heig-vd
Haute École d'Ingénierie et de Gestion du Canton de Vaud

Département TIN
Technologies Industrielles

Systèmes Industriels
et
Formation en emploi

Automatisme de séquences

Support de cours

Alain Beuret

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

9 septembre 2010 Version 1.4

Historique des modifications

Version	Date	Auteur	Modifications
1.4	09.09.10	Alain Beuret	Création document

Convention typographique

Exemples	Descriptions
Texte	Texte du cours, sans serif normal taille 12
Important	Termes importants, sans serif gras taille 12
English / *Deutsch*	Termes dans un langue étrangère, sans serif italique taille 12
Exemple, note	Exemple et note, sans serif taille 10
i, x, ω, φ I, X, Ω	Variables mathématiques, serif italique minuscules Constantes mathématiques, sérif italique majuscules
sin, tg, lim	Fonctions mathématiques, serif gras
`a := b + c;`	Code programme, chasse fixe normal
`if … then … else`	Mots réservés programme, chasse fixe gras
`' Commentaire`	Commentaires programme, chasse fixe italique

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Copyright © Alain Beuret, 2010

La copie de tout ou partie de ce document, quelle qu'en soit la forme et le support, n'est pas autorisée sans l'accord formel de l'auteur.

Par ailleurs, celui-ci ne prend aucune responsabilité relative à des erreurs éventuelles du contenu, ni aux droits de reproduction de certaines des images utilisées.

Toutes propositions d'améliorations et de corrections seront les bienvenus.

alain.beuret@heig-vd.ch

Table des matières

1 Logique combinatoire et séquentielle 1-2

1.1 Logique combinatoire....1-2

1.1.1 Fonctions incomplètement définies....1-3

1.2 Logique séquentielle....1-4

1.2.1 Analyse d'un système séquentiel....1-6

1.2.1.1 Chronogramme....1-6

1.2.1.2 Tableau de séquences....1-7

1.2.1.3 Graphe de fluence....1-8

1.2.1.4 Matrice primitive des états....1-9

1.2.2 Synthèse d'un système séquentiel....1-10

1.2.2.1 Réduction du nombre de variables....1-10

1.2.3 Machine de Moore....1-12

1.2.4 Méthode de Huffman....1-13

1.2.4.1 Contraction de la matrice d'état....1-13

1.2.4.2 Réduction des états équivalents....1-14

1.2.4.3 Détermination des variables d'état....1-15

1.2.4.4 Matrice d'excitation....1-16

1.2.4.5 Mise en équations des variables d'état....1-17

1.2.4.6 Équations des sorties....1-19

1.2.4.7 Représentation schématique....1-20

2 Le GRAFCET 2-2

2.1 Système automatisé de production....2-2

2.2 Le langage GRAFCET....2-4

2.2.1 Les étapes....2-4

2.2.1.1 Étapes particulières....2-5

2.2.2 Les transitions....2-6

2.2.2.1 Repère de liaison....2-6

2.2.3 Règles du GRAFCET....2-7

2.2.3.1 Syntaxe....2-7

2.2.3.2 Évolution....2-7

2.2.4 Structure du GRAFCET....2-10

2.2.4.1 Séquence linéaire....2-10

2.2.4.2 Sélection de séquences....2-11

2.2.4.2.1 Saut d'étapes....2-11

2.2.4.2.2 Reprise de séquence....2-12

2.2.4.3 Séquences parallèles....2-13

2.2.4.4 Structures particulières....2-14

2.2.5 Les actions....2-15

2.2.5.1 Action vide....2-15

2.2.5.2 Action continue....2-16

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.5.3 Action mémorisée....2-16
2.2.5.4 Action conditionnelle....2-17
2.2.5.5 Action dépendantes du temps....2-17
2.2.5.6 Action à l'activation et à la désactivation....2-18
2.2.5.7 Action au franchissement....2-18
2.2.6 Les réceptivités....2-19
2.2.6.1 Réceptivité toujours vraie....2-19
2.2.6.2 Font montant ou descendant....2-20
2.2.6.3 Réceptivité dépendante du temps....2-20
2.2.6.4 Prédicat....2-21
2.3 Structuration....2-22
2.3.1 Macro-étape....2-23
2.3.2 Grafcets partiels....2-25
2.3.2.1 Forçages....2-26
2.3.2.2 Réutilisation d'une séquence....2-28
2.3.3 Encapsulation....2-28
3 GEMMA 3-2
3.1 STRUCTURE DU GEMMA....3-4
3.1.1 Partie commande hors fonction....3-4
3.1.2 Partie commande en fonction....3-4
3.1.3 Familles de procédures....3-4
3.1.3.1 Procédures de fonctionnement (F)....3-5
3.1.3.2 Procédures d'arrêt (A)....3-6
3.1.3.3 Procédures en défaillances (D)....3-7
3.2 Utilisation du GEMMA....3-8
3.2.1 Sélection des modes de marche et d'arrêt....3-8
3.2.2 Notion de boucle opérationnelle....3-8
3.2.3 Analyse des boucles opérationnelles....3-8
3.2.3.1 Marche normale....3-8
3.2.3.2 Marche de réglage....3-8
3.2.3.3 Arrêt de sécurité....3-8
3.2.4 Conditions d'évolution....3-9
4 CoDeSys : Langage SFC 4-2
4.1 Langage SFC....4-2
4.1.1 Étape....4-2
4.1.2 Action....4-3
4.1.2.1 Étapes simplifiées....4-3
4.1.2.1.1 Action d'entrée et de sortie....4-3
4.1.2.1.2 Étape active....4-3
4.1.2.2 Étapes CEI....4-4
4.1.2.2.1 Qualificatifs....4-6

Ce chapitre n'est pas disponible en consultation publique

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Ce chapitre n'est pas disponible en consultation publique

4.1.2.2.2 Variables implicites....4-6
4.1.3 Transitions....4-7
4.1.4 Structures....4-7
4.1.4.1 Séquence alternative....4-7
4.1.4.2 Séquences simultanées....4-8
4.1.4.3 Saut....4-8
4.2 Environnement de développement CoDeSys....4-9
4.2.1 Fenêtre principale....4-9
4.2.2 Gestion de projets....4-11
4.2.3 Gestion des objets....4-17
4.2.3.1 Objet....4-17
4.2.3.2 Dossier....4-17
4.2.3.3 Commandes....4-18

5 Annexes 5-2

5.1 Glossaire....5-2
5.2 Bibliographie....5-4
5.3 Webographie....5-4

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Introduction

Ce document est le support du cours d'automatisme de séquences (**SEQ et AutoSeq**) pour les étudiants ingénieurs en systèmes industriels de la heig-vd. Ce cours vise à donner à l'étudiant les concepts généraux d'automatisation de séquences des machines. Le but et de lui fournir les éléments lui permettant d'analyser et de concevoir un système séquentiel.

Le premier chapitre résume les connaissances de base en logique booléennes et donne quelques outils d'analyse et de synthèse de systèmes séquentiels.

Le deuxième chapitre présente le langage de spécification GRAFCET pour l'établissement de diagrammes fonctionnels.

Le troisième chapitre décrit la méthode d'étude des modes de marche et d'arrêt des machines et l'utilisation du GEMMA.

Le quatrième chapitre introduit la programmation d'automates en langage IEC 61 131-3 SFC.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Chapitre 1

Logique combinatoire et séquentielle

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

1 Logique combinatoire et séquentielle

1.1 Logique combinatoire

Un système logique est dit **combinatoire** si l'état de ses sorties Q_k ne dépend que de l'état de ses entrées I_i. Le système combinatoire ne doit donc pas présenter de réactions de la sortie sur l'entrée, de sorte à ce que l'état de la sortie ne dépende pas de l'histoire du système.

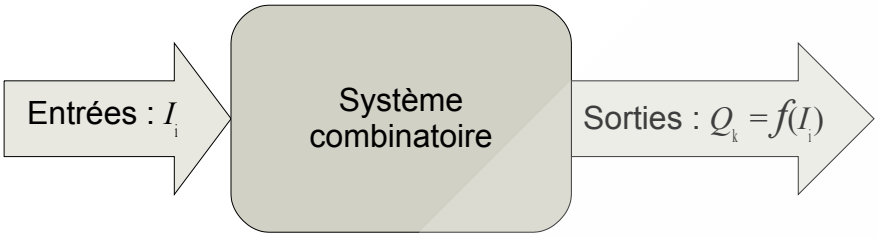


Figure 1.1.1: Système combinatoire

Les différents états d'un système combinatoire peuvent être représentés par une table de vérité. Cette table représente à tout instant l'état des entrées et des sorties d'un système combinatoire. Elle sert de base à l'établissement des équations logiques (booléennes) qui caractérisent le fonctionnement du système.

Chaque ligne de la table où l'état de la sortie est à '`1`' donne un terme partiel de l'équation qui est la fonction **`ET`** des variables entrées à '`1`' et du complément des variables d'entrée à '`0`'.

L'équation complète, exprimée sous sa forme canonique, est donnée par la fonction **`OU`** des termes partiels.

Exemple

Fonction majorité à 3 entrées: La sortie Q est l'état '`1`' si au moins 2 des entrées sont à '`1`'.

a	b	c	Q	Termes partiels
0	0	0	0	
1	0	0	0	
0	1	0	0	
1	1	0	1	$a \cdot b \cdot \bar{c}$
0	0	0	0	
1	0	1	1	$a \cdot \bar{b} \cdot c$
0	1	1	1	$\bar{a} \cdot b \cdot c$
1	1	1	1	$a \cdot b \cdot c$
L'équation canonique complète est : $Q = (a \cdot b \cdot \bar{c}) + (a \cdot \bar{b} \cdot c) + (\bar{a} \cdot b \cdot c) + (a \cdot b \cdot a)$				

Tableau 1: Exemple table de vérité - Fonction "majorité"

L'expression canonique de l'équation ainsi obtenue n'est pas forcément optimale car elle ne fait pas nécessairement intervenir un nombre minimal d'opérations logiques. Cette équation doit être simplifiée en appliquant les théorèmes de la l'algèbre booléenne. La simplification est importante pour toute réalisation technique car du nombre d'opérations logiques dépend la complexité du système.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Exemples

1. Simplification de l'équation canonique de la fonction majorité à 3 entrées :
   $Q=(a\cdot b\cdot\bar{c})+(a\cdot\bar{b}\cdot c)+(\bar{a}\cdot b\cdot c)+(a\cdot b\cdot a)$
   $Q=[(a\cdot b\cdot c)+(a\cdot b\cdot\bar{c})]+[(a\cdot b\cdot c)+(a\cdot\bar{b}\cdot c)]+[(a\cdot b\cdot c)+(\bar{a}\cdot b\cdot c)]$
   $Q=[(a\cdot b)\cdot c+(a\cdot b)\cdot\bar{c}]+[(a\cdot c)\cdot b+(a\cdot c)\cdot\bar{b}]+[(b\cdot c)\cdot a+(b\cdot c)\cdot\bar{a}]$
   $Q=[(a\cdot b)\cdot(c\cdot\bar{c})]+[(a\cdot c)\cdot(b\cdot\bar{b})]+[(b\cdot c)\cdot(a\cdot\bar{a})]$
   Équation simplifiée
   $Q=(a\cdot b)+(a\cdot c)+(b\cdot c)$
2. Réaliser la fonction $Q=\bar{a}+b+\bar{c}$ ($Q=$ **NON**(a) **OU** b **OU NON**(c)) uniquement avec des opérateurs **ET** et **NON** :
   $Q=\bar{a}+b+\bar{c}=\overline{\overline{\bar{a}+b}}+\bar{c}=\overline{\bar{a}\cdot\bar{b}}+\bar{c}=\overline{\overline{\overline{\bar{a}\cdot\bar{b}}+\bar{c}}}=\overline{\overline{\overline{\bar{a}\cdot\bar{b}}}\cdot\bar{\bar{c}}}=\overline{a\cdot\bar{b}\cdot c}$
   qui s'exprime $Q=$ **NON**$[a$ **ET NON**(b) **ET** $c]$

1.1.1 Fonctions incomplètement définies

Souvent, la variable de sortie d'un système que l'on souhaite exprimer par une relation booléenne n'est pas déterminée pour toutes les combinaisons des variables d'entrée. C'est généralement le cas lorsque ces combinaisons ne sont physiquement pas réalisables, ou pas spécifiées. Dans un tel cas, il n'est pas nécessaire d'imposer une valeur '**0**' ou '**1**' en établissant la table de vérité.

Un tel état indéterminé est noté '**x**'. Cette particularité permet souvent une meilleure simplification de l'équation logique. Celle-ci comprendra donc moins de terme, et sera techniquement plus simple à réaliser.

En déterminant l'équation, chaque case indéterminée se voit attribuer une valeur '**0**' ou '**1**', ce qui ne pose aucun problème puisqu'un tel état n'est pas possible. En d'autres termes, il est improductif de poser des contraintes supplémentaires lorsque ce n'est pas nécessaire.

Exemple

Les affichages numériques à 7 segments, qui permettent d'afficher les nombres de 0 à 9, doivent généralement correspondre à une combinaison de 4 signaux binaires. L'état de chaque segment (allumé ou éteint) est parfaitement déterminé pour les chiffres de 0 à 9, donc pour les combinaisons d'entrée comprises entre **0000** et **1001**.

Les combinaisons d'entrée comprises entre **1010** et **1111** ne peuvent normalement pas se produire (puisque l'on compte en décimal). Il n'est pas utile d'imposer un état allumé ou éteint à chaque segment pour ces combinaisons, et l'on indiquera un '**x**' pour chacun de ces état.

Évidemment, une fois que la simplification aura été réalisée, chaque '**x**' aura été remplacé par un '**0**' ou par un '**1**'.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2 Logique séquentielle

Un système logique est dit **séquentiel** si l'état de sa sortie dépend des variables d'entrée et des états antérieurs de ce système, c'est-à-dire qu'il se souvient de son histoire. La logique séquentielle prend en compte les **états** successifs du système.

L'histoire d'un système est représentée par une succession d'états que prend le système au cours du temps. Le changement d'état est provoqué par une variation des entrées I_i. Les sorties Q_k sont fonction de l'état du système. L'historique d'un système est décrit par un ensemble de variables appelées **variables d'état** S_j qui interviennent dans les équations caractéristiques du système.

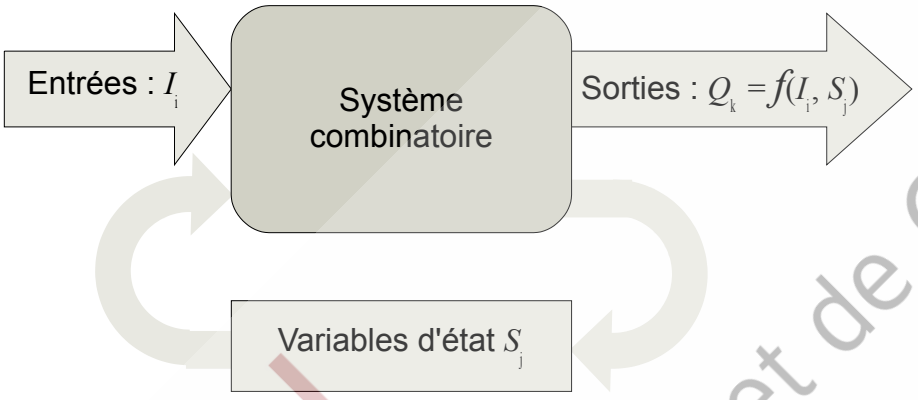


Figure 1.2.1: Système séquentiel

Pour décrire le fonctionnement d'un système séquentiel plusieurs outils d'analyse sont disponibles, principalement :

- le chronogramme;
- le graphe de fluence;
- la matrice d'états.

Différentes méthodes sont applicables pour faire la synthèse du système en vue de concevoir la partie commande :

- les machines de Moore et de Maely;
- le GRAFCET;
- les réseaux de Pétri[1].

Le but de ce cours n'est pas d'étudier en détail tous ces outils, nous nous contenterons d'en aborder des notions élémentaires afin d'appliquer ces méthodes à l'étude de cas pratiques.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1 Pour l'étude de systèmes complexes, ce sujet n'est pas abordé dans ce cours.

Exemple

Pour exemple prenons la commande d'un vérin hydraulique. Le piston du vérin est au repos à la position `1S3`, repérée, par un contact de fin de course.

Lorsqu'on appuie sur le bouton `START` les distributeurs `1M1` et `2M2` sont activés et le piston descend en vitesse rapide jusqu'à la position `1S2`.

A cette position le distributeur `2M1` est activé, le piston termine la descente à vitesse lente jusqu'en position `1S1`.

Le distributeur `1M2` est alors activé et le piston remonte en position `1S3`.

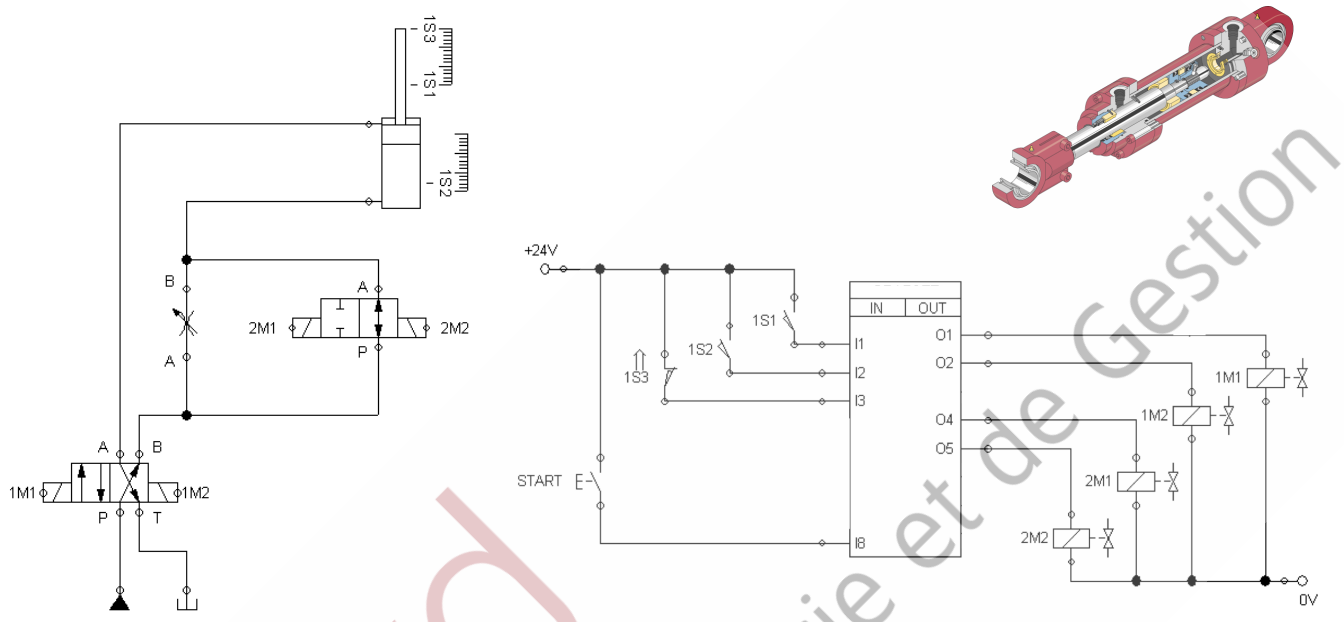


Figure 1.2.2: Exemple de schéma hydraulique et commande piston (Source : Festo)

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.1 Analyse d'un système séquentiel

L'analyse d'un système consiste à étudier en détail son fonctionnement et à déterminer ses propriétés afin de pouvoir le décrire avec précision et méthode.

1.2.1.1 Chronogramme

C'est un graphique qui représente l'évolution des valeurs prises par les variables d'entrée, de sortie et d'état du système en fonction du temps.

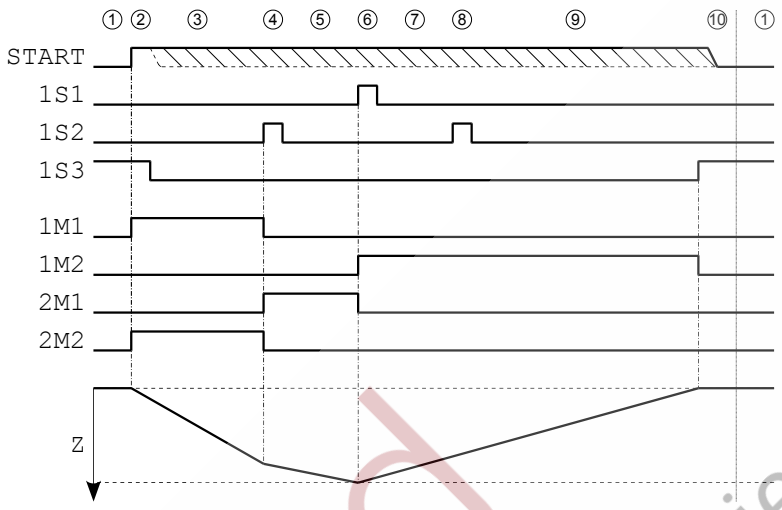


Figure 1.2.3: Exemple de chronogramme pour le vérin

Remarque : aucune condition n'est posée quand à la durée d'action sur le bouton `START`. La seule condition est qu'il ait été relâché à la fin du cycle. Cette particularité est indiquée par la zone hachurée sur le chronogramme.

