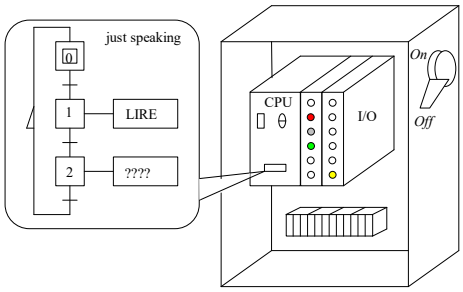
SOMMAIRE



I Présentation générale.

Les automates programmables du commerce peuvent grossièrement se diviser en deux catégories : ceux ayant un noyau GRAFCET (Ex TSX OMRON ..) et ceux ayant un noyau orienté manipulation de bits mots (Ex : Siemens ABB ...). L'implémentations du langage de description séquentiel d'automatisme GRAFCET sur cette dernière catégorie d'automate relève souvent de recettes de cuisine. Ces recettes fonctionnent relativement bien dans les cas simples, mais, posent des problèmes de non respect des 5 (CINQ!) règles du GRAFCET dans beaucoup de cas. De plus la relecture d'un programme pour modification n'est pas chose aisée.

Nous allons essayer d'éclaircir certaines méthodes de programmation, évaluer leurs problèmes, puis proposer une méthode très générale.

Durant ces pages, les exemples seront traités en utilisant soit le mnémonique des automates Siemens, soit un macro-langage issu de l'analyse informatique. Il est possible d'adapter ces exemples en un autre langage, je vous en laisse le soin.

II Notions préliminaires

2.1 Notion de table

Durant toute cette présentation, nous ferons appel à des tables de bits représentant les activités d'étapes et ou tout autres informations relatives à ces étapes. Dans un automate basé sur un microprocesseur, l'unité de stockage mémoire est l'octet ou le mot (2 octets), ce qui donne par table 8 ou 16 informations binaires que l'on peut manipuler indépendamment ou par paquet.

Exemple de table sur un automate siemens

Nom d'étape	0	1	..	7	8	9	..	15
Variable d'activité associée	X0	X1	..	X7	X8	X9	..	X15
Variable automate M10, M11	M10.0	M10.1		M10.7	M11.0	M11.1	..	M11.7

Par la suite nous parlerons des tables d'activité d'étape T_p et T_s. Dans les exemples, pour plus de clarté, nous utiliserons indistinctement les adressages automates (Mx.y) ou les variables d'activité associées (Xi). De même pour les transitions, nous utiliserons les pseudo adressages t_i

.

2.2 Opérations sur les tables

Toutes les opérations booléennes classiques ET (.), OU (+), NON (/) , OU exclusif (⊕) bit à bit s'appliquent aussi sue les tables. Ecrire M30=M10 et M20 revient à écrire pour chaque information booléenne de la table M30.x=M10.x et M20.x

Certains automates ne sont pas équipés d'une instruction de négation de mots (négation bit à bit). Il est possible de contourner ce problème en utilisant l'instruction de Ou exclusif. En effet il est facile de montrer que $\overline{T} = T \oplus FFFF$ ou FFFF représente un mot 16 bits tous à 1.

2.3 Les règles du GRAFCET

Règle N°1
Les étapes initiales, repérées par un double carré, sont activées inconditionnellement à l'initialisation de l'automatisme, au début du fonctionnement.

Règle N°2
Le franchissement d'une transition ne peut se produire que si la transition est validée (étapes précédentes actives) et si la réceptivité associée à la transition est vraie.
Si les deux conditions sont réunies, la transition devient franchissable et est alors obligatoirement franchie.

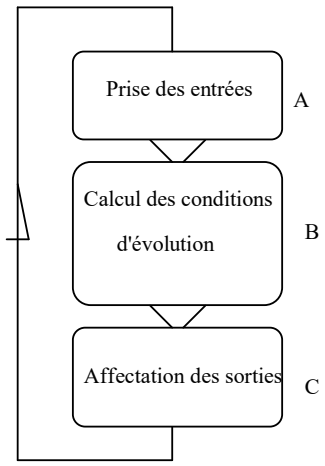
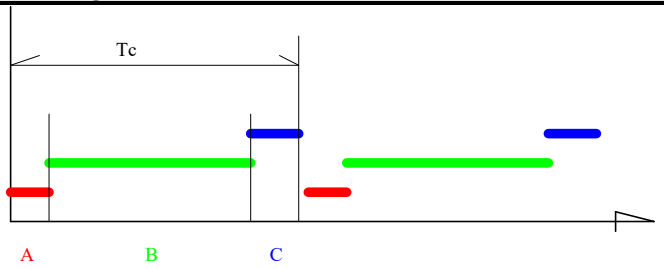
Règle N°3
Le franchissement d'une transition entraîne en même temps l'activation de toutes les étapes immédiatement suivantes et la désactivation de toutes les étapes immédiatement précédentes.

Règle N°4
Plusieurs transitions simultanément franchissables sont simultanément franchies.

Règle N°5
Si, au cours du fonctionnement une étape doit être à la fois désactivée et activée, elle reste active.

2.4 Le cycle classique d'un automate

Un automate réputé asynchrone vis à vis des événements se comporte comme suit :

Opérations	Chronogramme
	

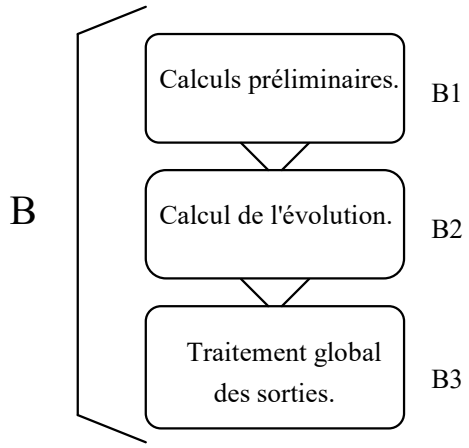
Tc Temps d'un cycle de calcul.

Le temps de réaction du système vis à vis de l'automate varie dans la fourchette : [Tc, 2 .Tc]

2.5 Travail du programmeur

Le programmeur n'a accès qu'a la case B. Afin d'éviter les problèmes de réaffectation des sorties, la méthode à utiliser pour la programmation doit avoir une certaine rigueur et se présenter comme suit :

Détail de la partie B



B1 : Dans cette partie on traite :
- Les fronts
- Les conversions analogiques numériques

B2 : Le traitement du corps du GRAFCET et ou toute autre méthode de description.

B3 : Différentes opérations sont à effectuer dans l'ordre pour un bon fonctionnement :
- Traitement des forçages
- Traitement des compteurs
- Lancement des temporisations
- Traitement global des sorties et figeages
- *Conversions numériques analogiques.*

soit la chronologie suivante :

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations

Traitement des affectations des sorties et figeages

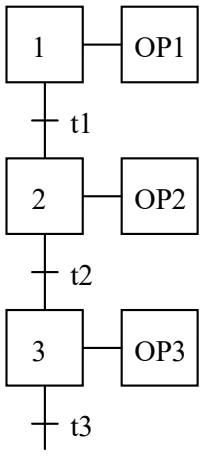
Nous reviendrons ultérieurement plus en détail sur le pourquoi et l'ordre impératif des actions de B3.

III La méthode 'SET RESET'

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

3.1 Présentation

Cette méthode repose sur un principe simpliste et une analyse assez succincte des règles du GRAFCET.
Les étapes sont décrites par une seul table d'activité.
Il est néanmoins possible avec un certain nombre de ruses de respecter les quatre (voire toutes les) règles.

Les programmations classiques d'un tel graphe par la méthode SET RESET sont les suivantes.	Méthode N°1	Méthode N°2	Prog. Siemens N°2
	***SI*** t1 ***ET*** X1 ***ALORS*** ***SET*** X2 ***RESET*** X1 ***FIN SI*** ***SI*** t2 ***ET*** X2 ***ALORS*** ***SET*** X3 ***RESET*** X2 ***FIN SI*** ***SI*** t3 ***ET*** X3 ***ALORS*** ***SET*** X? ***RESET*** X2 ***FIN SI***	***SI*** t3 ***ET*** X3 ***ALORS*** ***SET*** X? ***RESET*** X3 ***FIN SI*** ***SI*** t2 ***ET*** X2 ***ALORS*** ***SET*** X3 ***RESET*** X2 ***FIN SI*** ***SI*** t1 ***ET*** X1 ***ALORS*** ***SET*** X2 ***RESET*** X1 ***FIN SI***	***U*** t3 ***U*** X? ***S*** X? ***R*** X3 *** ***U*** t2 ***U*** X2 ***S*** X3 ***R*** X2 *** ***U*** t1 ***U*** X1 ***S*** X2 ***R*** X1 ***

Dans le cas ou les réceptivités transitions associées aux conditions logiques t1 et t2 seraient franchissables simultanément, les deux programmations ne conduisent pas au même résultat.
Dans le cas N°1, nous passons en un tour automate de l'étape N°1 à l'étape N°3. La sortie OP2 correspondant à l'étape N°2 ne sera jamais activée.

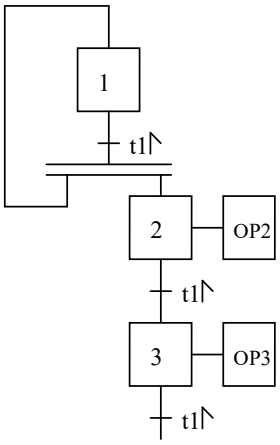
soit l'évolution X1 $\rightarrow$X3.

Dans le cas N°2, nous passons par l'étape N°2 au moins un tour automate. La sortie OP2 sera (si t2 reste durant Tc) une sortie du style impulsionnelle.

soit l'évolution X1 →X2 →X3.

La programmation N°1 ne respecte pas les règles du GRAFCET. Il est donc possible dans ce cas simple de tourner le problème en programmant les étapes de bas en haut.

