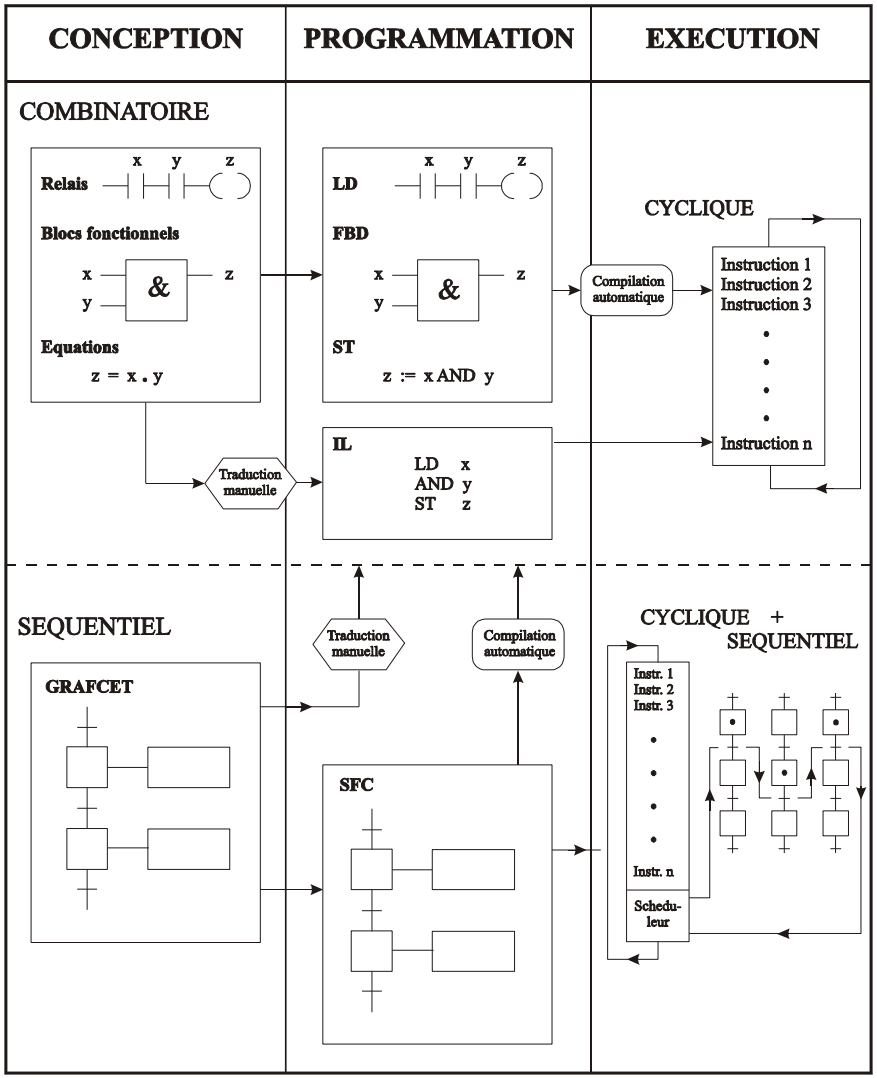
- 2/ ➔ Exécution de nature séquentielle
  - Langages GRAFCET
  - Le processeur ne vérifie que les conditions logiques qui intéressent l'évolution du système, c.-à-d. les réceptivités
  - Détermine les actions correspondantes à l'étape
  - Conséquences favorables sur le temps d'exécution: très grands programmes en mémoire tout en gardant un bon temps d'exécution

Exécution séquentielle

- En pratique dans un automate on trouve:
  - Plusieurs séquences se déroulant en parallèle
  - Des fonctions combinatoires (traitement d'alarmes)
- Partie combinatoire: placée dans une première branche exécutée linéairement et cycliquement
- Branches séquentielles: dans une série de branches réservées. Le scheduler vérifie la bonne évolution des branches, effectue les évolutions, mémorise l'état d'avancement.
- Cohérence entre la conception, la programmation et l'exécution d'un automatisme



Exécution séquentielle



LD : Ladder Diagram **FBD** : Function Block Diagram **ST** : Structured Text

IL : Instruction List **SFC** : Sequential Function Chart

GRAFCET: EXTENSION DE LA LOGIQUE DE BASE & FONCTIONS SPECIALES

GRAFCET: éléments complémentaires

- On associe généralement deux variables aux étapes:
  - La variable d'activité: variable booléenne gérée par le système qui vaut 1 si l'étape est active
    - Dans la norme CEI 1131-3, nom_étape.X
  - La variable de durée d'activité: variable de type temps gérée par le système qui indique depuis combien de temps l'étape est active
    - Dans la norme CEI 1131-3, nom_étape.T
    - Pas toujours disponible d'emblée sans déclaration a priori

