Réalisation technologique du GRAFCET

par **Daniel DUPONT**
Docteur en automatique
Enseignant chercheur à l'École supérieure des techniques industrielles et des textiles (ESTIT)

et **David DUBOIS**
Docteur en automatique et informatique industrielle
Chargé de cours à l'ESTIT

1.	**Vers la réalisation technologique**	S 8 032 – 2
1.1	Du GRAFCET fonctionnel au GRAFCET technologique	— 2
1.2	Cas particuliers	— 3
1.3	Exemple d'application	— 5
1.4	Dialogue entre GRAFCET	— 6
1.5	Conclusion	— 9
2.	**Équation logique d'une étape**	— 10
2.1	Rappel des règles d'évolution	— 10
2.2	Équation logique d'une étape	— 11
2.3	Choix de séquences	— 12
2.4	Séquences parallèles	— 12
2.5	Cas particulier : GRAFCET ou boucle à deux étapes	— 13
2.6	Gestion des actions	— 14
2.7	Prise en compte des modes de marche/arrêt	— 15
3.	**Technologies de réalisation**	— 17
3.1	Logique câblée	— 17
3.2	Automates programmables industriels (API)	— 20
3.3	Système informatique	— 22
3.4	Conclusion	— 24
Pour en savoir plus		Doc. S 8 032

Afin de réaliser l'implantation technologique du GRAFCET sur différents supports (câblés ou programmés), une conception en trois temps est préconisée :

*— l'**analyse fonctionnelle** qui débouche sur l'élaboration d'un GRAFCET indépendant de la réalisation technologique ;*

*— la **transcription** du GRAFCET en équations logiques utilisables lors de la réalisation technologique ;*

*— l'**implantation des équations logiques** sur le support de réalisation câblé ou programmé.*

Cette démarche met en évidence les difficultés de conception fonctionnelle pour des applications complexes et propose une solution par l'intermédiaire d'une décomposition en modules moins complexes et de dialogues entre les GRAFCET les décrivant.

La partie fonctionnelle ne prenant en compte que le mode de fonctionnement normal, la notion des modes de marche/arrêt est intégrée dans l'élaboration des équations logiques et de ce fait intervient dans la réalisation technologique.

Cet article vient en complément des articles sur le GRAFCET, notamment *Outil de description des automatismes séquentiels : le GRAFCET* [R 7 250].

1. Vers la réalisation technologique

1.1 Du GRAFCET fonctionnel au GRAFCET technologique

Le GRAFCET (graphe de commande étapes-transitions) est un outil graphique de représentation du cahier des charges d'un automatisme séquentiel. Il est à la fois simple à utiliser et rigoureux sur le plan formel.

Il est basé sur les notions d'**étapes** auxquelles sont associées des **actions** et de **transitions** auxquelles sont associées des **réceptivités**. Il décrit les ordres émis par la partie commande vers la partie opérative en mettant en évidence les actions engendrées et les événements qui les déclenchent. Cette représentation est étroitement liée à la notion d'évolution du processus.

Pour parvenir à la réalisation de la partie commande, il est conseillé de diviser le cahier des charges en deux niveaux successifs et complémentaires.

Le premier niveau décrit le **comportement de la partie commande** en fonction de l'évolution de la partie opérative : c'est le rôle des spécifications fonctionnelles décrivant ce que doit faire l'automatisme. À ce niveau, les contraintes technologiques (des actionneurs, des capteurs…) n'interviennent pas, seules comptent les fonctionnalités.

Le second niveau renseigne sur la **nature technologique des actionneurs et des capteurs** ; on y trouve donc leurs caractéristiques et les contraintes qui y sont associées. C'est aussi à ce niveau que les spécifications opérationnelles décrivant les conditions d'utilisation de l'application apparaissent.

L'automaticien, confronté à un problème de conception et de réalisation d'un automatisme, aborde donc l'étude en deux phases successives correspondant aux deux niveaux de spécification :

— un niveau fonctionnel ;
— un niveau technologique.

Cette approche en deux niveaux se retrouve dans la conception du GRAFCET :

— un premier GRAFCET dit fonctionnel ou de niveau 1, qui ne prend en compte que la partie fonctionnelle des spécifications et qui fait donc abstraction de toute réalisation technologique. Ainsi, s'il est bien conçu, il est valable pour tout type de réalisation ;
— un deuxième GRAFCET dit technologique ou de niveau 2, qui, en s'appuyant sur le GRAFCET de niveau 1, intègre les contraintes technologiques et opérationnelles.

Exemples :

Une application simplifiée, la **commande d'une presse** (figure **1**), permet d'illustrer cette approche.

Sur ordre (*m*) de l'opérateur, la presse descend (D), puis une fois arrivée en bas (*b*), elle remonte (M) jusqu'à la position haute (*h*). Le GRAFCET fonctionnel de cette application est fourni par la figure ***1b***. Il est dit GRAFCET de niveau 1 car il ne fait intervenir que l'aspect fonctionnel sans tenir compte d'aucune contrainte technologique (la technologie utilisée pour les actionneurs et les capteurs n'est pas utile à ce niveau).

Supposons maintenant que la technologie utilisée soit une technologie pneumatique et que la montée-descente de la presse soit commandée par un **vérin double effet** (figure **2**). L'arrivée d'air en A+ (resp. en A–) provoque la sortie du vérin et donc la descente de la presse (resp. la rentrée du vérin et donc la montée de la presse). Le GRAFCET technologique de cette application est la figure ***2b***.

Supposons maintenant que la montée-descente de la presse soit commandée par un **vérin simple effet** (figure **3**). L'arrivée d'air en A+ provoque la sortie du vérin et comprime le ressort. Le ressort en l'absence d'air en A+ provoque la rentrée du vérin et donc la montée de la presse. La figure ***3b*** fournit le GRAFCET technologique. L'étape 2 du GRACET de la figure ***3b*** ne possède aucune action associée car le ressort joue son rôle de rappel et la partie commande n'a aucune action à envoyer vers la partie opérative. Ce GRAFCET, possédant deux étapes (2 et 0) qui se suivent sans action associée, peut se transformer (comme bien souvent) en supprimant l'étape 2 (figure ***3c***). Dans ce cas, il ne faut pas oublier d'ajouter la condition *h* (qui peut être considérée comme la condition initiale) dans la réceptivité associée à la transition qui suit l'étape 0 ; sinon, l'opérateur peut réappuyer sur le bouton de marche (*m*) pendant le retour du vérin (donc étape 0 active), ce qui fait ressortir le vérin bien que la rentrée ne soit pas terminée.

Ce dernier exemple est cependant particulier car bien souvent, le GRAFCET de niveau 2 possède plus d'étapes que le GRAFCET de niveau 1. En effet, les spécifications technologiques et opérationnelles amènent des contraintes supplémentaires. Toutefois, il montre que le GRAFCET de niveau 2 peut être différent du GRAFCET de niveau 1.

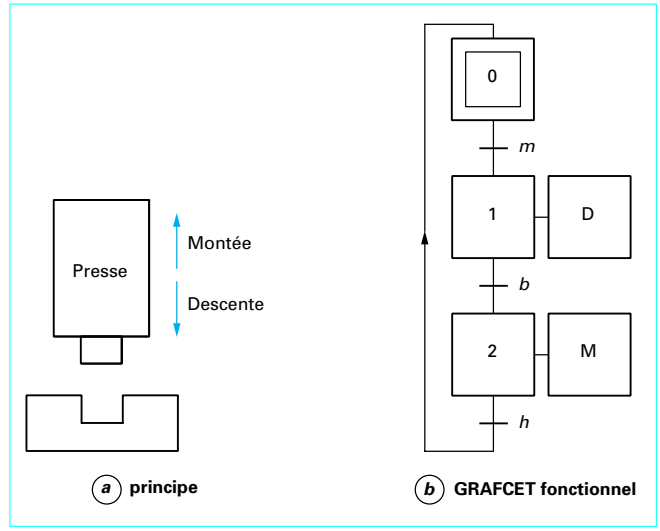


Figure 1 – Commande d'une presse

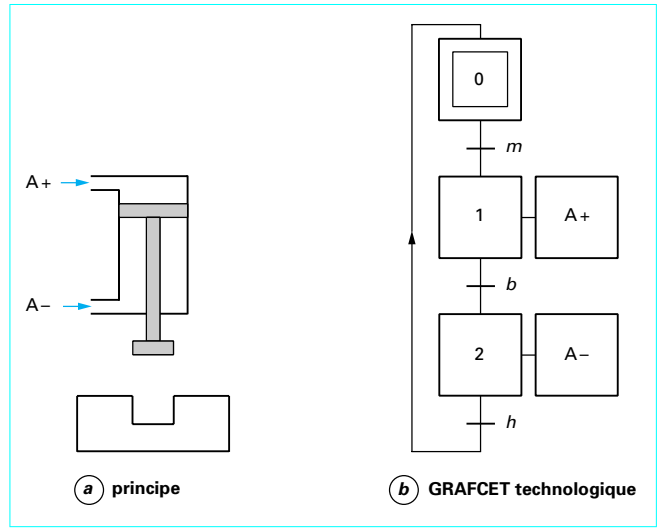
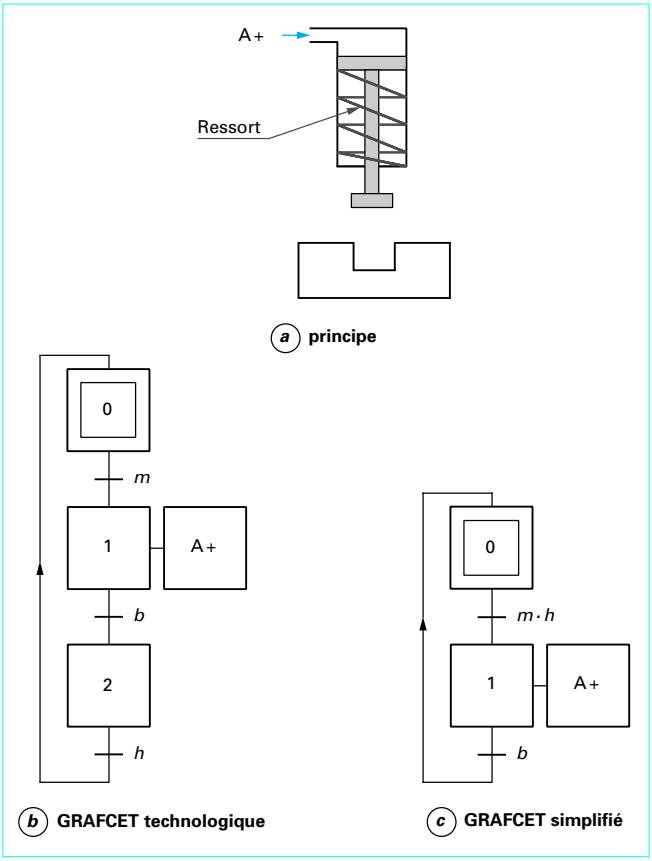


Figure 2 – Commande d'un vérin double effet

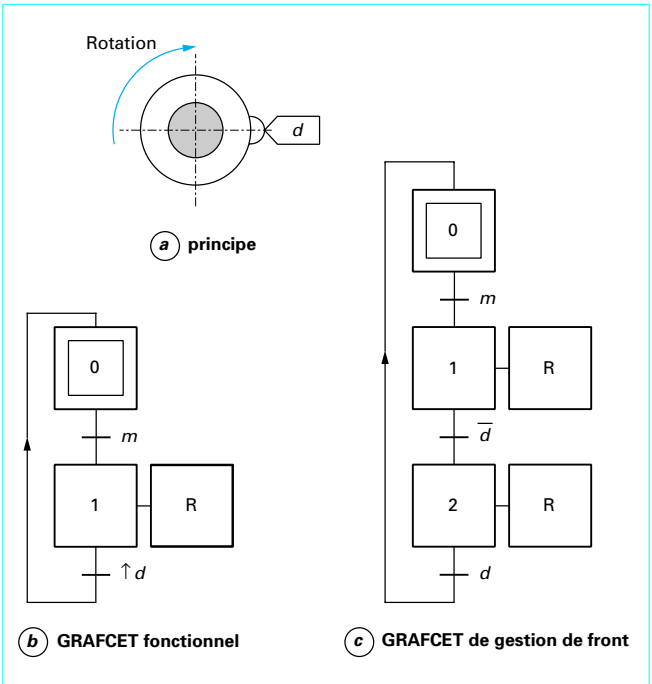


(a) principe

(b) GRAFCET technologique

(c) GRAFCET simplifié

Figure 3 – Commande d'un vérin simple effet



(a) principe

(b) GRAFCET fonctionnel

(c) GRAFCET de gestion de front

Figure 4 – Commande d'un moteur

1.2 Cas particuliers

Il est bien souvent intéressant, sinon indispensable, de modifier le GRAFCET de niveau 1 afin d'être sûr d'avoir un GRAFCET réellement **indépendant** de la technologie et donc de ne pas influencer les choix technologiques (si la partie opérative n'est pas déjà réalisée) et aussi d'être sûr d'avoir un GRAFCET qui répondra à la technologie (si la partie opérative est déjà réalisée).

1.2.1 Gestion des fronts

Dans certains GRAFCET, apparaissent au niveau des réceptivités des informations qui font intervenir l'apparition ou la disparition d'événements plutôt que de tester la présence ou l'absence d'une information. C'est le cas lorsque l'information est déjà présente avant que l'action soit commandée.

Exemple : commande d'un moteur (figure **4**)

Lorsque l'opérateur demande la rotation, l'information *d* (capteur de position) est déjà vraie. De fait, si la rotation ne doit faire qu'un tour, on ne peut pas tester *d* comme fin d'action ; il faut tester l'**apparition** de l'information *d*. Cela est fait en testant ce qu'on appelle un front montant sur *d*. La figure **4*b*** présente le GRAFCET fonctionnel. La flèche montante devant l'information *d* affirme que c'est le passage du niveau 0 (faux) au niveau 1 (vrai) qui permet de franchir la transition entre l'étape 1 et l'étape 0 : on parle de *front montant*.

D'un point de vue technologique, il n'est pas possible de tester directement les fronts. Il est nécessaire de les décomposer en deux étapes :

— pour un front montant (apparition d'information), on teste d'abord le niveau bas puis le niveau haut ;

— pour un front descendant (disparition d'information), on teste d'abord le niveau haut puis le niveau bas.

L' analyse du GRAFCET de niveau 2 conduit au GRAFCET de la figure **4*c***. L'étape 1 est doublée afin de pouvoir tester en premier lieu l'absence de l'information *d* (réceptivité de la transition entre l'étape 1 et 2), puis la présence de l'information *d* (réceptivité entre l'étape 2 et 0), ce qui revient à tester l'apparition de l'information *d*.

Nota : dans un tel cas, il ne faut pas oublier de répéter les actions dans l'étape qui est « doublée ».

1.2.2 Exclusivité au niveau des divergences

Certains GRAFCET de niveau 1 présentent des choix de séquences sans gérer l'exclusivité de ces choix (figure **5**) Or, le GRAFCET autorise l'activation de plusieurs branches.