Le chronogramme représente un certain nombre d'états du système qui correspondent à une configuration particulière des entrées sorties. L'état initial est choisi arbitrairement. Si le nombre de variables est grand, il existe un risque d'oublier certains états et certaines possibilités d'évolution. Ce mode de représentation ne permet pas d'aboutir à une synthèse. Le chronogramme servira plutôt pour représenter une particularité de fonctionnement.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.1.2 Tableau de séquences

Le tableau de séquences représente de manière exhaustive la succession des états par lesquels passe le système.

La première colonne indique le numéro des états, les suivantes représentent la valeur actuelle des différentes entrées du système et les dernières la valeur future des sorties de celui-ci.

	Entrées (état actuel)				**Sorties** (état futur)				**Remarques**
N°	START	1S3	1S2	1S1	2M2	2M1	1M2	1M1	Références schéma
N	I_{18}	I_3	I_2	I_1	Q_5	Q_4	Q_2	Q_1	*Variables*
1	**0**	1	0	0	0	0	0	0	Position de repos
2	**1**	1	0	0	1	0	0	1	Bouton START actionné
3	x	0	0	0	1	0	0	1	Piston descend GV
4	x	0	**1**	0	0	1	0	0	Piston descend PV
5	x	0	0	0	0	1	0	0	
6	x	0	0	**1**	0	0	1	0	Piston remonte
7	x	0	0	0	0	0	1	0	
8	x	0	1	0	0	0	1	0	
9	x	0	0	0	0	0	1	0	
10	x	**1**	0	0	0	0	0	0	Piston en haut

Tableau 2: Exemple de tableau de séquences

Note : Dans le tableau ci-dessus les événements qui entraînent un changement d'état sont marqués en gras.

Le tableau de séquences est une représentation analytique du fonctionnement. Cependant il devient ardu à établir lorsque le système comporte de nombreux états avec des choix de séquences qui dépendent de différentes situations.

heig-vd Haute Ecole d'ingénierie et de Gestion du Canton de Vaud

1.2.1.3 Graphe de fluence

Le graphe de fluence est une représentation graphique des séquences d'états. Il représente tous les **états stables** du système et l'ordre chronologique dans lequel il atteint successivement chacun des états à partir des autres en fonction des variations des variables d'entrée.

Un état stable est un état pour lequel les sorties du système restent inchangées tant que les combinaisons des entrées sont fixes.

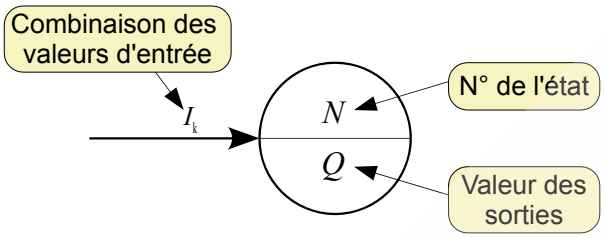


Figure 1.2.4: Symbole d'un nœud de graphe de fluence

Chaque nœud du graphe correspond à un état stable représenté par un cercle dans lequel est indiqué : en haut le numéro de l'état et en dessous la valeur des sorties.

Les branches du graphe indiquent tous les chemins possibles pour passer d'un état stable à l'autre. Ces chemins sont à sens unique, la flèche indique le sens de passage.

Sur chaque branche est indiqué l'état des variables d'entrée permettant d'effectuer la transition d'un état stable à l'autre.

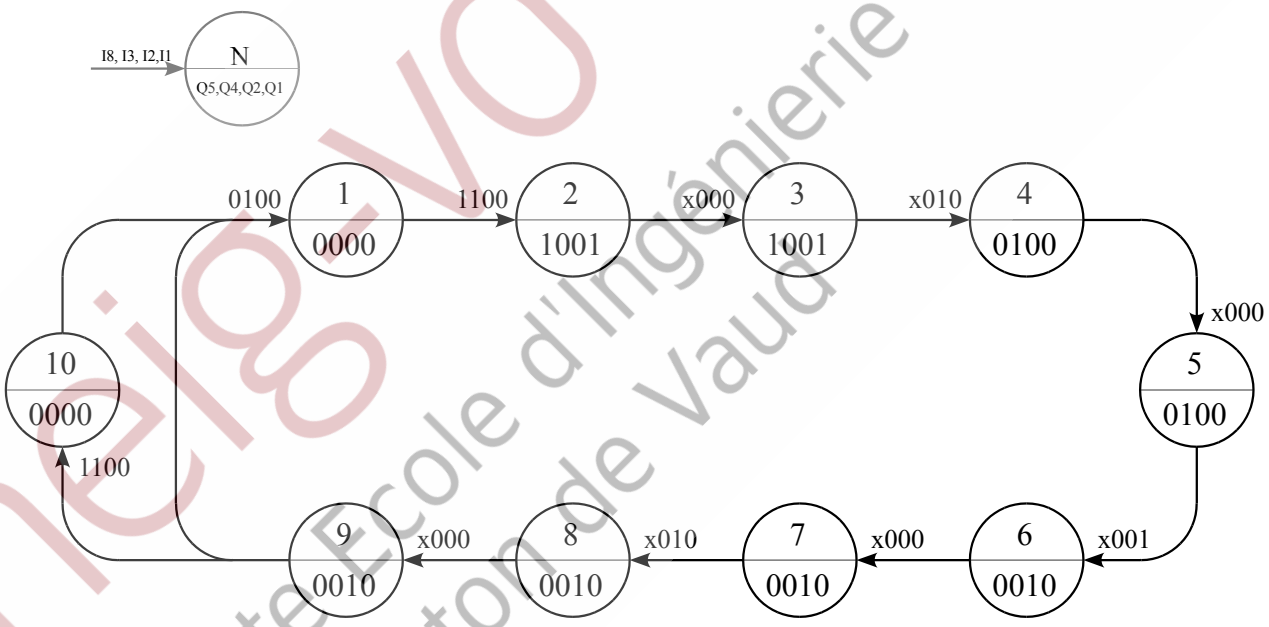


Figure 1.2.5: Exemple de graphe de fluence

Remarque : Dans l'exemple ci-dessus, la variable d'entrée I_8 n'a aucune influence pour le passage aux états 3 à 9. Le fonctionnement du système est indifférent à sa valeur, cette particularité est marquée par la lettre x.

Cette méthode de modélisation est systématique : pour chaque état toutes les variations possibles des entrées sont envisagées. Ce type de graphe montre bien la synthèse de tous les états d'un système, mais si le nombre d'états ou le nombre de variables est important cette représentation devient rapidement « touffue ».

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.1.4 Matrice primitive des états

La matrice primitive des états est une représentation tabulaire des états d'un système, elle peut être remplie à partir du graphe de fluence. La matrice primitive des état est l'outil de base pour la synthèse des systèmes séquentiels.

Les combinaisons des variables d'entrée du système sont représentées par les colonnes de cette matrice. A chaque ligne correspond la transition d'un état vers un autre. Dans la dernière colonne les valeurs des sorties sont indiquées à chaque ligne. La première colonne contient un repère d'identification.

	I_8,I_3,I_2,I_1																
N	0000	0001	0011	0010	0110	0111	0101	0100	1100	1101	1111	1110	1010	1011	1001	1000	Q_5,Q_4,Q_2,Q_1
A								①	2								0000
B	3								②							3'	1001
C	③			4									4'			③'	1001
D	5			④									④'			5'	0100
E	⑤	6													6'	⑤'	0100
F	7	⑥													⑥'	7'	0010
G	⑦			8									8'			⑦'	0010
H	9			⑧									⑧'			9'	0010
I	⑨							1	10							⑨'	0010
J								1	⑩								0000

Tableau 3: Exemple de matrice primitive des états

Remarque : Les états marqués ③'..⑨' correspondent à la séquence parcourue avec le bouton START activé.

Les chiffres entourés ①..⑩ correspondent aux états stables du système. Les autres sont des états transitoires, c'est-à-dire le passage d'un état stable vers l'état stable suivant. Cette transition est provoquée par la variation de l'entrée. L'évolution se fait toujours horizontalement puis verticalement. Le système ne peut évoluer que vers une case comportant un numéro.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.2 Synthèse d'un système séquentiel

La synthèse d'un système séquentiel consiste établir les équations caractéristiques décrivant son comportement. Puis à déterminer la structure de la partie commande, indépendamment de la technologie utilisée, en vue d'obtenir un comportement conforme aux spécifications.

1.2.2.1 Réduction du nombre de variables

La description d'un système physique réel comporte souvent un grand nombre de variables d'entrée-sortie. La plupart de ces variables sont dépendantes, c'est-à-dire qu'elles sont liées par des lois externes imposées par la partie opérative du système. Par conséquent, en raison de ces dépendances le nombre d'états possibles est généralement très inférieur au maximum théorique.

Exemple

Dans le cas de la commande de vérin vue plus haut le système comporte 4 entrées et 4 sorties. Théoriquement le nombre d'états possibles est de $2^4 \cdot 2^4 = 256$, or l'analyse montre qu'il n'y en a que 18 qui sont définis.

En réduisant le nombre de variable le système et ses équations caractéristiques deviennent plus simple à décrire.

La première étape de la réduction porte sur les variables d'entrées. Elle consiste à établir la liste des combinaisons possibles pour les variables d'entrée qui ont une dépendance technologique due à la partie opérative. Puis on effectue un changement de variables $\{I_1, .., I_n\} \Rightarrow \{x_1, .., x_k\}$ qui transforme l'ensemble des variables d'entrée dépendantes $\{I\}$ en un ensemble de variables internes $\{x\}$ qui seront utilisées pour établir les équations caractéristiques. Le codage se fait avec n variables binaires qui permettent de définir 2^n états.

Remarque : lorsqu'on réalise des commandes au moyen d'automates programmables on n'effectue généralement pas cette réduction pour des raisons de lisibilité et de compréhension des programmes.

La réduction du nombre de variable est par contre intéressante si on réalise la commande par câblage électrique ou avec des dispositifs mécaniques, pneumatiques ou hydrauliques. Dans ce cas le coût des composants justifie une étude plus poussée.

Exemple

Dans le cas de la commande du vérin les 3 entrées I_1, I_2 et I_3 sont dépendantes, en effet ces entrées ne peuvent pas être activées simultanément : le vérin ne peut pas être à la fois en position haute et en position basse.

Par contre l'entrée I_8 est indépendante, aucune loi technologique ne la lie aux 3 autres entrées. Cette entrée doit être traitée indépendamment.

Pour notre exemple il n'y a que 4 combinaisons possibles pour les 3 entrées, il faut donc 2 variables internes pour définir ces 4 états ($2^2 = 4$).

Entrées			Internes	
I_3	I_2	I_1	x_1	x_0
0	0	0	0	0
1	0	0	0	1
0	1	0	1	1
0	0	1	1	0

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Les équations liant les entrées et les variables internes peuvent être établies par la méthode des tables de Karnaugh:

x_1		$I_2 I_1$			
		00	01	11	10
I_3	0	0	1	x	1
	1	0	x	x	x

$x_1 = I_2 + I_1$

x_0		$I_2 I_1$			
		00	01	11	10
I_3	0	0	0	x	1
	1	1	x	x	x

$x_0 = I_3 + I_2$

Lorsque le changement de variables est effectué la matrice d'état réduite peut être écrite. Elle comprendra moins de colonnes que la matrice primitive.

	I_8, x_1, x_0								
N	000	001	011	010	110	111	101	100	Q_5,Q_4,Q_2,Q_1
A		①					2		0000
B	3						②	3'	1001
C	③		4			4'		③'	1001
D	5		④			④'		5'	0100
E	⑤			6	6'			⑤'	0100
F	7			⑥	⑥'			7'	0010
G	⑦		8			8'		⑦'	0010
H	9		⑧			⑧'		9'	0010
I	⑨	1					10	⑨'	0010
J		1					⑩		0000

Tableau 4: Exemple de matrice réduite des états

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.3 Machine de Moore

Tout système séquentiel peut être représenté par une machine de Moore. Une machine de Moore est un automate fini[2] pour lequel les valeurs de sortie Q_k ne peuvent dépendre que des variables d'état S_j donc de l'état actuel. Une machine de Moore est strictement synchrone car le changement des sorties ne se fait qu'avec le changement d'état.

Une machine de Moore se compose d'un système logique combinatoire qui détermine l'état futur S_j^+ par combinaisons logiques entre les entrées I_i et les variables d'état S_j du système. Il est suivi d'un dispositif de mémorisation qui enregistre l'état actuel. Un deuxième système combinatoire est parfois nécessaire pour effectuer le décodage des sorties en fonction des variables d'état.

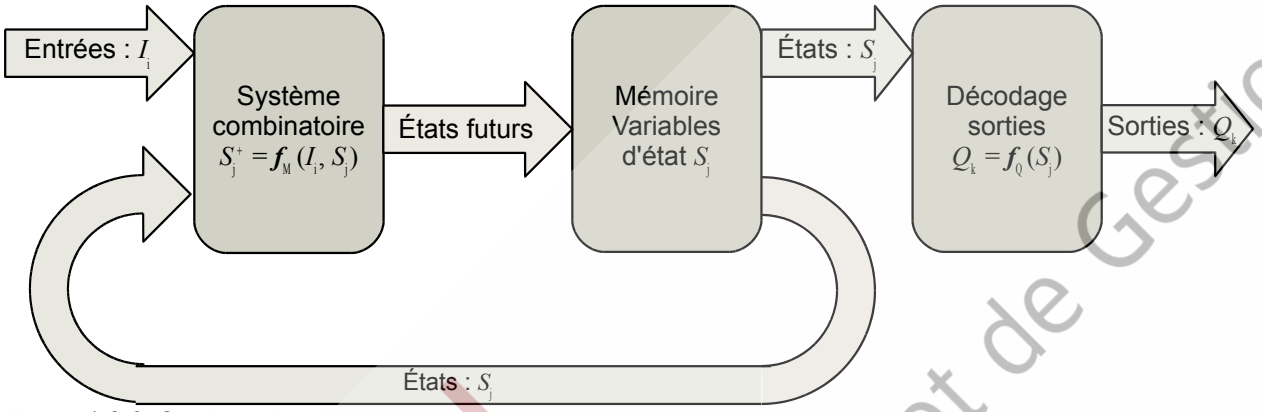


Figure 1.2.6: Schéma de principe d'une machine de Moore

Pour décrire une machine de Moore il faut déterminer :

- Les différents états de la machine, ceci est réalisé lors de l'analyse par l'établissement de la matrice d'état.
- Les variables d'état S_j. Le nombre de variables dépend du nombre d'état possibles de la machine. Les différent états sont représentés par un codage des variables d'état.
- Le système d'équations logiques f_M du système combinatoire d'entrée.
- Le système d'équations logiques f_Q du décodeur de sortie.

La détermination de ces différents éléments peut se faire par la méthode matricielle de Huffman.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2 Automate fini (*FSA : Finite State Automation*) ou machine à états finie (*FSM : Finite State Machine*) est un système séquentiel qui à chaque instant peut se trouver dans un état parmi un nombre fini d'états possibles.

1.2.4 Méthode de Huffman

Afin d'obtenir un système de commande le plus simple possible, il est évident que le nombre de variables d'état doit être minimisé ce qui, par conséquent réduit le nombre d'états stables. Si, lors de l'analyse du système on a déterminé plus d'états stables que nécessaire, il est possible de simplifier la matrice d'état pour éliminer les redondances.

1.2.4.1 Contraction de la matrice d'état

Lors de l'élaboration de la matrice primitive d'état, chaque ligne correspond à un état stable et un seul. Partant d'un état stable, une modification des variables d'entrée fait passer le système dans un état instable situé sur la même ligne. Après un certain temps et le système passe dans un nouvel état stable situé dans une ligne et une colonne différentes de l'état stable d'origine.

Si la modification des variables d'entrée n'entraîne pas de modification des variables de sortie, c'est-à-dire si $S_j^+(t) = S_j(t)$, il est évident que l'on pourra passer directement d'un état stable au suivant sans passer par un état instable intermédiaire. Dans ces conditions, la ligne correspondant à l'état stable suivant peut être fusionnée et les deux états stables peuvent ainsi être regroupés sur la même ligne. Cette simplification ayant pour effet de contracter le nombre de lignes de la matrice, et par conséquent le nombre de variables d'état.

Pour une machine de Moore deux lignes de la matrice ayant les mêmes valeurs de sortie peuvent être fusionnées si :

1. Les états stables à regrouper sont dans des colonnes différentes.
2. Les deux lignes sont identiques aux indifférences (cases vides) près.

Lorsque deux lignes répondent à ces conditions les règles de production de la contraction consistent à remplacer toute paire :

1. D'états indifférent par un état indifférent.
2. D'un état indifférent et l'autre non par l'état non indifférent stable ou instable.
3. D'un même état instable par cet état instable.
4. D'un même état stable et instable par l'état stable.
5. D'un même état stable par cet état stable.

Exemple

Dans l'exemple de la matrice d'état du système du vérin,
les lignes B et C, D et E, F et G, H et I répondent aux critères et peuvent être fusionnées.

	I_8, x_1, x_0								
N	000	001	011	010	110	111	101	100	Q_5,Q_4,Q_2,Q_1
A		①					2		0000
B→C	③		4			4'	②	③'	1001
D→E	⑤		④	6	6'	④'		⑤'	0100
F→G	⑦		8	⑥	⑥'	8'		⑦'	0010
H→I	⑨	1	⑧			⑧'	10	⑨'	0010
J		1					⑩		0000

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.2 Réduction des états équivalents

Il est possible que, au cours de la description du système permettant d'aboutir à la matrice primitive d'états, on ait utilisé un ou plusieurs états pour représenter en réalité un seul état stable.

Deux états stables seront reconnus **équivalents** si et seulement si :

- ils sont atteints par une même combinaison des variables d'entrée,
- ils produisent la même combinaison des variables de sortie,
- pour la même combinaison d'entrée, les deux états ont comme successeur soit le même état soit deux états équivalents.

Deux états stables sont **pseudo-équivalents** si

- ils sont atteints par une même combinaison des variables d'entrée,
- ils produisent la même combinaison des variables de sortie,
- pour la même combinaison d'entrée, un des états possède comme successeur un état stable et l'autre une indifférence.
- pour la même combinaison d'entrée, un des état présente comme successeur un état stable et l'autre le même état transitoire.

Les états équivalents ou pseudo équivalents peuvent être fusionnés en un seul état. Les règles de fusion consistent à remplacer toute paire :

1. D'états indifférent par un état indifférent.
2. D'un état indifférent et l'autre non par l'état non indifférent stable ou instable.
3. D'un même état instable par cet état instable.
4. D'un même état stable et instable par l'état stable.
5. De deux états stables par l'état stable de rang le plus élevé.

Exemple

Dans l'exemple de la matrice d'état du système du vérin,

les états ⑦ et ⑨, respectivement ⑦' et ⑨', des lignes G et I sont pseudo-équivalents et peuvent être fusionnées.

	I_8, x_1, x_0								
N	000	001	011	010	110	111	101	100	Q_5,Q_4,Q_2,Q_1
A		①					2		0000
C	③		4			4'	②	③'	1001
E	⑤		④	6	6'	④'		⑤'	0100
G→I	⑨	1	⑧	⑥	⑥'	⑧'	10	⑨'	0010
J		1					⑩		0000

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.3 Détermination des variables d'état

Après simplification de la matrice des états, puis sa contraction nous avons donc obtenu une matrice minimale, c'est-à-dire contenant un nombre minimal de lignes. Afin de distinguer chaque ligne des autres elle est identifiée de manière unique par une combinaison de variables d'état S_j.

Le codage idéal est un code qui ne change qu'une seule variable pour chaque transition (codage de Grey).

Le nombre de lignes de la matrice contractée détermine le nombre de variables d'état. Avec n variables il est possible de coder 2^n lignes.

Exemple

Dans notre exemple la matrice contractée des états comporte 5 lignes, il faut donc 3 variables d'état ($2^3 = 8 > 5$) pour coder les lignes.