- Remarque:
  - Ces variables sont une facilité pour la programmation.
  - On peut les recréer à partir de bits internes et de tempos

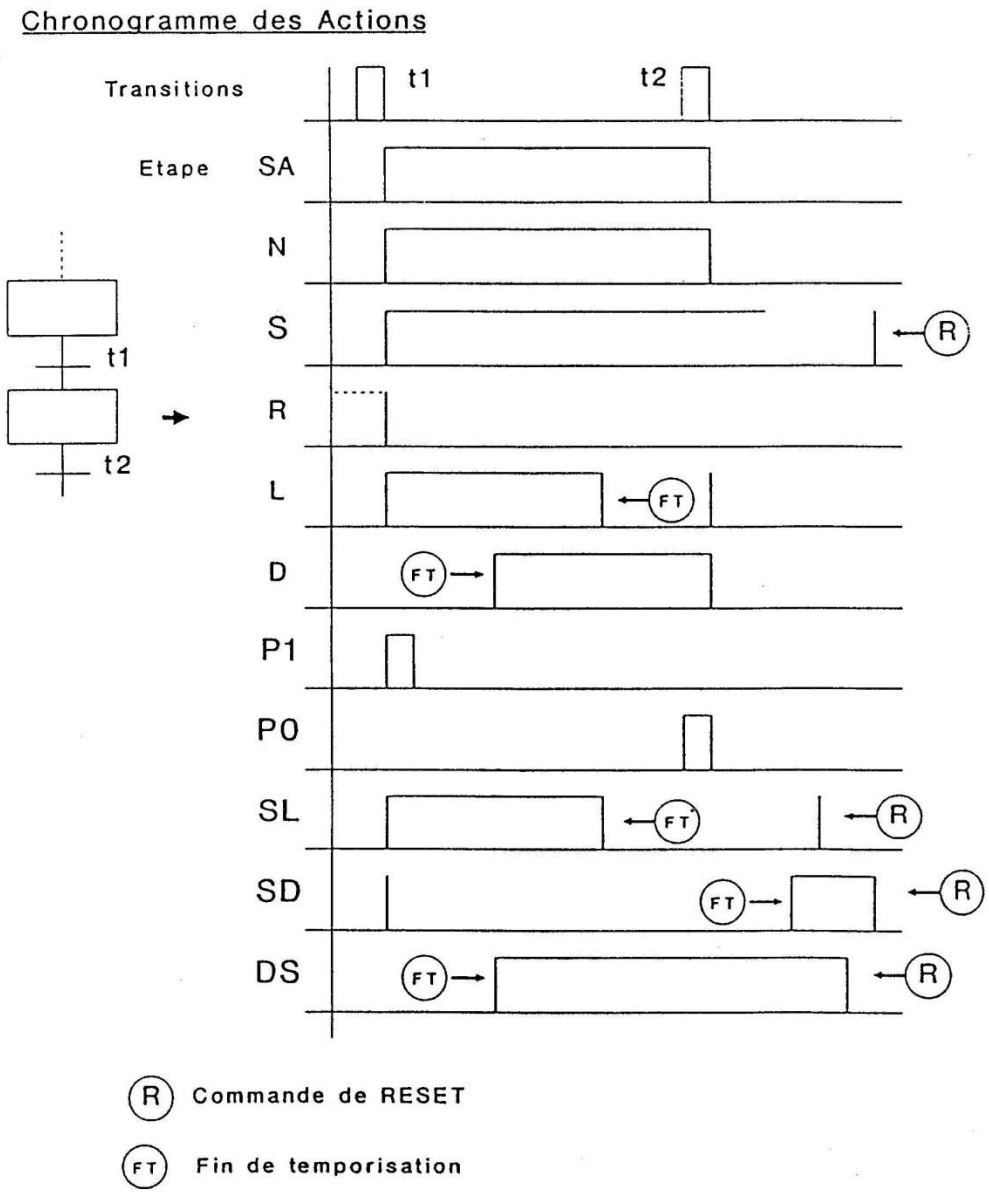
GRAFCET: éléments complémentaires

- TYPES D'ACTIONS:
  - On a supposé que les actions associées aux étapes étaient effectuées et maintenues durant toute la période d'activité de l'étape.
    - Ce sont des actions de types « non mémorisées »
  - Pour simplifier la tâche du programmateur, les constructeurs puis la norme CEI1131-3 ont introduit d'autres d'actions répondant à des besoins pratiques.
    - Ce sont des actions de types « mémorisées »
  - ➔ Extension des langages combinatoires
    - Fonction de relais bistables SR (set reset) ou relais à accrochage LATCH
    - Passe à 1 avec SET et y reste.
    - Retombe à 0 et y reste avec RESET
  - Par contre l'utilisation de ces actions rend plus difficile la lecture des GRAFCET
    - GRAFCET plus implicite
    - Difficulté de mise au point et de maintenance