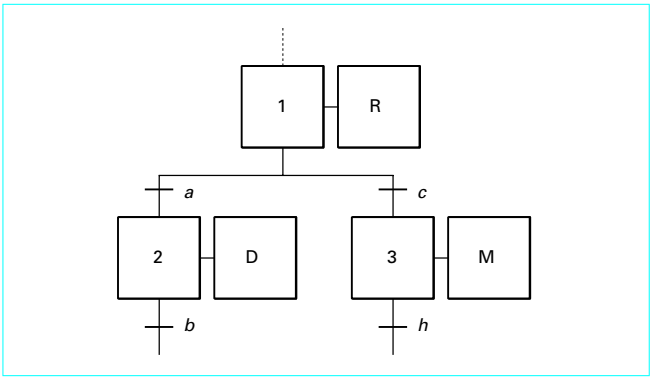
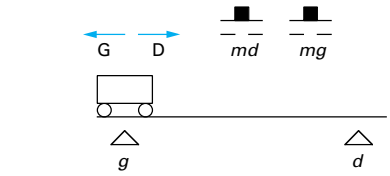
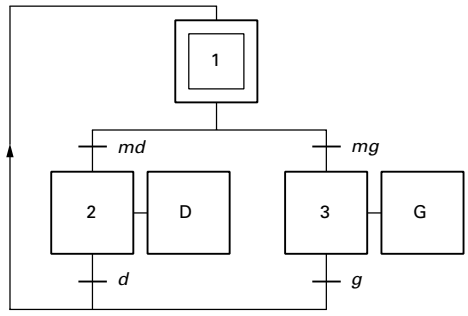


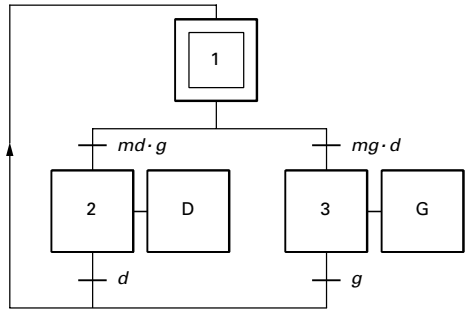
Figure 5 – Début de divergence



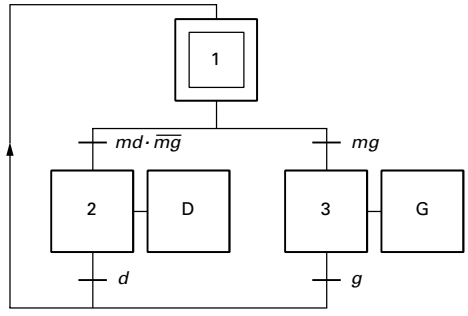
(a) principe



(b) activation simultanée possible



(c) interdiction d'activation simultanée



(d) priorité au déplacement à gauche

Figure 6 – Commande du déplacement latéral d'un chariot

Lorsque l'étape 1 est active et si les réceptivités *a* et *c* sont vraies simultanément, alors la transition entre les étapes 1 et 2 et la transition entre les étapes 1 et 3 sont simultanément franchissables. La règle 4 d'évolution du GRAFCET (§ 2.1) nous dit alors qu'elles sont simultanément franchies, donc les étapes 2 et 3 sont activées simultanément et les actions associées aux étapes 2 et 3 (D et M) sont envoyées simultanément à la partie opérative.

Exemple : soit un **chariot** (figure **6**) pouvant se déplacer entre deux positions (droite et gauche). Si au départ l'opérateur demande le déplacement à droite (*md*), le chariot se déplace à droite (D) jusqu'à la position droite (*d*) et s'il demande le déplacement à gauche (*mg*), le chariot se déplace à gauche (G) jusqu'à la position gauche (*g*).

Si au départ (figure **6*b***) l'opérateur appuie simultanément sur *md* et *mg*, alors les étapes 2 et 3 sont activées simultanément (règle 4 d'évolution du GRAFCET, § **2.1**). De fait, les actions D et G sont vraies simultanément et on demande au chariot de se déplacer vers la droite et vers la gauche simultanément, ce qui évidemment n'est pas une solution viable pour la partie opérative (risque de détérioration de l'actionneur). Il faut donc interdire, dans le GRAFCET, le fait que les étapes 2 et 3 puissent être activées simultanément (figure **6*c***).

Le fait de mettre $md \cdot g$ (resp. $mg \cdot d$) comme réceptivité entre les étapes 1 et 2 (resp. 1 et 3) interdit l'activation de l'étape 2 (resp. étape 3) et donc la demande de déplacement à droite (resp. à gauche) si le chariot n'est pas à gauche (resp. à droite). Comme le chariot ne peut pas être physiquement à droite et à gauche simultanément, il y a exclusivité dans le choix de séquence.

Pour certaines applications, il ne peut pas y avoir de choix exclusif basé sur les propriétés physiques de la partie opérative ; dans ce cas, il est possible de **donner une priorité à une séquence** (figure **6*d***).

Ici, le fait de mettre $md \cdot \overline{mg}$ entre l'étape 1 et 2 fait que, si l'opérateur appuie simultanément sur *md* et *mg*, alors cette réceptivité est fausse et donc uniquement l'étape 3 sera activée. La priorité est donc donnée à la séquence de droite.

1.2.3 Gestion des simultanéités en fin de convergence

Soit la partie de GRAFCET de la figure **7*a***. Lorsque les étapes 10 et 20 sont actives simultanément, alors les actions associées (A et B) sont vraies simultanément. La désactivation des étapes ne se fera que lorsque les informations *a* (fin de l'action A) et *b* (fin de l'action B) seront vraies simultanément.

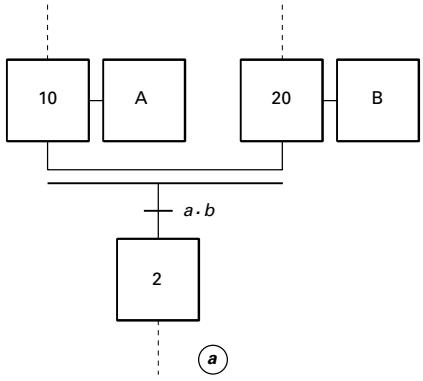
Que se passe-t-il si l'une des actions se termine avant l'autre (par exemple, l'action A) ?

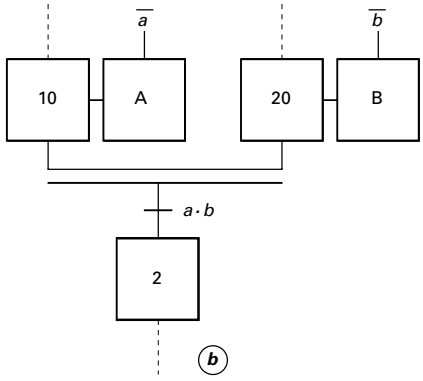
Compte tenu du GRAFCET, la partie commande continue à envoyer cette action à la partie opérative, ce qui peut être dommageable (actionneur en butée). Il est donc nécessaire de trouver une parade à ce problème. Le fait d'associer des actions conditionnelles aux étapes 10 et 20 le règle (figure **7*b***) car effectivement l'action A (resp. B) s'arrête quand l'information *a* fin de l'action A (resp. *b* fin de l'action B) est vraie.

Toutefois, un autre problème persiste : même si d'un point de vue fonctionnel, cette solution est tout à fait correcte, elle implique des contraintes au niveau de la réalisation technologique des capteurs (*a* et *b*) de fin d'action.

En effet, supposons que l'action A corresponde par exemple à un déplacement de chariot. Lorsque ce déplacement entraîne le chariot en position *a*, le capteur passe à 1 et la partie commande stoppe l'envoi du déplacement. Toutefois, par inertie, le chariot ne s'arrête pas immédiatement et risque de dépasser la zone de détection du capteur *a*. De fait, *a* repasse à 0 et l'action A est de nouveau vraie car l'étape 10 est toujours active. Donc le chariot repart en s'éloignant de la position *a* et la réceptivité $a \cdot b$ ne pourra plus être vraie. De fait, les étapes 10 et 20 resteront toujours actives, entraînant les actions A et B (d'où risque de butée, de casse, etc.). Il faut utiliser ici des capteurs à contacts maintenus.

La solution (figure **7*c***) consiste à ajouter des **étapes de synchronisation** en fin de séquences simultanées qui garantissent un GRAFCET de niveau 1 correct sans implication sur une quelconque réalisation technologique au niveau des capteurs *a* et *b*.





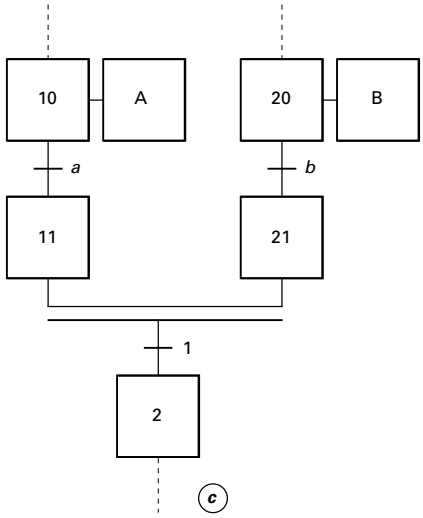


Figure 7 – Fin de divergence

Lorsque l'étape 10 est active, l'action A est vraie jusqu'au moment où l'information *a* est vraie. Dans ce cas, la transition entre l'étape 10 et 11 est franchie, l'étape 11 est activée et l'étape 10 désactivée. L'action A est stoppée. L'étape 11 permet à la séquence de gauche d'attendre que la séquence de droite se termine (étape 21). Si la séquence de droite se termine en premier, l'étape 21 est activée et permet d'attendre la fin de la séquence de gauche (étape 11 active).

Quand les deux séquences sont terminées (étapes 11 et 21 actives simultanément), alors l'étape 2 est activée et les étapes de synchronisation 11 et 21 sont désactivées.

Au niveau de la transition avant l'étape 2, il faut mettre une réceptivité toujours vraie (1) et non $a \cdot b$, sinon on retombe dans le même travers que précédemment.

1.3 Exemple d'application

L'exemple de la figure **8** permet de faire quelques remarques sur la réalisation de GRAFCET fonctionnels minimisant l'implication sur les GRAFCET technologiques correspondants. Lorsque l'opérateur appuie sur le départ cycle (*m*), les chariots partent pour un aller-retour.

Solution 1

Si l'étape 0 est active et si *m* est vraie, chaque chariot part pour un aller-retour (figure **9*a***). Le premier qui termine son trajet (par exemple, le chariot 1) réactive l'étape initiale 0. Supposons que le chariot 2 se trouve dans l'étape 22 (en phase de retour) et que l'opérateur appuie sur le départ cycle. De fait, la transition suivant l'étape 0 est franchie, ce qui entraîne l'activation des étapes 11 et 21. Comme l'étape 22 est active, les actions D2 et G2 sont envoyées simultanément à la partie opérative, d'où dysfonctionnement et risque de casse.

Solution 2

Cette fois-ci (figure **9*b***), l'ordre de départ cycle n'est effectif que si les chariots se trouvent dans les conditions initiales, ce qui interdit le problème précédent. Toutefois, on retombe sur le problème rencontré précédemment (§ 1.2.3) : en effet, les contacts *g*1 et *g*2 doivent être maintenus et donc ce GRAFCET impose une contrainte technologique à la réalisation.

Il faut remarquer que même si ces deux GRAFCET (figure **9**) sont corrects d'un point de vue syntaxique, généralement les séquences simultanées se terminent avec des étapes de synchronisation (fins de séquences simultanées, § 1.2.3).

Solution 3

L'introduction des étapes de synchronisation 13 et 23 (figure **10*a***) permet de supprimer le test des conditions initiales au niveau de la réceptivité associée à la transition suivant l'étape 0.

Solution 4

L'introduction de deux étapes initiales (figure **10*b***) permet de supprimer les étapes de synchronisation (13 et 23). En effet, les étapes 10 et 20 jouent aussi le rôle d'étapes de synchronisation.

Cette solution possède une qualité supplémentaire qui permet de séparer facilement ce GRAFCET en deux GRAFCET indépendants (figure **10*c***). La synchronisation des deux GRAFCET est réalisée en testant l'activité de l'étape initiale de l'autre GRAFCET (variables X10 et X20).

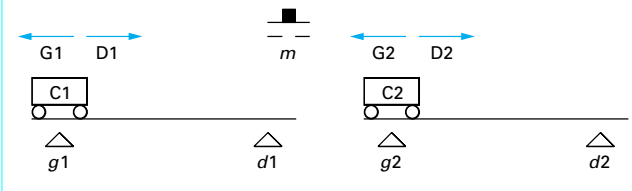
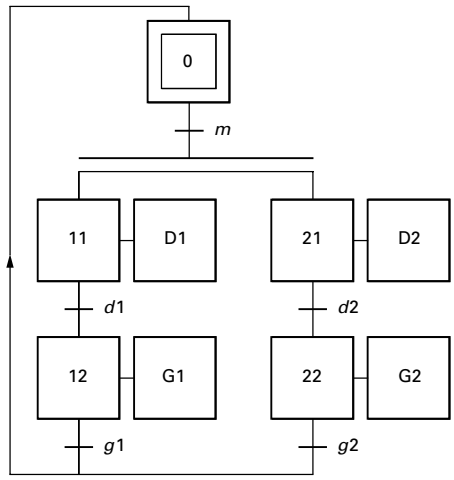
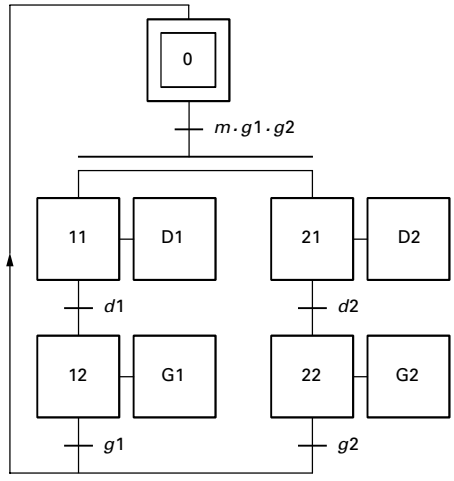


Figure 8 – Aller-retour de deux chariots



(a) solution 1



(b) solution 2

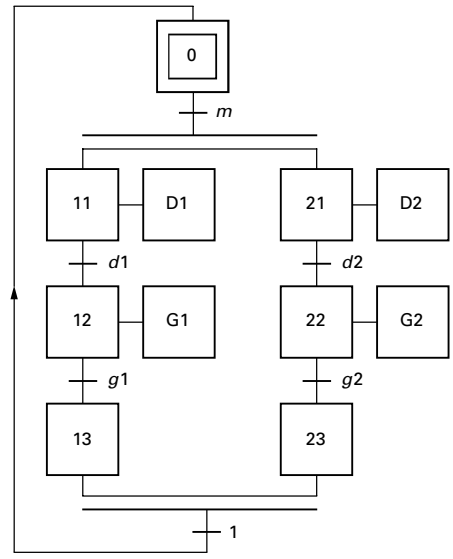
Figure 9 – Solutions à exclure

X*i* représente l'activité de l'étape *i* :
— X*i* = 1 ⇒ étape *i* active ;
— X*i* = 0 ⇒ étape *i* inactive.

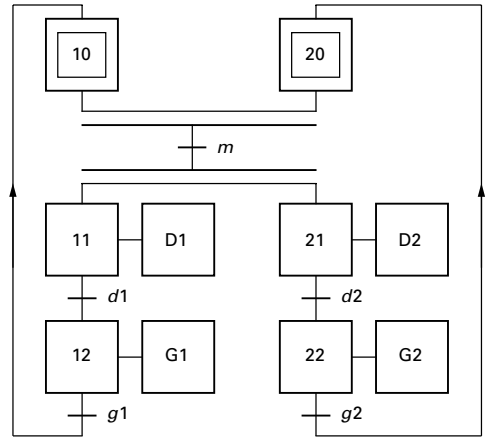
1.4 Dialogue entre GRAFCET

1.4.1 Problématique

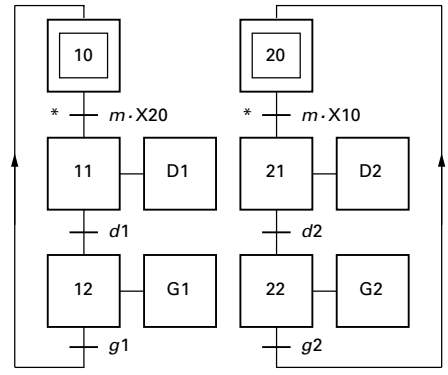
La solution 4 précédente introduit la notion de dialogue. Il se fait par passage d'informations entre les GRAFCET soit au niveau des réceptivités associées aux transitions (solution 4, § 1.3), soit au niveau des actions conditionnelles.



