N° d'ordre : 766

50376
1979
121

THÈSE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNIQUES DE LILLE

pour obtenir le titre de

DOCTEUR DE 3ème CYCLE

SPECIALITE ELECTRONIQUE - OPTION ELECTROTECHNIQUE

par

Jean DEFRENNE

IMPLANTATION DE RESEAUX DE PETRI SUR AUTOMATE BIPROCESSEUR A HAUTE SURETE DE FONCTIONNEMENT

Soutenue le 29 juin 1979 devant la Commission d'Examen

MM. MAIZIERES	Président
TOULOTTE	Rapporteur - Directeur du Travail
THELLIEZ	Examinateur
GENTINA	Examinateur
MANESSE	Directeur du Travail
LANDAS	Invité

Je prie Monsieur le Professeur MAIZIERES, Directeur du Laboratoire d'Electrotechnique, d'accepter mes plus vifs remerciements pour l'intérêt qu'il a témoigné à mes activités et pour son soutien permanent.

Je remercie pour l'aide efficace qu'ils m'ont apportée, Messieurs TOULOTTE, Maître de conférences au Laboratoire d'Automatique, et MANESSE, Maître assistant au Laboratoire d'Electrotechnique, qui ont assuré l'encadrement de ce travail.

Mes remerciements vont également à Messieurs les membres du jury qui ont bien voulu s'interesser à cette étude.

Je ne saurais terminer sans remercier Monsieur FRANCHAULT, technicien du Laboratoire d'Electrotechnique, pour sa collaboration à la réalisation de l'automate.

I N T R O D U C T I O N

Un automatisme logique comprend deux éléments qui coopérent. L'un est appelé partie opérative, l'autre partie commande.

Par exemple, dans une machine outil numérique, la partie opérative est la machine proprement dite, la partie commande est constituée par l'équipement de traitement numérique.

La partie opérative est chargée d'effectuer les tâches pour lesquelles l'équipement a été prévu (usinage dans notre exemple); c'est donc la partie exécutante du système. Les actionneurs (moteurs, vérins...) exécutent les ordres reçus de la partie commande.

Le rôle de celle-ci est de prendre des décisions en fonction des informations reçues de l'extérieur et des comptes rendus issus de l'appareillage de mesure chargé de suivre les évolutions de la partie opérative. Les décisions sont prises en vertu de spécifications élaborées lors de l'établissement du cahier des charges par le maître d'oeuvre.

Parmi les objectifs de l'industrie, nous trouvons: l'augmentation des cadences de production, l'amélioration de la qualité, la suppression des tâches ingrates. Ceci conduit bien souvent à des processus sophistiqués ayant des exigences au niveau de la commande de plus en plus lourdes. L'utilisation des composants électroniques modernes permet de suivre cette évolution grâce à ses possibilités de traitement, son faible encombrement et sa fiabilité. Il apparaît que les opérateurs logiques électroniques, très rapides, effectuent des traitements élémentaires en quelques dizaines de ns. Ils sont utilisés dans des systèmes câblés pour exécuter des tâches dont la répétitivité est généralement très faible.

A partir de cette constatation, il est normal de décomposer le traitement en tâches élémentaires exécutées les unes après les autres en séquence par un opérateur universel. Nous débouchons, tout naturellement, sur les systèmes de commande programmés, dont le coût se trouve nettement diminué depuis l'apparition sur le marché des microprocesseurs. Les systèmes programmés ont l'avantage de pouvoir s'adapter aux modifications du cahier des charges, ce qui est toujours délicat en câblés. Cette facilité d'adaptation permet même d'envisager l'utilisation d'un équipement standard pour réaliser la partie commande de systèmes très différents. Ceci permet d'abaisser les coûts de réalisation et d'exploitation des systèmes de commande.

Notre étude tente de montrer dans une première partie comment il est possible d'associer des microprocesseurs ayant des champs d'application différents pour réaliser la partie commande des automatismes logiques. Dans la deuxième partie, nous proposons de confier à la partie commande, en plus de ses attributions habituelles, une mission de contrôle des comptes rendus des tâches exécutées par la partie opérative. Ceci permet notamment de limiter les risques d'évolutions aberrantes en cas de défaut sur un capteur, donc d'augmenter la sécurité de fonctionnement. Ce résultat sera obtenu en introduisant des informations supplémentaires dans le programme du dispositif vu en première partie.