	I_8, x_1, x_0									
N	000	001	011	010	110	111	101	100	Q_5,Q_4,Q_2,Q_1	S_2, S_1, S_0
A		①					2		0000	000
C	③		4			4'	②	③'	1001	101
E	⑤		④	6	6'	④'		⑤'	0100	111
I	⑨	1	⑧	⑥	⑥'	⑧'	10	⑨'	0010	110
J		1					⑩		0000	100

Tableau 5: Exemple de matrice contractée des états

Sur la base de cette matrice on peut représenter le graphe de fluence réduit:

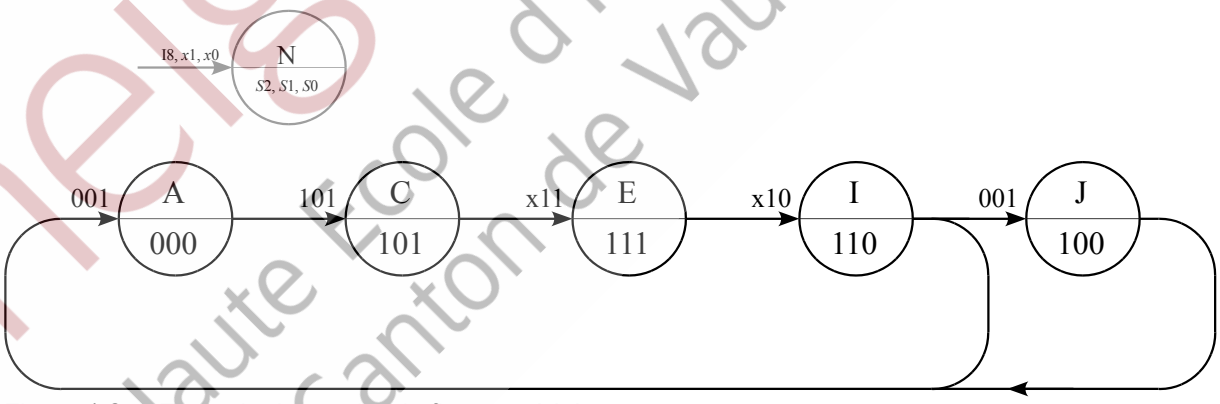


Figure 1.2.7: Exemple de graphe de fluence réduit

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.4 Matrice d'excitation

La matrice d'excitation des variables d'état est similaire à la matrice d'état contractée dans les case de laquelle on représente la valeur future des variables d'état S_j^+ pour chaque combinaison des entrées.

Dans cette matrice, chaque ligne est identifiée par une combinaison de valeur des variables de sortie à l'état présent S_j et les colonnes correspondent aux différente combinaison des valeurs d'entrées I_i.

Pour chaque état stable la valeur future de la variable S_j^+ sera identique à son état actuel S_j.

Pour chaque état instable, ou transitoire, la valeur de l'état futur sera celle de l'état stable qui doit être atteint.

Exemple

Pour l'état stable ③ (ligne C colonne 1) la valeur future des variables d'état est égale à leur valeur actuelle:

$$\begin{cases} S_0^+ = S_0(③) = 1 \\ S_1^+ = S_1(③) = 0 \\ S_2^+ = S_2(③) = 1 \end{cases}$$

Pour l'état transitoire 6 (ligne E colonne 4) la valeur future des variables d'état est égale celle de l'état stable ⑥ à atteindre :

$$\begin{cases} S_0^+ = S_0(⑥) = 0 \\ S_1^+ = S_1(⑥) = 1 \\ S_2^+ = S_2(⑥) = 1 \end{cases}$$

S_2^+, S_1^+, S_0^+	I_8, x_1, x_0								
	000	001	011	010	110	111	101	100	S_2, S_1, S_0
A		000					101		000
C	101		111			111	101	101	101
E	111		111	110	110	111		111	111
I	110	000	110	110	110	110	100	110	110
J		000					100		100

Tableau 6: Exemple de matrice d'excitation

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.5 Mise en équations des variables d'état

A partir de la matrice d'excitation, l'état futur de chaque variables d'état S_j^+ peut être déterminé par une relation logique combinatoire entre les entrées et leurs valeurs actuelles : $S_j^+ = f(I_i, S_j)$.

Dans les cases vides la valeur des variables d'état est indifférent. On utilisera ces indéterminations pour choisir une valeur de la variable permettant de simplifier les équations.

Les équations logiques décrivant le fonctionnement de la machine de Moore sont alors déterminées au moyen de tables de Karnaugh.

Exemple

Dans notre exemple les équations des états futurs S_2^+, S_1^+ et S_0^+ dépendent des 6 variables I_8, x_1, x_0, S_2, S_1 et S_0. et seules 23 valeurs sur les 64 possibles sont déterminées.

Les valeurs choisies pour lever les indéterminations sont indiquées par *(1)* ou *(0)*.

Table de Karnaugh pour S_0^+ :

S_0^+		I_8, x_1, x_0								
		000	001	011	010	110	111	101	100	
S_2, S_1, S_0	000		0			(1)	(1)	1	(1)	A
	001	(1)	(1)	(1)		(1)	(1)	(1)	(1)	
	011	(1)	(1)	(1)		(1)	(1)	(1)	(1)	
	010					(1)	(1)	(1)	(1)	
	110	0	0	0	0	0	0	0	0	I
	111	1	(1)	1	0	0	1	(1)	1	E
	101	1	(1)	1			1	1	1	C
	100		0					0		J

Le premier bloc est défini par l'équation partielle $x_0 \cdot S_0$

Le deuxième bloc est défini par l'équation partielle $\bar{x}_1 \cdot S_0$

Le troisième bloc est défini par l'équation partielle $\bar{I}_8 \cdot \bar{S}_2$

L'équation complète est : $S_0^+ = I_8 \cdot \bar{S}_2 + (\bar{x}_1 + x_0) \cdot S_0$

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Table de Karnaugh pour S_1^+ :

S_1^+		I_8, x_1, x_0								
		000	001	011	010	110	111	101	100	
S_2, S_1, S_0	000		0	(1)	(1)	(1)	(1)	0		A
	001			(1)	(1)	(1)	(1)			
	011	(1)		(1)	(1)	(1)	(1)		(1)	
	010	(1)		(1)	(1)	(1)	(1)		(1)	
	110	1	0	1	1	1	1	0	1	I
	111	1		1	1	1	1		1	E
	101	0		1	(1)	(1)	1	0	0	C
	100		0	(1)	(1)	(1)	(1)	0		J

Le premier bloc est défini par l'équation partielle $\bar{x}_1 \cdot \bar{x}_0 \cdot S_1$

Le deuxième bloc est défini par l'équation partielle x_1

L'équation complète est : $S_1^+ = x_1 + (\bar{x}_1 \cdot \bar{x}_0) \cdot S_1$

Table de Karnaugh pour $S2+$:

S_1^+		I_8, x_1, x_0								
		000	001	011	010	110	111	101	100	
S_2, S_1, S_0	000	(1)	0	(1)	(1)	(1)	(1)	1	(1)	A
	001	(1)		(1)	(1)	(1)	(1)	(1)	(1)	
	011	(1)		(1)	(1)	(1)	(1)	(1)	(1)	
	010	(1)		(1)	(1)	(1)	(1)	(1)	(1)	
	110	1	0	1	1	1	1	1	1	I
	111	1		1	1	1	1		1	E
	101	1		1	(1)	(1)	1	1	1	C
	100	(1)	0	(1)	(1)	(1)	(1)	1	-1	J

Le premier bloc est défini par l'équation partielle $\bar{x}_0$

Le deuxième bloc est défini par l'équation partielle I_8

Le troisième bloc est défini par l'équation partielle x_1

L'équation complète est : $S_2^+ = I_8 + x_1 + \bar{x}_0$

Remarque : on constate que cette équation ne dépend pas des variables d'état S_j.
Elle est purement combinatoire et ne dépend pas de l'historique du système.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.6 Équations des sorties

Pour une machine de Moore les variables de sorties Q_k sont définies par une combinaison des variables d'état S_j.

Ces équations dépendent du choix qui a été fait pour définir les variables d'état et on peut les établir à partir de la matrice contractée.

Exemple

De la matrice contractée on extrait la table de vérité des sortie Q_k en fonction des variables d'état S_j.

États			Sorties			
S_2	S_1	S_0	Q_5	Q_4	Q_2	Q_1
0	0	0	0	0	0	0
1	0	1	1	0	0	1
1	1	1	0	1	0	0
1	1	0	0	0	1	0
1	0	0	0	0	0	0

Les équations liant les sorties et les variables d'état peuvent être établies par la méthode des tables de Karnaugh:

Q_5		$S_1 S_0$			
		00	01	11	10
S_2	0	0	x	x	x
	1	0	1	0	0

$Q_5 = \bar{S}_1 \cdot S_0$

Q_4		$S_1 S_0$			
		00	01	11	10
S_2	0	0	x	x	x
	1	0	0	1	0

$Q_4 = S_1 \cdot S_0$

Q_2		$S_1 S_0$			
		00	01	11	10
S_2	0	0	x	x	x
	1	0	0	0	1

$Q_2 = S_1 \cdot \bar{S}_0$

Q_1		$S_1 S_0$			
		00	01	11	10
S_2	0	0	x	x	x
	1	0	1	0	0

$Q_1 = \bar{S}_1 \cdot S_0 = Q_5$

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

1.2.4.7 Représentation schématique

Finalement, le schéma logique de la réalisation technologique de la machine de Moore peut être établi sur la base des équations précédemment établies :

- Codage des entrées en variables internes $x_n = f(I_i)$,
- Variables d'état $S_j^+ = f(I_i, x_n, S_j)$,
- Décodage des sorties $Q_k = f(S_j)$.

La réalisation du système de commande peut être réalisée sur la base de ce schéma logique avec éléments combinatoires. Selon la technologie choisie ces composants seront mécaniques (cames, distributeurs hydrauliques ou pneumatiques), électromécaniques (relais), électroniques (circuits intégrés) ou informatiques par logique programmée.

Exemple

Pour le système de vérin les équations sont :

Entrées :

$$\begin{cases} x_0 = I_3 + I_2 \\ x_1 = I_2 + I_1 \end{cases}$$

États

$$\begin{cases} S_0^+ = I_8 \cdot \bar{S}_2 + (\bar{x}_1 + x_0) \cdot S_0 \\ S_1^+ = x_1 + (\bar{x}_1 \cdot \bar{x}_0) \cdot S_1 \\ S_2^+ = I_8 + x_1 + \bar{x}_0 \end{cases}$$

Sorties

$$\begin{cases} Q_1 = \bar{S}_1 \cdot S_0 \\ Q_2 = S_1 \cdot \bar{S}_0 \\ Q_4 = S_1 \cdot S_0 \\ Q_5 = \bar{S}_1 \cdot S_0 \end{cases}$$

Et le schéma logique :

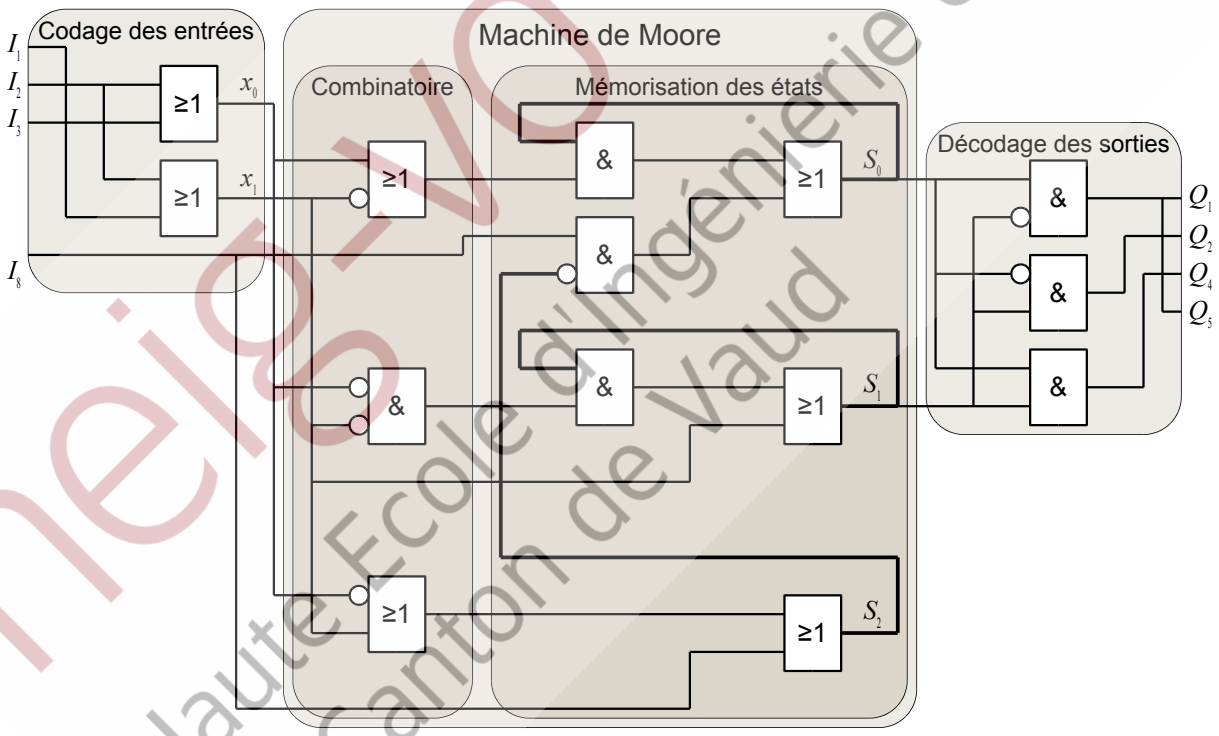


Figure 1.2.8: Exemple de schéma logique d'une commande séquentielle

On retrouve sur ce schéma les différents blocs de la machine séquentielle réalisés uniquement avec des opérateurs de logique combinatoire:

- Codage des entrées en logique combinatoire
- Machine de Moore avec le bloc combinatoire et la mémorisation des états. Les contre-réaction (*feedback*) des variables d'état sont marquées d'une liaison en gras.
- Décodage des sorties.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Chapitre 2

Le GRAFCET

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

2 Le GRAFCET

Le **GRAFCET** (GRAphe de Commande Étape - Transition) est la méthode de représentation et d'analyse des systèmes séquentiels qui s'est imposée dans le domaine industriel international. Créé en 1975 par l'AFCET (Association Française pour la Cybernétique Économique et Technique), un groupe d'industriels et d'universitaires français, cette méthode s'est rapidement imposée au niveau mondial par son applicabilité et sa simplicité.

Le GRAFCET est un langage de spécification pour les diagrammes fonctionnels en séquence. C'est un langage graphique, donc universellement compréhensible, sa syntaxe, ses règles et symboles sont spécifiés par la norme CEI 60848. Cet outil permet l'analyse du fonctionnement et la modélisation de commandes séquentielles pour systèmes automatisés industriels.

Remarque: Le GRAFCET est un langage général de description de séquences; il ne doit pas être confondu avec le langage de programmation *SFC (Sequential Flow Chart)* pour automates programmables défini par la norme IEC 61 131-3.

Le langage SFC est une implémentation particulière du GRAFCET pour réaliser des programmes d'automatismes. Son utilisation est recommandée mais il existe d'autres méthodes et d'autres langages de programmation.

2.1 Système automatisé de production

Un système automatisé de production est une machine qui crée de la valeur ajoutée. Il comporte une partie opérative et une partie commande qui sont étroitement liées et fonctionnent conjointement.

La partie opérative comprend le procédé qui agit sur la matière, la chaine d'énergie qui alimente les actionneurs et la chaine d'information qui transmet les données sur l'état du système et lui envoie des ordres.

La partie commande correspond à la chaîne d'information, elle traite les informations de différentes formes provenant du procédé et de sources externes. Elle génère et envoie les ordres en fonction des consignes reçues pour que le système fonctionne selon les besoins de l'exploitation. Ce traitement comprend de la logique combinatoire, des séquences, des opérations mathématiques et de la régulation.

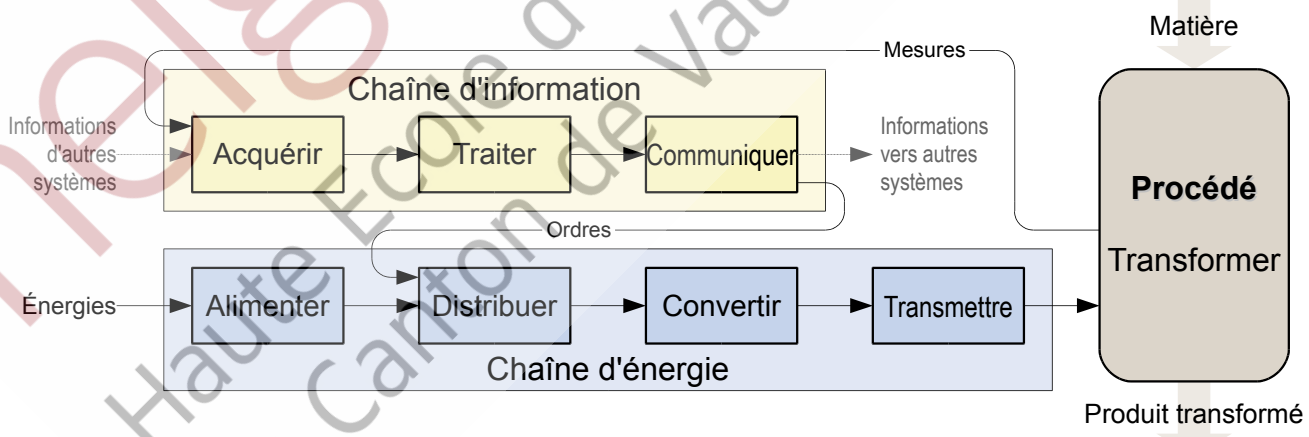


Figure 2.1.1: Système automatisé de production

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

L'évolution du comportement du procédé est contrôlé par la partie séquentielle de la commande en fonction des variables d'entrée, de sortie ou d'état. Ces variables sont généralement de type booléen,toutefois il est possible d'exploiter des variables d'autres types (numériques, flottant, temporelles) par l'évaluation de prédicats.

Le GRAFCET est utilisé pour décrire le comportement de la partie séquentielle.

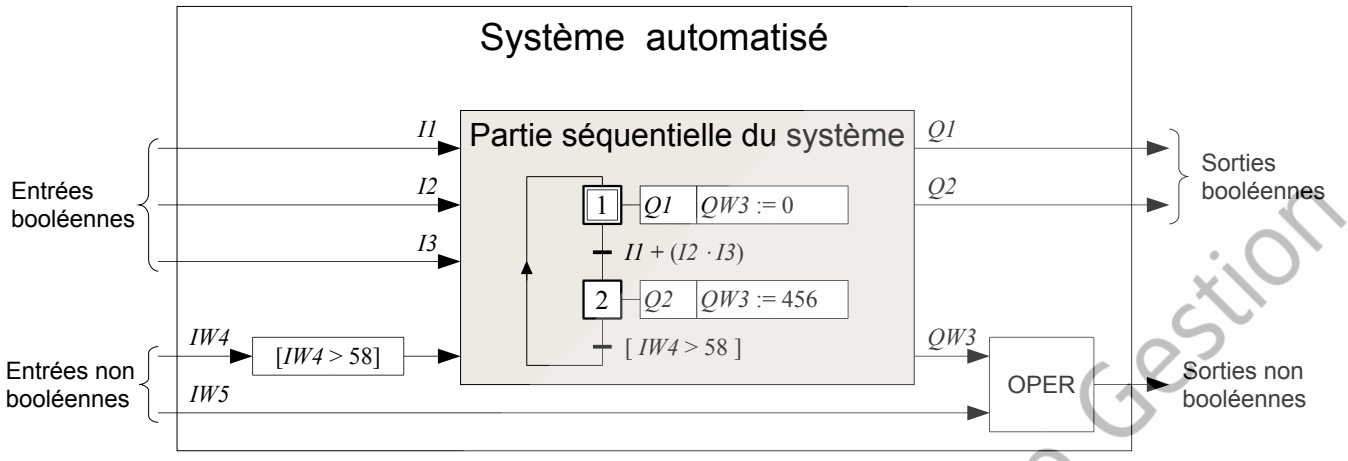


Figure 2.1.2: Partie séquentielle d'un système automatisé

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2 Le langage GRAFCET

Le langage GRAFCET permet de spécifier le comportement attendu d'un système de production automatisé.

C'est un graphe structuré, appelé grafcet[3], qui associé à des expressions mathématiques représente de manière synthétique les **séquences** d'opérations. Il comporte deux types de d'éléments graphiques : les **étapes** et les **transitions**.

Les étapes et transitions sont reliées par des liaisons orientées, c'est-à-dire que le parcours du graphe se fait dans un seul sens. Usuellement, si le sens de parcours n'est pas indiqué par une flèche, le cheminement se fait de haut en bas et de gauche à droite.

2.2.1 Les étapes

Une **étape** représente un **état stable** du système étudié. Un système de commande est dans un état stable tant que ses sorties ne changent pas.