(a) solution 3



(b) solution 4



(c) solution 4 en deux GRAFCET

Figure 10 – Solutions viables

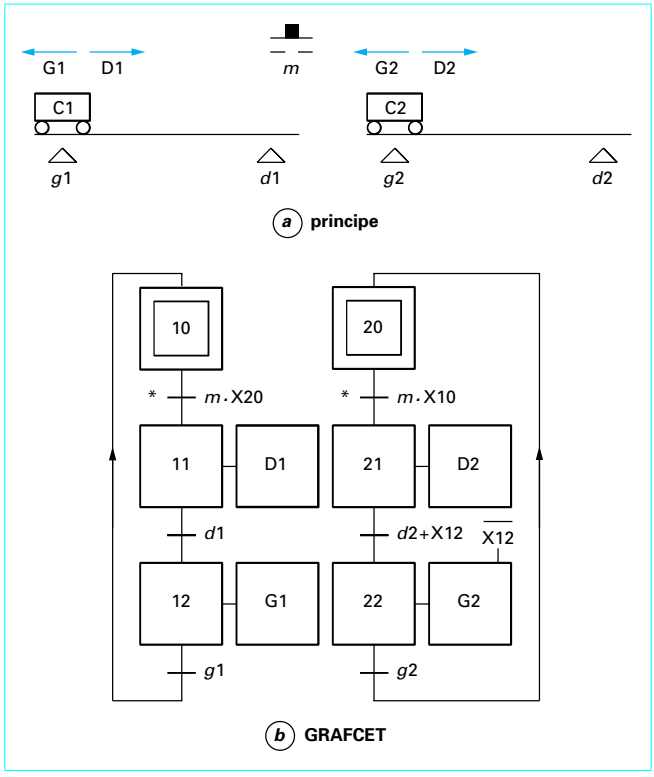


Figure 11 – Aller-retour de deux chariots avec contrainte

Exemple : reprenons l'exemple précédent (**§ 1.3**) en ajoutant la contrainte suivante : lorsque le chariot 1 arrive à droite, il stoppe le chariot 2 pendant son retour à gauche, puis il force le retour du chariot 2 (figure **11*a***). On retrouve les deux types de dialogue entre GRAFCET (figure **11*b***) : par les réceptivités et par les actions conditionnelles.

$\overline{X12}$ permet d'informer l'étape 22 que le chariot 1 n'est pas en déplacement à gauche (soit il a fini, soit il n'est pas encore arrivé à droite).

Il est conseillé de réaliser les dialogues par l'intermédiaire de l'activité des étapes plutôt que par l'état de capteurs car d'une part, on s'affranchit du problème des capteurs à contacts maintenus, d'autre part, il est parfois nécessaire de diviser la partie commande et/ou la partie opérative :

— application complexe divisée en sous-parties de moindre complexité ;

— synchronisation et dialogue entre sites répartis géographiquement ;

— intégration du concept CIM (*computer integrated manufacturing*) : stratification de l'industrie en différents niveaux de fonctionnalité suivant une structure pyramidale nécessitant des solutions optimales de communication entre les différents niveaux, basées bien souvent sur la notion de bus de terrain et/ou de réseau.

Cette division (figure **12*a***) implique une structuration afin d'éviter d'avoir des informations d'une partie opérative gérée par une partie commande qui soient reliées à une autre partie commande (figure **12*b***).

Dans la deuxième solution, il est donc nécessaire de tirer des connexions entre toutes les parties opératives et toutes les parties commandes, alors que dans la première solution, seules les parties commandes sont reliées (par exemple via un bus de terrain, un réseau, etc.). La première solution présente un avantage économique en terme de coût en connectique mais aussi en terme d'interface entre la partie opérative et la partie commande car une seule interface par capteur est nécessaire.

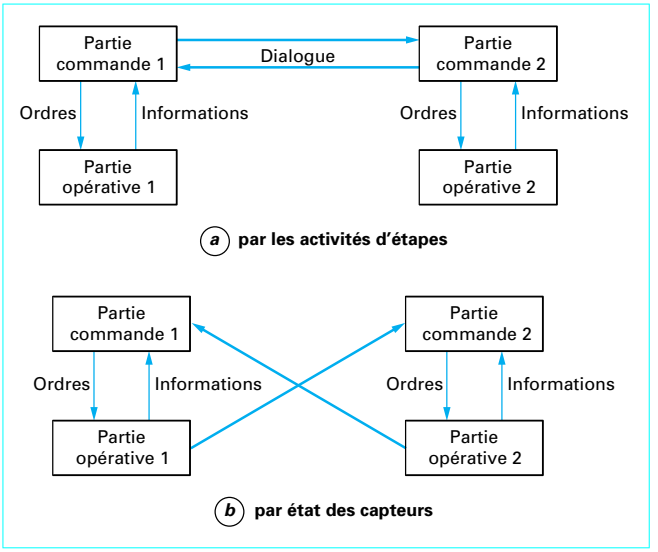


Figure 12 – Dialogue

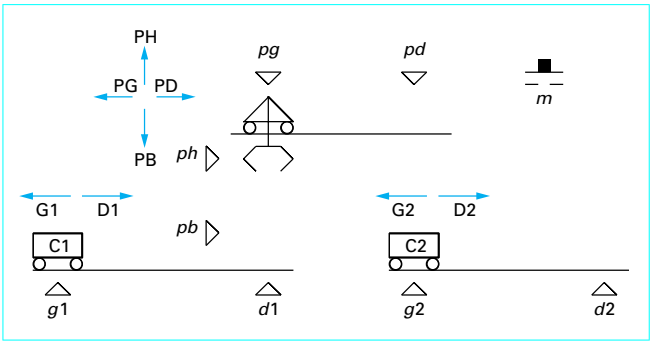


Figure 13 – Chargement par pince et déplacement de chariots

Dès la conception du GRAFCET fonctionnel, il est recommandé d'intégrer ce principe de **dialogue par les activités d'étapes** afin d'assurer la conception d'un GRAFCET le plus indépendant possible de la réalisation technologique et qui s'adaptera au mieux et avec un minimum d'effort aux diverses technologies de réalisation.

1.4.2 Décomposition d'une chaîne en sous-machines et synchronisation

Considérons l'application de la figure **13**. Sur ordre de l'opérateur (*m*), le chariot 1 se charge (CP1), puis se déplace à droite. Lorsqu'il est arrivé à droite, la pince descend (PB), se ferme (FP), remonte (PH), puis se déplace à droite (PD). Arrivée à droite, elle descend, s'ouvre (OP), se décharge dans le chariot 2, qui se déplace alors à droite et se vide (VC). Afin d'améliorer la productivité, les temps sont optimisés en introduisant du parallélisme dans le GRAFCET de commande (figure **14**).

Cette application peut se décomposer en trois sous-machines (le chariot 1, la pince, le chariot 2) ; à chacune correspond un GRAFCET qui dialogue avec les autres (figure **15**). Chaque GRAFCET peut être supporté par sa propre partie commande (figure **16**).

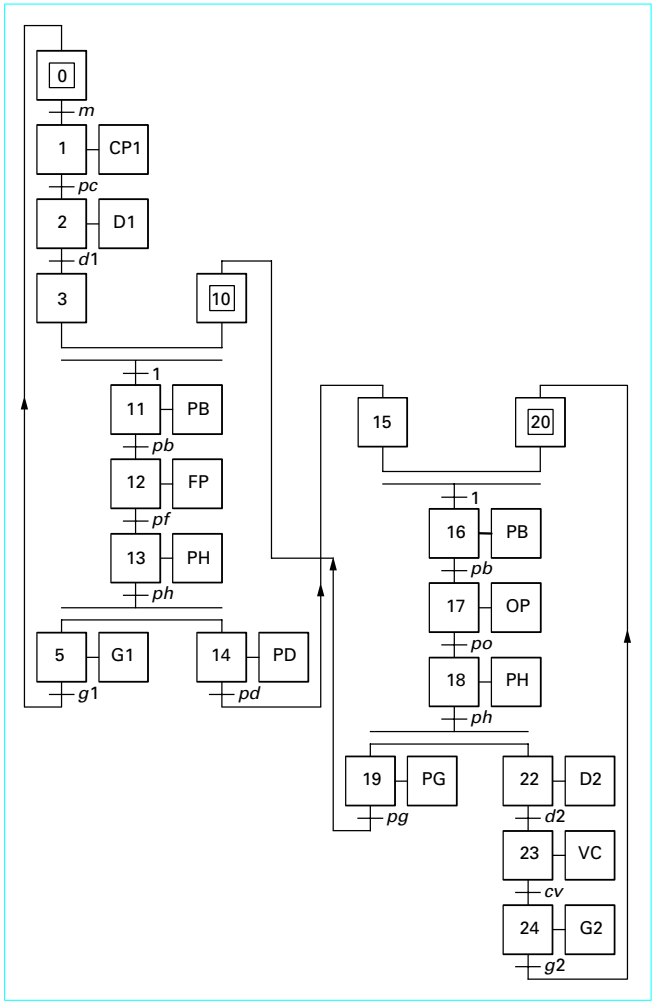


Figure 14 – GRAFCET de commande du chargement et du déplacement des chariots

1.4.3 Hiérarchisation de la partie commande

Un GRAFCET peut aussi servir à piloter d'autres GRAFCET, notamment pour les synchroniser ou pour mémoriser une information.

Les deux chariots 1 et 2 se chargent (CP*i*), se déplacent à droite (D*i*) puis se déchargent (DP*i*) dans le chariot 3, le chariot 1 en premier. Les chariots 1 et 2 retournent à gauche (G*i*) pendant que le chariot 3 va se décharger à droite (figure **17**).

L'étape 40 (figure **18**) permet de mémoriser le fait que le chariot 3 a fini son cycle et qu'il est prêt à en refaire un autre. Le front montant sur X30 (réceptivité de la transition entre l'étape 41 et 40) indique que le chariot 3 est de nouveau revenu dans son état initial symbolisé par l'étape initiale du GRAFCET de gestion du chariot 3.

Cette application peut se décomposer en trois sous-machines (une par chariot) ; à chacune correspond un GRAFCET qui dialogue avec les autres ; deux autres GRAFCET supplémentaires gèrent les synchronisations et la mémorisation (figure **19**).

On retrouve à nouveau un découpage de la partie commande : une partie commande par gestion d'un chariot et une partie commande qui gère les synchronisations et la mémorisation. Cette dernière supervise les trois autres (figure **20**) : c'est une **commande hiérarchisée**.

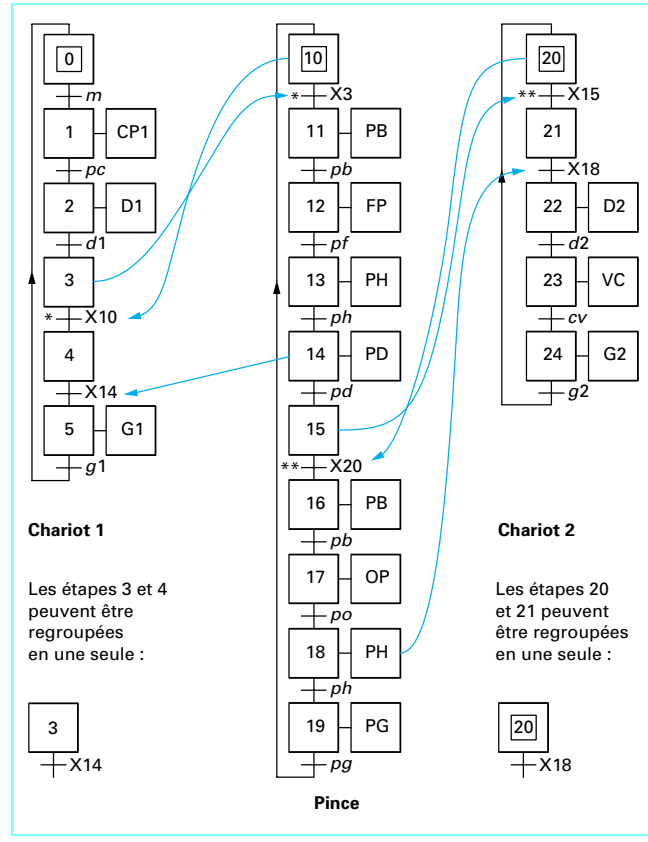


Figure 15 – Décomposition en sous-machines

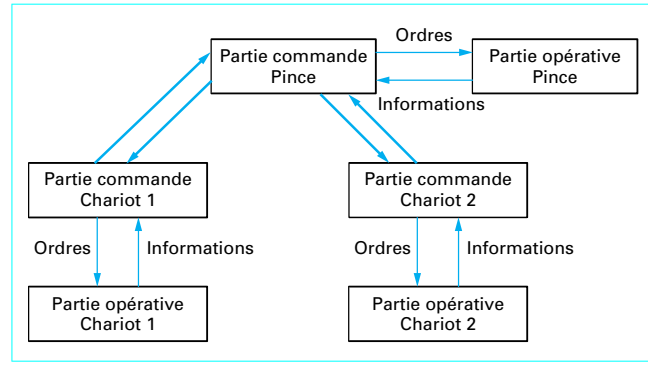


Figure 16 – Séparation des parties de commande

1.4.4 Gestion des ressources communes

Le GRAFCET permet de gérer le **partage de ressources communes**.

Reprenons l'exemple précédent (figure **17**) en supposant maintenant que chacun des chariots 1 et 2 peut déposer son chargement dans le chariot 3 et qu'une fois chargé, le chariot 3 part se décharger. Mais cette fois, le chariot 3 ne peut recevoir qu'un seul chargement, du chariot 1 ou du chariot 2.

Le chariot 3 est vu comme une **ressource partagée** entre les chariots 1 et 2 ; quand elle est occupée, un chariot qui demande son