S O M M A I R E

I N T R O D U C T I O N

C H A P I T R E I - CAHIER DES CHARGES DE L'EQUIPEMENT I

1.1- Caractéristiques générales du système de commande I.1
1.1.1- Domaine d'emploi I.1
1.1.2- Entrées de la partie commande I.2
1.1.3- Grandeurs de sorties I.2
1.1.4- Opérations sur les grandeurs d'entrées I.2
1.1.5- Contraintes de temps I.2
1.1.6- Réalisation des systèmes séquentiels I.3

1.2- Représentation desspécifications fonctionnelles d'un automatisme logique par un réseau de Pétri interprété I.3
1.2.1- Evolution du marquage d'un réseau de Pétri interprété sans conflits I.4
1.2.2- Réseaux de Pétri saufs I.4
1.2.3- Réseaux de Pétri non généralisés type S I.5
1.2.4- Réseaux de Pétri non généralisés type T I.5
1.2.5- Exemple de description par un réseau type S puis T I.8
1.2.6- Réceptivité et sensibilité I.10
1.2.7- Représentation matricielle des réseaux I.11

1.3- Réalisation d'un système de commande I.13
1.3.1- Choix des microprocesseurs I.13
1.3.2- Répartition des tâches entre microprocesseurs I.15
1.3.3- Définition du dialogue entre les automates numériques et logiques I.16

1.4- Schéma fonctionnel de l'ensemble I.18

1.5 Conclusion I.18

C H A P I T R E II - CHOIX D'UNE METHODE DE TRAITEMENT DES RESEAUX DE PETRI

2.1 - Données relatives à un réseau de Pétri II.1

2.2 - Traitement d'un graphe d'état II.2

2.3 - Traitement d'un réseau de Pétri II.4
2.3.1 - Décomposition d'un réseau de Pétri en graphes d'état II.4
2.3.2 - Traitement global d'un réseau de Pétri II.6

2.4 - Etablissement des sorties II.9
2.4.1 - Sorties de type T II.9
2.4.2 - Sorties de type S II.9
2.4.3 - Choix de la représentation réseau S ou réseau T II.12

2.5 - Choix des places clefs II.13
2.5.1 - Choix du critère II.14
2.5.2 - Méthode euristique de choix des places clefs II.15

2.6 - Aléas liés au traitement asynchrone dans les réalisations programmées II.20
2.6.1 - Comportement vis-à-vis des noeuds OU distributifs II.20
2.6.2 - Comportement lorsque plusieurs transitions sont franchissables simultanément II.21
2.6.3 - Réalisation synchrone II.23

2.7 Conclusion II.23

C H A P I T R E III - REALISATION DU DISPOSITIF DE COMMANDE

3.1 - Architecture du dispositif de commande III.1
3.1.1 - Structure synchrone III.1
3.1.2 - Description des périphériques du micro-processeur monobit III.2

3.2 - Accés aux pas III.4
3.2.1 - Test sur toutes les places clefs III.4
3.2.2 - Utilisation d'une table des places clefs marquées (table des pas) III.4

3.3 - Réalisation de l'automate logique III.7
3.3.1 - Réalisation des tables des places clefs marquées III.7
3.3.2 - Instructions nécessaires à la gestion du réseau III.8
3.3.3 - Schéma fonctionnel de l'automate logique III.10
3.3.4 - Présentation du logiciel permettant la gestion du réseau III.11

3.4 - Automate numérique III.17
3.4.1 - Dialogue entre les microprocesseurs III.17
3.4.2 - Temporisations III.21
3.4.3 - Organisation du programme de l'automate numérique III.22

3.5 - Conclusion III.23

C H A P I T R E IV - AMELIORATION DE LA SURETE DE FONCTIONNEMENT D'UNE MACHINE SEQUENTIELLE

4.1 - Causes et effets des pannes d'un système à commande numérique et logique IV.1