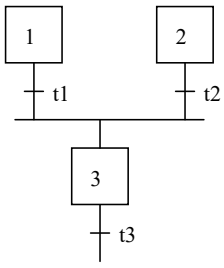
3.2 Problème de la règle N°5

Exemple de GRAFCET style registre à décalage	Pseudo-programmation	Prog. Siemens
	***SI*** t1↑ ***ET*** X3 ***ALORS*** ***SET*** X? ***RESET*** X3 ***FIN SI*** ***SI*** t1↑ ***ET*** X2 ***ALORS*** ***SET*** X3 ***RESET*** X2 ***FIN SI*** ***SI*** t1↑ ***ET*** X1 ***ALORS*** ***SET*** X2 ***FIN SI***	***U*** t1↑ ***U*** X3 ***S*** X? ***R*** X3 *** ***U*** t1↑ ***U*** X2 ***S*** X3 ***R*** X2 *** ***U*** t1↑ ***U*** X1 ***S*** X2 ***

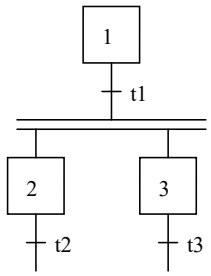
La méthode de programmation ci dessus (en décrivant le problème de bas en haut) permet (grâce à l'écriture incomplète de transition entre l'étape N°1 et N°2) de résoudre le problème.

3.3 Cas de divergences convergence en OU et ET

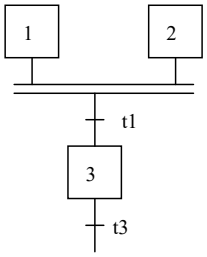
Cas N°1

Convergence en Ou	Méthode	Prog. Siemens
	***SI*** t1 ***ET*** X1 ***ALORS*** ***SET*** X3 ***RESET*** X1 ***FIN SI*** ***SI*** t2 ***ET*** X2 ***ALORS*** ***SET*** X3 ***RESET*** X2 ***FIN SI***	***U*** t1 ***U*** X1 ***S*** X3 ***R*** X1 *** ***U*** t2 ***U*** X2 ***S*** X3 ***R*** X2 ***

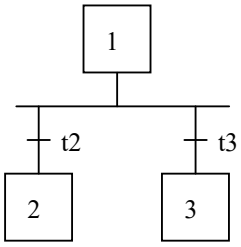
Cas N°2

Divergence en Et	Méthode	Prog. Siemens
	***SI*** t1 ***ET*** X1 ***ALORS*** ***SET*** X2 ***SET*** X3 ***RESET*** X1 ***FIN SI***	***U*** t1 ***U*** X1 ***S*** X2 ***S*** X3 ***R*** X1 ***

Cas N°3

Convergence en Et	Méthode	Prog. Siemens
	***SI*** t1 ***ET*** X1 ***ET*** X2 ***ALORS*** ***SET*** X3 ***RESET*** X1 ***RESET*** X2 ***FIN SI***	***U*** t1 ***U*** X1 ***U*** X2 ***S*** X3 ***R*** X1 ***R*** X2 ***

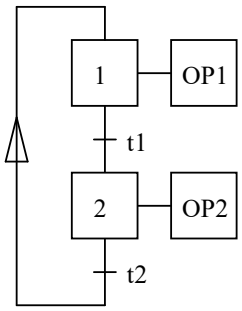
Cas N°4
Ce cas met en oeuvre la règle N°4

Divergence en Ou	Méthode	Prog. Siemens N°2
	***SI*** t2 ***ET*** X1 ***ALORS*** ***SET*** X2 ***FIN SI*** ***SI*** t3 ***ET*** X1 ***ALORS*** ***SET*** X2 ***FIN SI*** ***SI*** (t2 ***OU*** t3) ***ET*** X1 ***ALORS*** ***RESET*** X1 ***FIN SI***	***U*** t2 ***U*** X1 ***S*** X3 *** ***U*** t3 ***U*** X1 ***S*** X2 *** ***U*** X1 ***U(*** ***U*** t2 ***O*** t3 ***)*** ***R*** X1 ***

.

3.4 Graphes à deux étapes

Attention !!




```
SI t2 ET X2 ALORS
   SET   X1
   RESET X2
FIN SI

SI t1 ET X1 ALORS
   SET   X2
   RESET X1
FIN SI
```

Dans le cas t1 et t2 vrai, X2 active, le tour de calcul ne provoque aucune évolution du graphe et donc l'opération OP2 (qui peut déclencher le lancement d'une sous-tâche) ne sera pas exécuté. Il faudrait que l'opération d'évolution s'effectue en deux tours de calcul. Cette méthode de programmation est donc à bannir pour de tels graphes.
On peut remarquer que la méthode de programmation de haut en bas peut provoquer les mêmes problèmes.

3.5 Critique

Cette méthode peut fonctionner à condition que le programmeur soit parfaitement au clair avec les règles du GRAFCET et le traitement séquentiel d'un programme. Elle ne nécessite que peu de mémoire tant du point de vue tables que code opératoire.

On peut remarquer (notamment dans le cas N°4) une non-uniformité d'écriture des évolutions de graphe. Le fait de devoir utiliser la 'ruse' de décrire un graphe à plusieurs branches en parallèle de bas en haut, de traiter les cas particuliers des divergences et règles 5 oblige à une gymnastique d'esprit difficile à transmettre par un document écrit en vue de la maintenance logicielle de la machine.

Le principal problème vient de la destruction prématurée de l'état des étapes antérieures (par l'action du RESET) avant traitement pour une autre transition.

IV Méthode à deux tables et analyse globale.

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

Cette méthode se propose de rationaliser l'écriture de l'activation desactivation des étapes et de résoudre le problème de perte d'informations par destruction de l'état antérieur. La programmation réside dans l'étude approfondie de l'environnement de l'étape et de l'écriture des conditions d'activation desactivation de l'étape.

Dans cette partie nous nommerons T_p La table d'activité des étapes au tour automate précédent, T_s La table d'activité des étapes en cours d'actualisation.

4.1 Déroulement des opérations

Dans la partie 2.5, nous avons détaillé le traitement B. Sa composante B2 va maintenant faire l'objet de la subdivision suivante.

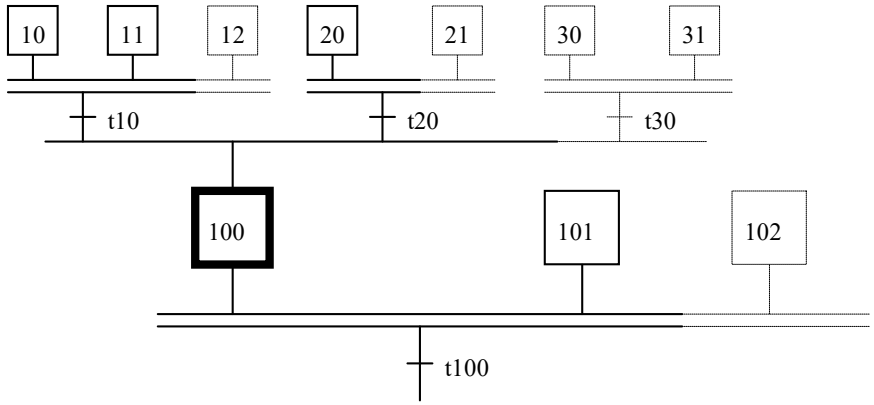
Détail de la partie B2	Opérations	Prog. Siemens
B2 : Actualisation des tables (B21) ; Calcul de l'évolution. (B22)	B21 : Mise à jour de la table Tp $T_s \rightarrow T_p$ B22 : Calcul de la nouvelle table Ts en fonction des valeurs de Tp et des conditions d'évolution.	***L*** T_s ***T*** T_p ***

Remarque : La mise à zéro de la table Ts dans la partie B21 n'est pas indispensable et même peut être indésirable suivant les modes de programmations de B22.
L'exemple de programmation présuppose une table de 8 bits seulement. On adaptera facilement dans le cas d'une table composée de plusieurs mots.

4.2 Schéma général autour d'une étape

Les complexités maximums autour d'une étape peuvent se résumer à deux cas simples

4.2.1



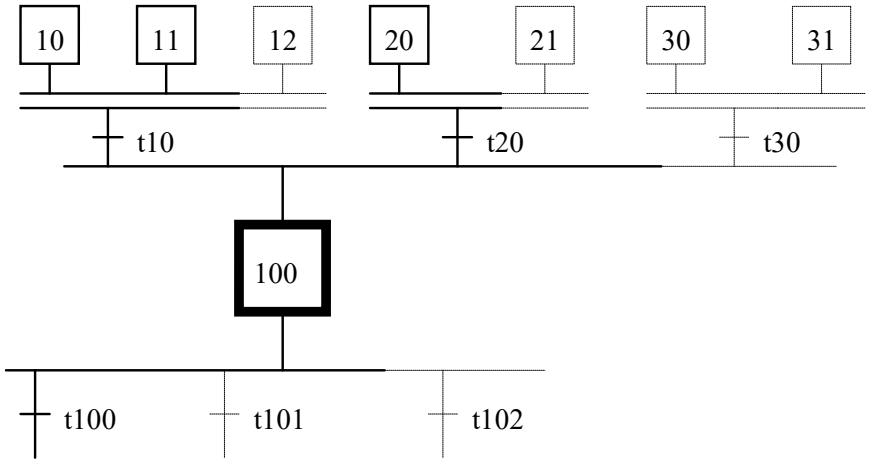
Une étape est active si elle est activée par les étapes antérieures ou si elle reste active.

La première proposition de cette affirmation nous permet de regarder globalement l'amont de l'étape considérée. La deuxième proposition nous amène à examiner les conditions de désactivation de l'étape soit :

si elle est activée	ou	si elle reste active
$(X10.X11.....).t10+(X20.X21...).t20+....$	+	$X100.\left(\overline{(X101.X102...t100)}\right)$

Le résultat de ce calcul est à mettre dans Ts et les valeurs Xn sont à prendre dans Tp.

4.2.2



Une étape est active si elle est activée par les étapes antérieures ou si elle reste active. De même que précédemment, nous avons :

si elle est activée	ou	si elle reste active
$(X10.X11.....).t10+(X20.X21...).t20+....$	+	$X100.\overline{(t100+t101+...)}$

Comme précédemment, le résultat de ce calcul est à mettre dans Ts et les valeurs Xn sont à prendre dans Tp.

4.3 Programmation des équations.

La programmation directe d'une équation compliquée relève (à la relecture) d'un certain mystère. Il est utile d'utiliser un certain nombre de bits intermédiaire de stockage pour permettre une meilleure lisibilité du programme. Ici, nous prendrons le bit M100.7 pour le résultat des conditions d'activation et M100.6 pour le résultat des conditions d'automaintient. Le bit M100.5 sert à éviter un niveau de parenthèses. Soit en langage SIEMENS :

4.3.1 cas du 4.2.1

Programme	***Commentaire***
U X101	*Programmation des*
U X102	*automaintiens*
.....	
U t100	Les Xi correspondent
= M100.5	aux adresses de la
***	table T_p[i].