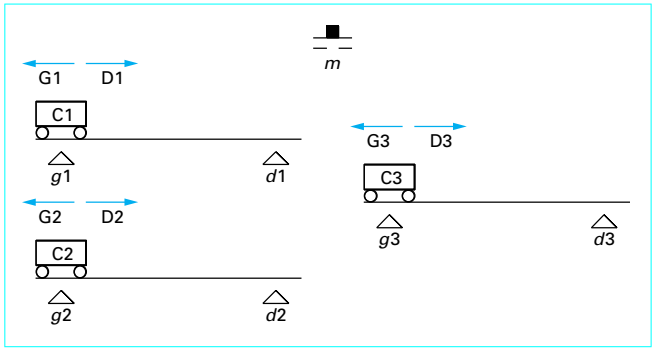


Figure 17 – Chargement et déplacement de trois chariots

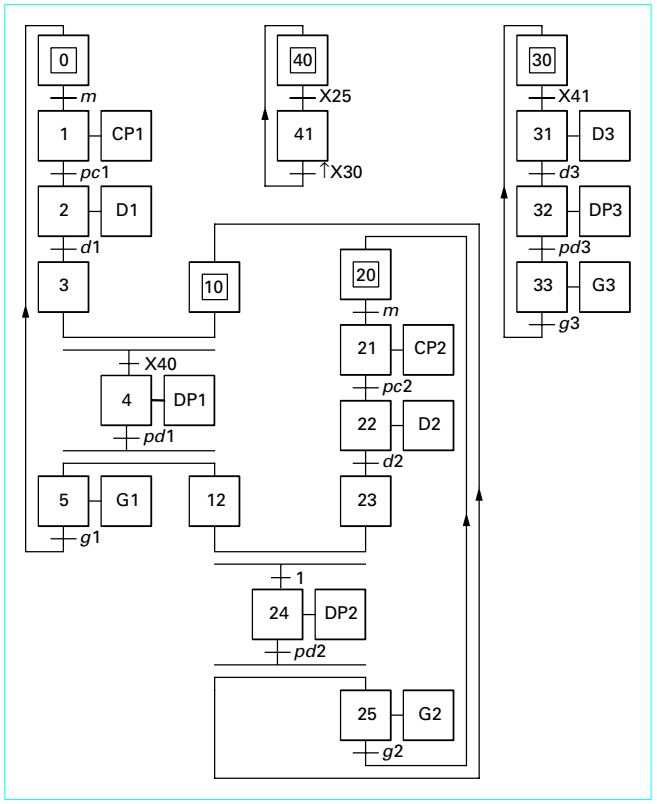


Figure 18 – GRAFCET de hiérarchisation

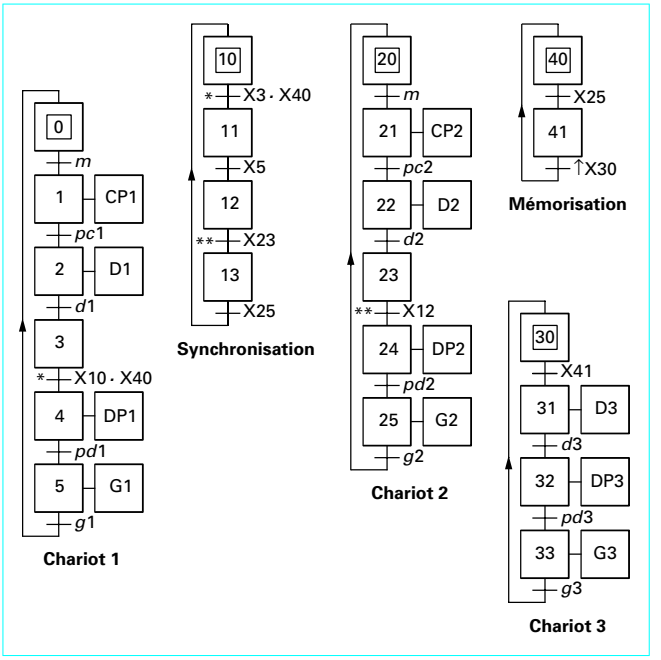


Figure 19 – Décomposition du GRAFCET de hiérarchisation

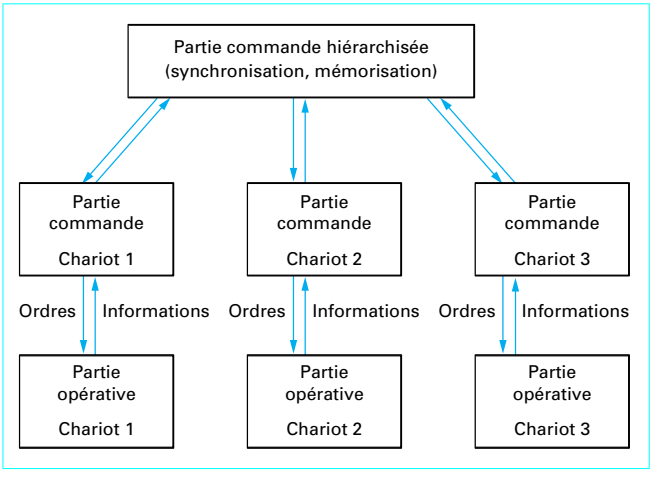


Figure 20 – Commande hiérarchisée

accès doit attendre sa libération. L'étape 10 (figure **21**) gère l'exclusion mutuelle au niveau de la ressource commune ; si l'étape 10 est active, alors la ressource est libre ; sinon, elle est occupée. La réceptivité NON(X3) associée à la transition entre les étapes 10 et 24 permet de donner la priorité au chariot 1 dans le cas où les deux chariots demandent l'accès simultanément à la ressource commune.

Cette application peut se décomposer en trois sous-machines (une par chariot). Le GRAFCET gérant le chariot 3 gère aussi la ressource commune (figure **22**).

Dans cet exemple, les étapes 10 et 11 ne peuvent pas être regroupées en une seule car elles ne sont pas que des étapes d'attente. L'étape 10 implique que la ressource commune (le chariot 3) soit libre, tandis que l'étape 11 indique qu'elle est occupée.

Chaque GRAFCET peut être supporté par sa propre partie commande (figure **23**).

1.5 Conclusion

Ainsi, certains principes doivent être respectés lors de la conception du GRAFCET fonctionnel afin qu'il soit le plus indépendant possible de la réalisation technologique et de pouvoir être supporté par tout type de réalisation.

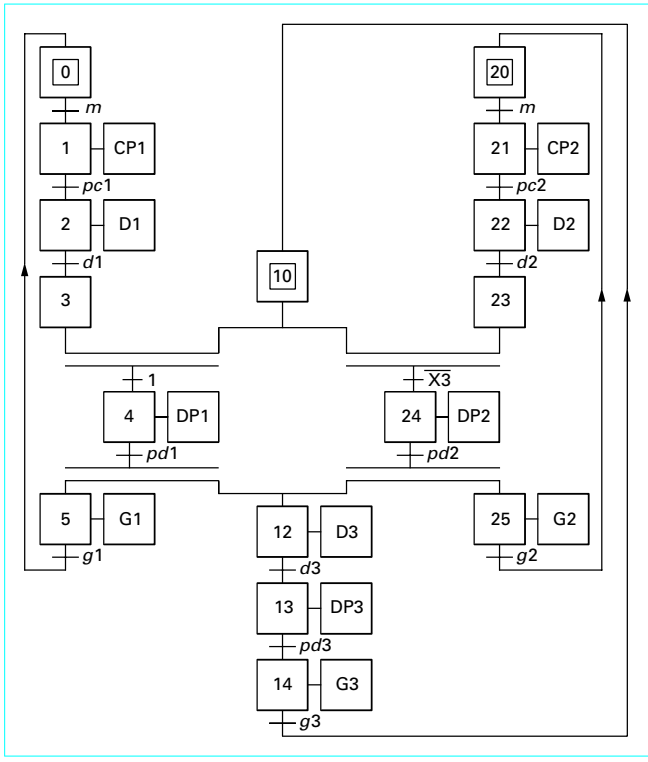


Figure 21 – GRAFCET de gestion de ressource commune

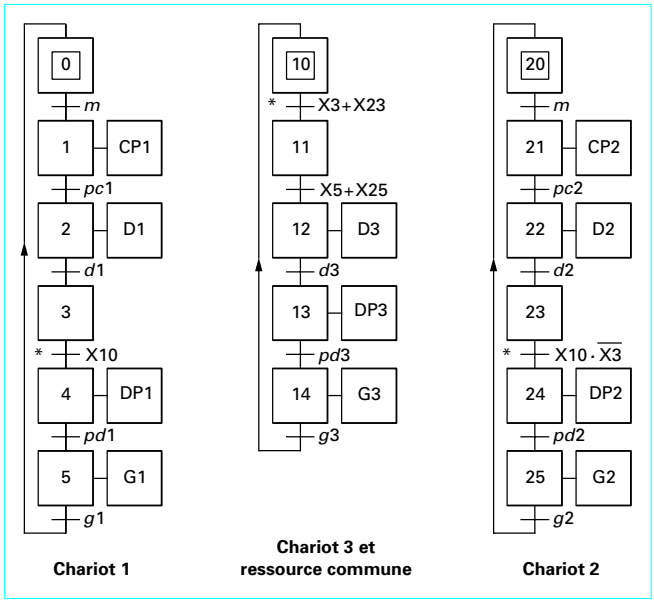


Figure 22 – Décomposition en trois GRAFCET

Un principe essentiel à respecter est d'**instaurer des processus de dialogue** entre GRAFCET ou entre séquences **basés sur l'activité des étapes** et non sur l'état des capteurs.

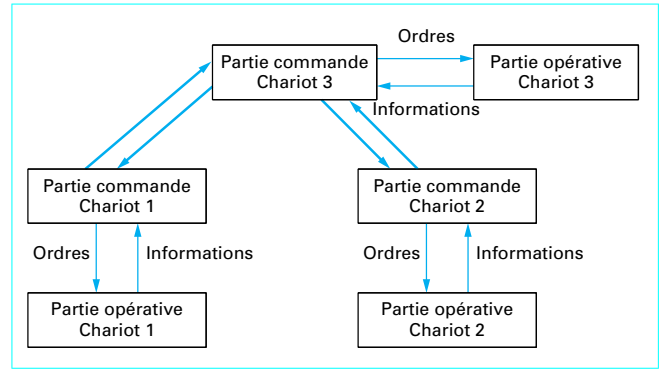


Figure 23 – Commande en trois parties

2. Équation logique d'une étape

Nous montrons ici les principes de base pour établir les équations logiques des étapes d'un GRAFCET et des actions associées, en introduisant les modes de marche/arrêt.

2.1 Rappel des règles d'évolution

La syntaxe du GRAFCET fait l'objet d'une norme (NF C03-190) et est basée sur cinq règles d'évolution qui définissent le caractère actif ou inactif d'une étape du GRAFCET.

Règle 1 : situation initiale

L'*initialisation* précise les étapes actives au début du fonctionnement. Elles sont activées inconditionnellement et repérées sur le GRAFCET en doublant les côtés des symboles correspondants.

Règle 2 : franchissement d'une transition

Une *transition* est soit validée, soit non validée. Elle est validée lorsque toutes les étapes immédiatement précédentes sont actives.

Elle ne peut être franchie que :

— lorsqu'elle est validée ;

— lorsque la réceptivité associée à la transition est vraie.

La transition est alors obligatoirement franchie.

Règle 3 : évolution 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.

Règle 4 : évolutions simultanées

Plusieurs transitions simultanément franchissables sont simultanément franchies. Cette règle permet de décomposer un GRAFCET en plusieurs parties en assurant leurs interconnexions. Dans ce cas, il est indispensable de faire intervenir, dans les réceptivités, les états actifs ou inactifs des étapes.

Règle 5 : activation et désactivation simultanées

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

Durées de franchissement d'une transition ou d'activité d'une étape

La durée de franchissement d'une transition est considérée théoriquement aussi petite que l'on veut, mais non nulle, même si en pratique, cette durée peut être imposée par la technologie utilisée pour la réalisation de l'automatisme.

De même, la durée d'activité d'une étape ne peut pas être nulle, mais suffisante, si besoin est, pour effectuer une action fugitive à la vitesse de la partie commande.

Ces règles ont été formulées pour des raisons de cohérence théorique interne au GRAFCET. L'implantation d'un GRAFCET sur une réalisation technologique se doit d'intégrer et de respecter toutes ces règles et leurs implications. C'est pourquoi elles sont à la base des équations logiques de réalisation.

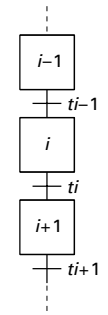


Figure 24 – GRAFCET partiel

2.2 Équation logique d'une étape

Pour décrire l'activité d'une étape, nous utilisons la notation proposée par la norme NF C03-190 :

— X*i* = 1 signifie que l'étape *i* est active ;

— X*i* = 0 signifie que l'étape *i* est inactive.

Soit la partie de GRAFCET présentée figure **24**. Déterminons quelles sont les variables qui interviennent dans l'activité de l'étape *i* : X*i* = *f* (?).

La règle 3 dit que le franchissement d'une transition entraîne la désactivation des étapes immédiatement précédentes et l'activation des étapes immédiatement suivantes. Or, une transition est franchissable si (règle 2) :

— elle est validée (toutes les étapes immédiatement précédentes sont actives) ;

— la réceptivité associée à la transition est vraie.

La traduction de la règle 2 donne la condition d'activation de l'étape *i* (CAX*i*) :

$$CAXi = Xi-1 \cdot ti-1$$

La traduction de la règle 3 donne la condition de désactivation de l'étape *i* (CDX*i*) :

$$CDXi = Xi+1$$

Si la condition d'activation et la condition de désactivation de l'étape *i* sont fausses, alors l'étape *i* reste dans son état (effet mémoire de la logique séquentielle) ; cela signifie que l'état de X*i* dépend aussi de X*i*. Donc :

$$Xi = f(Xi, CAXi, CDXi)$$

Il est possible de définir la table de vérité de X*i* (tableau **1**), et à partir du tableau de Karnaugh associé (tableau **2**), de déduire l'équation logique d'une étape :

$$Xi = CAXi + Xi \cdot \overline{CDXi} \qquad \textbf{(1)}$$

Nous avons vu que la condition de désactivation de l'étape *i* est CDX*i* = X*i* + 1 et non CDX*i* = X*i* · *ti* , alors que la règle 3 dit que c'est le franchissement de la transition qui entraîne la désactivation de toutes les étapes précédentes.

Si $Xi = CAXi + Xi \cdot \overline{CDXi}$ avec $CDXi = Xi \cdot ti$

$$Xi = Xi-1 \cdot ti-1 + Xi(\overline{Xi \cdot ti})$$

$$Xi = Xi-1 \cdot ti-1 + Xi[\overline{Xi} + \overline{ti}]$$

$$Xi = Xi-1 \cdot ti-1 + Xi \cdot \overline{Xi} + Xi \cdot \overline{ti}$$

$$Xi = Xi-1 \cdot ti-1 + Xi \cdot \overline{ti}$$

De même, $Xi+1 = Xi \cdot ti + Xi+1 \cdot \overline{ti+1}$.

Tableau 1 – Table de vérité d'une étape

X*i* (*t*)	CAX*i*	CDX*i*	X*i* (*t* + Δ*t*)
0	0	0	0 (2)
0	0	1	0
0	1	0	1
0	1	1	1 (1)
1	0	0	1 (2)
1	0	1	0
1	1	0	1
1	1	1	1 (1)

(1) Ces lignes garantissent la priorité à l'activation afin de respecter la règle 5.
(2) Ces lignes correspondent à l'effet mémoire.

Tableau 2 – Tableau de Karnaugh de l'équation d'une étape

	CAX*i* · CDX*i*			
X*i*	00	01	11	10
0	0	0	1	1
1	1	0	1	1

Quand la réceptivité *ti* passe de 0 à 1 :

$$Xi \cdot \overline{ti} = 0 \quad \text{(désactivation de } Xi\text{)}$$

$$Xi \cdot ti = 1 \quad \text{(activation de } Xi+1\text{)}$$

Donc d'un point de vue fonctionnel, cela fonctionne correctement.

Par contre, l'implantation technologique risque de poser problème dans le cas où la désactivation de X*i* se fait avant l'activation de X*i* + 1. On aura alors X*i* à 0 avant d'avoir X*i* + 1 à 1, ce qui implique la désactivation de X*i* sans avoir l'activation de X*i* + 1, donc un blocage car il n'y aura plus d'étape active. C'est pourquoi il est préférable de prendre la condition de désactivation de X*i* égale à X*i* + 1. En effet, l'étape X*i* est désactivée seulement quand l'étape X*i* + 1 est activée.

Conclusion

L'équation (1) donne l'équation générale pour réaliser une étape d'un GRAFCET en prenant garde d'utiliser l'activité des étapes immédiatement suivantes dans les conditions de désactivation afin d'assurer un fonctionnement exempt d'aléas.

2.3 Choix de séquences

Le début et la fin de choix de séquences impliquent des prises en compte particulières au niveau de l'équation d'une étape.

2.3.1 Début de choix de séquences

Soit le GRAFCET partiel de la figure **25**. L'activité de l'étape 10 valide les transitions entre les étapes 10/21 et 10/31.

Si la réceptivité *b* (resp. *c*) est vraie, alors la transition entre les étapes 10/21 (resp. 10/31) est franchie, entraînant l'activation de l'étape 21 (resp. 31) et la désactivation de l'étape 10.

Sachant que l'équation générale d'une étape est :

$$Xi = CAXi + Xi \cdot \overline{CDXi}$$

nous pouvons exprimer les équations logiques comme dans le tableau **3**. La condition de désactivation de l'étape 10 est bien l'activation de l'étape 21 ou de l'étape 31 car l'une ou l'autre des séquences est activée.

2.3.2 Fin de choix de séquences

Soit le GRAFCET partiel de la figure **26**. L'activation de l'étape 9 peut se faire soit par passage par l'étape 22 (il faut aussi que la réceptivité *g* soit vraie), soit par l'étape 32 (si *f* est vraie). L'activation de l'étape 9 implique alors la désactivation des étapes 22 et 32. Les conditions d'activation et de désactivation sont données dans le tableau **4**. La condition d'activation de l'étape 9 est une combinaison logique des deux fins de séquences car elle peut être activée en venant de l'une ou de l'autre des branches.