4.2 - Réseaux de Pétri pondérés IV.2

4.3 - Détection des pannes sur les capteurs et leurs liaisons IV.4
4.3.1 - Circuits combinatoires autotestables IV.4
4.3.2 - Description d'un système de commande à surveillance accrue IV.5
4.3.3 - Description d'une machine séquentielle à partir des variations de niveaux logiques des entrées IV.8
4.3.4 - Classification des variables primaires IV.13
4.3.5 - Surveillance des variations des entrées: détection de pannes IV.16

4.3.6 - Niveau de surveillance atteint IV.18
4.3.7 - Réseaux de Pétri temporisés IV.19

4.4 - Conclusion IV.19

C H A P I T R E V - ADAPTATION DU SYSTEME DE COMMANDE

5.1 - Comportement en cas de défaut V.1

5.2 - Adaptation du matériel V.2
5.2.1 - Automate d'entrée: détecteur de variations V.2
5.2.2 - Détection de défaut par modification non attendue d'une entrée V.3
5.2.3 - Sélection des variables de commande et de contrôle V.5
5.2.4 - Réseaux temporisés V.6

5.3 - Adaptation du logiciel V.6
5.3.1 - Description de la procédure d'arrêt V.7
5.3.2 - Modification du moniteur V.7

5.4 - Commande d'une presse V.8

5.5 - Conclusion V.13

C O N C L U S I O N

A N N E X E I - MATERIEL MIS EN OEUVRE A1.1

A N N E X E II - LOGICIEL DU DISPOSITIF DE COMMANDE A2.1

A N N E X E III - PROGRAMME DE COMMANDE D'UNE PRESSE AVEC DETECTION DE DEFAUTS A3.1

B I B L I O G R A P H I E B1

C H A P I T R E I

CAHIER DES CHARGES DE L'EQUIPEMENT

Après avoir placé notre étude dans son contexte industriel, nous présentons la structure générale de notre automate. Nous rappelons également les règles permettant la représentation d'un cahier des charges de machines séquentielles par un réseau de Pétri.

1.1 - Caractéristiques générales du système de commande

1.1.1 - Domaine d'emploi du système de commande

Il s'agit de mettre en oeuvre un système de commande logique et numérique*. Son rôle n'est pas d'effectuer la commande optimalisée de systèmes très complexes. Ce travail est normalement confié à des mini-ordinateurs. Les ordres sont fournis à la partie opérative essentiellement sous forme logique "tout ou rien" ou numérique. Si nous envisageons des sorties numériques codées sur plusieurs bits, c'est plutôt dans le but de fournir des consignes à des systèmes de régulation extérieurs à notre appareillage. En effet, il est préférable dans les systèmes travaillant en temps réel, de partager les tâches entre plusieurs microsystèmes. La nécessité des opérations numériques se fait sentir chaque fois que l'on utilise des capteurs numériques et notamment dans les applications de positionnement, pesage, dosage et comptage.
Elles sont utilisées aussi pour les temporisations.

* Nous dirons qu'un traitement portant sur des variables "tout ou rien" représentables par un chiffre en binaire est un traitement logique.
Par contre, pour des variables représentées par des mots, donc par des nombres, nous dirons que le traitement est numérique.

1.1.2 - Entrées de la partie commande

Le dispositif envisagé doit être capable de lire les niveaux de variables binaires, qu'elles soient de type "tout ou rien" ou organisées en mots.

1.1.3 - Grandeurs de sortie

Le système fixe les valeurs des grandeurs logiques ou numériques de sortie. Les grandeurs de sortie de type impulsionnel sont envisageables.

1.1.4 - Opérations sur les grandeurs d'entrée

Le système mis en oeuvre:

- calcule les valeurs de fonctions booléennes à partir des entrées "tout ou rien".
- effectue des opérations arithmétiques simples sur des mots d'entrée et éventuellement des mots donnés par programme. Les opérations envisagées sont: l'addition, la soustraction, la comparaison.
- gère des temporisations à partir d'une horloge.
- réalise des opérations de comptage.

1.1.5 - Contrainte de temps

Le système de commande travaille en temps réel. Si la vitesse de traitement des circuits électroniques permet d'envisager un traitement en séquence des tâches élémentaires, il n'en est pas moins vrai que le nombre de tâches à exécuter augmente avec la technicité de la partie opérative. Il apparaît alors que les systèmes programmés doivent avoir une

vitesse de traitement la plus grande possible, afin d'être applicables dans un maximum d'équipement.