GRAFCET: éléments complémentaires

- Mettre à la disposition du programmeur des fonctions couramment utilisées dans les automatismes
  - Bi stables
    - SR forcé dominant / RS réinitialisé dominant / SEMA sémaphore
  - Temporisateurs
    - Impulsion TP / temporisation à l'enclenchement TON / temporisation au déclenchement TOF / RTC horloge temps réel (date et heure)
  - Compteurs
    - Compteur CTU / décompteurs CTD / compteur-décompteur CTUD
  - Différentiateurs
    - Détecteurs de front montant R-TRIG ou descendant F-TRIG

GRAFCET: éléments complémentaires



Set / reset

Retard à la montée

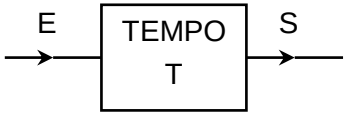
Différentiation +

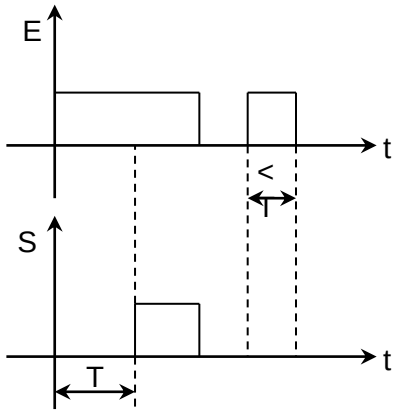
Différentiation -

Retard à la retombée

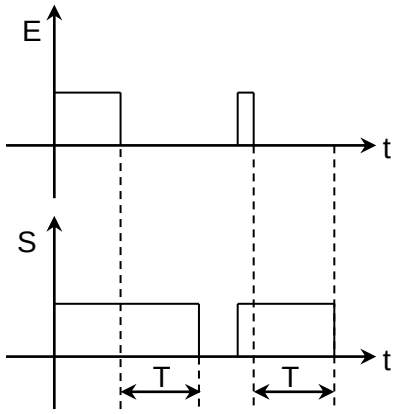
Temporisation

- La fonction temporisation permet de décaler un signal logique par rapport à un autre d'un délai fixé.
- Décalage à l'enclenchement ou au déclenchement
- Sur les automates modernes la temporisation est effectuée par une horloge interne à 100 ms, parfois à 10 ms





a. Retard à l'enclenchement

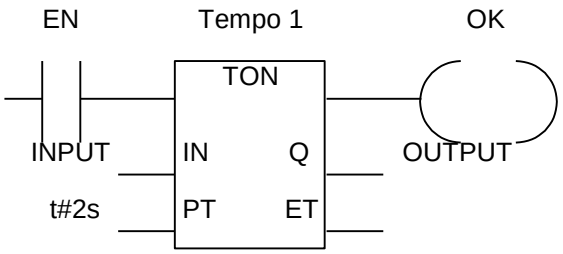


b. Retard au déclenchement

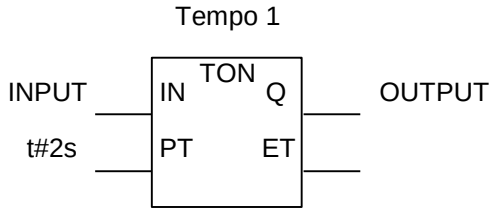
Temporisation

Programmation d'un bloc fonctionnel de temporisation dans les différents langages normalisés

LD



FBD



IL

```
Lancement     LD    t#2s
              ST    Tempo1.PT
              LD    INPUT
              ST    Tempo1.IN
              CAL   Tempo1

Utilisation   LD    Tempo1.Q
              ST    OUTPUT
```