Tableau 3 – Équations logiques : début de choix de séquences

Étape	CAX*i*	CDX*i*
10	$X9 \cdot a$	X21 + X31
21	$X10 \cdot b$	X22
31	$X10 \cdot c$	X32

Tableau 4 – Équations logiques : fin de choix de séquences

Étape	CAX*i*	CDX*i*
9	$X22 \cdot g + X32 \cdot f$	X10
22	$X21 \cdot d$	X9
32	$X31 \cdot e$	X9

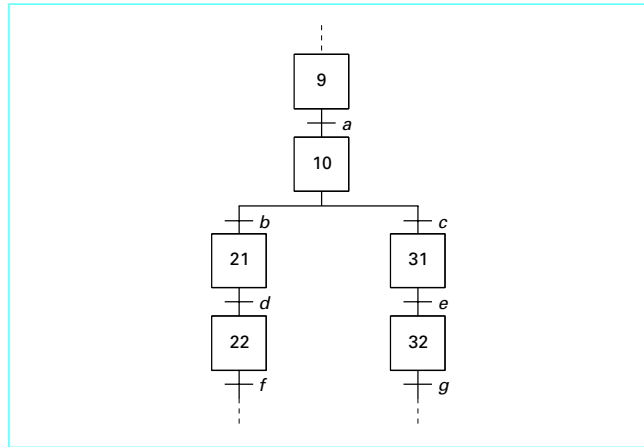


Figure 25 – GRAFCET partiel : début de choix de séquences

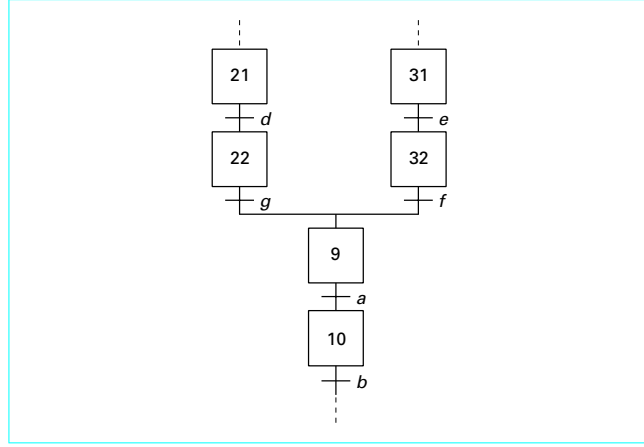


Figure 26 – GRAFCET partiel : fin de choix de séquences

2.4 Séquences parallèles

Le début et la fin de séquences parallèles impliquent, comme les choix de séquences, des prises en compte particulières au niveau de l'équation d'une étape.

2.4.1 Début de séquences parallèles

Soit le GRAFCET partiel de la figure **27**. L'activité de l'étape 10 entraîne la validation de la transition entre les étapes 10/21/31. Lorsque la réceptivité associée est vraie ($b = 1$), la transition est franchie, entraînant l'activation des étapes 21 et 31 et la désactivation de l'étape 10. Les conditions d'activation et de désactivation pour les étapes 10, 21 et 31 sont données dans le tableau **5**.

Les conditions d'activation des étapes 21 et 31 sont évidemment les mêmes, car elles sont en début de séquences simultanées (donc activées en même temps). Par contre, la condition de désactivation de l'étape 10 est $X21 \cdot X31$, il faut que ces étapes soient actives avant de désactiver l'étape 10.

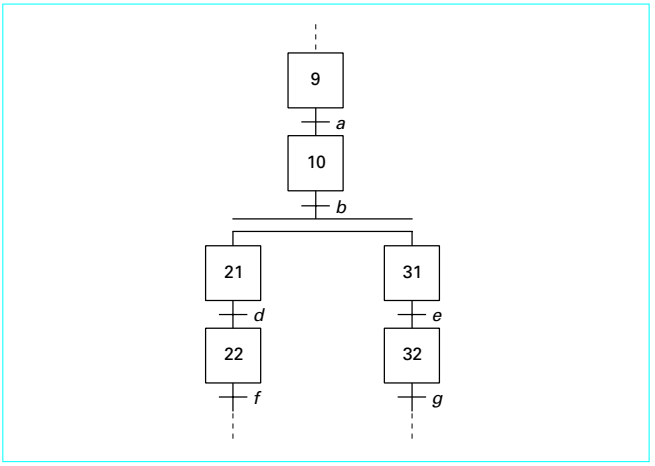


Figure 27 – GRAFCET partiel : début de séquences parallèles

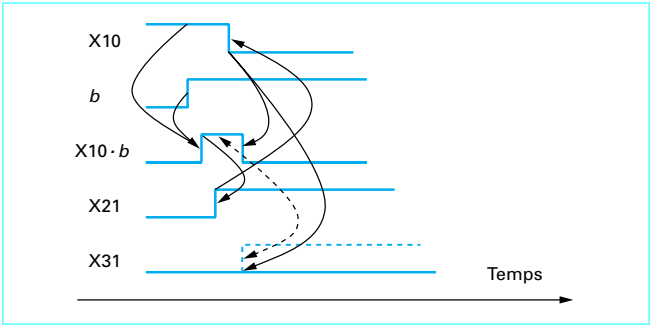


Figure 28 – Chronogramme de séquences parallèles

Tableau 5 – Équations logiques : début de séquences parallèles

Étape	CAX*i*	CDX*i*
10	X9 · *a*	X21 · *X*31
21	X10 · *b*	X22
31	X10 · *b*	X32

Dans la littérature, il est parfois indiqué que cette condition de désactivation est X21 ou X31. D'un point de vue fonctionnel et d'après la règle 3 d'évolution du GRAFCET, ces solutions sont correctes. Mais d'un point de vue technologique, elles peuvent entraîner des aléas de fonctionnement. Supposons que la condition de désactivation de l'étape 10 soit l'activité de l'étape 21, et que l'activation de l'étape 21 soit plus rapide que l'activation de l'étape 31 (figure **28**). Étant donné que l'activation de l'étape 21 entraîne la désactivation de l'étape 10, la condition d'activation (X10 · *b*) de l'étape 31 n'est plus vraie et donc la seconde branche ne sera pas activée, contrairement au GRAFCET de niveau 1, d'où un dysfonctionnement et un risque de blocage à la fin des séquences simultanées.

Ainsi, le fait de dire que la condition de désactivation de l'étape 10 est X21 · X31 assure que les deux étapes sont activées avant de désactiver l'étape 10.

2.4.2 Fin de séquences parallèles

Soit le GRAFCET partiel de la figure **29**. L'activité des étapes 21 et 31 entraîne la validation de la transition entre les étapes 21/31/10. Lorsque la réceptivité de cette transition est vraie ($b = 1$), alors la transition est franchie, impliquant l'activation de l'étape 10 et la désactivation des étapes 21 et 31. Le tableau **6** décrit les conditions d'activation et de désactivation pour les étapes 10, 21 et 31.

2.5 Cas particulier : GRAFCET ou boucle à deux étapes

Un tel GRAFCET (figure **30*a***) possède les particularités suivantes :

— l'activité de l'étape 20 et la réceptivité *a* vraie impliquent l'activation de l'étape 30 et la désactivation de l'étape 20 ;

— l'activité de l'étape 30 et la réceptivité *b* vraie impliquent l'activation de l'étape 20 et la désactivation de l'étape 30.

Tableau 6 – Équations logiques : fin de séquences parallèles

Étape	CAX*i*	CDX*i*
10	X21 · X31 · *b*	X11
21	X20 · *d*	X10
31	X10 · *e*	X10

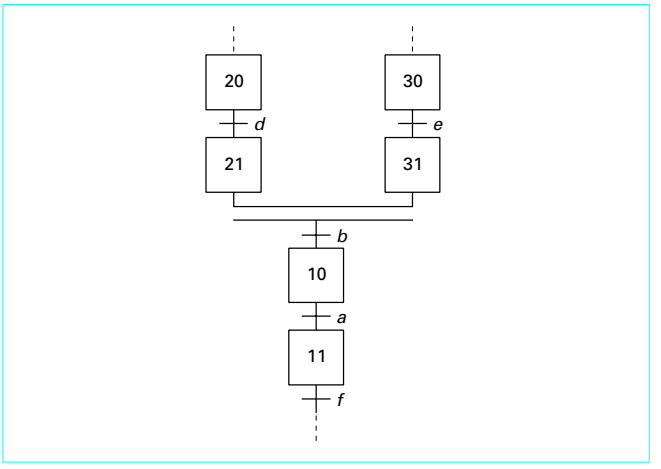


Figure 29 – GRAFCET partiel : fin de séquences parallèles

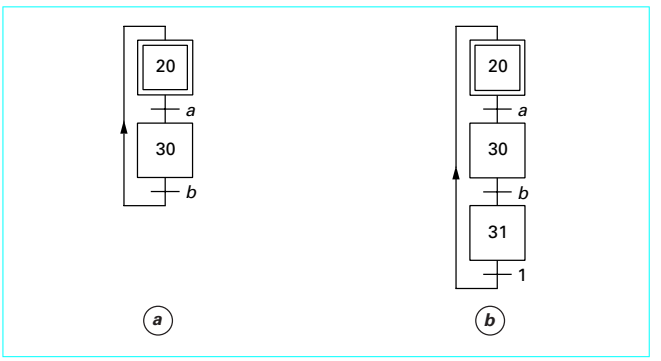


Figure 30 – GRAFCET à deux étapes

Les conditions d'activation et de désactivation pour les étapes 20 et 30 sont données dans le tableau **7**. Ce tableau révèle que :

— l'activité de l'étape 30 intervient dans la condition d'activation de l'étape 20 mais aussi dans sa condition de désactivation ;

— l'activité de l'étape 20 intervient dans la condition d'activation de l'étape 30 mais aussi dans sa condition de désactivation, ce qui représente une source de dysfonctionnement.

La solution est d'insérer une troisième étape (figure **30*b***). Cette troisième étape évite d'avoir l'activité d'une étape qui intervient simultanément dans la condition d'activation et dans la condition de désactivation d'une autre étape. Les conditions d'activation et de désactivation pour les étapes 20, 30 et 31 sont données dans le tableau **8**.

Ce problème se pose de façon similaire lorsqu'un GRAFCET présente une boucle à deux étapes (figure **31**). Il est possible d'atteindre l'étape 1 directement de l'étape 0 en passant par la réceptivité *b*, de fait l'étape 1 peut suivre l'étape 0 et l'étape 0 suit l'étape 1. Ce GRAFCET possède une boucle à deux étapes présentant les mêmes inconvénients qu'un GRAFCET à deux étapes. La solution est similaire au cas précédent : introduire une étape supplémentaire dans la boucle, à la suite de l'étape 1, ou bien entre l'étape 0 et l'étape 1, à la suite de la transition ayant pour réceptivité *b* par exemple.

Tableau 7 – Équations logiques : GRAFCET à deux étapes

Étape	CAX*i*	CDX*i*
20	X30 · *b*	X30
30	X20 · *a*	X20

Tableau 8 – Équations logiques : solution à trois étapes

Étape	CAX*i*	CDX*i*
20	X31	X30
30	X20 · *a*	X31
31	X30 · *b*	X20

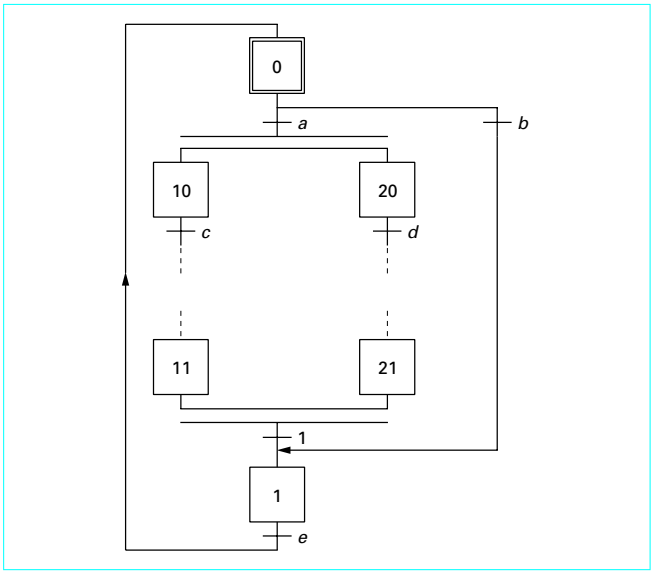


Figure 31 – GRAFCET avec boucle à deux étapes

2.6 Gestion des actions

Comme il existe deux classes d'actions (continues ou conditionnelles) dans un GRAFCET, nous allons voir leur traduction au niveau technologique.

2.6.1 Actions continues

L'activité d'une action continue ne dépend que de l'activité des étapes auxquelles elle est associée (figure **32**). Donc, l'action A est vraie quand l'étape 1 (X1) est active et l'action B est vraie quand l'étape 2 (X2) est active. Donc :

$$\begin{cases} A = X1 \\ B = X2 \end{cases}$$

Si une même action est associée à plusieurs étapes (figure **33**), son équation est une somme logique (OU) des étapes auxquelles elle est associée :

$$\begin{cases} A = X1 + X3 \\ B = X2 \end{cases}$$

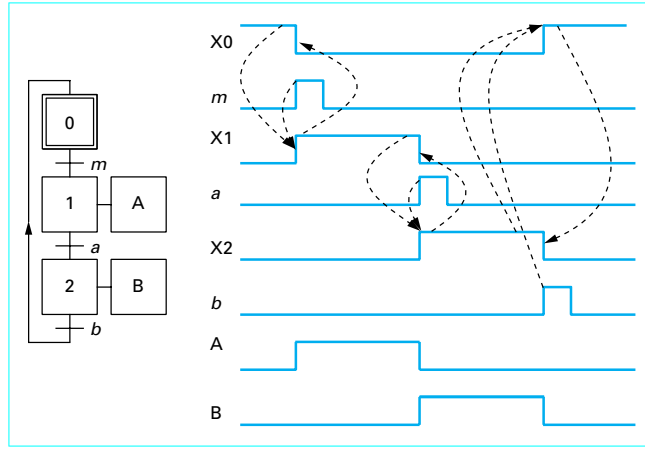


Figure 32 – Actions continues

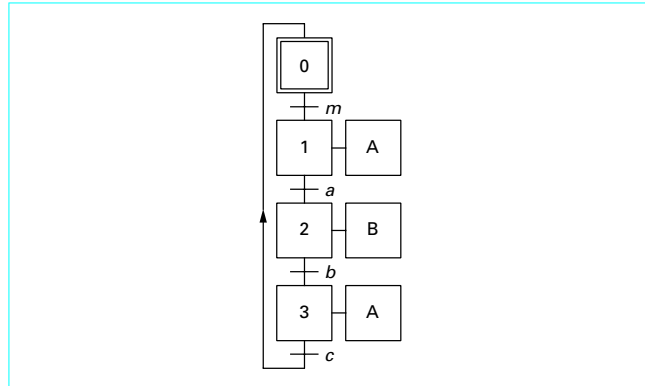


Figure 33 – Actions continues à plusieurs étapes

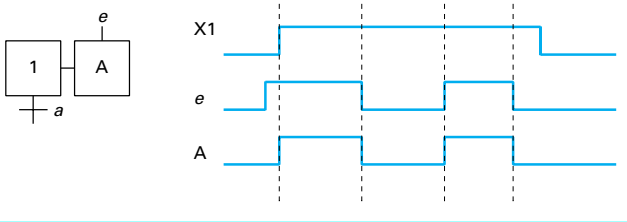


Figure 34 – Actions conditionnelles

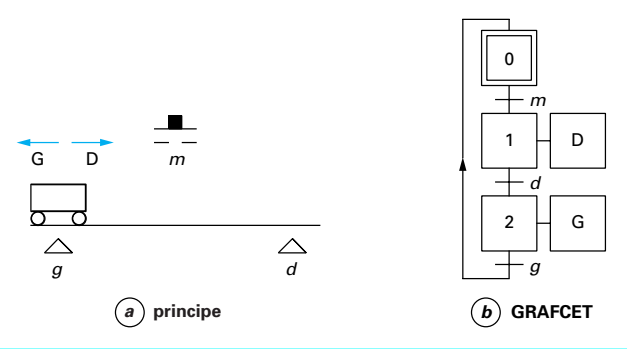


Figure 35 – Aller-retour d'un chariot

2.6.2 Actions conditionnelles

L'activité d'une action conditionnelle dépend de l'activité des étapes auxquelles elle est associée et des conditions (figure **34**). L'action A est vraie si l'activité de l'étape 1 est vraie (X1) et si la condition *e* est vraie, donc :

$$A = X1 \cdot e$$

Exemple : lorsque l'opérateur appuie sur *m*, le chariot fait un aller-retour (figure **35*a***). Le GRAFCET est celui de la figure **35*b***. Les conditions d'activation et de désactivation pour les étapes 0, 1 et 2 sont celles du tableau **9**, d'où les équations logiques des étapes et des actions :

$$\begin{cases} X0 = X2 \cdot g + X0 \cdot \overline{X1} \\ X1 = X0 \cdot m + X1 \cdot \overline{X2} \\ X2 = X1 \cdot d + X2 \cdot \overline{X0} \\ D = X1 \\ G = X2 \end{cases}$$

Tableau 9 – Équations logiques : aller-retour d'un chariot

Étape	CAX*i*	CDX*i*
0	$X2 \cdot g$	X1
1	$X0 \cdot m$	X2
2	$X1 \cdot d$	X0

2.7 Prise en compte des modes de marche/arrêt

2.7.1 Initialisation

L'observation des équations des étapes de l'exemple précédent montre qu'il n'est pas fait de différence entre les étapes initiales et les autres. Or, d'après la règle 1 (§ 2.1), il est nécessaire d'ajouter une information qui gère l'initialisation du GRAFCET.