4.3.2 cas du 4.2.2

Programme	***Commentaires***
U t100	*Programmation des*
O t101	*automaintiens*
.....	
= M100.5	Les Xi correspondent
***	aux adresses de la
UN M100.5	table T_p[i].

UN M100.5 ***U*** X100 = M100.6 ***		***U*** X100 = M100.6 ***	
U X10 ***U*** X11 .. ***U*** t10 ***O*** ***U*** X20 ... ***U*** t20 ***O*** .. .. = M100.7 ***	*Programmation des conditions d'activation*	***U*** X10 ***U*** X11 .. ***U*** t10 ***O*** ***U*** X20 ... ***U*** t20 ***O*** .. .. = M100.7 ***	*Programmation des conditions d'activation*
U M100.6 ***O*** M100.7 = M Ts[100] ***	Mise en commun et constitution de la table T_s	***U*** M100.6 ***O*** M100.7 = M Ts[100] ***	Mise en commun et constitution de la table T_s

Les Bits M100.6 M100.7 M100.5 peuvent être réutilisés dans un autre calcul.

Seul la partie correspondant au bit M 100.5 change entre les deux cas.

4.4 Analyse des règles du GRAFCET.

Le problème de la méthode du 3 due à la perte des informations antérieures est désormais réglé, et, il est facile de montrer que cette méthode respecte *TOUTES* les règles du GRAFCET.

4.5 Mise en oeuvre sur automate SIEMENS.

Nous proposerons ici une mise en oeuvre en utilisant les unités logicielles nommées PB. Pour un graphe, nous utiliserons cinq PB distincts. L'un des PB ne sera appelé qu'une seul fois au lancement du système et ou après tout redémarrage provoqué par une commande stop-run de l'automate. Ce PB permettra de respecter la règle N°1 du GRAFCET concernant les étapes initiales. Pour simplifier la programmation, lorsque une réceptivité est une équation logique et ou le résultat d'une comparaison, il peut être pratique de dissocier le calcul de cette réceptivité et de son utilisation. Nous utiliserons donc un PB spécial (PB n4) pour ces calculs dont les résultats seront stockés dans une table T_t utilisée dans le PB n5. Ce PB peut très bien se résumer à l'ordre BE (fin de PB). Pour une homogénéité de programmation, il serait bon que chaque sous-graphe en possède un même vide.
Nous séparons le traitement des fronts du type $X_n \uparrow$ du graphe (PB n1) de la phase calcul des receptivités des transitions (PB n4). Ce traitement sera effectué en priorité avant tout calcul d'évolution de graphe. Une mécanique inverse ou aléatoire de calcul conduirait à une incertitude dans le calcul d'une transition du type $X_n \uparrow$ (elle fait appel aux tables T_p et T_s avant toutes modifications de celles-ci).

Nom	***Rôle***	***Programmation***
PB n0	Mise à un des étapes initiales si besoin est.	***L*** KH???? ***T*** Ts

Image de la mémoire au lancement de la machine.

		T T_p ***BE***
PB n1	Traitement des fronts de type $X_n\uparrow$ (préliminaire)	
PB n3	Réactualisation du tour précédent	***L*** T_s ***T*** T_p ***BE***
PB n4	Calcul des réceptivités associés aux transitions	***L*** KH0000 ***T*** T_t ... ***BE***
PB n5	Calcul des étapes Cf 4.3	
PB n2	Le maître à danser des PB n3 n4 et n5	***SPA*** PB n3 ***SPA*** PB n4 ***SPA*** PB n5 ***BE***

il est utile de réinitialiser la table de bits avant de l'utiliser.

Les OB (Organisation Block) de démarrages (OB 22 et OB 21) devront posséder au moins la ligne suivante :

SPA PB n0
...
BE

Pour permettre l'initialisation des grafcets et le respect de la règle N°1.

L'OB 1 (Block d'organisation général) doit posséder la ligne d'appel du PB n1 et PB n2 soit

Appel des traitements préliminaires dont
SPA PB n1
Appel du traitement des graphes dont :
SPA PB n2
..
Appel du traitement des forçages
Appel du traitement des compteurs
Appel du lancement des temporisations
Appel du traitement des affectations de sorties et figeages
BE

4.6 Critique

Cette méthode plus rigoureuse mais plus élégante admet difficilement une programmation à vue. En effet, il est indispensable d'écrire pour chaque transition son équation d'évolution associée ainsi que de préparer ses tables de bits. Elle permet néanmoins de programmer de façon efficace et surtout de pouvoir relire son programme au bout de quelques mois.

Une modification du GRAFCET oblige une réétude complète, et parfois un peu lourde en phase de mise au point, des conditions d'activation-desactivation de l'étape.

V Méthode à deux tables et analyse transition par transition

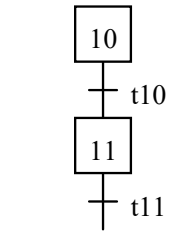
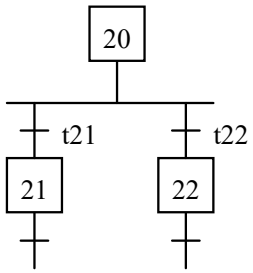
Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

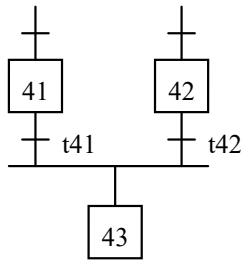
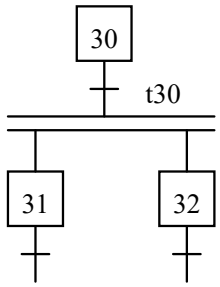
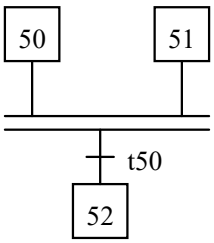
Nous proposerons ici une méthode équivalente à celle proposée dans le chapitre précédent. Dans le chapitre IV, nous avons fait une analyse globale de l'activité de l'étape. Nous allons maintenant nous intéresser à la phase de franchissement des transitions, et, nous allons regarder l'influence de ces franchissements via deux tables : T_a table d'activation et T_d table de désactivation.

Nous modifierons ici la partie B22 citée dans le IV tout en gardant une compatibilité parfaite avec les traitements suivants et précédants cette partie.

5.1 Traitement des transitions

Le franchissement d'une transition provoque simultanément l'activation des étapes suivantes (liées à cette transition) et la désactivation des étapes précédentes. Nous allons stocker ces informations dans les deux tables présentées ci-dessus T_a et T_d. Nous traitons successivement toutes les transitions du graphe.

Exemple de graphe	*Pseudo - code*	Prog. Siemens
	***SI*** t10 et X10 ***ALORS*** *SET* T_a[11] *SET* T_d[10] ***FIN SI***	***U*** t10 ***U*** T_p[10] ***S*** T_a[11] ***S*** T_d[10] ***
	***SI*** t21 et X20 ***ALORS*** *SET* T_a[21] *SET* T_d[20] ***FIN SI*** ***SI*** t22 et X20 ***ALORS*** *SET* T_a[22] *SET* T_d[20] ***FIN SI***	***U*** t21 ***U*** T_p[20] ***S*** T_a[21] ***S*** T_d[20] *** ***U*** t22 ***U*** T_p[20] ***S*** T_a[22] ***S*** T_d[20] ***

	***SI*** t41 et X41 ***ALORS*** *SET* T_a[43] *SET* T_d[41] ***FIN SI*** ***SI*** t42 et X42 ***ALORS*** *SET* T_a[43] *SET* T_d[42] ***FIN SI***	***U*** t41 ***U*** T_p[41] ***S*** T_a[43] ***S*** T_d[41] *** ***U*** t42 ***U*** T_p[42] ***S*** T_a[43] ***S*** T_d[42] ***
	***SI*** t30 et X30 ***ALORS*** *SET* T_a[31] *SET* T_a[32] *SET* T_d[30] ***FIN SI***	***U*** t30 ***U*** T_p[30] ***S*** T_a[31] ***S*** T_a[32] ***S*** T_d[30] ***
	***SI*** t50 et X50 et X51 ***ALORS*** *SET* T_a[52] *SET* T_d[50] *SET* T_d[51] ***FIN SI***	***U*** t50 ***U*** T_p[50] ***U*** T_p[51] ***S*** T_a[52] ***S*** T_d[50] ***S*** T_d[51] ***

5.2 Calcul de l'évolution du graphe

Nous rappelons que : une étape qui est activée et désactivé reste active. Il est facile de montrer que la table d'activité T_s répond à l'équation (globale sur tables ou sur bits) :

$$T_s = T_a \text{ ou } (T_p \text{ et } \overline{T_d})$$

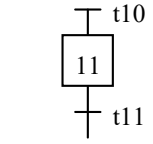
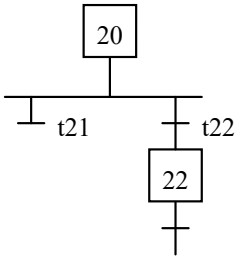
Il est important de remettre les tables d'activation et désactivation à 0 au début du calcul du graphe.

5.3 Traitement des structures Source-Puits

Pour une 'étape' source, il suffit de remplacer dans les calculs l'activité T_p[i] par une ressource toujours à un (ou de l'omettre dans une combinaison et) et d'éliminer la désactivation de l'étape antérieure T_d[i] (C.F. remarque au 12.1).

Pour une fin en puits, il suffit d'omettre d'activer la table d'activation T_a[i+1] en fin de calcul.

Exemple de graphe	***Pseudo-code***	***Prog.Siemens***

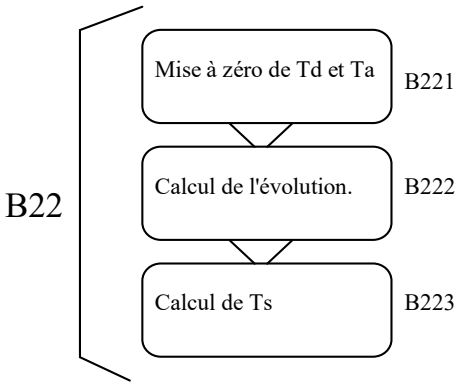
	***SI*** t10 ***ALORS*** *SET* T_a[11] *SET* T_d[10] ***FIN SI*** C.F. 12.1	***U*** t10 ***S*** T_a[11] ***
	***SI*** t21 et X20 ***ALORS*** *SET* T_d[20] ***FIN SI*** ***SI*** t22 et X20 ***ALORS*** *SET* T_a[22] *SET* T_d[20] ***FIN SI***	***U*** t21 ***U*** T_p[20] ***S*** T_d[20] *** ***U*** t22 ***U*** T_p[20] ***S*** T_a[22] ***S*** T_d[20] ***

.