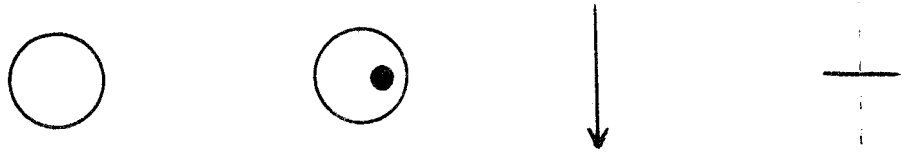
1.1.6 - Réalisation des systèmes séquentiels

Les systèmes logiques de commande industriels sont essentiellement de type séquentiel. Nous supposerons que le cahier des charges est présenté sous forme d'un réseau de Pétri interprété, sauf, sans conflits. Pour faciliter la programmation, nous choisirons un langage permettant de déduire le programme du réseau de Pétri. Le matériel et le logiciel devront permettre la gestion du marquage, même dans le cas d'évolutions simultanées.

1.2 - Représentation des spécifications fonctionnelles d'un automatisme logique par un réseau de Pétri interprété. [1] [2] [22]

Le réseau de Pétri est formé d'un ensemble de places $\mathcal{P}$, de transitions $\mathcal{T}$, d'arcs $\mathcal{A}$. Un arc relie une place à une transition ou une transition à une place.

Chaque place peut être marquée ou vide. A chaque instant il est possible de définir le marquage M du réseau et en particulier le marquage initial M_0. La représentation graphique est donnée Fig. I.1



Place non marquée — Place marquée — Arc — Transition

Symboles utilisés.

- fig. I.1 -

Le réseau est, sauf indication contraire, représenté dans son marquage initial.

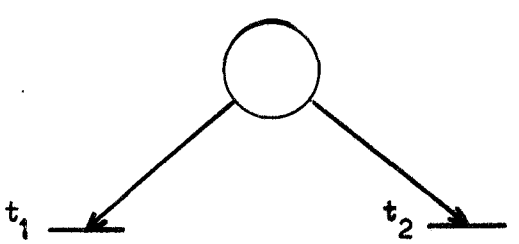
1.2.1 - Evolution du marquage d'un réseau de Pétri interprété sans conflits

Le réseau de Pétri peut être utilisé pour représenter le cahier des charges de la partie commande d'un système logique et numérique. A chaque transition est alors associée une proposition logique, ou prédicat, appelée événement.

Le franchissement d'une transition n'est possible qu'aux conditions suivantes:

- la transition est validée, c'est-à-dire que toutes les places antécédentes sont marquées;
- la condition associée à la transition est vraie. Au franchissement d'une transition, les places antécédentes perdent une marque, les places subséquentes en gagnent une.

Un réseau de Pétri qui contient au plus une place marquée, pour tout marquage accessible à partir de M_0, est appelé graphe d'état.
En cas de noeud OU distributif (Fig I.2) il y a conflit pour le réseau non interprété. Ce conflit peut être éliminé dans le réseau interprété en associant aux transitions critiques des événements exclusifs.



- fig. I.2 -

1.2.2 - Réseaux de Pétri saufs [8]

Un réseau est dit sauf si, pour le marquage initial M_0 et pour tout marquage du réseau accessible à partir de M_0, aucune place ne peut posséder plus d'un marqueur.

Nous nous interesserons uniquement au cas des Réseaux de Pétri interprétés saufs sans conflits qui sont suffisants dans la majorité des cas.

1.2.3 - Réseaux de Pétri non généralisés type S

1.2.3.1 - Définition

Dans un réseau de type S, l'état des grandeurs de sortie est associé au marquage des places. Les sorties qui prennent la valeur "1" lorsqu'une place est marquée, sont inscrites à côté de celle-ci. Lorsqu'une sortie n'est pas mentionnée à côté d'une place, cela implique que son marquage ne force pas à "1" cette sortie. Pour un marquage du graphe donné, toutes les sorties non forcées à "1" sont à "0".
Une condition supplémentaire, liée à l'état des entrées, peut être introduite pour définir l'état des sorties.