Lorsque cette information (que nous appellerons Init) est vraie, il faut inconditionnellement :

— activer les étapes initiales ;
— désactiver les étapes non initiales.

Cette procédure d'initialisation est prioritaire. On a donc :

— pour les étapes initiales : $Xi = CAXi + Xi \cdot \overline{CDXi} + Init$

— pour les étapes non initiales : $Xi = (CAXi + Xi \cdot \overline{CDXi}) \cdot \overline{Init}$

2.7.2 Arrêts d'urgence

Un arrêt d'urgence provoque l'**arrêt de toutes les actions envoyées vers la partie opérative** (sauf celles agissant dans le sens de la sécurité, commande d'un frein par exemple). Il existe deux manières d'opérer :

— soit on joue uniquement sur l'activité des actions sans toucher à l'activité des étapes. On parlera d'arrêt d'urgence de type *doux* ;
— soit on joue sur l'activité des étapes. On parlera d'arrêt d'urgence de type *dur*.

Arrêt d'urgence de type dur

Lorsque l'arrêt d'urgence de type dur (AUD) est vrai, il désactive toutes les étapes du GRAFCET, et donc l'activité des actions associées. Son arrêt ne signifie pas le redémarrage du cycle, étant donné que toutes les étapes sont inactives, il faut faire une initialisation avec éventuellement une remise manuelle ou automatique de la partie opérative dans les conditions initiales. L'arrêt d'urgence est prioritaire sur toutes les informations même sur l'initialisation. On a donc :

— pour les étapes initiales :

$$Xi = (CAXi + Xi \cdot \overline{CDXi} + Init) \cdot \overline{AUD}$$

— pour les étapes non initiales :

$$Xi = (CAXi + Xi \cdot \overline{CDXi}) \cdot \overline{Init} \cdot \overline{AUD}$$

Arrêt d'urgence de type doux

Lorsque l'arrêt d'urgence de type doux (AUd) est vrai, il stoppe l'activité des actions sans modifier l'activité des étapes du GRAFCET. Son arrêt signifie que le cycle peut redémarrer là où il s'était interrompu car le GRAFCET a conservé son état, étant donné que l'activité des étapes n'est pas modifiée. Seules les équations des actions sont modifiées, les équations des étapes ne sont pas concernées : $A = Xi \cdot \overline{AUD}$ (A étant une action associée à l'étape *i*).

Exemple : de fait, l'exemple des chariots décrit au paragraphe 2.6 donne, en supposant qu'il y ait les deux types d'arrêt d'urgence :

$$\begin{cases} X0 = (X2 \cdot g + X0 \cdot \overline{X1} + Init) \cdot \overline{AUD} \\ X1 = (X0 \cdot m + X1 \cdot \overline{X2}) \cdot \overline{Init} \cdot \overline{AUD} \\ X2 = (X1 \cdot d + X2 \cdot \overline{X0}) \cdot \overline{Init} \cdot \overline{AUD} \\ D = X1 \cdot \overline{AUd} \\ G = X2 \cdot \overline{AUd} \end{cases}$$

2.7.3 Gestion des défaillances et/ou modes de fonctionnement

Dans certaines applications, le GRAFCET est utilisé pour décrire et gérer les modes de fonctionnement (normal, dégradé, arrêt d'urgence, etc.) et/ou traiter les défaillances. Des GRAFCET sont alors développés afin de piloter le ou les GRAFCET de fonctionnement normal. On retrouve une **hiérarchisation** non plus de la commande mais des GRAFCET.

Afin d'illustrer ce propos, nous allons décrire une **ligne de séchage de ruban** (figure **36**). Cet exemple s'inspire d'un cas industriel concernant une entreprise textile.

Cahier des charges

La ligne de séchage est composée de plusieurs éléments. Autour d'un *cylindre* de grande inertie thermique est enroulé le ruban ; il l'entraîne par rotation. Plusieurs tours de ruban autour du cylindre sont nécessaires pour opérer un séchage correct. Un *brûleur linéaire* se trouve à l'intérieur du cylindre sur toute sa longueur. Il est alimenté en gaz par l'intermédiaire d'une *électrovanne* (tout ou rien, simple effet). Un *thermocouple* permet de mesurer la température du cylindre et donc de la comparer à une température de consigne représentant la température optimale pour le séchage. L'arrivée du gaz dans le brûleur ne suffit pas à son inflammation. C'est pourquoi une *veilleuse* se trouve au début du brûleur et permet l'inflammation du gaz lorsque l'électrovanne est ouverte. Par sécurité, un *thermocouple* est placé à proximité de la veilleuse afin de s'assurer qu'elle ne soit pas éteinte et d'éviter des émanations de gaz non brulé. En entrée de tambour (cylindre), le ruban se déplace sur une glissière munie d'un *capteur d'absence de ruban* permettant de détecter la fin de ruban.

Les **spécifications de fonctionnement** sont les suivantes :

— le *démarrage* permet à l'opérateur de lancer l'automatisme (on suppose que la température du tambour est proche de la température de consigne) ;

— lorsque la température du tambour est inférieure à la température de consigne, il est nécessaire de *chauffer le tambour* en alimentant le brûleur en gaz (ouverture de l'électrovanne) ;

— quelle que soit sa température, le tambour tourne ;

— dans le cas où l'on détecte une fin de ruban, on arrête l'automatisme et on génère l'alarme « fin de ruban » jusqu'à ce que l'opérateur acquitte l'alarme ;

— dans le cas où la veilleuse est éteinte, on arrête l'automatisme et on génère l'alarme « veilleuse éteinte » jusqu'à ce que l'opérateur acquitte l'alarme ;

— dans le cas où l'opérateur appuie sur l'arrêt d'urgence, on arrête l'automatisme jusqu'à ce que l'opérateur désactive l'arrêt d'urgence.

Notation pour les actions et les informations

Actions

Ouverture de l'électrovanne ..... OE
Rotation du tambour ..... R
Alarme veilleuse éteinte ..... AVE
Alarme fin de ruban ..... AFR

Informations

Température de tambour inférieure à température de consigne ..... $T < T°c$
Veilleuse éteinte ..... *ve*
Détecteur de fin de ruban ..... *dfr*
Démarrage ..... *d*
Acquittement alarme ..... *aa*
Arrêt d'urgence ..... *au*

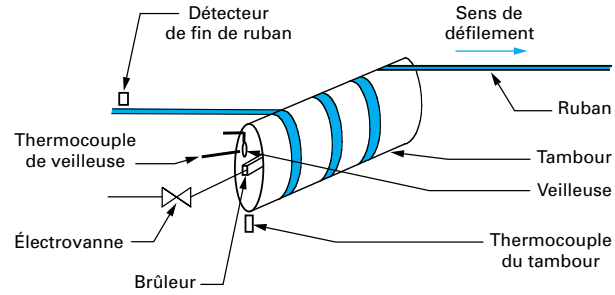


Figure 36 – Ligne de séchage de ruban

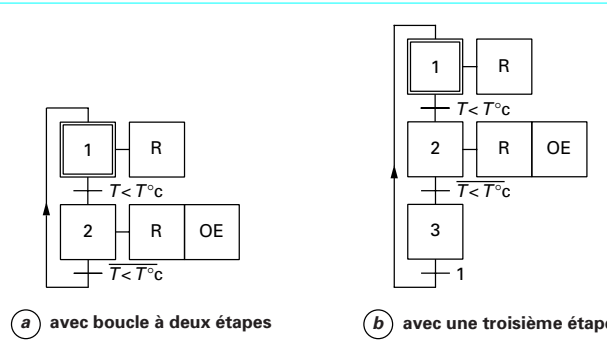


Figure 37 – GRAFCET 1 : fonctionnement normal

GRAFCET

Cette application peut se décomposer en deux GRAFCET, l'un gérant le fonctionnement normal (figure **37*a***) (la rotation du tambour et l'inflammation du gaz), l'autre gérant les arrêts et les défaillances (figure **38*a***).

Équations des étapes et des actions

Comme chaque GRAFCET possède une boucle à deux étapes, il est nécessaire de leur adjoindre une troisième étape, conformément à ce qui est décrit au paragraphe 2.5 (figures **37*b*** et **38*b***).

- *GRAFCET 1*

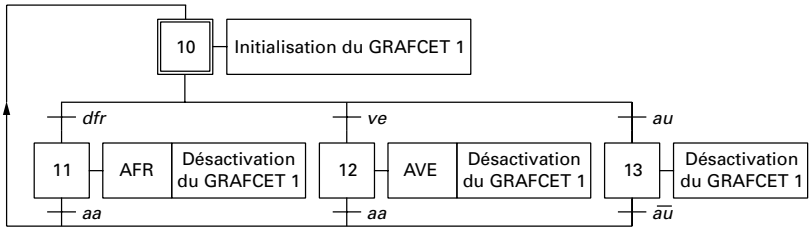
L'initialisation de ce GRAFCET se fait grâce à l'activité de l'étape 10 du GRAFCET 2. De même, l'arrêt d'urgence de type dur se fait par l'activité de l'étape 11, 12 ou 13 du GRAFCET 2.

$$\begin{cases} X1 = (X3 + X1 \cdot \overline{X2} + X10) \cdot \overline{X11 + X12 + X13} \\ X2 = (X1 \cdot (T < T°c) + X2 \cdot \overline{X3}) \cdot \overline{X10} \cdot \overline{X11 + X12 + X13} \\ X3 = (X2 \cdot \overline{(T < T°c)} + X3 \cdot \overline{X1}) \cdot \overline{X10} \cdot \overline{X11 + X12 + X13} \end{cases}$$

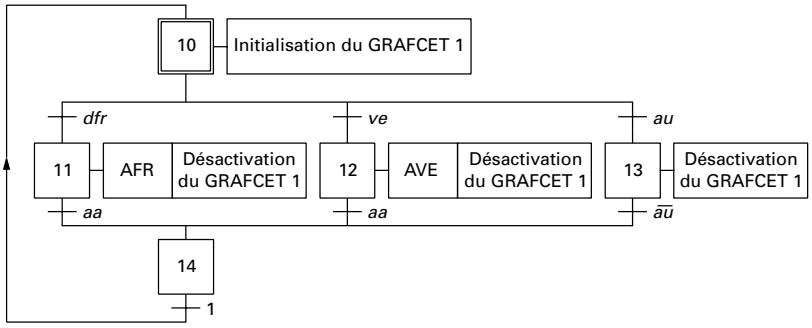
- *GRAFCET 2*

L'initialisation du GRAFCET 2 se fait par la mise en marche *d* déclenchée par l'opérateur.

$$\begin{cases} X10 = X14 + X10 \cdot \overline{(X11 + X12 + X13)} + d \\ X11 = (X10 \cdot dfr + X11 \cdot \overline{X14}) \cdot \overline{d} \\ X12 = (X10 \cdot ve + X12 \cdot \overline{X14}) \cdot \overline{d} \\ X13 = (X10 \cdot au + X13 \cdot \overline{X14}) \cdot \overline{d} \\ X14 = (X11 \cdot aa + X12 \cdot aa + X13 \cdot \overline{au} + X14 \cdot \overline{X10}) \cdot \overline{d} \end{cases}$$



(a) avec boucle à deux étapes



(b) avec une troisième étape

Figure 38 – GRAFCET 2 : gestion des modes de fonctionnement et traitement des défaillances

● *Actions*

Pour cet exemple, l'arrêt d'urgence de type doux n'est pas pris en compte dans les actions.

$$\begin{cases} R = X1 + X2 \\ OE = X2 \\ AFR = X11 \\ AVE = X12 \end{cases}$$

3. Technologies de réalisation

Nous montrons ici comment réaliser l'implantation technologique sur différents supports (câblé ou programmé).

Le GRAFCET, décrivant le déroulement séquentiel d'un automatisme, inclut dans les équations de ses étapes la **fonction mémoire**. En effet, celle-ci est réalisée par l'introduction de l'état du système dans les équations de sortie. Afin de réaliser technologiquement un GRAFCET en logique câblée, il faut pouvoir mettre en œuvre la fonction mémoire.

3.1 Logique câblée

3.1.1 Portes de la logique combinatoire

Considérons une fonction X logique de deux variables *a* et *b* réalisant une fonction mémoire. La variable *a* permet l'activation (mise à 1) et la variable *b* la désactivation (mise à 0). Dans le cas où les deux variables sont actionnées simultanément, il existe deux représentations en fonction du temps de la fonction mémoire X : mémoire à marche prioritaire et mémoire à arrêt prioritaire représentée par les chronogrammes de la figure **39**. La figure **40** donne la représentation de la fonction mémoire à l'aide de logigrammes. L'équation logique est :

— pour une mémoire à arrêt prioritaire :

$$X = (a + X) \cdot \overline{b}$$

— pour une mémoire à marche prioritaire :

$$X = a + X \cdot \overline{b}$$

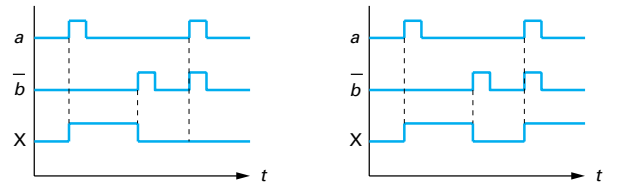
En câblant l'équation logique X*n* d'une étape du GRAFCET $Xn = Xn-1 \cdot tn-1 + \overline{Xn+1} \cdot Xn$, la fonction mémoire est réalisée (figure **41**). Celle-ci est à marche prioritaire (règle d'évolution 5 du GRAFCET, § 2.1).

Une étape de GRAFCET se symbolise sous forme d'un module de phase (figure **42**).