5.4 Mise en oeuvre sur automate SIEMENS

Les opérations de calcul n'étant possibles que dans des blocs de type FB, nous transformerons l'appel au PB105 mis en place dans le IV par un appel au FB105 équivalent Due à l'absence de FB dans l'automate 100U (bas de game), cette méthode n'est pas adaptée à cet automate.

La partie B22 devient donc :

Détail de la partie B22	Programmation SIEMENS FB n5
	***L*** KH 0000 ***T*** T_a ***T*** T_d *** calcul suivant graphe tel que décrit en 5.2 *** ***L*** T_d ***KEW*** ***L*** T_p ***UW*** ***L*** T_a ***OW*** ***T*** T_s ***

5.5 Critique -réflexion

Les deux méthodes proposées en IV et V étant équivalentes, nous pouvons envisager un automatisme avec différents graphes traités indistinctement par la méthode du chapitre N°4 ou celle du chapitre N°5.

Cette méthode bien qu'elle demande deux tables temporaires de traitement supplémentaire apporte une aisance accrue lors de la modification d'un graphe. Il n'est pas nécessaire de réécrire les équations (parfois lourdes) d'activité d'étapes.

Le traitement systématique des transitions permet d'envisager une méthode fiable de compilation du langage GRAFCET.

Les habitués des automates Merlin-Gerin (PB 15 ...) y retrouveront l'équivalent de leurs mnémoniques EANT RCEP EPOS.

Cette méthode peut constituer la base d'un noyau GRAFCET pour un automate futur ou une émulation. L'émulateur d'automate Merlin-Gerin PB15 du même auteur fonctionne suivant ce principe.

Par rapport à la méthode proposée au IV, Seul le PB n5 change.

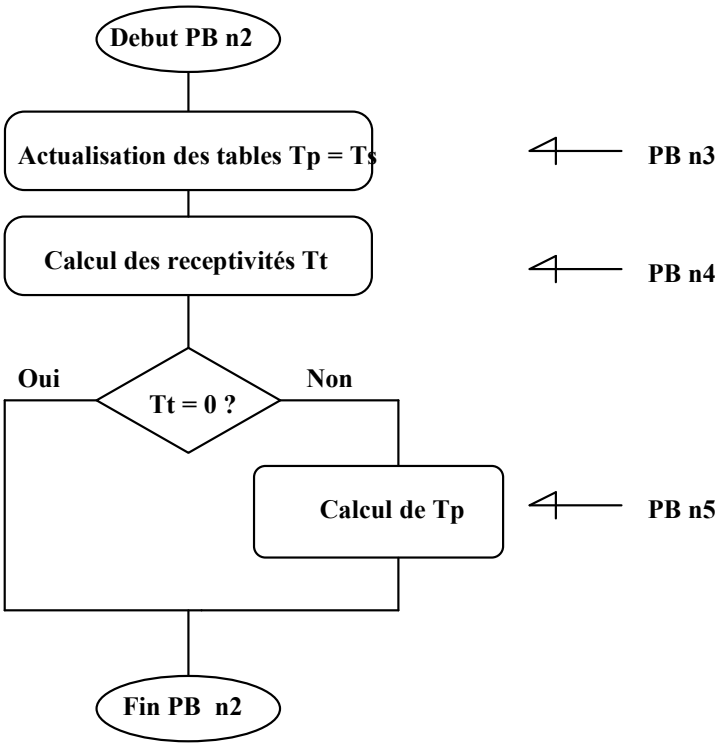
VI Méthode à deux tables, analyse globale et accélération du cycle (Automate SIEMENS).

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

Cette méthode repose en grande partie sur la méthode précédente. Nous pouvons remarquer que dans le cas ou aucune réceptivité du graphe n'est vraie, les conditions d'activations et d'automaintiens des étapes se résument à une recopie pure et simple de la table d'activité précédente.

6.1 Modifications à apporter

Pour une facilité de traitement, il est indispensable de traiter le PB n4 et d'établir une table COMPLETE de toutes les réceptivités associées aux transitions du graphe considéré T_t.



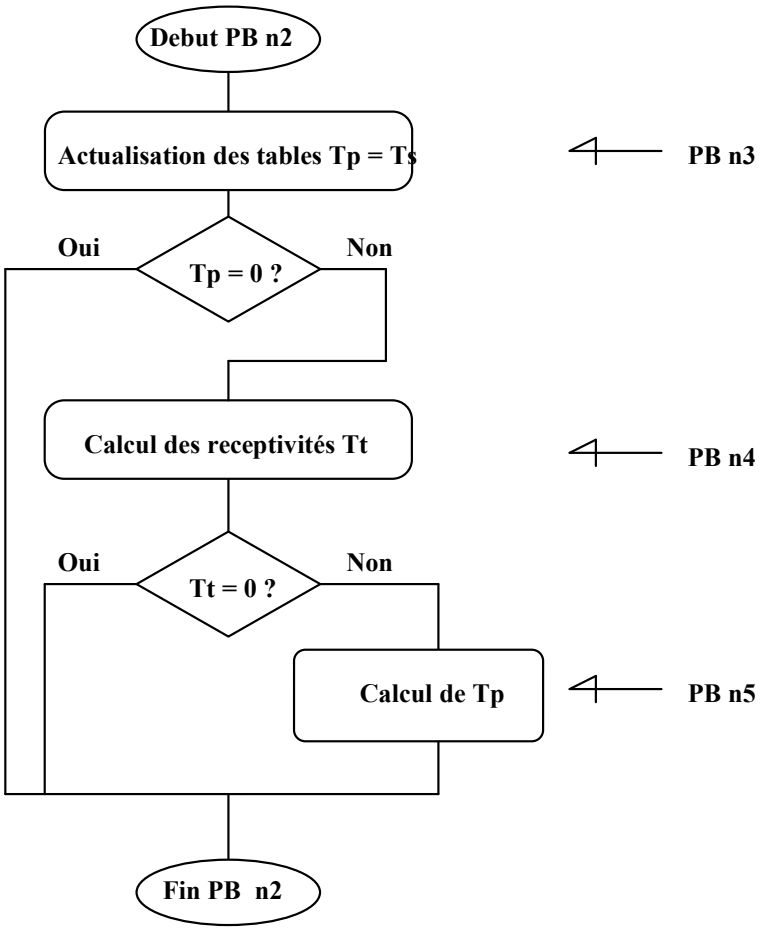
6.2 Programmation SIEMENS correspondante.

La seule partie de programme à modifier correspond au PB n5 soit désormais :

SPA PB n3	*Initialisation du tour*
SPA PB n4	*Calcul des réceptivités*
L T_t	*Comparaison de la table des réceptivités à 0*
L KH0000	
!=F	*Si égalité alors*
BEB	*fin de bloc*
	Traitement normal :
SPA PB n5	*- calcul de la nouvelle table d'étapes cas du IV*
ou	
SPA FB n	*- calcul suivant V*
BE	*Fin de bloc normale*

6.3 Critique améliorations

Ce style de programmation impose une rigueur plus grande dans le calcul et l'utilisation des réceptivités (obligation d'utiliser une table). Pour un automatisme lent ou une CPU rapide, on peut se dispenser de cette accélération du cycle de calcul. Dans le cas ou la rapidité de réponse de l'automatisme est cruciale, il peut être nécessaire d'avoir recours à de tels stratagèmes voire à encore accélérer le système comme suit :



Soit ne traiter le calcul des réceptivités que si une étape du graphe est active.

En programmation SIEMENS, le PB n2 devient alors :

SPA PB n1	*Initialisation du tour*
L T_p	*Comparaison de la table des activités d'étape à* 0
L KH0000	
!=F	*Si égalité alors*
BEB	*fin de bloc*
***	
SPA PB n4	*Calcul des réceptivités*
L T_t	*Comparaison de la table des réceptivités à 0*
L KH0000	
!=F	*Si égalité alors*
BEB	*fin de bloc*
***	*Traitement normal :*
SPA PB n5	*- calcul de la nouvelle table d'étapes cas du IV*
ou	
SPA FB n5	*- calcul suivant V*
BE	*Fin de bloc normale*

VII Les Forçages Figeages

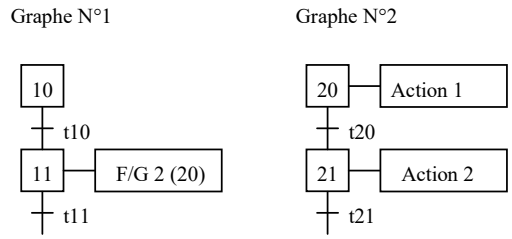
Traitement préliminaire
Traitement des graphes

Traitement des forçages

Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

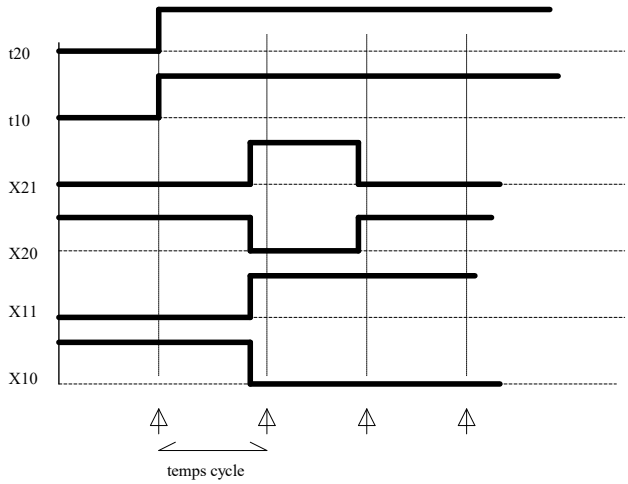
7.1 Dualité Forçage-action, ordre de traitement

Au niveau écriture, un forçage est une action comme une autre. Son traitement ne requière normalement aucunes précautions. Comme nous travaillons avec des automates programmables asynchrone vis à vis des événements extérieurs, nous devons prendre quelques précautions de traitement. Prenons l'exemple suivant :



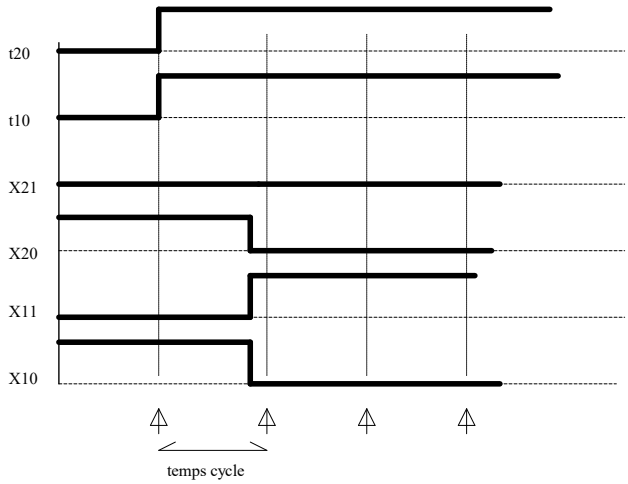
Les étapes 10 et 20 sont actives.
Les réceptivités t10 et t20 deviennent vraies dans le même tour automate.