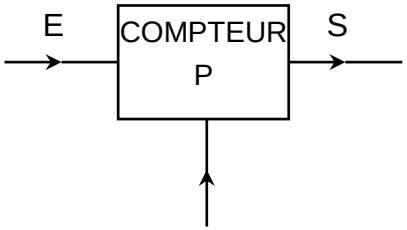
ST

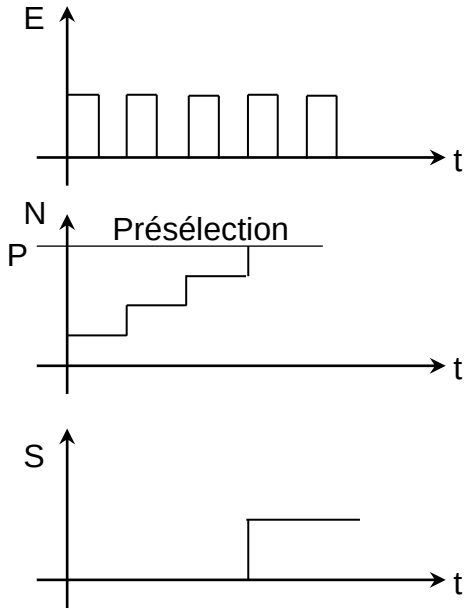
```
Lancement     Tempo1 (IN : = INPUT , PT : = t#2s)

Utilisation   OUTPUT : = Tempo1.Q
```

Comptage

- Les fonctions de comptage et de décomptage sont des fonctions couramment utilisées
- Il existe divers compteurs logiciels, mais ils ne peuvent compter que des impulsions espacées d'une durée significativement supérieure au temps de cycle de l'automate (10 à 100 ms)
- Pour les comptages rapides, on recourt à des cartes spéciales (horloges à plusieurs kHz)

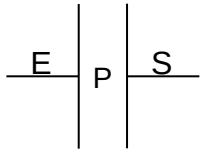


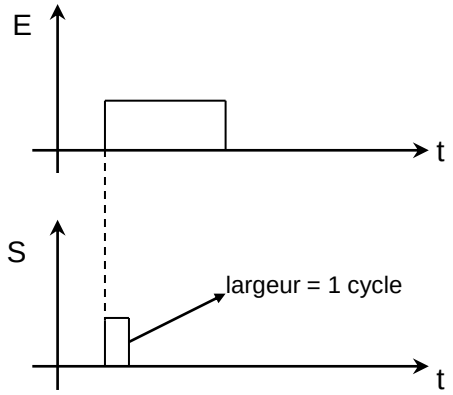


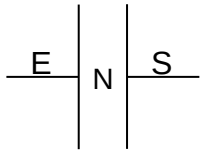
Compteur avec présélection: devient vrai lorsque le nombre d'entrées est dépassé.

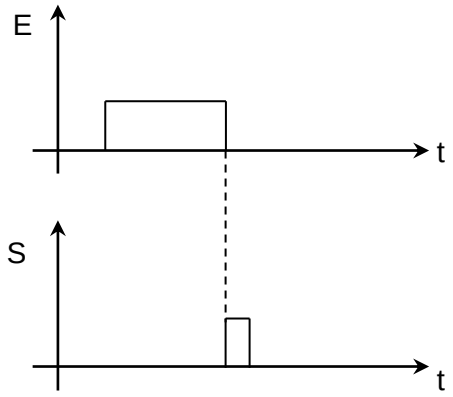
Différentiateurs

- Délivrent des impulsions d'une durée égale à un cycle d'automate lors du changement d'état d'une variable logique