Exemple : GRAFCET à séquence unique

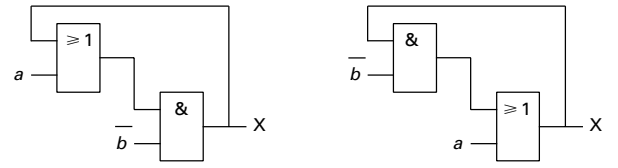
Chaque étape du GRAFCET sera câblée comme le module de phase décrit figure **42**. On réalise alors un séquenceur à base de portes logiques (figure **43**). Afin de tenir compte des modes de marche et d'arrêt du GRAFCET, le module d'une étape doit intégrer l'initialisation et l'arrêt d'urgence dur comme indiquent les équations logiques des étapes du GRAFCET qui le précède :

$$\begin{cases} X0 = (X2 \cdot b + \overline{X1} \cdot X0 + Init) \cdot \overline{AUD} \\ X1 = (X0 \cdot m + \overline{X2} \cdot X1) \cdot \overline{Init} \cdot \overline{AUD} \\ X2 = (X1 \cdot a + \overline{X0} \cdot X2) \cdot \overline{Init} \cdot \overline{AUD} \end{cases}$$



(a) mémoire à arrêt prioritaire (b) mémoire à marche prioritaire

Figure 39 – Chronogrammes



(a) mémoire à arrêt prioritaire (b) mémoire à marche prioritaire

Figure 40 – Logigrammes

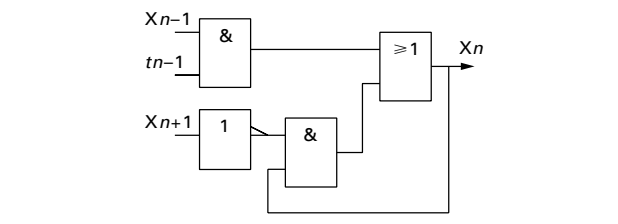


Figure 41 – Câblage d'une étape

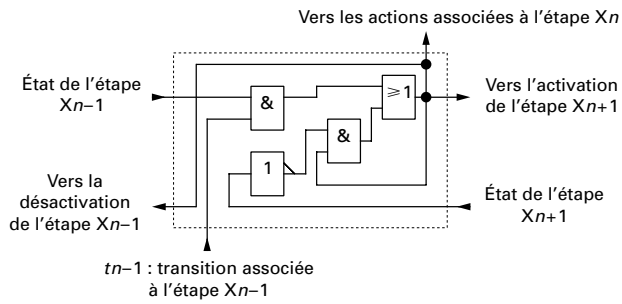


Figure 42 – Module de phase d'une étape

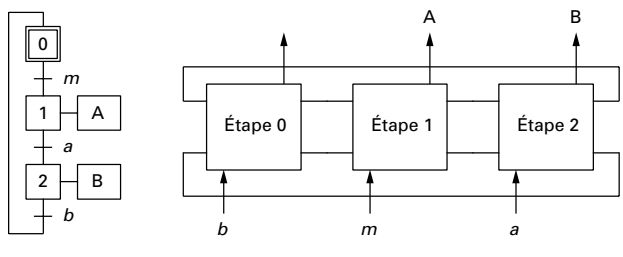
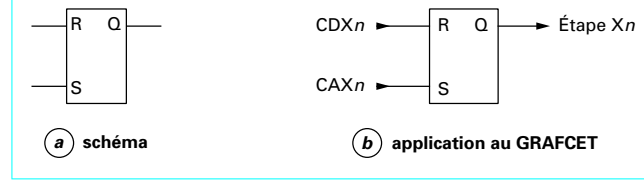


Figure 43 – Séquenceur à base de portes logiques



(a) schéma (b) application au GRAFCET

Figure 44 – Bascule RS

3.1.2 Bascules RS

La bascule RS (*reset*/*set*) (figure **44*a***) est le composant de base de la logique séquentielle. Dans la table de vérité (tableau **10**), on considère que R = S = 1 n'est pas une combinaison d'entrée possible. En effet, on ne peut demander simultanément l'activation et la désactivation. De manière générale, la table de vérité d'une bascule RS se met sous la forme du tableau **10** où $Qn + 1$ représente l'état qui suit Qn.

Tableau 10 – Table de vérité d'une bascule RS

R	S	$Qn + 1$
0	0	Qn
0	1	1
1	0	0
1	1	–

Nota : pour l'état R = S = 1, la valeur de la sortie de la bascule est indéterminée. Elle dépend uniquement des composants ou des retards dans la commande entre S et R. Si la bascule RS est fabriquée technologiquement avec des portes NOR, il y aura mémoire à activation prioritaire ; dans le cas de réalisation de bascules RS avec des portes NAND, il y aura mémoire à désactivation prioritaire.

Si la bascule RS est appliquée au GRAFCET (figure **44*b***) :

— la condition d'activation d'une étape est alors câblée sur le SET de la bascule ;

— la condition de désactivation d'une étape est câblée sur le RESET de la bascule.

Exemple : GRAFCET à séquence unique (figure **45*a***)

Intéressons-nous à l'étape 1 (l'arrêt d'urgence n'est pas représenté) :

$$X1 = X0 \cdot m + \overline{X2} \cdot X1 \cdot \overline{Init}$$

ou encore : $X1 = X0 \cdot m \cdot \overline{Init} + \overline{X2} \cdot \overline{Init} \cdot X1$

soit : $X1 = X0 \cdot m \cdot \overline{Init} + \overline{X2 + Init} \cdot X1 = CAX1 + \overline{CDX1} \cdot X1$

Il en résulte alors pour l'étape X1 (figure **45*b***) :

— sa condition d'activation est : $CAX1 = X0 \cdot m \cdot \overline{Init}$

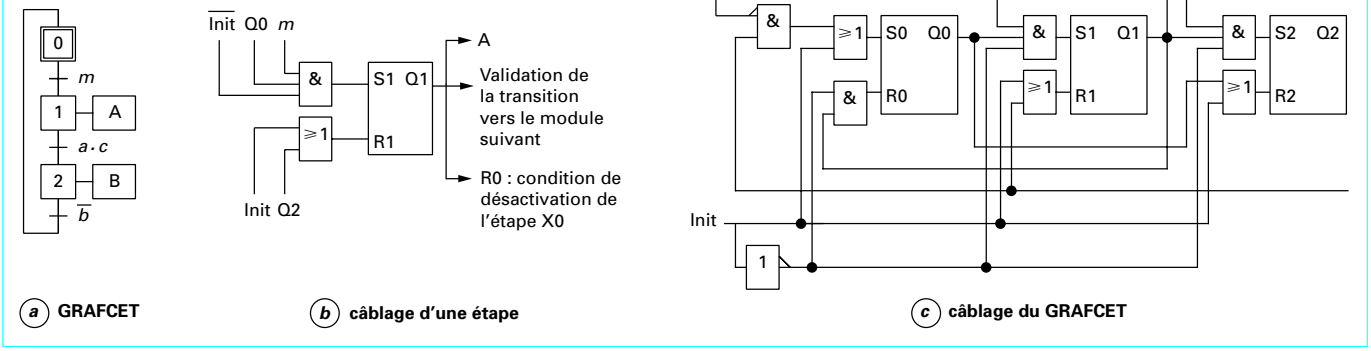
— sa condition de désactivation est : $CDX1 = X2 + Init$

Nota : lorsque la variable Init (initialisation du GRAFCET) est à 1, il y a désactivation de l'étape 1, quel que soit l'état de $X0 \cdot m$ (priorité à l'initialisation).

La figure **45*c*** présente le câblage du GRAFCET de la figure **45*a***.

3.1.3 Séquenceur électrique

Le séquenceur électrique est un système de commande d'automatisme séquentiel basé sur l'emploi de relais à accrochage qui mémorise chaque étape du cycle.



(a) GRAFCET — (b) câblage d'une étape — (c) câblage du GRAFCET

Figure 45 – GRAFCET à séquence unique

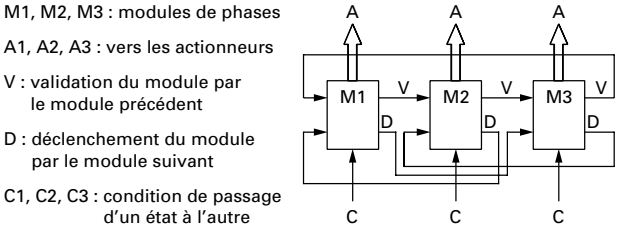


Figure 46 – Séquenceur électrique

Le **principe de fonctionnement** est décrit figure **46** :

— chaque module comprend un relais à accrochage ;

— l'association de modules n'autorise la réalisation d'actions que dans l'ordre de câblage du séquenceur.

Chaque module permet de :

— commander une action par un contact fermé pendant l'exécution de l'action ;

— déclencher le module ayant commandé l'action précédente ;

— valider le module suivant, c'est-à-dire permettre à ce module de recevoir l'ordre d'enclenchement.

3.1.4 Séquenceur pneumatique

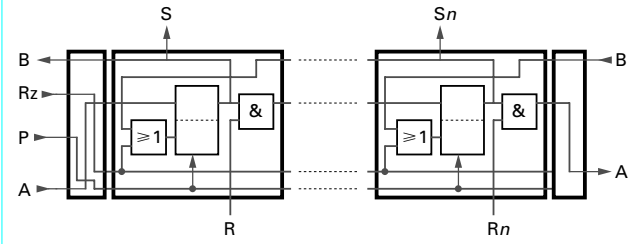
Le module séquenceur pneumatique permet, comme le module séquenceur électrique (§ 3.1.3), de résoudre directement tous les schémas de postes automatiques à partir d'un GRAFCET. Cette technologie, relativement coûteuse, tend à disparaître. Elle reste néanmoins utilisée dans certaines applications où le milieu de fonctionnement est hostile et n'admet pas le passage de courant électrique (risque d'explosion par exemple).

Le **principe de fonctionnement** est décrit figure **47*a*** :

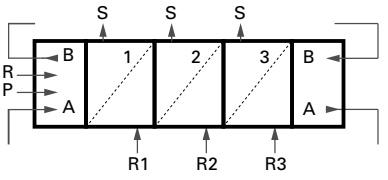
— un séquenceur est une association de modules ;

— le nombre de modules correspond au nombre d'étapes du cycle à réaliser.

La mémoire du module de phase est mise à l'état 1 par le signal arrivant de la cellule ET du module de phase précédent. La sortie de cette mémoire provoque alors trois actions :



(a) principe de fonctionnement



(b) schéma d'usage

Figure 47 – Séquenceur pneumatique

— elle assure le signal de commande S vers l'extérieur prévu à cette phase du cycle ;

— elle remet à zéro le module de phase précédent à travers la cellule OU ;

— elle alimente une entrée de la cellule ET.

La figure **47*b*** donne le schéma d'usage suivant une représentation Télémécanique.

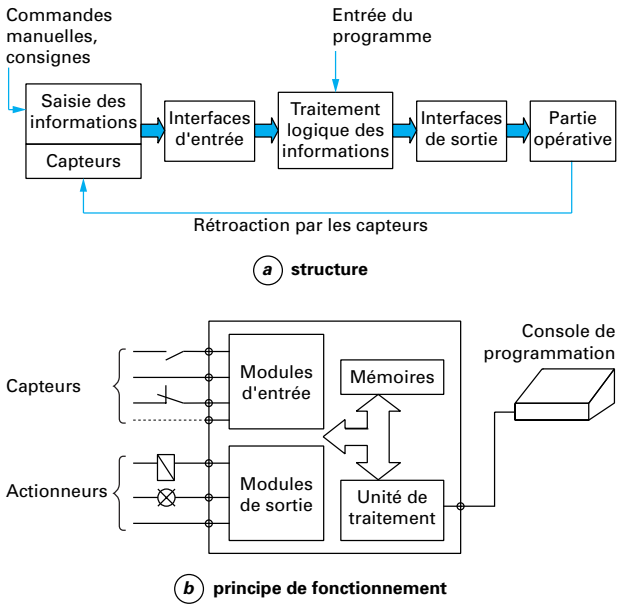


Figure 48 – Automate programmable industriel

3.2 Automates programmables industriels (API)

L'automate programmable est un système de traitement logique d'informations dont le programme de fonctionnement est effectué à partir d'instructions établies en fonction du processus à réaliser. C'est un système informatique dédié aux applications d'automatisme.

La **structure des systèmes automatiques** (figure **48*a***) comprend les éléments suivants :

— saisie des informations : consignes, capteurs mécaniques (contacts), pression, température, déplacement, etc. ;
— interfaces d'entrée : isoler électriquement avec une isolation galvanique (découplage) le circuit puissance et le traitement, mise en forme du signal, système antiparasite, etc. ;
— traitement logique : effectuer des opérations ET, OU, lire l'état d'une variable, ranger le résultat dans une variable, mémoire, etc. ;
— interfaces de sortie : elles permettent de commander des actions (relais, électrovannes, contacteurs, moteurs, etc.) avec une isolation galvanique.

Le **principe général de fonctionnement** d'un API est décrit figure **48*b***. L'API est généralement construit autour d'un microprocesseur. Les entrées sont nombreuses et acceptent des signaux venant de capteurs industriels et les sorties sont traitées pour actionner des contacteurs, des relais, etc. Les langages de programmation sont simples et accessibles rapidement par les automaticiens.

La mémoire est en partie prise par le système d'exploitation (contrôle du fonctionnement de l'API, gestion interne des traitements, gestion des ressources, etc.).

Le programme de traitement des informations est *stocké en mémoire*, l'*unité de traitement* pilote le fonctionnement de l'automate et la **console de programmation** assure le dialogue entre l'opérateur et l'automate pendant la phase d'écriture et de mise au point du programme.

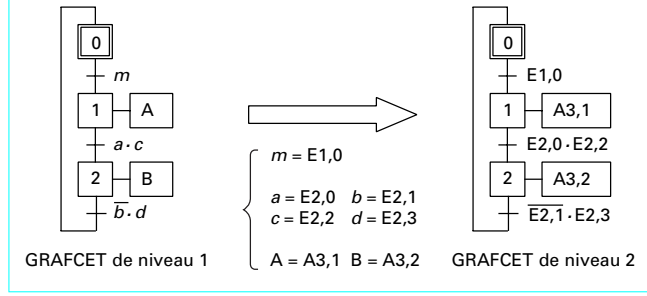


Figure 49 – Programmation en langage booléen

Un automate traite des variables booléennes et numériques. En général, on distingue les entrées (en lecture seule), les sorties (en écriture) et les variables internes (mémorisation des calculs intermédiaires).

3.2.1 Programmation en langage booléen

Le langage booléen de l'automate programmable dispose généralement de cinq fonctions combinatoires élémentaires :

— ET logique entre une variable indiquée et le résultat précédent ;
— OU logique entre une variable indiquée et le résultat précédent ;
— LIRE l'état de la variable indiquée ;
— RANGER le résultat dans la variable indiquée ;
— NON qui s'utilise en combinaison des quatre opérations précédentes (inverser l'état de la variable sélectionnée).

L'exemple de la figure **49** montre comment programmer un GRAFCET à séquence unique. La programmation est faite en langage STEP 5 sur une console PG605 pour les automates de type SIMATIC S5 de Siemens. Pour chaque étape X*i* du GRAFCET, il faut écrire sa condition d'activation, sa condition de désactivation et l'action associée (tableau **11**). Pour la programmation, on utilise des variables internes de l'automate notées M*i,j* (*i* = 0 à 255 et *j* = 0 à 7).

Tableau 11 – Équations des étapes pour le langage booléen

Étape	CA	CD	Remarques
0	M0,2 · /E2,1 · E2,3 + E1,1	M0,0 · /E1,1	E1,1 : initialisation du GRAFCET /E1,1 : complément de E1,1
1	M0,0 · E1,0 · /E1,1	M0,2 + E1,1	
2	M0,1 · E2,0 · E2,2 · /E1,1	M0,0 + E1,1	

Notations
CAX*i* : condition d'activation de l'étape *i*
CDX*i* : condition de désactivation de l'étape *i*
M0,*i* : variable interne associée à l'étape *i*

Langage de programmation
U : fonction ET
UN : fonction NON
O. : fonction OU
S : SET (mise à 1)
R : RESET (mise à 0)

Programmation d'une étape

```
U « CAXi »
S  M0, i     (mise à 1 : SET )
U « CDXi »
R  M0, i     (mise à 0 : RESET )
```

Programme

Étape 0

```
U   M0, 2
UN  E2, 1
U   E2, 3
O.  E1, 1      Condition d'activation de l'étape 0

S   M0, 0  ⟶   Mise à 1 (SET) de l'étape 0

U   M0, 1
UN  E1, 1      Condition de désactivation de l'étape 0

R   M0, 0  ⟶   Mise à 0 (RESET) de l'étape 0
```

Étape 1

```
U   M0, 0
U   E1, 0
UN  E1, 1      Condition d'activation de l'étape 1

S   M0, 1  ⟶   Mise à 1 (SET) de l'étape 1

U   M0, 2
O.  E1, 1      Condition de désactivation de l'étape 1

R   M0, 1  ⟶   Mise à 0 (RESET) de l'étape 1
```

Étape 2

```
U   M0, 1
U   E2, 0
U   E2, 2
UN  E1, 1      Condition d'activation de l'étape 2

S   M0, 2  ⟶   Mise à 1 (SET) de l'étape 2

U   M0, 0
O.  E1, 1      Condition de désactivation de l'étape 2

R   M0, 2  ⟶   Mise à 0 (RESET) de l'étape 2
```

Actions

```
U   M0, 1
=   A3, 1     Activation de l'action A3,1

U   M0, 2
=   A3, 2     Activation de l'action A3,2
```

3.2.2 Programmation en diagramme en échelle (*ladder*)

Comme pour la programmation en langage booléen, il faut établir pour chaque étape du GRAFCET les équations des conditions d'activation et de désactivation. La condition d'activation d'une étape est câblée à une bobine SET (S) et la condition de désactivation est câblée sur une bobine RESET (R). La figure **50** montre le principe de programmation d'une étape en *ladder*.