Cas de traitement N°1



Si nous traitons les sorties avant les forçages (en préliminaire par exemple) l'action 2 associée à l'étape 20 sera effective durant un tour automate (un temps de cycle). Cette configuration peut être gênante voire dangereuse s'il s'agit d'une action de puissance ou d'actionner un périphérique rapide du style compteur électronique externe.

Cas de traitement N°2



Le traitement des forçages avant les actions de sorties, ce qui revient à hiérarchiser le niveau d'importance des actions, permet d'éviter les aléas précédents.

Il convient donc de respecter l'ordre suivant :

1 : Traitement préliminaire
2 : Traitement des graphes

3 : Traitement des forçages

4 : Traitement des compteurs
5 :Lancement des temporisations
6 : Traitement des affectations des sorties et figeages

7.2 Solution simplifiée

Attention, Il y a priorité au forçage à 1. Une étape forcée à 1 par un graphe et à 0 par un autre reste active.

Dans un automatisme simple, il est possible de traiter les forçages d'étapes à la sauce SET-RESET en gérant manuellement tous les problèmes de perte d'information en cours de calcul. Cette méthode est acceptable car il est relativement facile d'avoir une vue globale de tous les graphes et de toutes les étapes.

7.3 Solution systématique

7.3.1 problème théorique

Dans le cas d'un automatisme complexe ou voué à une évolution, nous développerons une méthode systématique. Pour chaque graphe outre les tables T_p et T_s, nous adjoindrons deux autres tables :

T_{f0} : Table des forçages à 0
T_{f1} : Table des forçages à 1

Par graphe, nous ajouterons une unité logicielle qui rempli les tables de forçage à 0 et 1.

Si une étape est forcée à 0, nous mettrons 1 dans le bit correspondant de la table $T_{f0.}$
Si une étape est forcée à 1, nous mettrons 1 dans le bit correspondant de la table $T_{f1.}$

Une étape est active si :

elle à étée calculée comme active et qu'il n'y a pas de forçages à zéro,
ou,
elle est forcée à 1

La nouvelle table d'activité d'étape peut alors se calculer comme suit :

$$T_s \leftarrow T_s . \overline{T}_{f0} + T_{f1}$$

7.3.2 programmation sur automate SIEMENS

Par graphe nous rajoutons un PB n1 d'initialisation des tables de forçage, un PB n6 de traitement des forçages, un PB n7 de calcul de résultat des forçages.
En reprenant 4.5 nous ajoutons le traitement du PB n6 dans l'appel général des PB.
En outre le PB n1 devra être appelé au début du programme cyclique (OB1).

Nom	***Rôle***	***Programmation***
PB n0	Mise à un des étapes initiales si besoin est.	***L*** KH???? ***T*** Ts ***T*** T_p ***BE***
PB n1	Calcul des fronts de type Xi↑ Mise à zéro des tables de forçage	*calcul des fronts* ***L*** KH0000 ***T*** T_{f0} ***T*** T_{f1} ***BE***
PB n3	Réactualisation du tour précédent	***L*** T_s ***T*** T_p ***BE***
PB n4	Calcul des réceptivités associés aux transitions	***L*** KH0000 ***T*** T_t ... ***BE***
PB n5 ou FB n5	Calcul des étapes Cf IV et V	
PB n6	Calcul des tables de forçage	
PB n7	Calcul du résultat des forçages	
PB n2	Le maître à danser des PB n3 n4 n5 et n6	***SPA*** PB n3 ***SPA*** PB n4 ***SPA*** PB n5 ***SPA*** PB n6 ***BE***

Image de la mémoire au lancement de la machine.

UN T_p[i]
U T_s[i]
= front de Xi

Traitement sans accélérations

Le PB n6, dans le cas ou aucuns ordre de forçage n'émane du graphe se réduit à un ordre : BE

Exemple de PB n6

Exemple de graphe	***Tables***	***PB de forçages du graphe N°1***
Graphe N°1 10 — F/G 3 ( ) t10 11 — F/G 2 (20) t11 Graphe N°2 20 t20 21 t21	Graphe N° 1 $T1_p$ $T1_s$ $T1_{f0}$ $T1_{f1}$ Graphe N° 2 $T2_p$ $T2_s$ $T2_{f0}$ $T2_{f1}$ Graphe N° 3 $T3_p$ $T3_s$ $T3_{f0}$ $T3_{f1}$	***U*** X10 ***L*** KHFFFF ***T*** $T3_{f0}$ *** ***U*** X11 ***L*** KHFFFF ***T*** $T2_{f0}$ ***S*** $T2_{f1}$[20] ***BE***

Le PB n7 respecte l'équation du 7.3.1
soit en travaillant avec des opérations sur bytes :

L T_{f0}

OW
KEW négation du résultat T_{f0}
L T_s
UW et bit a bit
L T_{fl}
OW ou bit a bit
T T_s
***

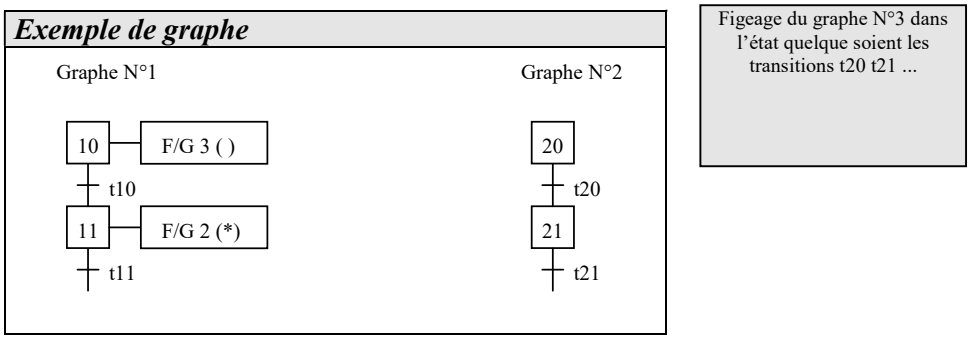
L'OB 1 (Block d'organisation général) doit être complété par l'appel du bloc de calcul des forçages

Appel des traitements préliminaires dont la mise à zéro des tables de forçages
SPA PB n1

Appel du traitement des graphes

Appel du calcul des forçages dont :
SPA PB n7 calcul de la table d'activité d'étape compte tenu des forçages
...
Appel du traitement des compteurs
Appel du traitement des affectations de sorties
BE

7.4 Les figeages de graphe



Cette possibilité du langage GRAFCET est une des moins utilisée car peu évidente à mettre en oeuvre, et peu souvent implémenté sur nos automates orientés GRAFCET. Pour figer un grafcet, il suffit de suspendre momentanément les calculs d'évolution du graphe considéré. Un graphe figé ne peut plus évoluer par le biais des recptivités. Il peut par contre évoluer par forçages. Nous développerons ici une solution dans le cadre de l'implémentation sur automate SIEMENS.

7.4.1 Principe

Nous associons à tous les graphes une table supplémentaire d'autorisation d'évolution du graphe soit T_e . Les calculs d'évolution (PBn4 n5) seront désormais assujettis au bit correspondant au graphe de la table.

SI graphe N°n non figé ***ALORS***
Calcul des transitions
Calcul des étapes
FIN SI

La mise à jour de la table, sera traitée dans le traitement des affectations de sortie.

Bit d'autorisation du graphe N° n = figeage graphe N°n demandé

7.4.2 Programmation SIEMENS

Ce traitement amène les modifications suivantes dans les PB des graphes (avec n numéro du graphe considéré):

Nom	***Rôle***	***Programmation***
PB n0	Mise à un des étapes initiales si besoin est. Autorisation d'évolution du graphe (défigeage).	***L*** KH???? ***T*** Ts ***T*** T_p *** ***U*** M0.0 ***R*** T_e[graphe] ***BE***
PB n3	Réactualisation du tour précédent	***L*** T_s ***T*** T_p ***BE***
PB n4	Calcul des réceptivités associés aux transitions	***L*** KH0000 ***T*** T_t .. .. ***BE***
PB n5 ou FB n5	Calcul des étapes Cf 4.3	
PB n1	Traitement des fronts Mise à zéro des tables de forçage	***L*** KH0000 ***T*** T_{f0} ***T*** T_{f1} ***BE***
PB n6	Calcul des tables de forçage	
PB n7	Calcul du résultat des forçages	
PB n2	Le maître à danser des PB n3 n4 n5 et n6 avec figeage et sans accélérations.	***SPA*** PB n3 ***UN*** T_e[graphe] ***SPB*** PB n4 *** ***UN*** T_e[graphe] ***SPB*** PB n5 *** ***SPA*** PB n6 ***BE***

C.F. remarque au paragraphe 12.1 sur les OB21 et OB22

L'unité logicielle de traitement des sorties devra (dans le cas de l'exemple introductif du 6.4) contenir les lignes suivantes

U $T1_s$[11]
= M T_e[2]

***

VIII Les compteurs

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

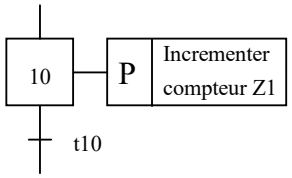
Dans cette partie, nous supposerons que les traitements antérieurs ont été effectués à l'aide d'au moins les deux tables T_p et T_s.