Une étape est représentée par un carré encadrant un numéro. Les étapes sont numérotées arbitrairement, la seule condition est que les numéros soient uniques.

Pour une situation donnée de l'évolution du grafcet, une étape peut être active ou inactive. L'activité d'une étape est marquée par un point ou par le remplissage du carré.

L'état d'activité d'une étape est donné par une variable booléenne Xn, n est le numéro de l'étape. Cette variable est vrai quand l'étape est active, faux dans le cas contraire.

Remarque : Dans le langage IEC 61131-3 SFC, la variable binaire associée à l'état de l'étape `<Step>` se nomme `<Step>.x` . Elle vaut `True` si l'étape est active et `False` dans le cas contraire.

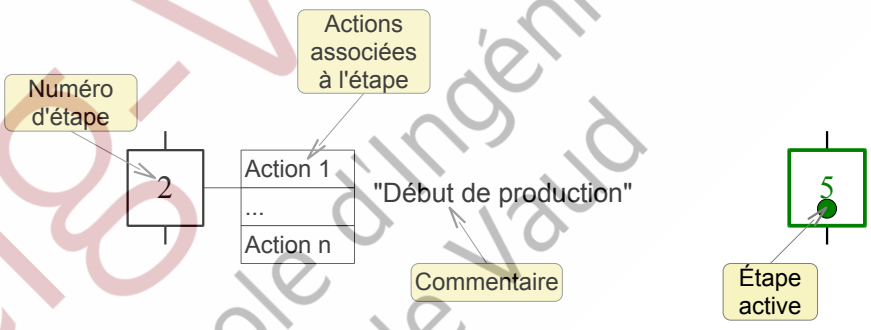


Figure 2.2.1: Étapes GRAFCET

Les **actions**, du système de commande sur la partie opérative, sont associées aux étapes. Elles sont indiquées dans un rectangle à droite du carré symbolisant l'étape. Le libellé des actions à réaliser peut être spécifiées littéralement ou symboliquement.

Une action est effectuée tant qu'une étape est active, elle n'est plus effectuée quand l'étape est désactivée.

Un commentaire, placé entre guillemets "...", peut être ajouté pour préciser le fonctionnement d'une étape.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3 En minuscule c'est le graphe; en majuscules c'est le langage GRAFCET

2.2.1.1 Étapes particulières

Les étapes actives à l'instant suivant la mise en service de la commande (initialisation) sont appelées **étapes initiales** et sont présentées par un double carré.

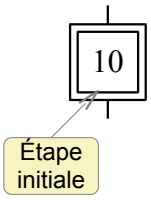


Figure 2.2.2: Étape initiale

Les **étapes sources** sont des étapes qui ne sont pas précédées d'une transition. Pour qu'une étape source puisse devenir active il faut soit:

1) Qu'elle ai été définie comme étape initiale,
2) Qu'elle soit forcée active par un autre grafcet,
3) Qu'elle soit une étape activée d'une encapsulation.

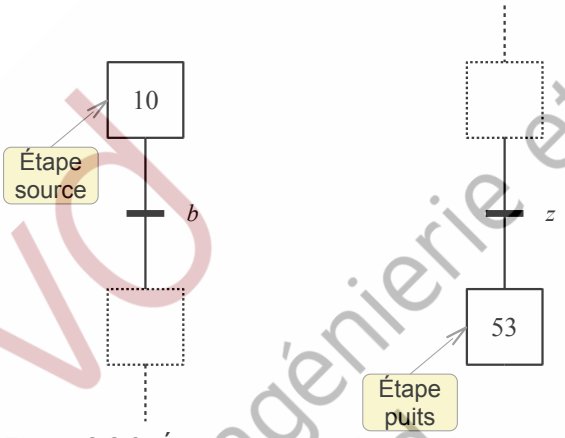


Figure 2.2.3: Étape source et étape puits

Les **étapes puits** sont des étapes qui ne sont pas suivies de transition. Ces étapes terminent des séquences dans une situation qui ne peut plus évoluer, par exemple arrêt faisant suite à un dysfonctionnement.

Une étape puits reste active tant:

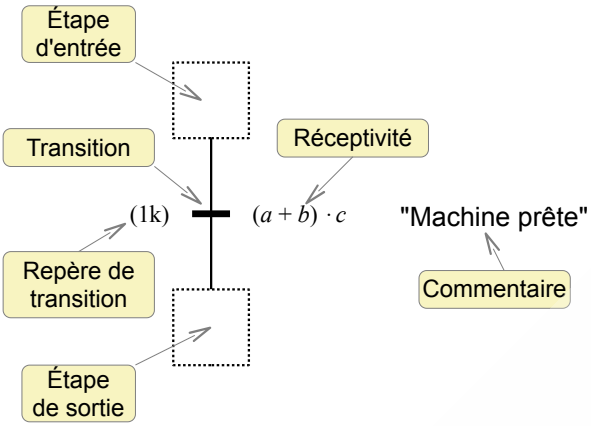
1) Qu'elle n'a pas été forcée inactive par un autre grafcet,
2) Que son étape encapsulante est active.

heig-vd Haute Ecole d'ingénierie et de Gestion du Canton de Vaud

2.2.2 Les transitions

Une **transition** modélise un **changement d'état** du système. Elle indique une unique possibilité d'évolution entre deux ou plusieurs étapes.

Elle est représentée par un trait horizontal court sur la liaison orientée.



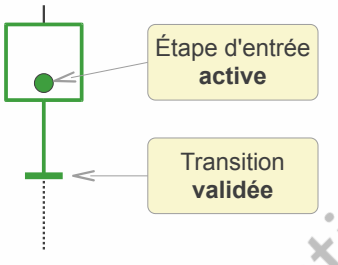


Figure 2.2.4: Transitions GRAFCET

Une transition relie l'ensemble des **étapes d'entrée**, c'est-à-dire qui précèdent la transition, à l'ensemble des **étapes de sorties**, qui suivent la transition.

Une transition précise les conditions dans lesquelles les étapes de sortie deviendront actives. Ces conditions de franchissement d'une transition sont appelées **réceptivité**. Elles sont exprimées d'une manière littérale, symbolique ou à l'aide d'une expression booléenne.

Lorsque toutes les étapes d'entrée sont actives, la transition est **validée**, elle peut alors être franchie dès que la réceptivité est vraie.

2.2.2.1 Repère de liaison

Lorsqu'une liaison orientée doit être interrompue (par exemple dans des dessins complexes ou dans le cas de représentation sur plusieurs pages) le repère de l'étape de destination ainsi que le repère de la page à laquelle elle apparaît doivent être indiqués.

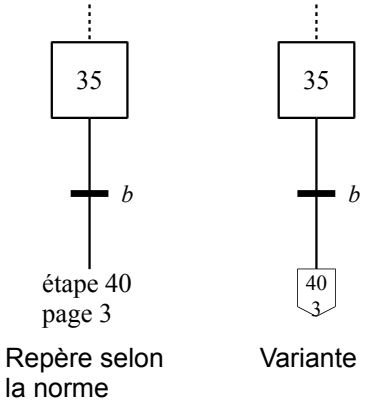


Figure 2.2.5: Repère de liaison

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.3 Règles du GRAFCET

Un système séquentiel évolue dans le temps, il passe d'un état stable à un nouvel état stable. Ce changement d'état est traduit dans le GRAFCET par une évolution de la situation qui doit respecter six règles fondamentales.

2.2.3.1 Syntaxe

Règle 1 : Succession

L'alternance étape-transition et transition-étape doit toujours être respectée quelle que soit la séquence parcourue.

Deux étapes ou deux transitions ne peuvent pas être directement reliées par une liaison orientée.

2.2.3.2 Évolution

Règle 2 : Situation initiale.

La situation initiale précise quelles étapes doivent être activées lors de la mise en service du système de commande de la machine. Cette situation est choisie par le concepteur. Elle est caractérisée par l'ensemble des étapes initiales.

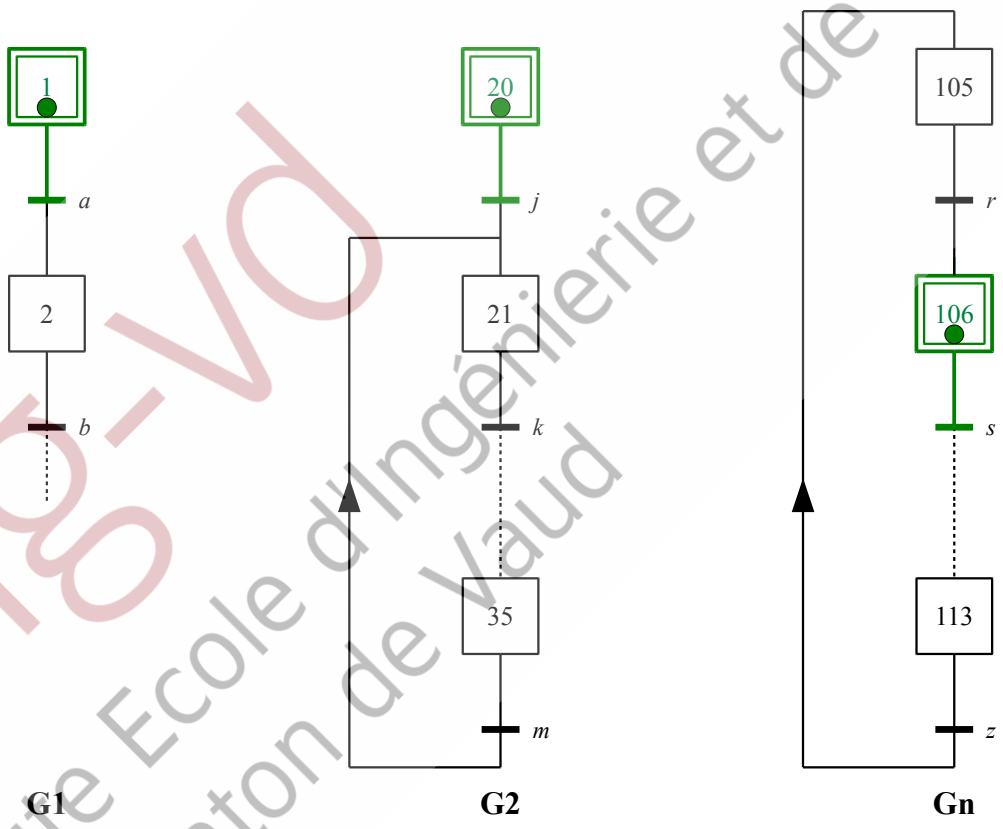


Figure 2.2.6: Situation initiale

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Règle 3 : Franchissement d'une transition.

Une transition est franchissable si les deux conditions suivantes sont satisfaites :

- toutes les étapes d'entrée de la transition sont actives, on dit alors que la transition est validée,
- la réceptivité associée à cette transition est vraie.

Une transition franchissable est obligatoirement franchie.

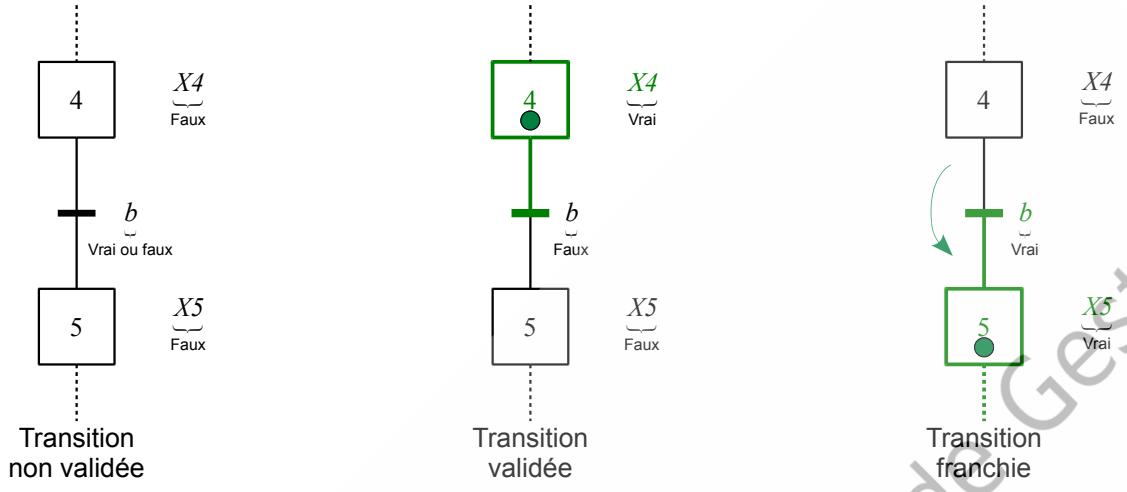


Figure 2.2.7: Franchissement d'une transition

Règle 4 : Évolution des étapes actives.

Le franchissement d'une transition entraîne simultanément l'activation de toutes les étapes de sortie de la transition et la désactivation de toutes les étapes d'entrée.

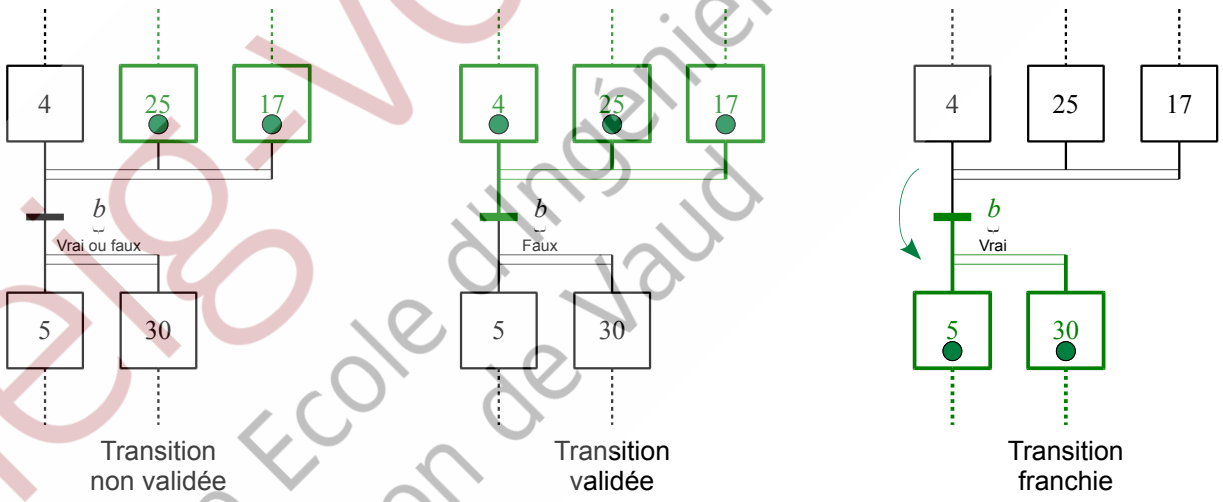


Figure 2.2.8: Évolution des étapes actives

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Règle 5 : Évolutions simultanées.

Plusieurs transitions simultanément franchissables sont simultanément franchies.
L'évolution entre deux situations du grafcet implique qu'aucune situation intermédiaire n'est possible. Le graphe passe instantanément d'une représentation d'une situation à une autre.

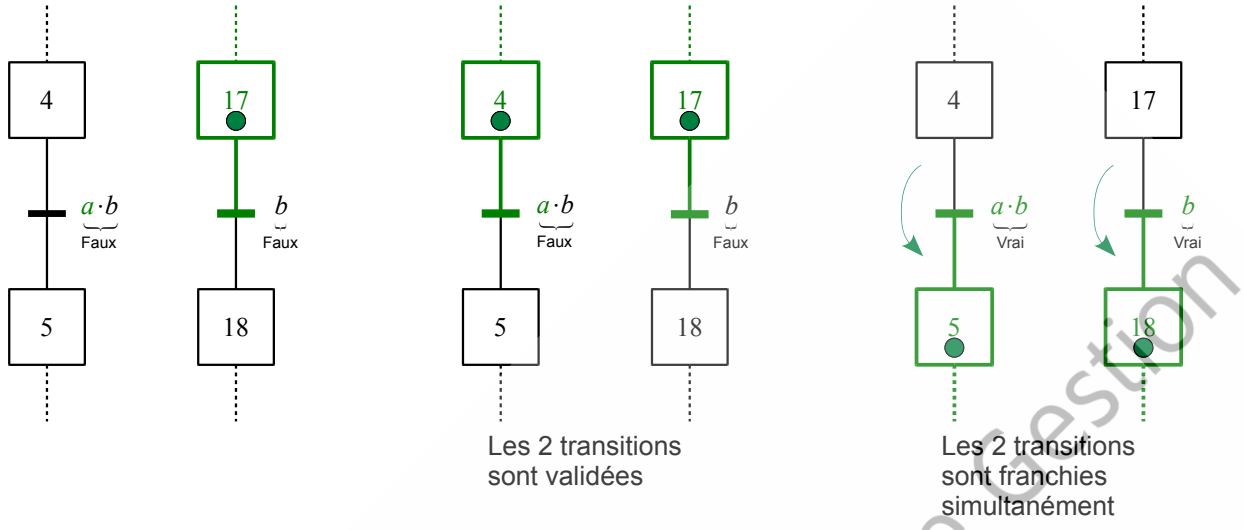


Figure 2.2.9: Franchissement simultané

Règle 6 : Activation et désactivation simultanées.

Si au cours du franchissement d'une ou de plusieurs transitions simultanément, une même étape doit être désactivée et activée, alors elle reste active.
Si une même étape participe à la représentation de la situation précédente et à celle de la situation suivante, elle ne peut que rester active.

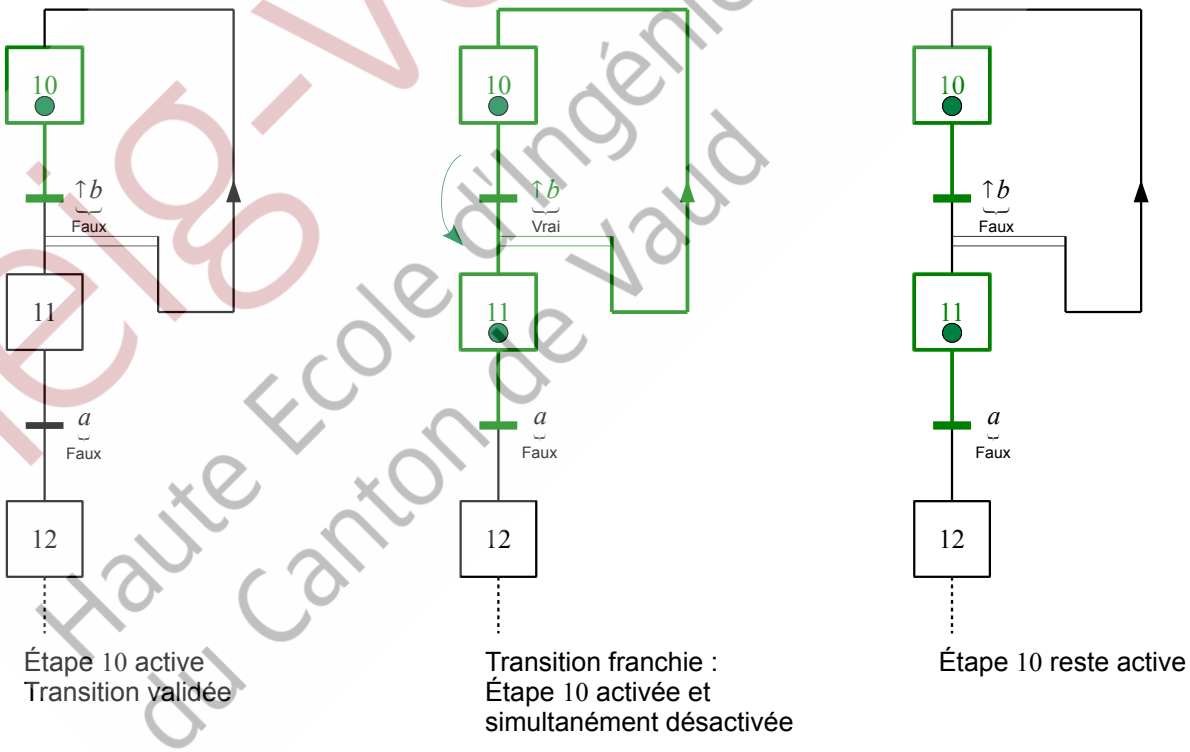


Figure 2.2.10: Activation et désactivation simultanées

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.4 Structure du GRAFCET

Un grafcet est composé de **séquences**, c'est-à dire une succession d'étapes et de transitions.

Une séquence est active lorsqu'au moins une de ses étapes est active.