1.2.3.2 - Utilisation

Ce type de représentation est bien adapté à la description du système pour lequel les actions ont une durée d'application contrôlée. Elle s'applique donc à la majorité des problèmes industriels de commande logique et numérique.

1.2.4 - Réseaux de Pétri non généralisés type T [23][24]

1.2.4.1 - Définition

Dans un réseau de type T, une action est lancée au franchissement d'une transition. Ceci se représente pour la sortie Y_1 en écrivant (Y_1) à droite de la transition (la partie gauche étant réservée à l'événement associé).

1.2.4.2 - Utilisation

Ce type de représentation est bien adapté à la description des systèmes pour lesquels les actions ont une durée propre non contrôlée (sorties impulsionnelles). Ceci se rencontre lorsque la sortie correspond à un lancement de calcul numérique, à l'enclenchement d'une temporisation, à une autorisation de comptage...

Un réseau de type T peut toutefois être utilisé si nous associons implicitement, à chaque action de durée contrôlée, une mémoire.

Si l'action Y_1 est lancée au franchissement d'une transition nous écrivons alors (Y_1) à droite de cette transition. Si l'action Y_1 est arrêtée au franchissement d'une transition nous écrivons alors $(\bar{Y}_1)$ à droite de cette transition.

Un exemple de description par réseau de Pétri de type S et de type T est donné au paragraphe I.2.5.

1.2.4.3 - Initialisation

Pour tout réseau de Pétri, il faut définir un marquage initial et l'état des actionneurs correspondants. Avec un réseau de Pétri type T, il n'est pas possible de fixer les sorties de manière absolue. Nous conviendrons donc que tous les actionneurs sont inactifs pour le marquage initial. Nous créons au besoin une place unique marquée initialement. Elle est alors suivie d'une transition permettant de marquer les places qui étaient, au départ, initialement marquées,et de lancer les actions nécessaires.

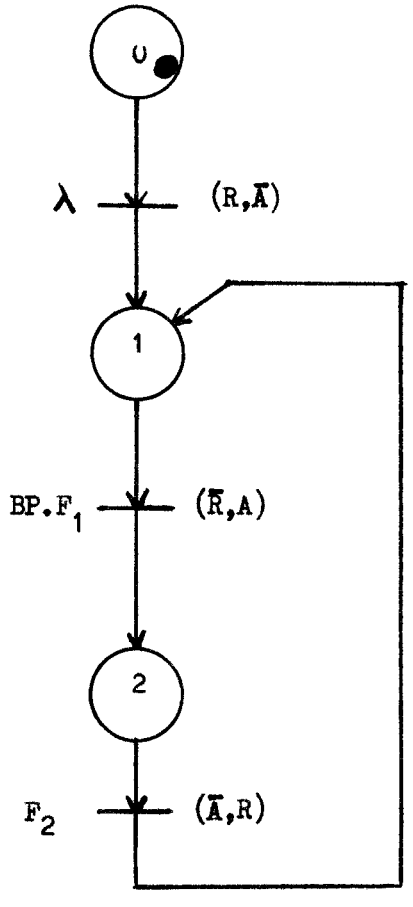
Nous créons ainsi artificiellement un état initial, valable pour tous les problèmes, accessible par une remise aux conditions initiales affectant aussi bien le marquage que les sorties.

A la transition que nous avons ajoutée, nous associons un

événement certain (pas de condition à réaliser) ou un événement permettant de vérifier, soit que les circuits de puissance sont bien alimentés, soit que l'état de la partie opérative est conforme aux conditions de mise en route de l'ensemble.

Exemple:

Un bouton poussoir BP fait faire un aller et retour à un vérin dont les positions extrêmes sont repérées par F_1 et F_2. F_1 est la position de départ. A et R indiquent que la commande du distributeur permet le déplacement vers F_2(A) ou vers F_1(R). Le réseau de Pétri est le suivant (Fig I.3).