8.1 Problème général des opérations sur compteurs

Un travail sur compteur est par essence une action impulsionnelle liée à l'activité de l'étape. Nous devons donc la traiter comme tel. Le front montant de l'activité de l 'étape peut se calculer facilement à l'aide des deux tables T_p et T_s (état au tour précédent et état calculé pour le tour courant) comme suit :

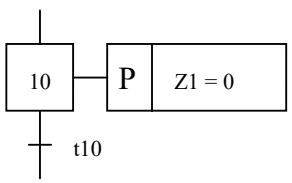
$$X10\uparrow = \overline{T_p[10]} \quad et \quad T_s[10]$$

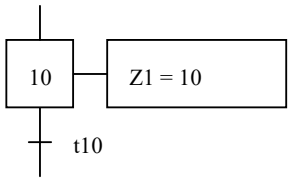
8.2 incrémentation et decrementation

Représentation GRAFCET	***Pseudo - code***	***Programmation SIEMENS***
	***SI*** X10↑ ***ALORS*** ***INC*** Z1 ***FIN SI***	***UN*** T_p[10] ***U*** T_s[10] ***ZV*** Z1 comptage ou ***ZR*** Z1 décomptage ***

8.3 Initialisation

Une Initialisation de compteur peut suivant le graphe considéré être une action impulsionnelle ou une action inconditionnelle. Les opérations simultanées Initialisation et de travail sur compteur sont par expérience des cas très rares. Nous ne développerons donc pas de méthode systématique pour traiter ces cas.

Représentation GRAFCET	***Pseudo - code***	***Programmation SIEMENS***
	***SI*** X10↑ ***ALORS*** Z1 = 0 ***FIN SI***	***UN*** T_p[10] ***U*** T_s[10] ***L*** KZ 0 ***S*** Z1 ***

	***SI*** X10 ***ALORS*** Z1 = 10 ***FIN SI***	***U*** T_s[10] ***L*** KZ 10 ***S*** Z1 ***

8.4 Remarque

Sur la majorité des automates, les actions sur compteurs sont des " boites noires " dans lesquelles la notion de front montant de l'entrée est déjà prise en compte. La programmation proposée ci dessus peut alors se simplifier.

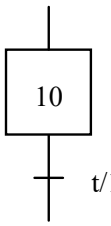
Pseudo - code	***Programmation SIEMENS simplifiée***
SI X10↑ ***ALORS*** ***INC*** Z1 ***FIN SI***	***U*** T_s[10] ***ZV*** Z1 comptage ou ***ZR*** Z1 décomptage ***

Dans le cas de travail sur des compteurs 'Home made', à partir de mots, la programmation initiale est à conserver. C'est pourquoi par soucis de compatibilité entre automates et par systématisation de programmation la méthode n'utilisant pas les fronts intégrés est préférable (les utilisateurs des premières versions de SCOLA7 qui péchait en la matière ne me contrediront pas !).

IX Lancement des temporisations

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

Cette partie s'apparente beaucoup au traitement des compteurs (Initialisation). En effet une temporisation monostable est souvent activée sur un front montant d'une activité d'étape. Nous ne reprendrons donc ici que des exemples de programmation. Nous supposerons comme précédemment que le traitement des étapes à été effectué à l'aide des deux tables T_p et T_s.

Représentation GRAFCET	*Pseudo - code*	*Programmation SIEMENS*
 utilisation de la temporisation automate T3	***SI*** X10↑ ***ALORS*** T3 = 20s ***FIN SI***	***UN*** T_p[10] ***U*** T_s[10] ***L*** KT 200.1 ***SV*** T3 *** l'utilisation de la temporisation dans les unités logicielle PB n2 ou PB n3, se fera en testant que T3 est repassé à 0. Pour la transition t10 : ***UN*** T3 = t10 ***

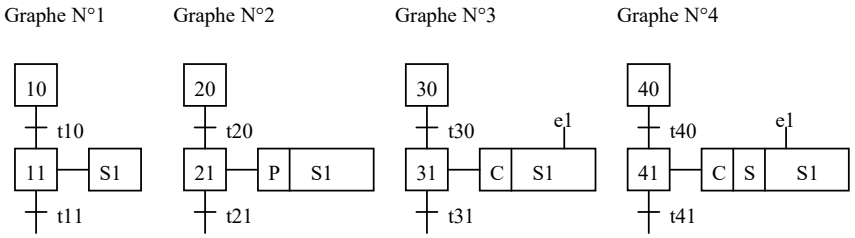
X Affectation des sorties

Traitement préliminaire
Traitement des graphes
Traitement des forçages
Traitement des compteurs
Lancement des temporisations
Traitement des affectations des sorties et figeages

10.1 Généralité

Cette partie est en fait relativement simple. Il suffit d'examiner une à une toutes les sorties et d'écrire l'équation logique régissant son activité. Dans l'exemple ci dessous, les équations traduisant la sortie sont :

Sto = S1
Si X41 et e1 Alors Set Sto
S1 = (X11 + X31 . e1 + X21↑)+(Sto.X41)



S1 est une action conditionnée dans la description de l'étape 31, correspond à une sortie impulsionnelle dans l'étape 21 et est considérée comme bistable et conditionnée dans l'action associée à l'étape N°41.

Dans cette partie, il importe de mettre à jour les bits mémoire pour le calcul des fronts correspondant aux événements en entrée de l'automate.

10.2 Programmation SIEMENS correspondante

Le PB d'affectation des sorties correspondant à l'exemple ci-dessus devra posséder les lignes suivantes :

```
U  S1
= Sto
U  X41
U  e1
S  Sto
```

```
U  X11
O(
UN T2p[21]
U  T2s[21]
)
O(
U  T3s[31]
U  e1
)
```

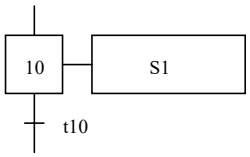
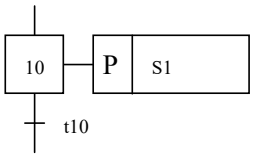
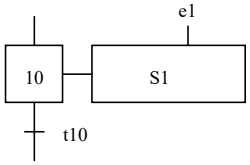
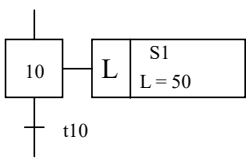
```
O(
U  X41
U  Sto
)
= S1
***
```

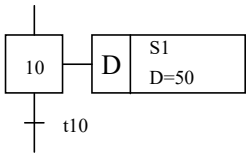
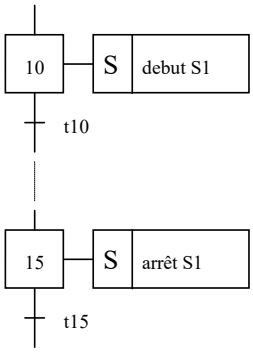
Une partie réactualisation des bits peut se présenter ainsi

```
U  E32.2
=  M2.0
***
```

10.3 Traitement des différents types de sorties

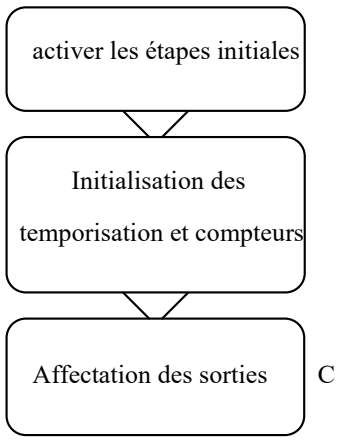
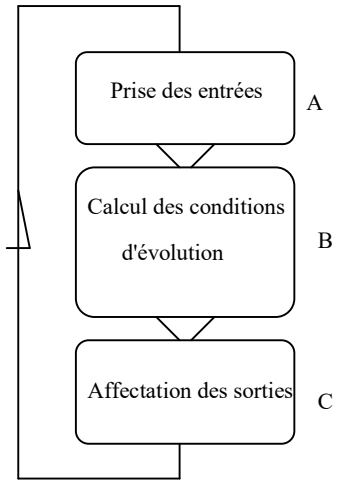
Dans les exemples proposés ci-après, seul l'étape 10 mets en oeuvre l'action S1.

Représentation GRAFCET	***Pseudo - code***	***Programmation SIEMENS***
Action inconditionnelle 	***SI*** X10 ***ALORS*** S1 = 1 ***SINON*** S1 = 0 ***FIN SI***	***U*** T_s[10] **=** S1 ***
Action impulsionnelle (pulse) 	***SI*** X10↑ ***ALORS*** S1 = 1 ***SINON*** S1 = 0 ***FIN SI***	***UN*** T_p[10] ***U*** T_s[10] **=** S1 ***
Action conditionnée 	***SI*** X10 et e1 ***ALORS*** S1 = 1 ***SINON*** S1 = 0 ***FIN SI***	***U*** T_s[10] ***U*** e1 **=** S1 ***
Action limité dans le temps (limited) 	***SI*** X10↑ ***ALORS*** lancer tempo T2 ***FIN SI***	***U*** T_s[10] ***L*** KT 500.1 ***SI*** T2 ***U*** T2 **=** S1 ***

	S1=X10 et (T2 <>0)	
Action retardée (delay) 	***SI*** X10↑ ***ALORS*** lancer tempo T2 ***FIN SI*** S1=X10 et (T2 =0)	***U*** T_s[10] ***L*** KT 500.1 ***SE*** T2 ***U*** T2 = S1 ***
Action mémorisé (set) 	***SI*** X10 ***ALORS*** Set S1 ***FIN SI*** ***SI*** X15 ***ALORS*** Reset S1 ***FIN SI***	***U*** T_s[10] ***S*** S1 *** ***U*** T_s[15] ***R*** S1 ***

XI Problème de l'initialisation du processus.

Lors de la mise en route de l'automate, il est nécessaire d'initialiser les étapes initiales (règle N°1) puis d'affecter une première fois les sorties en fonction de ces étapes (marche de préparation de l'automate). L'affectation de ces sorties peut se faire par l'intermédiaire de la même unité logicielle que lors du traitement normal. En ce qui concerne les temporisations et compteurs, il peut être nécessaire, lors de l'initialisation ou du redémarrage à la suite d'une opération 'stop-run' de l'automate, de les remettre à 0.

Préparation	***Fonctionnement normal***
	

Cette partie étant étroitement liée à l'automate, nous proposerons une mise en oeuvre pour automate SIEMENS dans le chapitre suivant.

XII Structure générale logicielle sur automate SIEMENS

12.1 Cas général

Nous proposerons ici une structure logicielle intégrant toutes les particularités décrites dans les chapitre précédents .