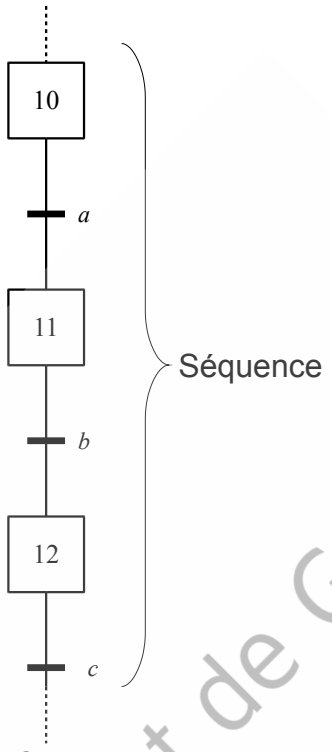


Figure 2.2.11: Séquence, succession d'étapes-transitions

2.2.4.1 Séquence linéaire

Une séquence est dite **linéaire** quand il n'existe qu'un seul et unique chemin pour la parcourir.

Chaque étape comporte au plus une transition en amont et en aval.

Exemple

Le vérin hydraulique vu au chapitre précédent.

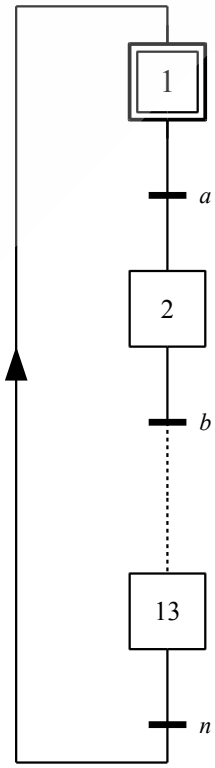


Figure 2.2.12: GRAFCET linéaire

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.4.2 Sélection de séquences

La **sélection de séquences** permet de faire un choix de séquence à partir d'une étape. Elle est composée d'une **divergence en OU**, symbolisée par un trait de liaison simple, qui relie plusieurs transitions comportant les conditions de choix. Chaque transition est suivie d'une séquence. Une **convergence en OU** termine la sélection.

Lorsque l'étape en amont de la sélection est active, toutes les transitions en aval de la divergence sont validées simultanément.

Remarque: Bien que la norme ne l'impose pas, les conditions de franchissement des transitions composant la divergence doivent être ou seront rendues **exclusives**, la sélection des évolutions possible doit être unique. Si cette exclusivité n'est pas assurée, des aléas (évolutions non souhaitées) peuvent se produire lors de l'interprétation du grafcet. Par exemple plusieurs étapes peuvent devenir simultanément actives dans un même grafcet.

Dans ce type de structure une seule branche du graphe peut être parcourue, c'est-à-dire qu'une seule étape peut être activée. Une des branches peut ne pas contenir d'étape, ce qui permet de ne pas exécuter une partie du graphe sous certaines conditions.

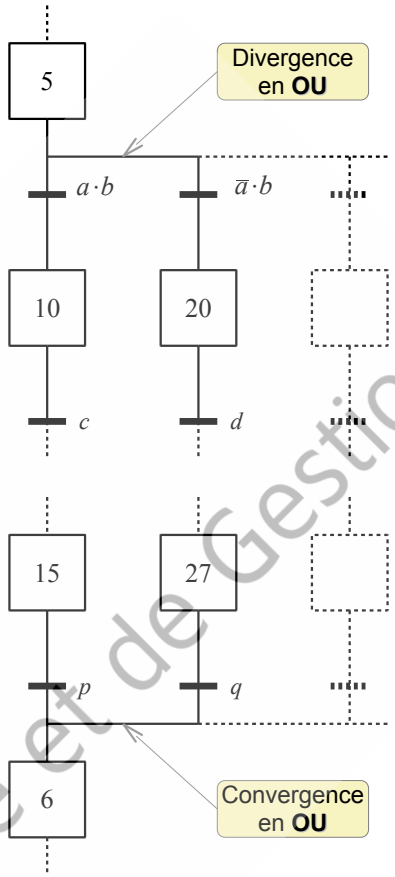


Figure 2.2.13: GRAFCET avec conditionnelle

2.2.4.2.1 Saut d'étapes

Le **saut d'étapes** est un cas particulier de la sélection de séquences. Il permet soit d'exécuter une séquence, soit de sauter des étapes.

Une des branches de la sélection ne contient donc aucune étape.

Exemple

Exécution d'une séquence de dégivrage de pompe à chaleur en cas de danger de gel.

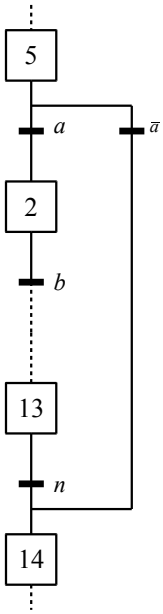


Figure 2.2.14: Saut d'étapes

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.4.2.2 Reprise de séquence

La **reprise de séquence** est un autre cas particulier.

Cette structure est utilisée lorsqu'il faut répéter la même séquence jusqu'à ce qu'une condition soit satisfaite.

Exemple

Reprise d'une séquence d'usinage sur une rectifieuse jusqu'à ce que la pièce soit dans les tolérances.

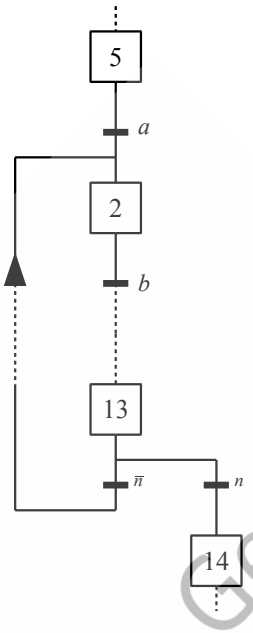


Figure 2.2.15: Reprise de séquence

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.4.3 Séquences parallèles

Le **parallélisme** est l'exécution simultanée de plusieurs séquences, il est réalisé par une transition qui possède plusieurs étapes de sortie. L'exécution en parallèle de plusieurs séquences débute par une **divergence en ET**, représentée par un trait de liaison double. Les séquences parallèles débutent simultanément mais l'évolution dans chaque branche est indépendante.

Les séquences parallèles se terminent par une **synchronisation**. Celle-ci consiste en une **convergence en ET**, symbolisée par un trait de liaison double, qui est une transition possédant plusieurs étapes d'entrée.

La transition qui suit la synchronisation n'est validée que lorsque toutes les étapes d'entrée sont actives.

Exemple

L'exécution de séquences simultanées est principalement utilisé sur des machines de transfert ou constituées d'un ensemble de sous-machines travaillant de manière indépendante.

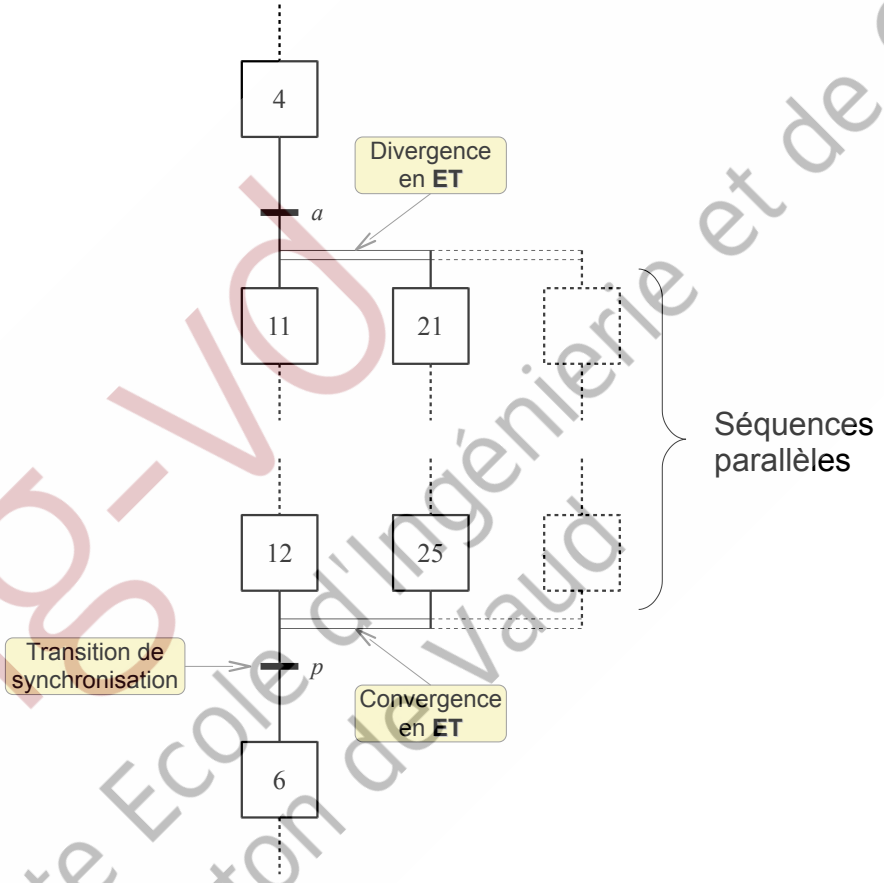
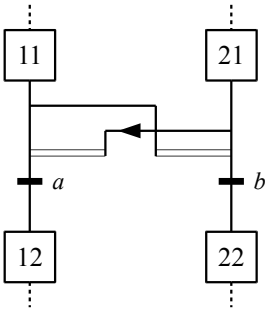


Figure 2.2.16: GRAFCET avec parallélisme

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

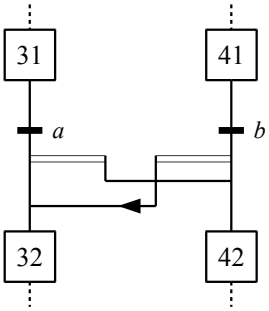
2.2.4.4 Structures particulières

L'évolution des structures combinées particulières suivantes sont présentées sous forme d'exemples.



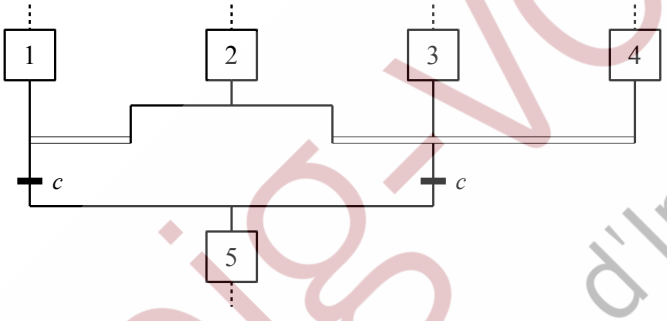
Exemple 1

Quelle doit être la situation de départ et quelles conditions doivent être remplies pour que le grafcet ci-contre puisse évoluer vers la situation où seule l'étape 22 est active ?



Exemple 2

Comment fonctionne le grafcet ci-contre ?



Exemple 3

Comment fonctionne le grafcet ci-contre ?

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.5 Les actions

Une ou plusieurs actions élémentaires ou complexes peuvent être associées à une étape. Les actions décrivent quelles opérations doivent être exécutées, éventuellement sous condition, lorsque l'étape à laquelle elles sont associée est active.

Une action sert à émettre un ordre de la commande vers la partie opérative, elle agit généralement sur une sortie de la commande. Elle peut être décrite de différentes manières:

- Dans un cahier des charges général elle sont expliquées par un texte qui doit contenir un verbe à l'infinitif.
  Exemples : ouvrir la porte, enclencher pompe, fermer vanne de purge, positionner outil, etc.
- Lorsque la conception de la machine est plus avancée, les actions spécifiées par le descriptif de fonctionnement peuvent être notées sous forme symbolique. La signification des symboles est alors définis dans un tableau, sur des plans ou des schémas.
  Exemples :
  M105+ {Enclencher moteur 105, sens direct},
  V47- {Fermer vanne 47},
  E14 {Enclencher agrégat 14}, etc.
- Pour la documentation d'un programme d'automatisme les actions sont décrites en utilisant les symboles des variables informatiques.
  Exemples : `%Q10; SET %Q12; BIT01 := TRUE; CMPT := CMPT+1;` etc.

Ces différents niveaux de descriptions interviennent à des phases distinctes de l'avancement d'un développement. Pour assurer une bonne compréhension du descriptif on évitera de mélanger les notations.

Lorsque plusieurs actions sont associées à une étape elles sont indiquées dans des rectangles situés à droite du carré symbolisant l'étape. Chaque rectangle décrit une seule et unique action. Ces rectangle peuvent être disposés indifféremment l'un sur l'autre ou l'un à côté de l'autre. L'ordre de disposition n'a aucune importance toutes les actions étant exécutées simultanément.

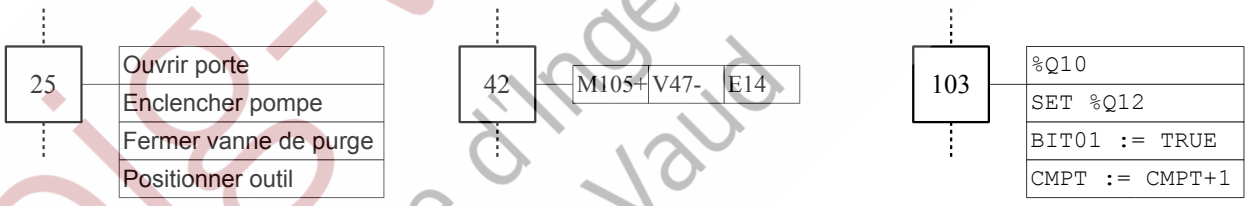


Figure 2.2.17: Différentes possibilités de description des actions

2.2.5.1 Action vide

Une action vide signifie qu'aucune action n'est effectuée lorsque l'étape est active.

Une action vide est utilisées dans certaines conditions, par exemple attente qu'un événement se produise ou synchronisation de deux graphes.

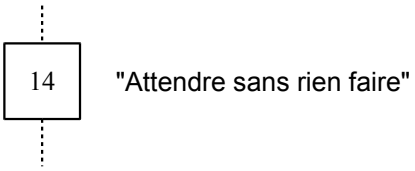


Figure 2.2.18: Action vide

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.5.2 Action continue

Une action **continue** est exécutée tant que l'étape à laquelle elle est associée est active.

Avant que l'étape ne soit activée l'action n'est pas exécutée. Après la désactivation de l'étape l'action n'est plus exécutée.

La mise à l'état vrai du symbole ou de la variable noté dans le rectangle d'action est implicite.

Remarque
La description d'un processus séquentiel se fait principalement sur ce type d'actions, les autres types ne devraient être utilisés que pour traiter des cas particuliers.

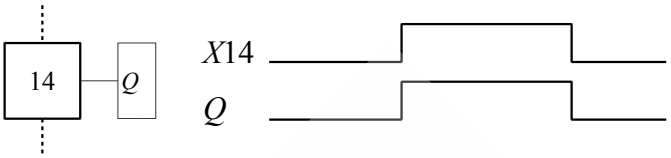


Figure 2.2.19: Action continue

La variable Q est à l'état vrai tant que l'étape 14 est active.

2.2.5.3 Action mémorisée

Une action **mémorisée** est débutée dans une étape et se termine par une autre étape.

Contrairement à l'action continue le début et la fin de l'action mémorisée doit être **explicité** dans le libellé de l'action.

Une action mémorisée est généralement associée à une variable d'état à laquelle on attribue une valeur déterminée. Pour cela on utilise l'opération d'affectation notée « := » .

L'opération d'affectation peut être utilisée pour des variables booléennes, mais elle peut également être utilisée pour effectuer des opérations mathématiques, par exemple incrémenter ou décrémenter un compteur. Ceci est également une action mémorisée.

La valeur attribuée à la variable d'état n'est pas modifiée un fois que l'étape est désactivée.

Exemple

Incrémentation d'un compteur : `CMPT := CMPT + 1;`

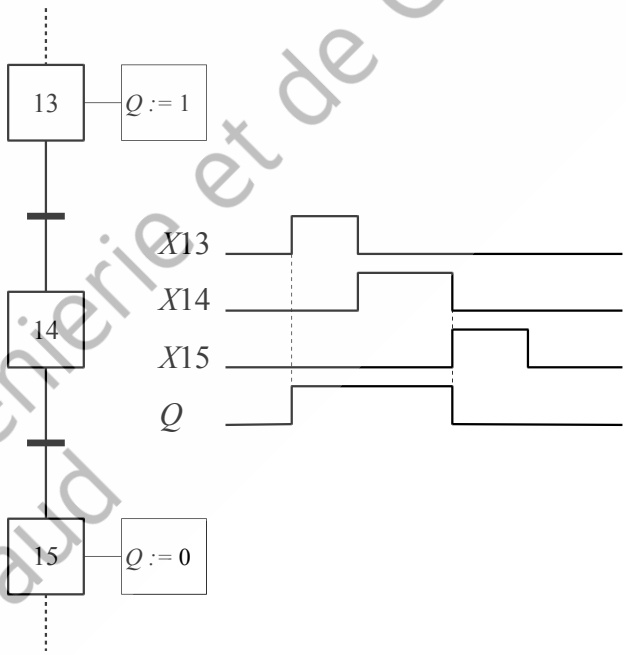


Figure 2.2.20: Action mémorisée

La variable Q est mise à l'état vrai quand l'étape 13 devient active. Elle est mise à l'état faux quand l'étape 15 devient active.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.5.4 Action conditionnelle

L'exécution d'une action continue peut dépendre d'une condition d'assignation. Cette condition, écrite au dessus du rectangle d'action, s'exprime sous forme d'un texte ou d'une proposition logique qui peut être vraie ou fausse.

Une action conditionnelle est réalisée quand l'étape est active ET que la condition d'assignation est vraie.

Cette condition ne doit pas comporter d'éléments dynamiques tels que détection de front.

Les actions mémorisées ne peuvent pas être conditionnées.

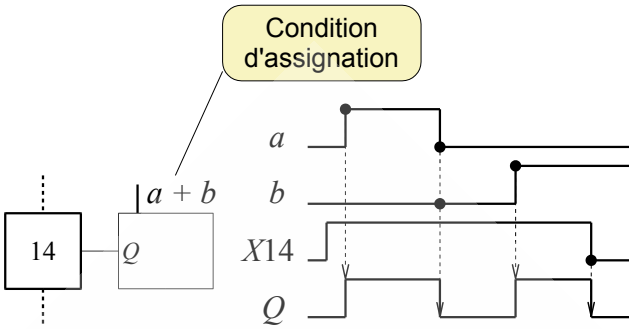


Figure 2.2.21: Action conditionnelle

Exemple

Dans le graphe ci-dessus, l'état de la variable de sortie Q est décrit par l'expression logique:

$$Q = X14 \cdot (a + b)$$

2.2.5.5 Action dépendantes du temps

Les actions continues peuvent dépendre du temps. Elles peuvent être retardées par rapport à l'apparition d'une condition ou prolongées après sa disparition.

Cette dépendance au temps se note :

<temps retard> / <condition> [/ <temps maintien>]

L'étape doit être active et la condition rester vraie pendant un temps supérieur au retard pour que l'assignation puisse se faire.

La spécification du temps de maintien est optionnelle. Si elle est omise, l'assignation se termine lorsque la condition redevient fausse ou lorsque l'étape se termine.

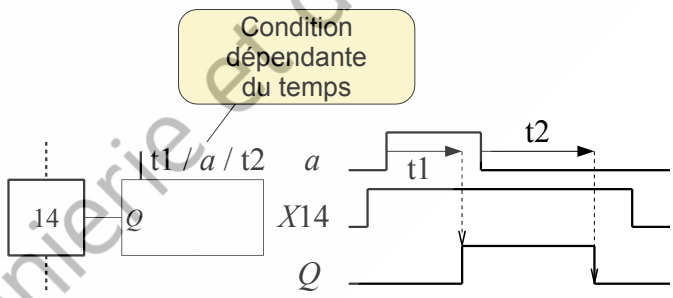


Figure 2.2.22: Action dépendante du temps

Exemple

Dans le graphe ci-dessus (2.2.22), la variable de sortie Q devient vrai après un délai t1 lorsque la condition a est devenue vrai et elle reste vrai un temps t2 après que a soit devenue faux.

Une action peut être limitée dans le temps. L'assignation de la variable de sortie se fait tant que la temporisation n'est pas écoulée.

Remarque : La condition peut être l'état d'activité de l'étape, on utilise dans ce cas sa variable associée Xn.

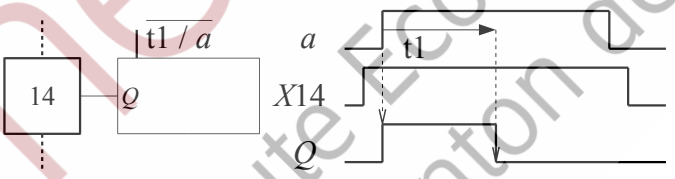


Figure 2.2.23: Action limitée dans le temps

heig-vd Haute Ecole d'ingénierie et de Gestion du Canton de Vaud

2.2.5.6 Action à l'activation et à la désactivation

Une action mémorisée peut être exécutée à l'instant d'activation ou de désactivation de l'étape.