Exemple : GRAFCET à séquence unique

Ce genre de programmation peut se faire en langage PL7-2 sur un terminal TSX-T407 pour les automates Télémécanique TSX 17-20/27/47-J/47-10/20.

Les entrées de l'automate sont notées de I0,0 à I0,21 et les sorties de O0,0 à O0,10 (figure **51*a***).

Les conditions d'activation et de désactivation des étapes sont définies dans le tableau **12**.

La programmation se fait par label (L) comportant au maximum quatre lignes (figure **51*b***).

Tableau 12 – Équations logiques pour la programmation en *ladder*

Étape	CA	CD	Remarques
0	B2 · /I2,1 · I2,3 + I1,1	B1 · /I1,1	I1,1 : initialisation du GRAFCET /I1,1 : complément de E1,1
1	B0 · I1,0 · /I1,1	B2 + I1,1	
2	B0 · I2,0 · I2,2 · /I1,1	B0 + I1,1	

3.2.3 Programmation en GRAFCET

Le GRAFCET est programmé sur un automate Télémécanique TSX-47 avec le langage GRAFCET PL7-2 sur un terminal TSX-T407. La programmation est structurée en trois parties (préliminaire, séquentiel et postérieur) que l'automate exécute consécutivement et en cycle (figure **52**).

Nota : la gestion des sorties se programme dans le traitement postérieur plutôt que dans le traitement séquentiel, comme l'autorise le langage PL7-2. En effet, cette façon d'opérer est plus conforme au principe du GRAFCET et permet de traiter plus facilement les actions conditionnelles.

Considérons l'exemple décrit figure **53** et définissons les différents traitements.

Traitement préliminaire (PRE)

L'automate possède des bits systèmes tels que :

— SY21 : initialisation du GRAFCET. Si SY21 = 1, alors il y a désactivation des étapes non initiales et activation des étapes initiales ;
— SY22 : RAZ du GRAFCET (= 1 : AUD) ;
— SY09 : mise à 0 des sorties. (= 1 : AUd).

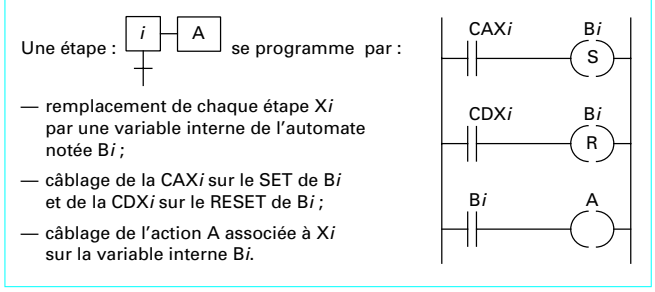


Figure 50 – Principe de programmation d'une étape en *ladder*

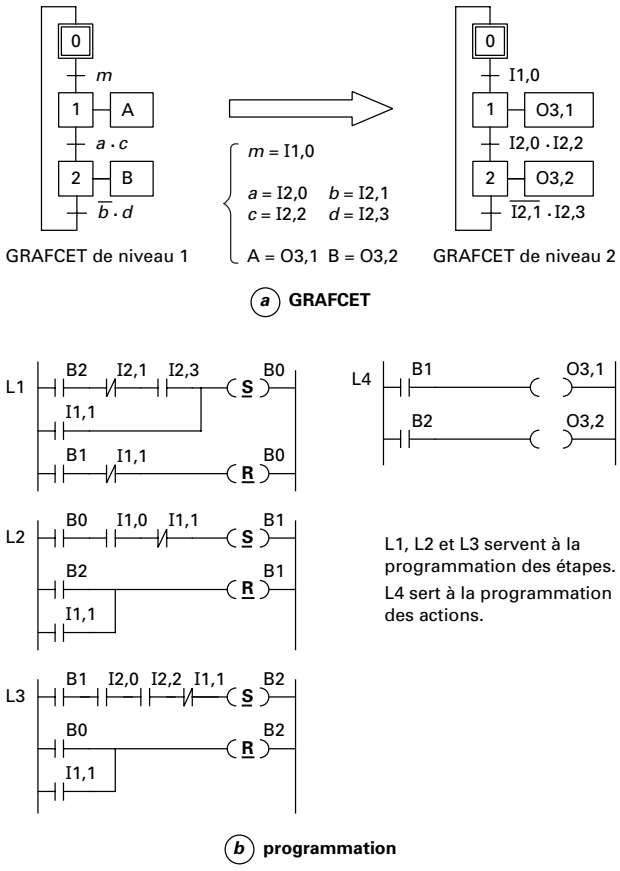


Figure 51 – Programmation d'un GRAFCET en *ladder*

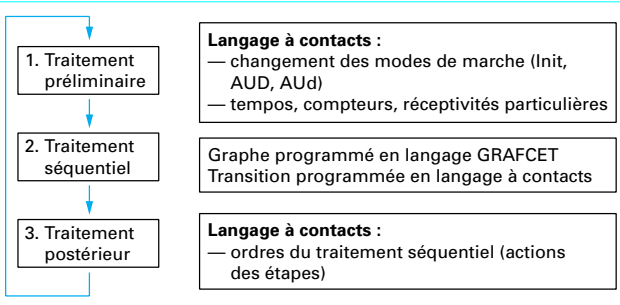


Figure 52 – Traitement exécuté par l'API

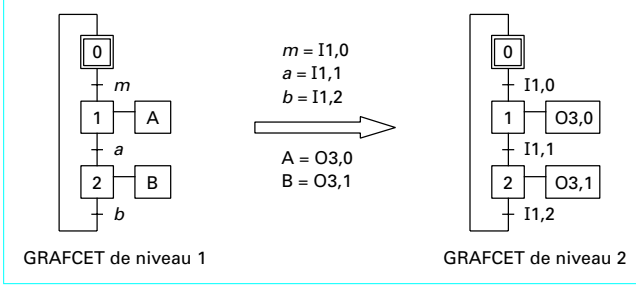


Figure 53 – Principe de programmation d'un API en GRAFCET

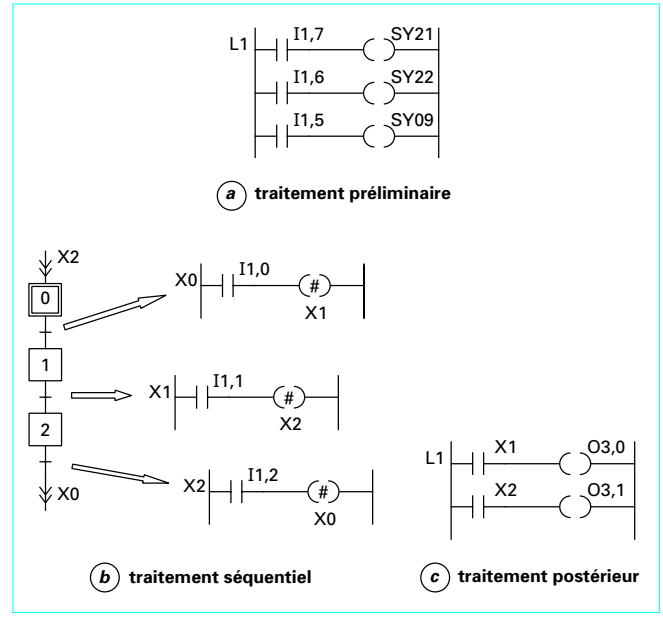


Figure 54 – Programmation des traitements

Pour l'exemple, on prend SY21 câblé sur I1,7, SY22 câblé sur I1,6 et SY09 câblé sur I1,5 (figure **54*a***).

Traitement séquentiel (SEQ)

Le traitement séquentiel de l'automate (figure **54*b***) contient la partie séquentielle du GRAFCET. On y trouve les étapes, les transitions et les réceptivités associées aux transitions.

Traitement postérieur (POS)

Le traitement postérieur (figure **54*c***) est réservé à la programmation des actions du GRAFCET.

3.3 Système informatique

La programmation d'un GRAFCET en **langage évolué** sur ordinateur représente, comme pour l'automate, une solution plus flexible que la logique câblée. En effet, il est beaucoup plus aisé de modifier des lignes de programme sur un ordinateur ou sur une console d'automate que de modifier un câblage.

Nous allons détailler la transcription d'un GRAFCET possédant une divergence en **langage Pascal**. Comme il faut tenir compte du traitement séquentiel des informations, le programme est structuré de façon séquentielle par une lecture des données (entrées), la gestion des étapes (activation et désactivation) et l'activation des sorties. Ce schéma de programmation en langage évolué (figure **55**) montre la même évolution que la structure des systèmes automatiques traitée par l'automate programmable.

À titre d'exemple, nous présentons la programmation d'un GRAFCET à séquences simultanées (figure **56**).

On suppose que la liaison du PC à l'automate se fasse par un port parallèle possédant un port A d'adresse $1AC de 8 bits utilisé en entrée et un port B d'adresse $1AD de 8 bits utilisé en sortie.

Ce programme présente les principes de programmation d'un GRAFCET. Il peut être modifié en fonction de l'application et notamment pour la gestion des modes de marche et d'arrêt.

```
{ -- début du programme GRAFCET -- }
Program GRAFCET ;
Uses crt, dos ;
Var   x,y : array[0..6] of boolean ;          { étapes actives : 1 - étapes non actives : 0 }
      entree, sortie : byte ;                 { variables d'état des ports d'entrée et de sortie }
      i : integer ;                           { variable de boucle for}
      AUdoux, AUdur : boolean ;               { variables désignant les arrêts d'urgence}
      touche : char ;                         { touches du clavier }

{ -- procédure d'initialisation -- }
Procedure initialisation ;
begin
   x[0] := true ;                             { initialisation du GRAFCET : mise à 1 de l'étape 0 }
   for i := 1 to 6 do x[i] := false ;         { et mise à 0 des autres étapes }
   AUdur = false ;
   AUdoux = false ;
   ...                                        { mettre ici la configuration des ports entrée/sortie}
end ;

{ -- programme principal -- }
begin
   initialisation ;
   repeat
        for i := 0 to 6 do y[i]:=x[i] ;       { image de l'état d'activité des étapes du GRAFCET }
        entree := port[$1AC] ;                { lecture des entrées }
        if ( x[0] and (entree and 1) = 1 ) then   { si m vrai (bit 0 du port A est à 1)...  }
           begin                              { ... et si l'étape 0 est active  }
              y[0] := false ; y[1] := true ; y[4] := true ;   { alors activation des étapes 1 et 4... }
           end ;                              { ... et désactivation de l'étape 0    }
        if ( x[1] and (entree and 2) = 2 ) then   { si a vrai (bit 1 du port A est à 1)... }
           begin                              { ... et si l'étape 1 est active }
              y[1] := false ; y[2] := true ;  { alors, activation de l'étape 2... }
           end ;                              { ... et désactivation de l'étape 1  }
        if ( x[2] and (entree and 4) = 4 ) then   { si b vrai (bit 2 du port A est à 1)...  }
           begin                              { ... et si l'étape 2 est active }
              y[2] := false ; y[3] := true ;  { alors, activation de l'étape 3... }
           end ;                              { ... et désactivation de l'étape 2  }
        if ( x[4] and (entree and 8) = 8 ) then   { si c vrai (bit 3 du port A est à 1)...  }
           begin                              { ... et si l'étape 4 est active }
              y[4] := false ; y[5] := true ;  { alors, activation de l'étape 5 ...}
           end ;                              { ... et désactivation de l'étape 4  }
        if ( x[5] and (entree and 16) = 16 ) then   { si d vrai (bit 4 du port A est à 1)...  }
           begin                              { ... et si l'étape 4 est active }
              y[5] := false ; y[6] := true ;  { alors, activation de l'étape 6... }
           end ;                              { ... et désactivation de l'étape 5  }
        if (x[3] and x[6] ) then              { test de l'activité des étapes  4 et 6  }
           begin
              y[3] := false ; y[6] := false ; y[0] := true ;   { si vraies, activation de l'étape 0 }
           end ;                              { ... et désactivation des étapes 3 et 6  }
        for i := 0 to 6 do x[i]:=y[i] ;       { restitution de l'image de l'état }
                                              { d'activité des étapes du GRAFCET }
        sortie := 0 ;                         { Établissement du port de sortie }
        if  not AUdoux then
        begin
              if y[1] then sortie := sortie or 1 ;   { Si étape 1 active, faire action A }
              if y[2] then sortie := sortie or 2 ;   { Si étape 2 active, faire action B }
              if y[4] then sortie := sortie or 4 ;   { Si étape 4 active, faire action C }
              if y[5] then sortie := sortie or 8 ;   { Si étape 5 active, faire action D }
        end
        port[$1AC+1] := sortie ;              { Activation du port de sortie : port B }
        if  keypressed                        { Une touche du clavier est actionnée }
            then touche := upcase(readkey) ;  { Cette touche est lue }
        if touche = 'A' then AUdur = true ;   { 'A'    = touche de l'AUdur }
        if touche = 'I' then initialisation ; { 'I'    = touche de l'initialisation }
        if touche = 'S' then AUdoux = true ;  { 'S'    = touche de l'AUdoux }
        if touche = 'R' then AUdoux = false ; { 'R'    = touche de fin de l'AUdoux }
until AUdur  ;
end. { fin du programme }
```

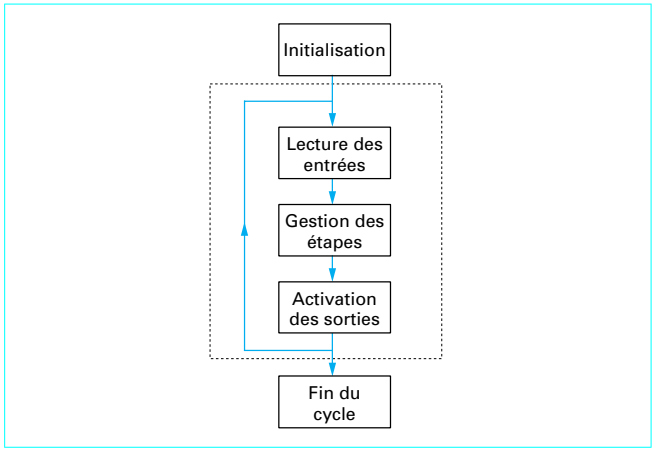


Figure 55 – Schéma de programmation en langage évolué

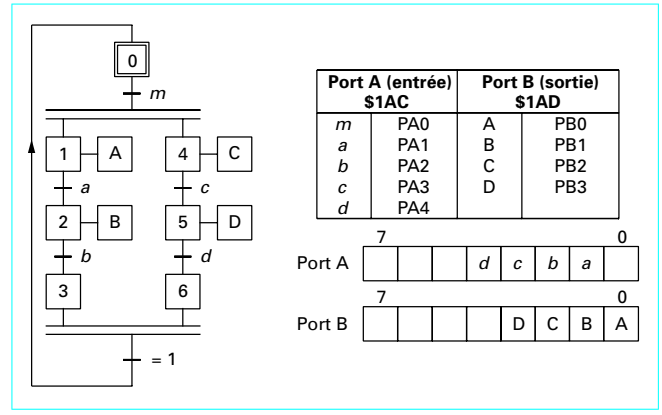


Figure 56 – Programmation d'un GRAFCET à séquences simultanées en langage évolué

3.4 Conclusion

Ainsi, il est possible de réaliser de manière technologique le GRAFCET décrivant le traitement séquentiel de processus. La réalisation est soit rigide et peu évolutive (câblage avec des éléments de logique combinatoire ou séquentielle, de séquenceur électrique ou pneumatique), soit flexible par l'utilisation d'un système informatique dédié aux automatismes (API) ou d'un système informatique classique.