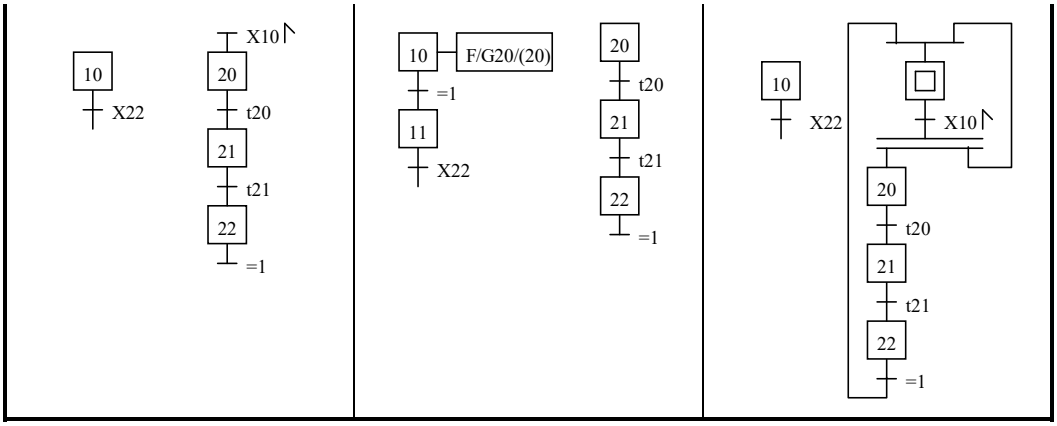
Pour le graphe N°n nous aurons défini les huits PB (s) ou FB (s) suivants

Nom	***Rôle***
PB n0	Mise à un des étapes initiales si besoin est. Défigeage du graphe.
PB n1	Fronts d'activité d'étapes + initialisation des tables de forçage du graphe.
PB n3	Reactualisation du tour précédent
PB n4	Calcul des receptivités (Table T_t).
PB n5 ou FB n5	Calcul des étapes Cf IV et V
PB n6	Calcul des tables de forçage
PB n7	Calcul du résultat des forçages

PB n2	Le maître à danser des PB n3 n4 n5 et n6	***SPA*** PB n3 ***SPA*** PB n4 ***SPA*** PB n5 *ou* ***SPA*** FB n5 ***BE***	***SPA*** PB n3 *** ***UN*** T_e[n] ***SPB*** PB n5 *ou* ***SPB*** FB n5 *** ***SPA*** PB n6 ***BE***	***SPA*** PB n3 *** ***UN*** T_e[n] ***SPB*** PB n4 *** ***L*** T_t ***L*** KH 0000 ***!= F*** *= mémoire* *** ***UN*** T_e[n] ***UN*** mémoire ***SPB*** PB n5 *ou* ***SPB*** FB n5 *** ***SPA*** PB n6 ***BE***	***SPA*** Pbn3 *** ***L*** T_p ***L*** KH 0000 ***!= F*** ***BEB*** *** ***UN*** T_e[n] ***SPB= fin*** ***SPA*** Pbn4 ***L*** T_t ***L*** KH 0000 ***!= F*** ***SPB= fin*** ***SPB*** PB n5 *ou* ***SPB*** FB n5 ***fin :*** ***SPA*** PB n6 ***BE***
		le plus simple	***avec figeages***	***figeages et accceleration partielle. Prog. en PB***	***figeages et accceleration complète. Prog. en FBn2***

Remarque concernant les structures Source-Puits

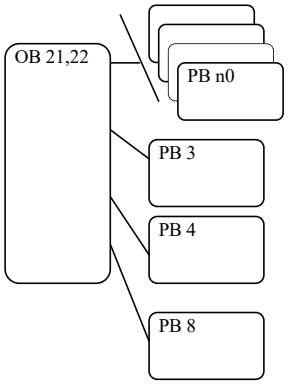
Solution N°1	***Solution N°2***	***Solution N°3***



Les trois solutions ci-dessus sont équivalentes du point de vue du fonctionnement. La solution N°1 ne peut être traitée par la méthode 'accélération complète'. En effet si la branche grafcet de droite n'a aucunes étapes actives, T_p=0 provoque l'éviction du traitement du code correspondant à cette branche. La solution N°2 ne pose pas de problèmes et, sera donc préférée. La solution N°3 utilise la règle N°5 du grafcet et peut être traitée indifféremment par l'une ou l'autre méthode. Elle est néanmoins plus compliquée.

Les OB de démarrage OB21 et OB22

Ces OB répondent au problème évoqué correspondent au problème évoqué en XI.
Leurs rôles sont de réinitialiser les tables d'automate ils contiennent l'appel aux PB d'initialisation des étapes du grafcet suivi des initialisation des ressources communes (compteurs temporisations ...) puis une première affectation des sorties.

Structure	***Rôle***	***Programmation***
	- Création d'une ressource 1 permanente (C.F. remarque) - Initialisation de chaque graphe (PB10, PB20, ...) - Initialisation des compteurs. - Initialisation des temporisateurs. - Affectation du combinatoire de sortie.	***UN*** M0.0 ***S*** M0.0 ***SPA*** PB10 ***SPA*** PB20 **...** ***SPA*** Pbn0 ***SPA*** PB3 ***SPA*** PB4 ***SPA*** PB8 ***BE***

Remarque :

Les instructions Set et Reset sont toujours conditionnée au *RLG* qui est le résultat de l'opération précédente. Pour effectuer un forçage inconditionnel à 0 ou à 1 d'un bit, nous avons plusieurs solutions :

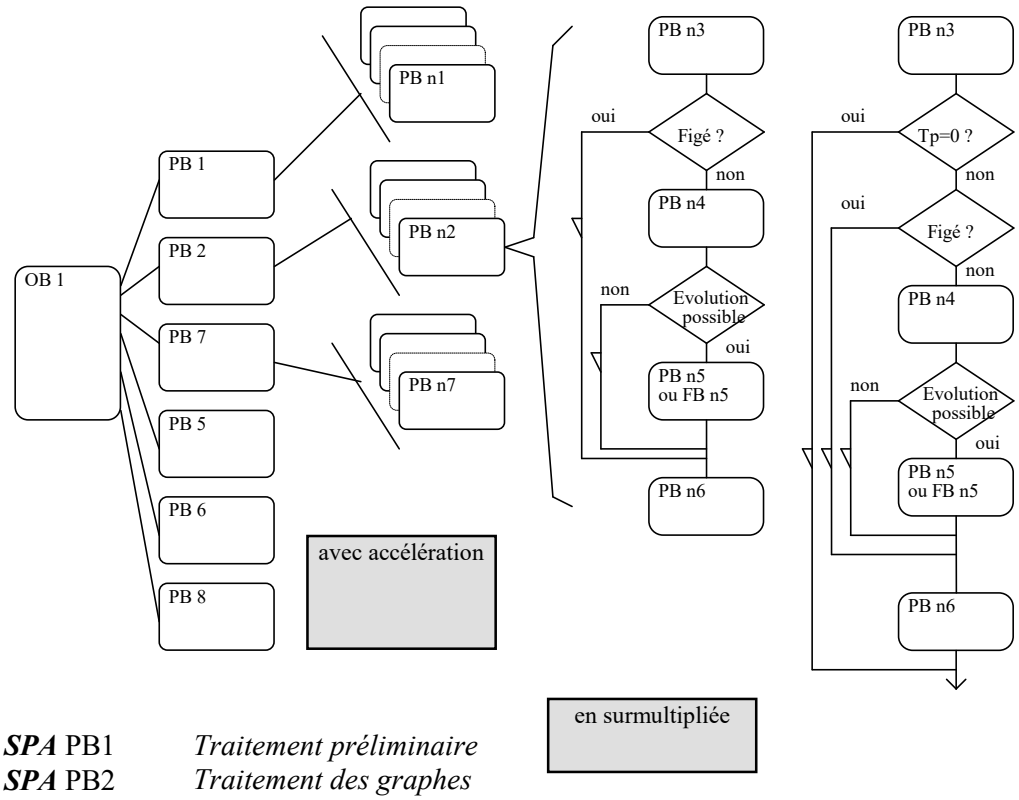
Set	bit =M0.0 ***U*** M0.0 = bit	***SI*** vrai ***ALORS*** ***SET*** bit	***U*** M0.0 ***S*** bit ***U*** M0.0	***SI*** $\overline{\text{bit}}$ =1 ***ALORS*** ***SET*** bit	***UN*** bit ***S*** bit

Reset	bit = $\overline{M0.0}$ ***UN*** M0.0 = bit	***SI*** vrai ***R*** bit ***ALORS*** ***RESET*** bit	***SI*** bit =1 ***U*** bit ***ALORS*** ***R*** bit ***RESET*** bit

Les deux premières solutions nécessitent l'utilisation d'un bit ressource toujours à 1 M0.0 mis à un au démarrage de l'automate par les structures logicielles OB21 ou OB22.

L' OB d'organisation général OB1

Il contient l'appel de tous les PB constituant l'ossature de la méthode



SPA PB1 *Traitement préliminaire*
SPA PB2 *Traitement des graphes*
SPA PB7 *Traitement des forçages*
SPA PB5 *Traitement des compteurs*
SPA PB6 *Lancement des temporisations*
SPA PB8 *Traitement des affectations des sorties et*
BE *figeages*

Le traitement préliminaire *PB1*

SPA PB n4	*Par graphe :*
..	*fronts*
BE	*mise à zéro des tables de forçage.*

Le traitement des graphes *PB2*

SPA PB n2	*Traitement des graphes*

..	*graphe par graphe.*
BE	

Le traitement des forçages *PB7*

SPA PB n7	*Traitement des équations*
..	*finales des forçages pour*
BE	*chaque graphe.*

Le traitement des compteurs *PB5*

..	*Traitement général des*
BE	*compteurs*

Le Lancement des temporisations *PB6*

..	*Traitement général des*
BE	*temporisations*

-

Le traitement des affectations des sorties *PB8*

..	*Traitement général des*
BE	*sorties et figeages*

L'initialisation des compteurs *PB3*

..	*Initialisation des compteurs.*
BE	

L'initialisation des temporisations *PB4*

..	*Raz des temporisations.*
BE	

12.2 Automatisme simple

Souvent sur des petits automatismes, il n'est pas nécésssaire d'implementer les structures de forçage et figeages et accélérations de cycle. La structure générale peut alors se simplifier (en conservant les numéros de PB) comme suit

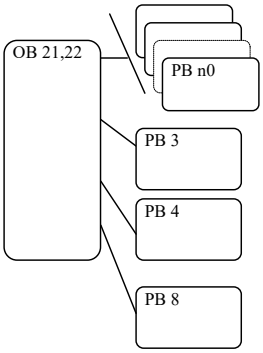
Pour le graphe N°n nous aurons défini les PB suivants

Nom	***Rôle***
PB n0	Mise à un des étapes initiales si besoin est.
PB n1	Fronts d'activité d'étapes du graphe
PB n3	Réactualisation du tour précédent
PB n4	Calcul des réceptivités associés aux transitions
PB n5 ou	Calcul des étapes Cf IV et V