Les actions à l'activation et à la désactivation ne sont exécutées qu'**une seule fois**.

Une action à l'activation est notée par une flèche montante en haut à gauche du rectangle d'action.

Une action à la désactivation est notée par une flèche descendante en bas à gauche.

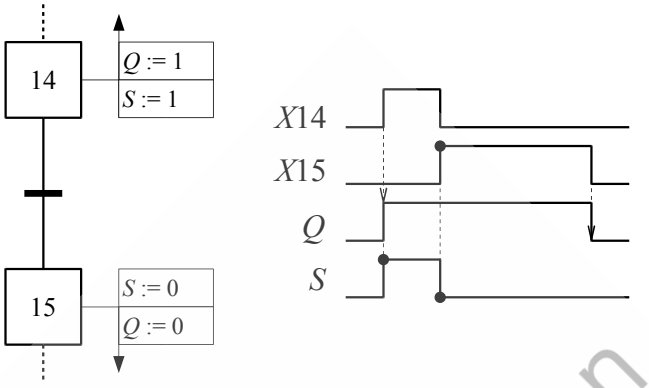


Figure 2.2.24: Action à l'activation / désactivation

Exemple

La variable Q est mise à vrai à l'activation de l'étape 14 et mise à faux à la désactivation de l'étape 15.

La variable S est mise à vrai quand l'étape 14 est active puis mise à faux quand l'étape 15 est active.

2.2.5.7 Action au franchissement

Pour traiter certains cas particuliers, une action mémorisée peut être associée à une transition. Cette action sera exécutée si la transition est franchie.

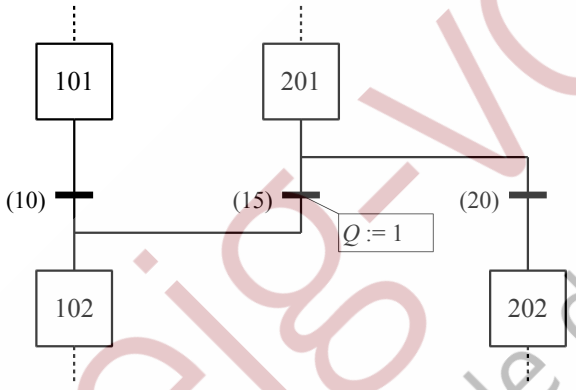


Figure 2.2.25: Action au franchissement

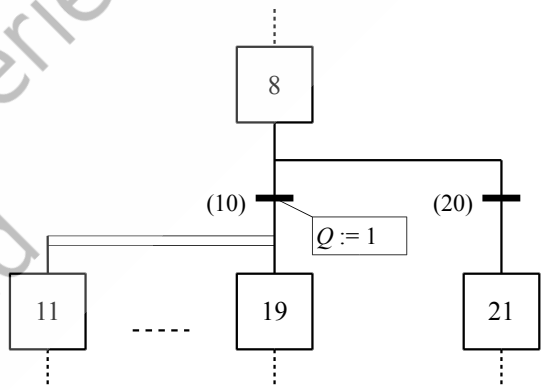


Figure 2.2.26: Action au franchissement

Exemple 1

La valeur vrai est affectée à la sortie Q quand la transition (15) est franchie.

Un fonctionnement équivalent ne pourrait pas être obtenu en effectuant l'action à la désactivation de l'étape 201 ou à l'activation de l'étape 102.

Exemple 2

La valeur vrai est affectée à la sortie Q quand la transition (10) est franchie.

Un fonctionnement équivalent pourrait être obtenu, mais de manière moins optimale, en effectuant cette action à l'activation de toutes les étapes parallèles 11 .. 19.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.6 Les réceptivités

A chaque transition est associé une réceptivité qui est spécifiée par une proposition logique pouvant être vraie ou fausse. Les éléments de cette proposition sont d'une part des variables qui représentent les états du système automatisé et d'autre part des opérateurs logiques.

Ces propositions peuvent être décrites :

- Dans un cahier des charges, généralement définies par un texte contenant un qualificatif.
  Exemples : porte ouverte, pompe en marche ET niveau bas, temps écoulé OU pression haute, etc.
- Quand les spécifications du descriptif de fonctionnement sont plus avancées, les conditions de réceptivité sont spécifiées de manière symbolique.
  Exemples :

G10	{Contact de position fermé},
M105 • LL243	{Moteur 105 marche ET niveau bas 243},
(t2 / X37) + [P2 > 6 bar]	{(Temporisation de t2 sur l'étape 37) ou pression P2 supérieure à 6 bar}

- Pour documenter un programme d'automatisme les propositions seront décrites en utilisant les symboles des variables informatiques.
  Exemples : `%I47; iM105_1 AND iLL243; (X37.x >= T#25s) OR (P2 > 6.0);`

Ces différentes notations interviennent à différentes phases d'avancement d'un projet, on évitera donc de les mélanger.

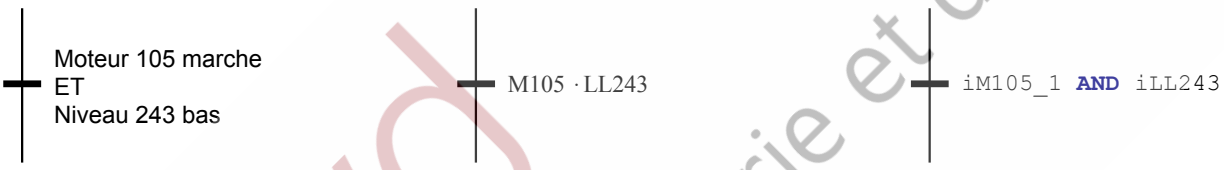


Figure 2.2.27: Exemples de description de réceptivités

2.2.6.1 Réceptivité toujours vraie

Certaines transitions ont une réceptivité qui est toujours vrai, elles sont donc franchies dès qu'elles sont validées.

Elles sont notées « 1 ».

Ce type de transition est souvent utilisé pour la synchronisation de séquences parallèles.

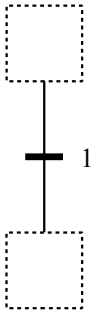


Figure 2.2.28: Réceptivité toujours vraie

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.6.2 Font montant ou descendant

Des événements dynamiques peuvent être utilisés pour spécifier des réceptivités.

Par définition, les événements dynamiques ont une durée infinitésimale.

Ces événements sont des flancs montants ↑ c'est-à-dire passage de l'état faux à l'état vrai. Ce qui est noté par "↑" placé devant la variable.

Les flancs descendants ↓, passage de l'état vrai à faux sont notés par "↓" placé devant la variable.

Cette notation peut être utilisée pour qualifier une variable élémentaire ou une expression booléenne.

Exemple

La transition (I) est franchie quand l'état de la variable a est vrai.

La transition (II) est franchie quand la variable b passe de faux à vrai.

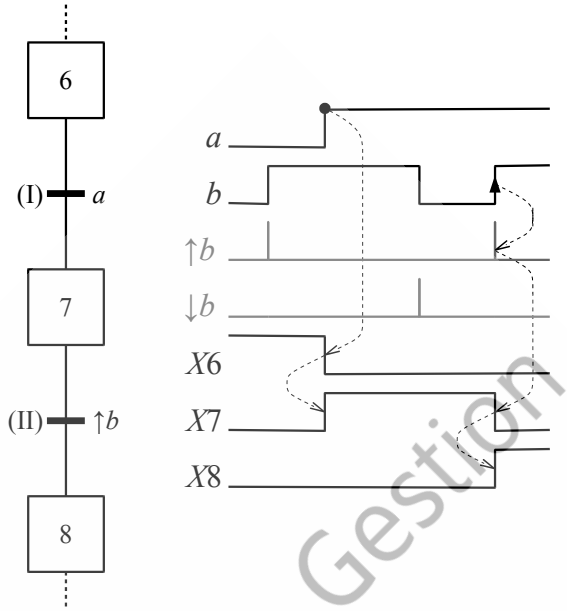


Figure 2.2.29: Transition sur flanc montant

2.2.6.3 Réceptivité dépendante du temps

Une réceptivité peut être temporisée, elle sera franchie après un délai prédéterminé quand la condition est devenue vraie. Cette temporisation se note :

<temps retard> / <condition>

Généralement la condition utilisée est l'activité de l'étape amont.

La condition doit rester vraie pendant un temps supérieur au retard défini pour que la réceptivité puisse devenir vraie.

La temporisation d'une transition peut être utilisée pour définir un **temps enveloppe**, c'est-à-dire que la transition doit être franchie dans un intervalle de temps donné. Les temps enveloppe sont employés pour contrôler un mouvement d'une machine.

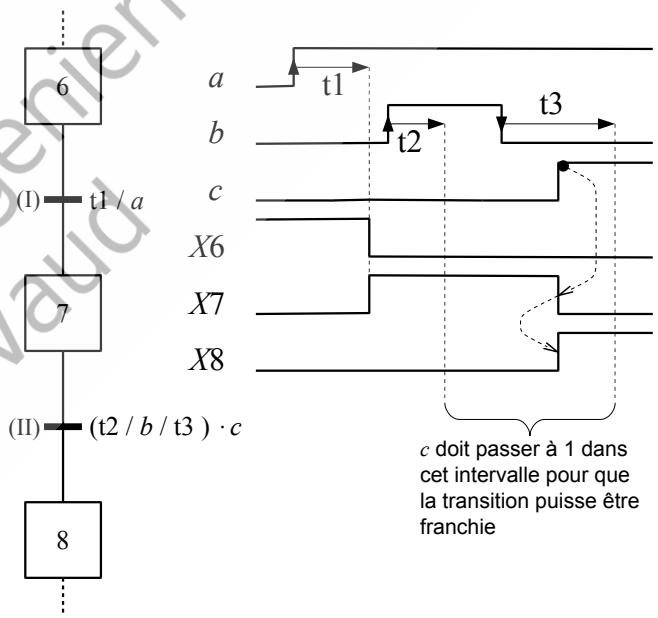


Figure 2.2.30: Réceptivité temporisée et temps enveloppe

Cette dépendance au temps se note :

<temps retard> / <condition> / <temps maintien>

La réceptivité devient vraie après le temps de retard suivant l'occurrence du front montant de la condition et redevient fausse quand le temps de maintien est écoulé.

heig-vd Haute-Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.2.6.4 Prédicat

Une condition de transition peut être déclarée sur la base de variables autres que booléennes, variables de type entier ou flottant. La condition est alors constituée d'un **prédicat**.

Un prédicat est une expression mathématique qui comprend deux arguments et un opérateur de comparaison (=, ≠, <, ≤, ≥, > ou encore ∈, ∉). Le résultat d'un prédicat peut prendre la valeur vrai ou faux.

En GRAFCET un prédicat est noté entre crochets [...] . Le prédicat peut être associé à d'autres variables logiques pour constituer une proposition logique de réceptivité.

Exemples

En langage littéral : [Température inférieure ou égale à 4°C],

Expression symbolique : $[P56 > 6]$

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.3 Structuration

Lorsque les systèmes automatisés de production sont complexes, la description de leur fonctionnement avec un grafcet unique est quasi impossible. Une structuration de la partie commande est alors nécessaire afin d'en simplifier l'étude, la mise en œuvre et la maintenance du système. L'objectif principal de cette structuration est de permettre une approche modulaire de l'analyse et de la représentation fonctionnelle du système.

Une approche modulaire consiste à décomposer un système en éléments partiels qui seront plus simples à étudier et à décrire. Cette décomposition peut se faire soit de manière fonctionnelle, les modules correspondent alors à une fonction particulière du système (sécurité, modes de marche, communication, supervision, etc.), soit selon la topologie de la machine (chargeur, usinage, contrôle, transfert, etc.). En général un système est structuré en utilisant une combinaison de ces deux formes.

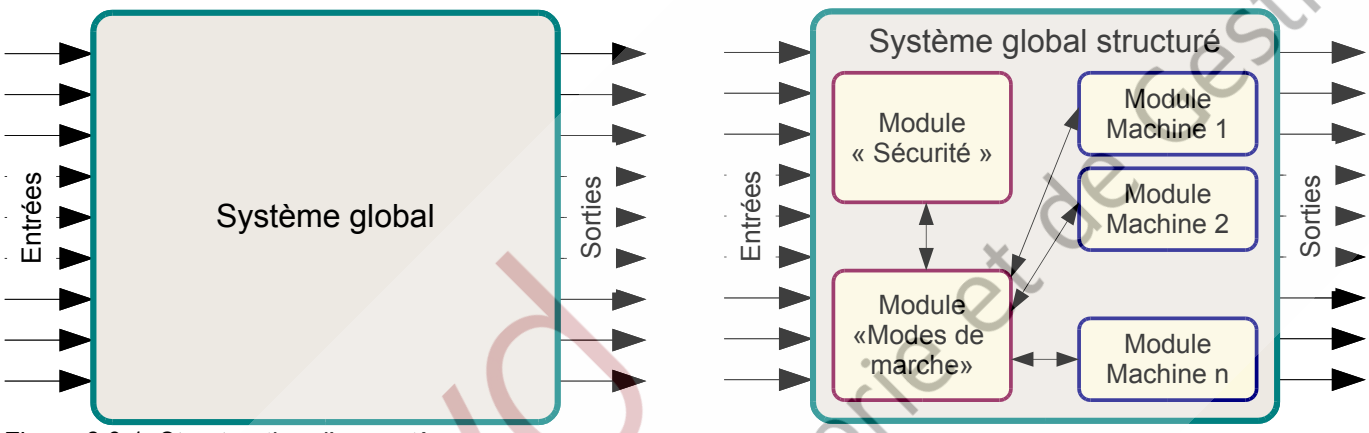


Figure 2.3.1: Structuration d'un système

L'analyse structurée d'un système consiste à le décrire en ayant une approche partant du niveau le plus général vers des niveaux de plus en plus détaillés. Une structuration peut être hiérarchique avec supérieurs et subordonnés ou heuristique avec un réseau relationnel plus complexe. Des outils comme le GEMMA ou l'analyse de risques sont des aides précieuses pour structurer un système.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Le GRAFCET spécifie plusieurs moyens de structuration :

- les macro-étapes,
- les grafcets partiels,
- l'encapsulation.

Cette structuration peut se limiter à un simple découpage ou intégrer des notions de hiérarchisation par forçage ou encapsulation.

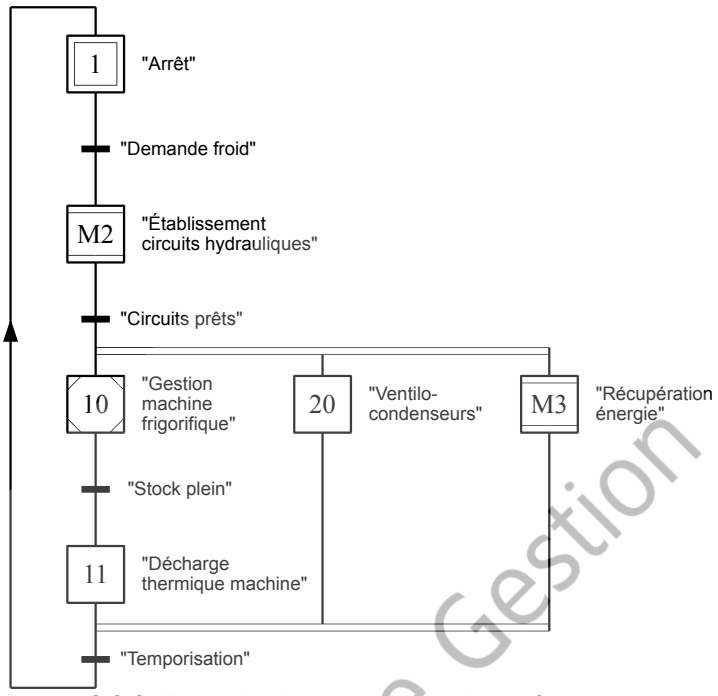


Figure 2.3.2: Exemple de structuration de grafcet

2.3.1 Macro-étape

Lorsque des grafcets contiennent un grand nombre de séquences, pour améliorer leur compréhension, il est possible de les représenter à plusieurs niveaux en utilisant des **macro-étapes**. Celles-ci expriment la fonction à remplir sans présenter de détails superflus au niveau de représentation supérieur. Les macro-étapes permettent une description progressive du général au particulier.

Une macro-étape `Mi`, désignée par un carré avec deux barres horizontales, est un ensemble **unique** d'étapes et de transitions. Cet ensemble appelé **expansion** de la macro-étape représente une partie détaillée du grafcet. Une macro-étape se substitue à une séquence du grafcet.

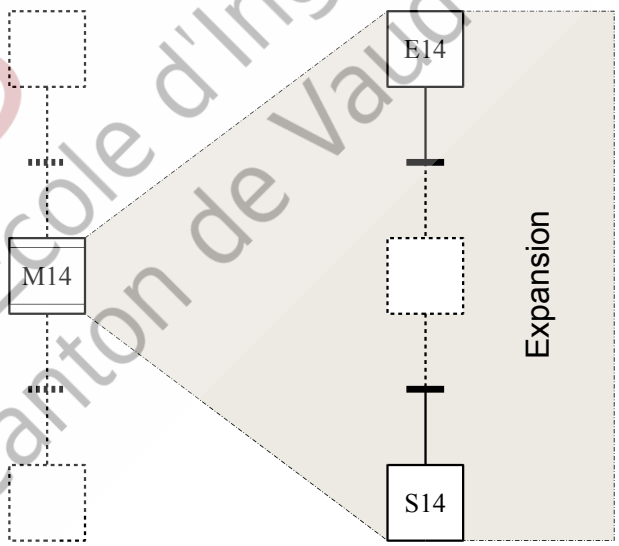


Figure 2.3.3: Macro-étape et son expansion

L'expansion de la macro-étape commence par <u>une seule</u> étape d'entrée notée `Ei` et se termine par <u>une seule</u> étape de sortie notée `Si`. L'étape d'entrée `Ei` devient active lorsque l'une des transitions amonts de la macro-étape est franchie. Les transitions en aval de la macro-étape ne sont validées que lorsque l'étape de sortie `Si` est activée. L'étape de sortie est désactivée lorsqu'une des transitions en aval de la macro-étape est franchie.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

L'expansion d'une macro-étape peut comporter une ou plusieurs macro-étapes. Une macro-étape peut comporter une étape initiale.

Il n'y a généralement pas d'action associée aux étapes d'entrée et de sortie des macro-étapes.

Une macro-étape est dite active lorsque au moins une de ses étapes est active, elle est inactive quand aucune de ses étapes est active. L'activité d'une macro-étape est représentée par une variable booléenne XMn où n est le numéro de la macro-étape.

Remarque : Une macro-étape n'est pas un sous-programme. Une macro-étape est une extension du concept d'étape, à chaque macro-étape correspond une et une seule expansion. Si plusieurs séquences identiques sont à représenter, autant de macros et d'expansions seront nécessaires.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.3.2 Grafcets partiels

La partition d'un grafcet complexe en plusieurs grafcets plus petits et plus compréhensibles est une approche judicieuse. Le grafcet est scindé, d'une part selon la topologie du système pour faire correspondre les modules à des éléments de la partie opérative, d'autre part selon les fonctions à exécuter. Ces grafcets constituent des modules qui sont reliés et en interaction. Les relations peuvent être hiérarchisées en différents niveaux partant du général au détail, ou structurées de façon heuristique.

Le GEMMA est une des méthodes pour réaliser cette structuration, les grafcets de niveau hiérarchiquement supérieur sont des grafcet de gestion qui pilotent les grafcets subordonnés décrivant les séquences. Les principaux grafcets sont:

- Le **grafcet de surveillance** (sécurité) décrit l'ensemble des procédures de sécurité du système. Ce grafcet est au niveau hiérarchique supérieur. Il contient principalement la gestion des arrêts d'urgence et les procédures de mise en route.
- Le **grafcet de conduite** (modes de marche) décrit l'ensemble des procédures de marche: automatique, cycle, pas-à-pas, manuel, ...
- Le **grafcet de maintenance** précise les procédures des opérations de service et de réglage.
- Le **grafcet de production** modélise le fonctionnement normal du système automatisé de production. Ce grafcet est souvent décomposé en plusieurs tâches représentant les différentes fonctions de l'automatisme.

Un grafcet complexe peut être subdivisé en plusieurs **grafcets connexes** plus petits qui respectent les règles de syntaxe du GRAFCET. Le fonctionnement d'un des fonctions ou éléments du système est décrite par un **grafcet partiel**, constitué d'un ou plusieurs grafcets connexes. L'ensemble des grafcets partiels constitue le **grafcet global** qui modélise le fonctionnement du système complet.