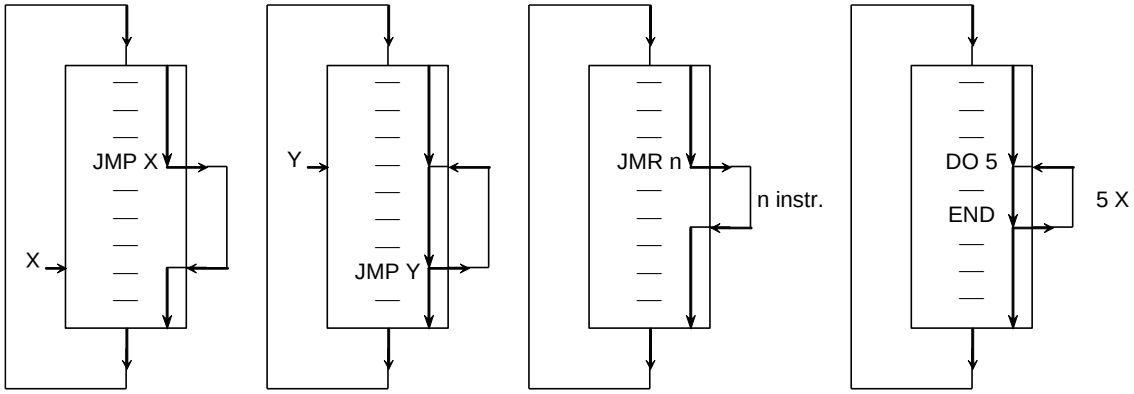




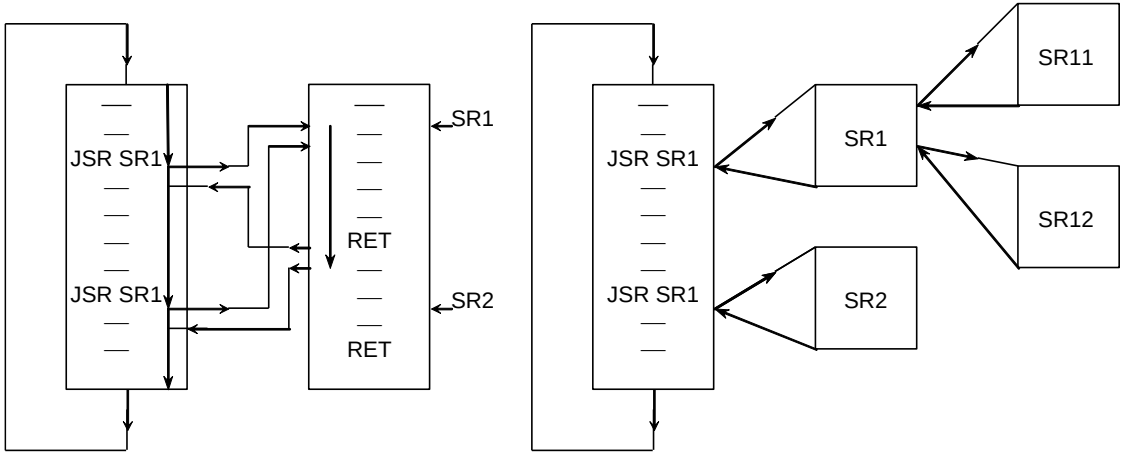
Fonctions d'organisation du cycle

- BRANCHEMENTS ET REPETITIONS
  - Fonctions qui permettent d'intervenir sur le déroulement du cycle de l'automate
  - Fonctions de branchement
    - Permettent de sauter conditionnement ou inconditionnellement une série d'instructions
    - Saut avant / saut arrière / saut absolu / saut relatif (nombre d'instructions)
  - Boucles de répétition
    - Permettent de répéter un groupe d'instructions un nombre de fois donné

Fonctions d'organisation du cycle



a. Saut direct b. Saut inverse c. Saut relatif d. Boucle DO



e. Sous-routines (économie mémoire) e. Sous-routines (structuration des programmes)

Fonctions d'organisation du cycle

- SOUS ROUTINES
  - A pour effet de faire abandonner la séquence d'instructions en cours par le processeur au profit d'une séquence d'instructions (la sous routine) située ailleurs dans la mémoire.
  - La deuxième séquence se termine par une instruction de retour RETURN qui ramène le processeur à l'endroit où il avait quitté la séquence initiale
- Deux avantages
  - Économiser la place mémoire
    - Séquence reproduite plusieurs fois mais une seule fois présente en mémoire
  - Structurer les programmes
    - Décomposer les problèmes complexes en sous ensembles plus simples, correspondant par ex. à des entités technologiques
    - Structure modulaire et hiérarchique
    - Faciliter la maintenance et la lisibilité

Fonctions d'organisation du cycle

- INTERRUPTIONS
  - L'automate ne scrute le monde extérieur que de manière cyclique
  - Pour les évènements urgents, cela demanderait des temps de cycle très courts et donc des automates surpuissants.
  - Les procédures d'interruptions sont des solutions beaucoup plus efficaces.
  - Les sources d'interruptions sont connectées individuellement ou par une fonction OU à des lignes spéciales du bus, les lignes d'interruption.
  - Le processeur scrute automatiquement les lignes d'interruption après chaque instruction (quelques microsecondes).
  - Si on détecte une demande d'interruption, on interrompt l'exécution du programme, on réalise les traitements désirés puis on reprend l'exécution du programme interrompu.
  - Au moment de l'interruption on a accès direct à tous les I/O.

GRAFCET: éléments complémentaires

- NATURE DES ACTIONS
  - Le plus souvent, une action = changer une ou plusieurs variables booléennes
  - La norme permet également de réaliser des opérations arithmétiques ou autres dans un langage quelconque de l'automate
  - Ces instructions sont effectuées à chaque cycle pendant la durée de l'action
  - Une action peut être constituée d'un autre GRAFCET.
    - Démarrage au moment du démarrage de l'action
    - Tué lorsque l'action est arrêtée (toutes les actions désactivées)

References

- Hugh Jack. Automating Manufacturing Systems with PLCs. 7th edition.