FB n5		
PB n2	Le maître à danser des PB n3 n4 et n5.	***SPA*** PB n3 ***SPA*** PB n4 ***SPA*** PB n5 ou ***SPA*** FB n5 ***BE***

Les OB de démarrage OB21 et OB22

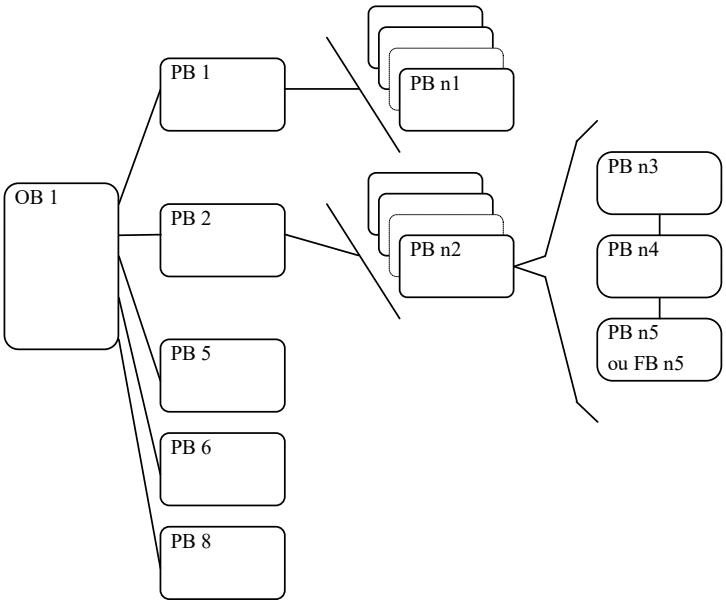
Ces OB ne subissent aucunes modifications par rapport à la solution complete

Structure	***Rôle***	***Programmation***
	- Création d'une ressource à 1 (C.F. remarque du 12.1) - Initialisation de chaque graphe (PB10, PB20, ...) - Initialisation des compteurs. - Initialisation des temporisateurs. - Affectation du combinatoire de sortie.	***UN*** M0.0 ***S*** M0.0 ***SPA*** PB10 ***SPA*** PB20 .... ***SPA*** PB3 ***SPA*** PB4 ***SPA*** PB8 ***BE***

.

L' OB d'organisation général OB1

Il contient l'appel de tous les PB constituant l'ossature de la méthode Il se simplifie légèrement comme suit:



SPA PB1 *Traitement préliminaire*

SPA PB2 *Traitement des graphes*
SPA PB5 *Traitement des compteurs*
SPA PB6 *Lancement des temporisations*
SPA PB8 *Traitement des affectations des sorties et*
BE *figeages*

Le traitement préliminaire *PB1*

SPA PB n1
..
BE

Le traitement des graphes *PB2*

SPA PB n2 *Traitement des graphes*
.. *graphe par graphe.*
BE

Le traitement des compteurs *PB5*

.. *Traitement général des*
BE *compteurs*

Le Lancement des temporisations *PB6*

.. *Traitement général des*
BE *temporisations*

-

Le traitement des affectations des sorties *PB8*

.. *Traitement général des*
BE *sorties et figeages*

L'initialisation des temporisations *PB7*

.. *Raz des temporisations.*
BE

L'initialisation des compteurs *PB7*

.. *Initialisation des compteurs.*
BE

12.3 Résumé des tables mises en jeu

12.3.1 Les tables générales à tous les graphes :

Nom	***Rôle***
T_t	Table des transitions
T_e	Table d'autorisation d'évolution (figeages)
T_a	Table d'activation des étapes
T_d	Table de désactivation des étapes

La table T_t correspond en fait à une table transitoire utilisée dans le calcul d'évolution du graphe. Elle est mise à zéro avant chaque nouvelle utilisation par le graphe en cours de traitement. Il en est de même pour T_a et T_d . Ces deux dernières tables sont utilisées dans le cas d'une programmation des transitions (Chapitre V).

12.3.2 Les tables utilisées par graphe :

Nom	***Rôle***
T_p	Table d'activité du tour précédent
T_s	Table d'activité du tour suivant
T_{f0}	Table des forçages à 0
T_{f1}	Table des forçages à 1

XIII Structure mémoire

Les automates SIEMENS série 100 102 103 95u possèdent au minimum une mémoire adressage de 128 mémento de 8 bits M0 à M127, 64 PB, 64 FB. Pour une clarté accrue lors de la relecture des graphes, il est de bon ton de 'normaliser' les tables d'activité d'étapes par rapport au numéro des graphes. Nous proposerons ci dessous une structure mémoire qui permet de satisfaire les cas simples. Pour un automatisme plus complexe, on utilisera une CPU de modèle 103 ou 95U qui outre sa rapidité de calcul, possède une mémoire interne plus importante. L'architecture mémoire utilisée est issue de celle proposée ci-dessous.

Possibilités :

6 Graphes.
16 Etapes par graphe.
32 Transitions par graphe.

Mémoire utilisé pour les tables de figeages des graphes

Table	***Mots mémoire associés***
$\mathbf{T_e}$	M0

M0.0 ressource à 1

Graphe 10 : M0.1
Graphe 20 : M0.2
....

Mémoire utilisé pour les graphes

Table	***Mots mémoire associés***
$\mathbf{Tn_s}$	M n0, M n1
$\mathbf{Tn_p}$	M n2 M n3
$\mathbf{Tn_{f0}}$	M n4 M n5
$\mathbf{Tn_{f1}}$	M n6 M n7

Il est important que les bits de même rang dans les mots correspondent aux mêmes étapes pour pouvoir effectuer des calculs globaux sur les mots C. F. PB n6.

Mémoire utilisé pour les tables de transition

La table T_t peut être commune à tous les graphes (elle est remise à zéro puis recalculé dans chaque PB n4).

Table	*Mots mémoire associés*
$\mathbf{T_t}$	M120 M121 M122 M123

Mémoire utilisé pour les tables d'activation désactivation

Si nous utilisons la méthode décrite au V, il est nécessaire de réserver en mémoire deux tables T_a et T_d commune à tous les graphes (elles sont remises à zéro puis recalculé dans chaque PB n5 ou FB n5).

Table	*Mots mémoire associés*
$\mathbf{T_d}$	M124 M125
$\mathbf{T_a}$	M126 M127

Mémoire utilisée pour les calculs de fronts et résultats des calculs préliminaires

M1 M2 M3 M4 M5 M6 M7 M8 M9

Soit 9x8 = 72 bits (largement suffisant pour les applications courantes).

Mémoire utilisée pour les bits intermédiaires de calculs

M120 M121 M122 M123

XIV Utilisation des structures de calcul de type FB (SIEMENS)

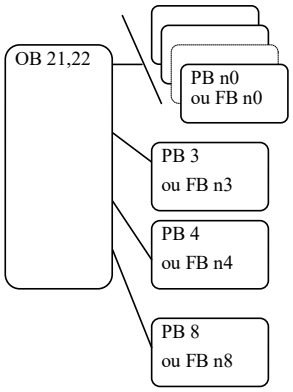
14.1 Présentation

La gamme des CPU SIEMENS, outre leurs vitesses de traitement différentes possèdent des possibilités de plus en plus étendues. Nous discuterons ici des blocs fonctionnels de type FB.

Ces blocs sont en fait une amélioration des structures de type PB. Tout ce qui est écrit en PB peut être écrit en FB. Toutes les instructions de calcul sur mots sont accessibles en FB et non en PB. Il existe des instructions de saut avant dans les FB et non dans les PB. Les FB (sur CPU 103 et 95U) sont paramétrables et possèdent un nom (au sens 'rabelaisien' et non 'infomagicien' du terme) qui nous permet d'éclairer la programmation. Pour toutes ces raisons, il est possible et même souhaitable de transformer la multitude de PB cité dans les quelques (oui je sais j'ai été bavard !) lignes ci dessus en FB si l'on utilise une CPU 103 ou 95U.

14.2 Exemple

Nous donnerons ici l'exemple de l'implémentation de l'OB de démarrage OB21,22. Le lecteur fera de lui-même l'adaptation pour les autres unités logicielles.

Structure	***Rôle***	***Prog. en PB***	***Prog. en FB***
	- Initialisation de chaque graphe - Initialisation des compteurs. - Initialisation des temporisateurs. - Affectation du combinatoire de sortie.	***UN*** M0.0 ***S*** M0.0 ***SPA*** PB10 ***SPA*** PB20 .... ***SPA*** PB3 ***SPA*** PB4 ***SPA*** PB8 ***BE***	***UN*** M0.0 ***S*** M0.0 ***SPA*** FB10 ***nom :*** initG1 ***SPA*** FB20 ***nom :*** initG2 .... ***SPA*** FB3 ***nom :*** raz compteur ***SPA*** FB4 ***nom :*** raz tempo ***SPA*** FB8 ***nom :*** Actions ***BE***

Nous avons déjà présenté l'implémentation des Pbn2 en utilisant les FB dans le chapitre 12.1

XV Conclusion

Ces quelques lignes (que je voulais au départ succinctes) se sont transformées au fil de l'écriture en un plaidoyer indigeste, je m'en excuse.

Malgré diverses remises en forme, il m'a été impossible de dérouler le fil de mon exposé de manière continue surtout en ce qui concerne les exemples de programmation. Une lecture en deux temps sera sans doute utile pour comprendre les tenants et aboutissants des OB PB FB et conséquences sur la programmation.

Les idées développées dans les pages précédentes sont issues d'une analyse du fonctionnement du coeur d'un automate (le très célèbre PB15) lorsque j'en ai écris un clone fonctionnant sur P.C. La méthode proposée peut paraître lourde à mettre en place pour un automatisme simple (1 ou 2 graphes) mais, possède l'avantage de la systématisation et du partage du problème global en tâches simples facilement compréhensibles une à une. Elle peut donc être utilisée dans l'implémentation d'un automatisme un peu complexe (Graphe d'AU, Graphe de conduite, divers Graphes de fonctionnement, Graphes indépendants de plusieurs parties de machines) réalisé par une ou plusieurs personnes simultanément. Sa structure modulaire et systématique permet un développement partie par partie, un remplacement de tout module par un équivalent du point de vue entrée-sortie (excusez l'informaticien qui revient au galop).

Enfin je me permet de remercier les collègues automaticiens qui ont bien voulu lire ma prose (très peu shakespearienne) et indigeste et apporter leurs critiques et améliorations.