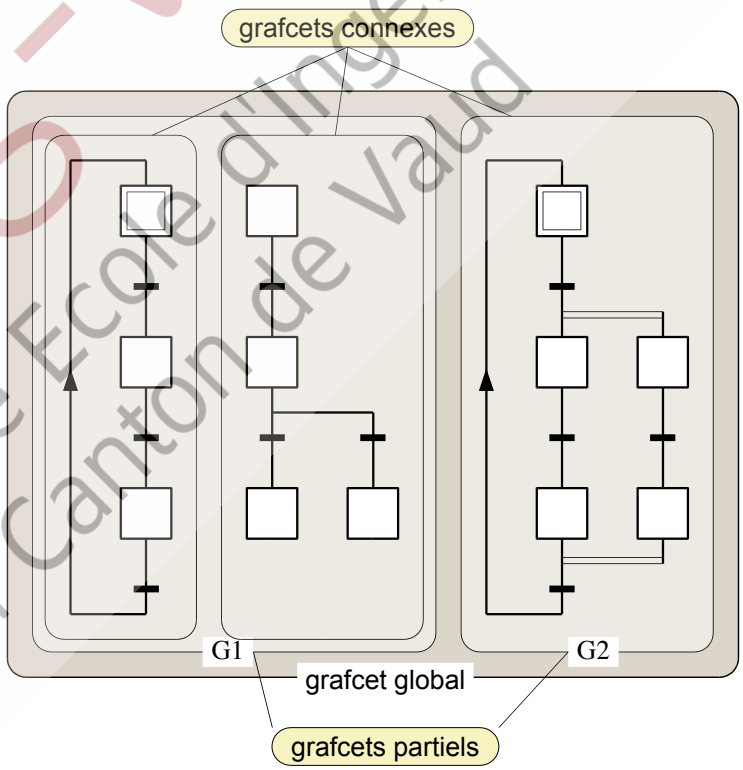


Figure 2.3.4: Partition d'un grafcet

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Les grafcets hiérarchisés forment une structure dans laquelle le supérieur donne des ordres à un ou plusieurs subordonnés qui sont chargés d'exécuter une tâche. Les grafcets subordonnés retournent un accusé d'exécution en fin de tâche. Ces échanges d'informations assurent la synchronisation du fonctionnement.

La structuration par grafcets hiérarchisés permet plus de liberté que l'utilisation de macro-étapes. Une séquence peut être appelée depuis plusieurs étapes du graphe supérieur, voir même d'autres grafcet. L'inconvénient principal est que les mécanismes de synchronisation ne sont pas vérifiés et une erreur de conception peut provoquer des aléas dans le fonctionnement du système.

Les grafcets partiels sont identifiés par le symbole Gn où n est un numéro d'ordre unique quelconque. Une variable booléenne *XG*n associée à chaque grafcet partiel indique son état d'activité. Un grafcet partiel est actif lorsqu'au moins une de ses étapes est active.

La situation d'un grafcet partiel représente l'état d'activité de ses étapes. Pour représenter une situation la notation Gn{p, ... w} est utilisée Gn est le grafcet partiel et entre les accolades se trouve la liste des étapes actives.

Exemple

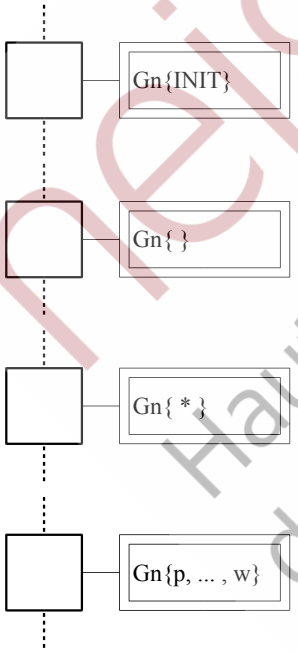
G17{3,7,12,14} signifie que les étapes 3, 7, 12 et 14 du grafcet partiel G17 sont actives.

2.3.2.1 Forçages

La gestion des relations entre les grafcets se fait au moyen d'ordres de forçage. Un ordre de forçage se représente par un double rectangle à droite de l'étape à laquelle il est associé. Dans ce double rectangle est indiqué une situation d'un grafcet partiel.

L'ordre de forçage est associé à l'activité d'une étape d'un grafcet hiérarchiquement supérieur, il impose une situation à un grafcet partiel hiérarchiquement subordonné. L'exécution d'un forçage est prioritaire sur l'application des règles d'évolution. Cela signifie que le grafcet subordonné ne peut plus évoluer et reste dans la situation qui lui est imposée tant que l'étape du grafcet supérieur est active. Le grafcet subordonné est figé.

Quatre possibilités de forçage sont possibles:



Initialisation, les étapes initiales du grafcet partiel forcé n sont activées, toutes les autres étapes sont désactivées.

Désactivation, toutes les étapes du grafcet partiel n sont désactivées.

Le redémarrage du grafcet ne pourra se faire que par un autre ordre de forçage.

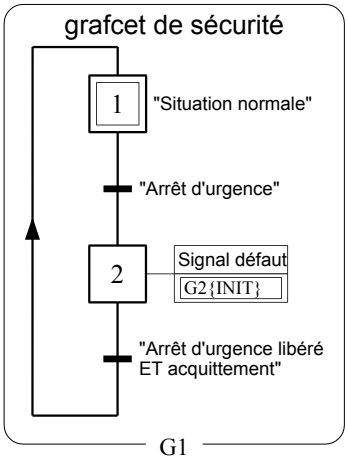
Figeage, le grafcet partiel n est "gelé" dans sa situation courante. Il ne peut plus évoluer tant que l'ordre est présent.

Forçage dans une situation donnée, le grafcet partiel n est forcé dans une situation déterminée. Les étapes repérées entre les accolades sont activées, toutes les autres sont désactivées.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Exemple

L'exemple ci-dessous présente la structuration en grafcets partiels d'un système de commande.



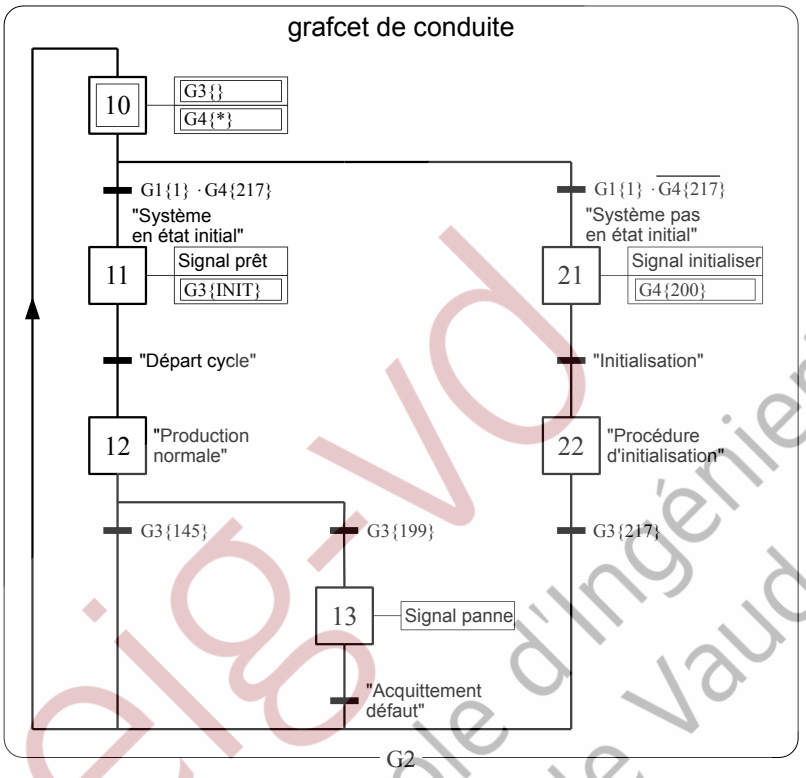
Au niveau hiérarchique supérieur se trouve le grafcet de sécurité.

Ce grafcet gère l'arrêt d'urgence.

En situation normale l'étape 1 est active.

Quand l'arrêt d'urgence est actionné, l'étape 2 s'active : un signal de défaut est actionné et le grafcet de conduite G2 est forcé en situation initiale.

Pour revenir en situation normale l'arrêt d'urgence doit être levé et le défaut acquitté.



Au niveau intermédiaire le grafcet de conduite gère le fonctionnement général du système.

L'étape initiale 10 désactive le grafcet de production G3 et fige le grafcet d'initialisation.

Ce grafcet peut évoluer quand le grafcet de sécurité G1 est en situation normale.

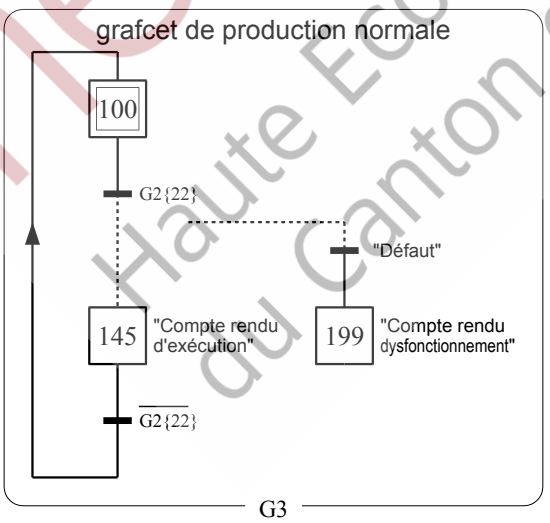
Si le système n'est pas en état initial l'étape 21 force le grafcet d'initialisation G4 à l'étape 200 et attend que l'opérateur donne un ordre d'initialisation.

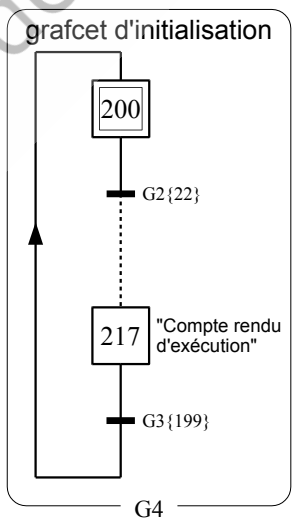
Lorsque le système à été initialisé l'étape 11 signale qu'il est prêt et initialise le grafcet de production G3.

Quand l'ordre de départ cycle est donné l'étape 12 s'active et le grafcet de production est libéré.

A la fin normale du cycle de production l'étape 10 se réactive.

Si une panne se produit l'étape 13 s'active et la panne est signalée. le cycle normal reprend après acquittement.





Au niveau inférieur les grafcets de production normale et d'initialisation représentent les séquences de fonctionnement et d'initialisation du système.

Les dernières étapes de ces grafcets G3{145}, G3{199} et G4{217} donnent les comptes rendus qui sont utilisés pour synchroniser les autres grafcets.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

2.3.2.2 Réutilisation d'une séquence

Une séquence qui se produit plusieurs fois dans un cycle peut être représentée par un grafcet secondaire indépendant. Ce grafcet secondaire est lancé par une étape du grafcet principal chaque fois que c'est nécessaire.

Cette forme d'écriture[4] simplifie l'analyse et optimise la programmation. Les mécanismes de synchronisation doivent être réalisé correctement pour éviter de provoquer des aléas de fonctionnement.

La séquence 30..45 est lancée par l'activation des étapes 3 et 7 du graphe principal. L'étape d'appel reste active pendant le déroulement de la séquence. Lorsque la séquence est terminé la variable $X45$ devient vraie, cette variable est utilisée pour envoyer un compte rendu d'exécution au grafcet principal.

Remarque : La notation [n] ne fait pas partie de la norme CEI 60848 mais est utile pour clarifier la lecture des graphes.

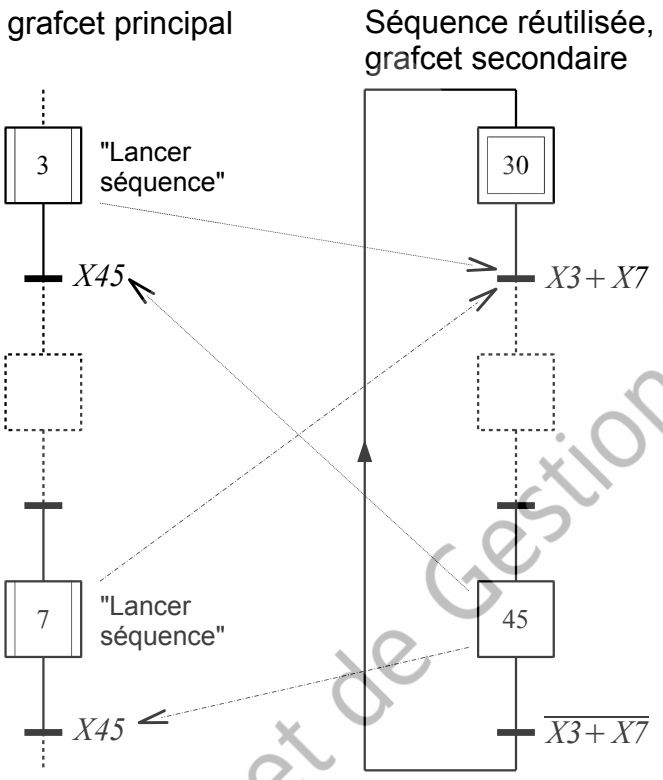


Figure 2.3.5: Réutilisation d'une séquence

2.3.3 Encapsulation

L'encapsulation est un outil permettant la structuration des grafcets complexes. Il regroupe les notions de macro-étapes et de forçage. L'encapsulation associe un ou plusieurs grafcets encapsulés à une étape encapsulante.

L'activation de l'étape encapsulante implique l'activation, dans le(s) grafcet(s) encapsulés, de l'étape ou des étapes possédant un lien d'activation symbolisé par un astérisque (*) à droite de l'étape. Les étapes activées ne sont pas nécessairement des étapes initiales.

La désactivation de l'étape encapsulante entraîne la désactivation de toutes les étapes des grafcets encapsulés.

Le grafcet encapsulé est représenté dans un cadre mentionnant en haut le numéro de l'étape encapsulante et en bas le nom du graphe encapsulé. Chaque grafcet encapsulé ne dépend que d'une seule et unique étape encapsulante. Plusieurs grafcets encapsulés peuvent être associés à la même étape encapsulante. Une étape initiale peut être une étape encapsulante.

Notations

Un grafcet encapsulé est désigné par Xn/Gp ou Xn désigne l'étape encapsulante et Gp le grafcet encapsulé (on peut, s'il n'y a pas d'ambiguïté le designer directement par Gp). Une étape d'un grafcet encapsulé est désigné par Xn/Xm ou Xn désigne l'étape encapsulante et Xm l'étape encapsulée, s'il n'y a pas ambiguïté on peut directement la nommer Xm.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

4 Certains auteurs appellent cette structure "sous-programme".

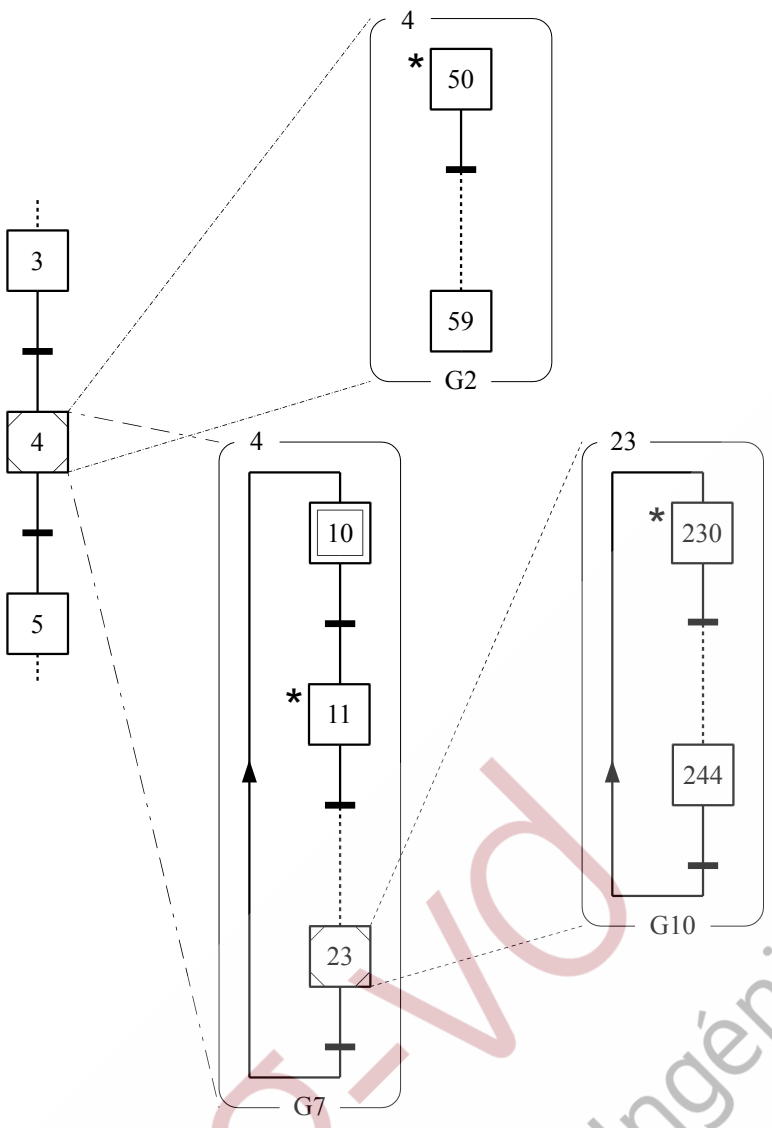


Figure 2.3.6: Encapsulation de grafcets

Une encapsulation se comporte comme une macro-étape lors de l'activation, en effet, l'activation de l'étape encapsulante entraîne l'activation des étapes sélectionnées (par l'astérisque) dans le grafcet encapsulé. Contrairement à l'expansion d'une macro-étape, plusieurs étapes peuvent avoir un lien d'activation, elles seront activées simultanément.

La désactivation du grafcet encapsulé est équivalent à un forçage de désactivation du grafcet encapsulé, cette désactivation intervient dès la désactivation de l'étape encapsulante et une simple évolution du grafcet réalise cette action.

Exemple

Les grafcets X4/G2 et X4/G7 sont deux grafcets encapsulés dans l'étape 4 du graphe principal. Le grafcet X23/G10 est encapsulé dans l'étape 23 de X4/G7.

L'activation de l'étape 4 entraîne l'activation l'activation des étapes X4/X50 de X4/G2 et X4/X11 de X4/G7.

L'étape X4/X23/X230 (étape 230 du grafcet encapsulé dans l'étape 23 du grafcet encapsulé dans l'étape 4 du grafcet principal) de X4/X23/G10 est activée quand X4/X23 devient active.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Chapitre 3

Guide GEMMA

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

3 GEMMA

Le GEMMA, acronyme de Guide d'Étude des Modes de Marche et d'Arrêt, est un outil[5] graphique complémentaire au GRAFCET qui permet d'exprimer de façon claire et complète les besoins en modes de marche d'un système automatisé.

L'objectif du GEMMA est de structurer l'élaboration et la description du fonctionnement d'un système. Par un vocabulaire simple, il facilite le dialogue entre tous les techniciens qui auront à intervenir sur le système. Par une approche guidée, il permet de lister toutes les procédures de marches et d'arrêts de la partie commande et de la partie opérative.

Le GEMMA en définit les états principaux ainsi que trois familles de procédures pour le passage d'un état à l'autre :

(F) procédures de fonctionnement,

(A) procédures d'arrêt,

(D) procédures en défaillance.

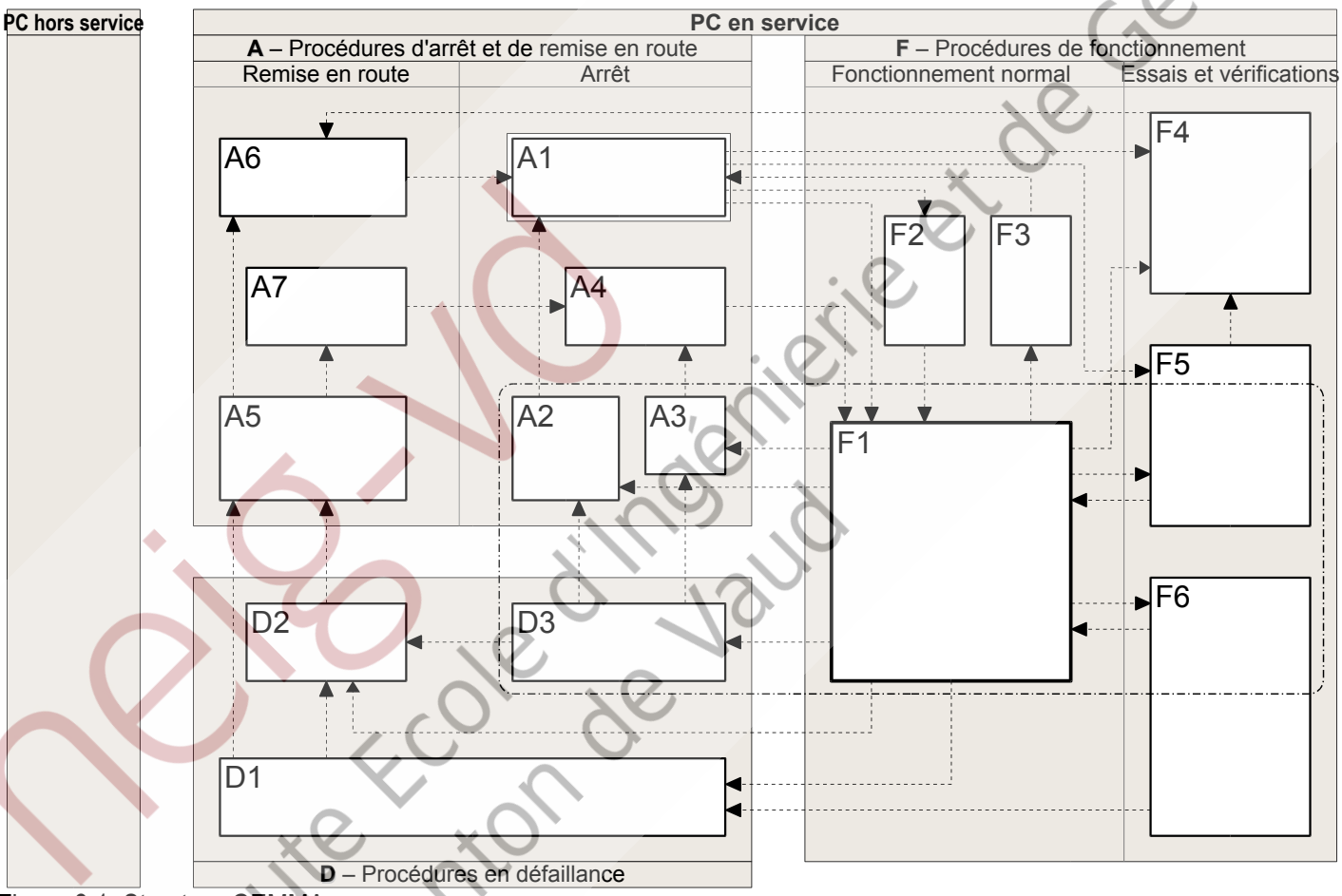


Figure 3.1: Structure GEMMA

Le mode de marche normal d'un système est celui pour lequel il a été conçu. Des modes de marche particuliers peuvent exister, ils doivent cependant être définis et les procédures de changement de mode décrites. Le GEMMA présente sous forme d'un graphique structuré ces différents modes et les liaisons qui sont possibles. Les différents états sont représentés par des rectangles regroupés par famille. Les évolutions sont représentées par des arcs orientées sur lesquelles figurent les conditions. Ce graphique n'est pas figé et peut être adapté en fonction des particularités du système.

5 Outil élaboré par l'ADEPA Agence pour le Développement de la Productique Appliquée

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

Exemples

Avant d'être en production automatique, une machine d'injection de plastic nécessite une phase de pré-chauffage et de purge de sa vis d'extrusion.

Si un incident survient en cours de production un arrêt d'urgence peut être demandé et la machine doit être sécurisée. La phase de redémarrage peut être différente d'un démarrage normal.

Pour effectuer des opérations de maintenance il est parfois nécessaire d'outrepasser certains verrouillages.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.1 STRUCTURE DU GEMMA

Le guide graphique du GEMMA est constitué de deux zones principales qui définissent l'état dans lequel se trouve la partie commande du système automatisé de production :

- Partie commande hors fonction
- Partie commande en fonction

3.1.1 Partie commande hors fonction

La partie commande hors fonction, zone PZ à gauche, ne traite aucun mode de fonctionnement. Seules des actions réflexes peuvent se réaliser. Le choix des composants du système sera prépondérant dans ce mode pour des raisons de sécurité.

3.1.2 Partie commande en fonction

La partie commande en fonction, zones A,D,F à droite, traite les modes de marche et d'arrêt du système classés en trois familles de procédures.

Un critère supplémentaire distingue les procédures de production et de non production. La zone de production recouvre les trois familles de procédures. Le système est en production quand il réalise la valeur ajoutée pour lequel il a été conçu.

3.1.3 Familles de procédures

Les familles de procédures qui décrivent les modes de marche et d'arrêt sont au nombre de trois :

- Les procédures de fonctionnement (F) regroupent tous les états du système indispensables à l'obtention de la valeur ajoutée.
- Les procédures d'arrêt (A) spécifient les états conduisant à un arrêt pour des causes externes au système. Elles correspondent aux arrêts normaux.
- Les procédures en défaillances (D) définissent les états conduisant à un arrêt pour des causes internes au système. Elles correspondent à des défaillances de la partie opérative.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.1.3.1 Procédures de fonctionnement (F)

Les procédures de fonctionnement, au nombre de 6, définissent les états de fonctionnement du système. Dans ces états, le système peut produire, mais il peut également être en réglage ou en test. L'état caractérisant la production normale de tout système fait partie de cette famille de procédures.

F1 : production normale.

Le système produit normalement, c'est l'état pour lequel il a été conçu. Ce rectangle d'état est marqué par un cadre renforcé. On peut souvent faire correspondre à cet état un GRAFCET de production normale.

F2 : marche de préparation.

Cet état permet au système d'atteindre les conditions nécessaires pour pouvoir accéder à la production normale. Il est utilisé pour les machines nécessitant une préparation préalable à la production normale : préchauffage de l'outillage, remplissage de la machine, mises en routes diverses...

Exemples : Préchauffage du fourreau d'une presse à injecter, mise en place d'une boîte avant remplissage.

F3 : marche de clôture

Cet état permet au système d'atteindre une certaine position avant un arrêt prolongé. Cet état est nécessaire pour certaines machines qui doivent être vidées ou nettoyées en fin de journée ou en fin de campagne de production.

Exemple : Avant l'arrêt de la presse à injecter les plastiques thermodurcissables, il est nécessaire de vider le fourreau afin de pouvoir réutiliser la presse.

F4 : marche de vérification dans le désordre

Cet état permet de vérifier certaines fonctions ou certains mouvements sur la machine, sans respecter l'ordre du cycle. Il correspond le plus souvent au mode manuel des organes de la machine.

F5 : marche de vérification dans l'ordre

Dans cet état, le cycle de fonctionnement peut évoluer au rythme souhaité par l'opérateur effectuant la vérification, la machine pouvant produire ou ne pas produire. Le cycle de production normale évolue tâche par tâche. La machine peut être ou non en production. Cet état correspond au mode manuel des groupes fonctionnels.

F6 : marche de test

Cet état permet le réglage de différents éléments du système qui doivent l'être. Ces réglages peuvent être effectués en ou hors production.

Exemple : Les machines de contrôle, de tri, comportent des capteurs qui doivent être réglés ou étalonnés périodiquement.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.1.3.2 Procédures d'arrêt (A)

Les procédures d'arrêt, au nombre de 7, définissent les états de d'arrêt du système. Les causes de l'arrêt sont externes au système. Ces état permettent d'amener le système à un arrêt normal ou de préparer le système à une procédure de remise en route.

A1 : arrêt dans l'état initial

C'est l'état repos du système avant qu'il passe en production normale. Cet état est particulier, il correspond en général à la situation initiale du GRAFCET : c'est pourquoi, comme une étape initiale, ce rectangle est marqué d'un double cadre.

Remarque: Pour une analyse plus aisé de l'automatisme, il recommandé de représenter la machine dans cet état initial.

A2 : arrêt demandé en fin de cycle

Cet état permet de conduire le système à un arrêt en fin d'un cycle de production. Le système continue de produire et s'arrête lorsque le cycle de production sera terminé. A2 est un état transitoire vers A1.

A3 : arrêt dans un état déterminé

Cet état permet de conduire le système à un arrêt différent du précédent. La machine continue de produire jusqu'à un arrêt dans une position autre que la fin du cycle : c'est un état transitoire vers A4.

Il permet par exemple d'arrêter le système dans un état permettant une intervention sur un système.

A4 : arrêt obtenu

Cet état permet de conduire le système à un arrêt différent de l'arrêt en fin de cycle.
Par exemple cet état est utilisé lorsque l'on souhaite réalimenter en matière première un système.

A5 : préparation pour remise en route après défaillance

Cet état permet de ramener le système après une défaillance dans une position qui lui permettra de remettre en route le système. Dans cet état l'opérateur intervient en général manuellement pour dégager, nettoyer ou vider le système.

A6 : mise de la partie opérative dans son état initial

Cet état permet, en général après une remise en route après défaillance du système, de ramener le système, manuellement ou automatiquement, en position initiale pour un redémarrage.

A7 : mise de la partie opérative dans un état déterminé

Cet état permet d'arrêter le système dans une position autre que la position initiale. Le redémarrage du système se fera donc dans un état autre que l'initial.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.1.3.3 Procédures en défaillances (D)

Les trois procédures en défaillance définissent les états que devra avoir la partie opérative en cas de défaillance. Ces état permettent gérer les défaillances du système tel que par exemple l'arrêt d'urgence. Les cause de l'arrêt sont internes au système. Les états de défaillance sont les états de défaillance de la partie opérative.

D1 : marche ou arrêt en vue d'assurer la sécurité

Cet état permet de gérer le système lors d'un arrêt d'urgence. Il prévoit toutes les mesures visant à protéger le système, cycle de dégagements et précautions pour limiter les conséquences de la défaillance.

D2 : diagnostic et/ou traitement de défaillance

Cet état permet d'examiner la machine, de diagnostiquer l'origine de la défaillance et d'envisager le traitement approprié qui permettra le redémarrage du système après suppression de la défaillance.

D3 : production tout de même

Cet état permet, en cas de nécessité, de continuer la production après défaillance du système. La production n'est plus forcément automatisée, elle peut être alors accompagnée par un opérateur. On parlera alors de marche dégradée ou de production forcée.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.2 Utilisation du GEMMA

3.2.1 Sélection des modes de marche et d'arrêt

Avec le GEMMA l'étude des modes de marches et d'arrêts est prévue dès la conception et intégrée dans la réalisation. Après établissement du GRAFCET opérationnel de base, on met en œuvre le guide graphique GEMMA pour la sélection des modes de marches et d'arrêts.

Sur le Gemma, chaque état est caractérisé par son nom et son repère. Dans un premier temps on sélectionne les états nécessaires à la description du système automatisé étudié, puis on défini les liaisons entre les état. Une brève définition de l'effet attendu pourra être utilisé pour décrire le comportement attendu dans chaque d'état.

Les rectangles d'états du guide graphique constituent une liste de contrôle (*check-list*) des différents modes de marches et d'arrêts nécessaires pour un automatisme courant. Si le mode proposé est retenu, il sera précisé sous forme de descriptif de fonctionnement de la machine, dans le rectangle d'état et, au besoin, plusieurs variantes de ce mode peuvent être distinguées.

3.2.2 Notion de boucle opérationnelle

Sur le GEMMA on caractérise plusieurs "boucles". Une boucle est une succession d'états caractérisant le fonctionnement du système. En effet il n'est possible de passer d'un état à un autre que si les conditions d'évolutions sont respectées, mais il est parfois impossible de passer d'un état à un autre sans utiliser un état intermédiaire. Cet état intermédiaire permettra d'atteindre l'état final sans risque pour le système, ce qui est l'objectif premier.

3.2.3 Analyse des boucles opérationnelles

3.2.3.1 Marche normale

La suite d'états A1→F2→F1→A2→A1 est la boucle de marche normale. C'est en suivant celle-ci que le système va pouvoir fonctionner correctement. Elle décrit le fonctionnement normal du système. En fin de cycle lors d'un arrêt de fabrication, le système vient se remettre en position initiale et sera donc prêt pour un redémarrage.

3.2.3.2 Marche de réglage

La suite A1→F4→A6→A1 est la boucle de marche de réglage. Le système quitte l'état A1 (arrêt dans conditions initiales) et passe en F4 (Marches de vérification dans le désordre) ce qui permet à l'opérateur de pouvoir tester les actionneurs, pré-actionneurs, capteurs, etc., du système sans devoir respecter l'ordre des séquences. Une fois les vérifications effectuées, le système passe de l'état F4 à l'état A6 (Mise de la P.O. dans l'état initial). Dans cet état le système va atteindre les conditions initiales. Quand les conditions seront atteintes, le système passera de l'état A6 à l'état A1.

3.2.3.3 Arrêt de sécurité

La suite F1→D1→A5→A6→A1→F1 est la boucle d'arrêt de sécurité. Cette boucle permet de gérer tous les états successifs d'un système automatisé depuis un arrêt d'urgence lors d'une production normale jusqu'à la reprise de la production normale.

L'état D1 a une particularité intéressante. Sur la flèche de liaison entre l'état F1 et l'état D1 vient se greffer une extrémité de flèche. Cette flèche associée à son commentaire signifie que cette case est accessible depuis tous les états du GEMMA. Autrement dit quel que soit l'état dans lequel se situe le GEMMA, si les conditions nécessaires pour passer dans l'état D1 sont réunies alors le système se mettra en D1.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

3.2.4 Conditions d'évolution

On peut passer d'un état à l'autre de deux manières :

- Avec condition d'évolution
  La condition d'évolution est portée sur la liaison orientée entre états ; la condition peut être liée à une action de l'opérateur au pupitre de commande, ou à une information d'un capteur situé sur la machine.
- Sans condition explicite
  Pour certaines évolutions entre états, l'écriture d'une condition n'apporterait aucune information utile : particulièrement lorsque celle-ci est évidente, (par exemple pour le passage de A5 à A6).

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Chapitre 4

Chapitre non disponible en consultation publique

CoDeSys : Langage SFC

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Annexes

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

5 Annexes

5.1 Glossaire

Les définitions du glossaire sont extraites du « Grand dictionnaire terminologique » de l'Office québécois de la langue française ou du Larousse et adaptées aux besoins de ce cours.

action — Élément du langage GRAFCET associé à une étape décrivant le comportement d'une fonction, d'une tâche ou d'une variable.

actionneur — Dispositif qui permet de modifier le fonctionnement ou l'état d'une machine ou d'un système. Organe de commande d'un mécanisme.

activation — Mise à l'état actif d'une étape.

actuateur — Dispositif qui engendre par son action la force motrice nécessaire pour effectuer le mouvement linéaire ou rotatif d'un mécanisme.

bit — Unité élémentaire d'information de nature binaire. De l'anglais *binary digit*.

capteur — Dispositif sensible à une grandeur physique et permettant de la transformer en un signal de sortie mesurable (généralement un signal électrique).

chronogramme — Diagramme représentant l'évolution temporelle d'un phénomène.

condition — Résultat de l'évaluation d'une proposition logique.

consigne — Valeur d'un point de fonctionnement donnée par l'opérateur à un système. (Anglais : *set point*; Allemand *Sollwert*)

coordination — Possibilité de faire évoluer un grafcet à partir d'information liées à la situation d'autres grafcets.

défaillance — Cessation de l'aptitude d'une unité fonctionnelle à accomplir une fonction requise.

désactivation — Mise à l'état inactif d'une étape.

diagramme — Représentation graphique décrivant le comportement d'un système, par exemple les relations entre deux ou plus de deux grandeurs variables, actions ou états.

événement — Variable qui permet, par ses différents états, d'indiquer la situation ou l'évolution d'une partie d'un système. Tout fait significatif pour un traitement.

- ~ d'entrée — événement caractérisé par le changement de valeur d'une ou plusieurs variables d'entrée de la partie séquentielle du système.
- ~ interne — événement caractérisé par un événement d'entrée associé à la situation de la partie séquentielle du système.

étape — Élément du langage GRAFCET symbolisant un état ou une partie de l'état du système et qui correspond à une phase durant laquelle une action est effectuée.

évolution — Passage d'une situation d'un grafcet à une autre.

- ~ fugace — évolution caractérisée par le franchissement de plusieurs transitions successives sur occurrence d'un unique événement d'entrée.

expansion — Sous-séquence d'un grafcet associée à une macro-étape

figeage — Blocage de toute évolution d'un grafcet lorsqu'une condition particulière se présente.

forçage — Passage à une situation imposée d'un grafcet par un grafcet hiérarchiquement

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

	supérieur.
franchissement	Passage d'une transition.
front	Changement d'état d'un signal.
GEMMA	Acronyme de Guide d'Étude des Modes de Marche et d'Arrêt.
grafcet	Diagramme fonctionnel utilisant le langage GRAFCET pour décrire le comportement d'un système séquentiel.
GRAFCET	Langage de spécification, décrit par la norme CEI 60848, pour la description fonctionnelle du comportement d'un système séquentiel. Acronyme de GRAphe Fonctionnel de Commande, Étape, Transition.
incrémenter	Ajouter un incrément (une unité) au contenu d'une variable.
interprétation	Partie du GRAFCET permettant de faire la relation entre - les variables d'entrée et la structure, par les réceptivités - les variables de sortie et la structure, par les actions.
interface	Limite commune à deux systèmes permettant des échanges entre ceux-ci.
liaison orientée	Élément du langage GRAFCET, ces liaisons indiquent les voies d'évolution possibles en reliant les étapes aux transitions et les transitions aux étapes.
machine	Ensemble constitué de pièces, d'organes et de dispositifs formant un tout, qui effectue des opérations mécaniques.
macro-étape	Représentation symbolique d'une partie du grafcet permettant des descriptions par raffinements successifs où plusieurs niveaux de représentation peuvent être mis en œuvre.
mécatronique	Intégration de produits qui sont issus de la mécanique, de l'électronique et de l'informatique, au sein d'un système, dans le but de réaliser une fonction précise.
mesure	Valeur d'une grandeur comparativement à une grandeur constante de même espèce prise comme référence. (Anglais : *measure*; Allemand *Istwert*)
parallélisme	structure particulière du GRAFCET comportant des branches à évolution simultanée.
PC	Partie Commande : partie du système automatisé de production qui regroupe les composants assurant le traitement des informations d'entrée en vue de commander les sorties.
PO	Partie Opérative : partie du système automatisé de production qui regroupe les capteurs et actionneurs.
prédicat	Fonction mettant en œuvre un ou plusieurs arguments, utilisée pour déclarer quelque chose concernant des objets, et dont le résultat est vrai ou faux.
procédé	Méthode à suivre pour obtenir un résultat.
processus	Ensemble des opérations d'élaboration d'un produit selon un procédé déterminé au moyen d'unités de traitement et de transformation.
réceptivité	Élément du langage GRAFCET, proposition logique associée à une transition.
SAP	Système Automatisé de Production : machine qui crée de la valeur ajoutée.
séquence	Dans un GRAFCET, suite linéaire d'étapes et de transitions.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud

SFC	Sequential Function Chart : transposition du GRAFCET en langage informatique spécifié par la norme IEC 61131-3.
seuil	Valeur qu'une grandeur physique doit atteindre pour être prise en compte.
situation	Désignation de l'état du système spécifié par un grafcet et caractérisé par les étapes actives à l'instant considéré. Ensemble des étapes actives d'un grafcet.
structure	Partie du GRAFCET permettant de décrire l'évolution possible entre les situations.
structuration	Organisation modulaire.
synchronisation	Opération consistant à faire concorder ou à coordonner deux ou plusieurs opérations, processus ou phénomènes.
système	Ensemble d'éléments reliés entre eux, considérés dans un contexte défini comme un tout et séparés de leur environnement.
transition	Élément graphique de base du langage GRAFCET symbolisant le passage entre deux étapes. Indique la possibilité d'évolution entre deux ou plusieurs étapes.

5.2 Bibliographie

[1] **Langage de spécification GRAFCET pour diagrammes fonctionnels en séquence,** 2002-02
Norme CEI 60848

[2] **Le GRAFCET, Conception-Implantation dans les Automates**, 2009
Simon Moreno, Edmond Peulot
Casteilla, ISBN 978.2.7135.3064.7

[3] **Manuel de développement de programmes pour automates programmables avec CoDeSys 2.3**, 2006
3S-Smart Software Solutions GmbH, D-87439 Kempten

[4] **Automatique, Informatique industrielle**, 2001
François Benielli, Gilles Cerato
Foucher, ISBN 2-216-08590-1

[5] **Informatique industrielle II**, 1986
Henri Nussbaumer
PPUR, ISBN 2-88074-101-7

5.3 Webographie

[A] http://www.granddictionnaire.com
Le grand dictionnaire terminologique de l'office québécois de la langue française.

[B] http://fr.wikipedia.org
Le projet d'encyclopédie libre

[C] http://www.epsic.ch/pagesperso/maccaudo/Schema/Exercices/AnimationsFlash/Grafcet.swf
Quelques animations flash intéressantes.

heig-vd Haute Ecole d'Ingénierie et de Gestion du Canton de Vaud