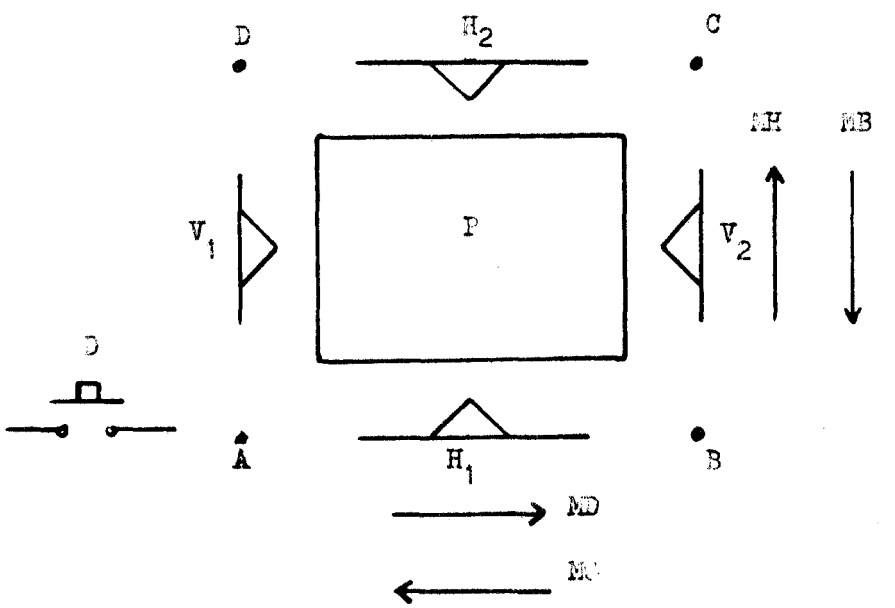
Introduction de la place marquée initialement

- Fig. I.3 -

1.2.5 - Exemple de description par un réseau de Pétri type S puis T

Cherchons à représenter le système séquentiel associé au dispositif de la figure I.4 et dont le fonctionnement doit être le suivant [8]

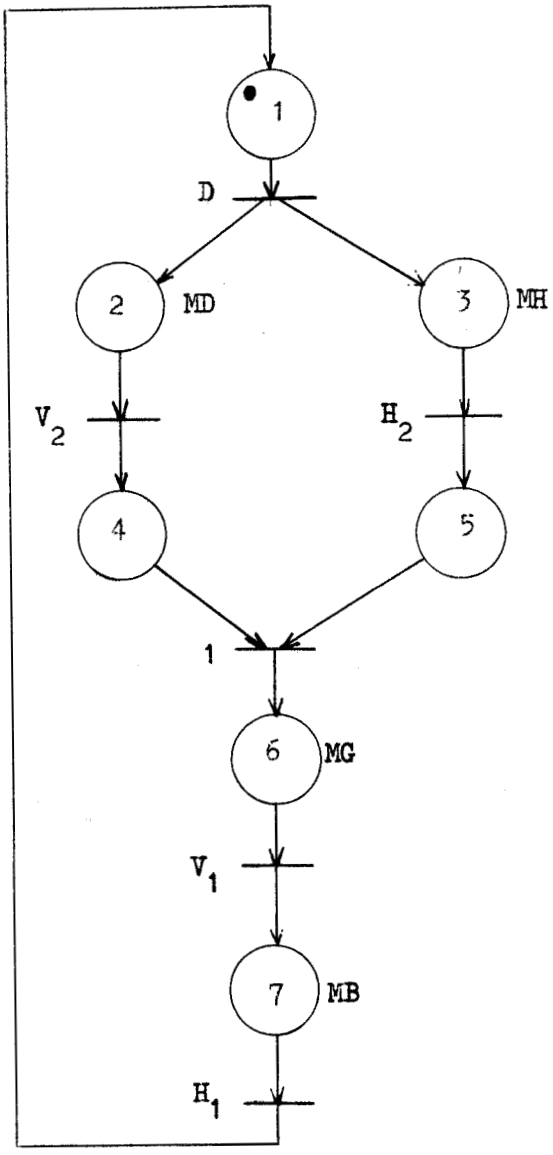
- au départ, le plateau P est en A ($H_1 = V_1 = 1$ tous moteurs arrêtés)
- en appuyant sur D, nous provoquons le mouvement du plateau suivant le cycle A C D A
- de retour en A, le plateau doit d'arrêter si D est relâché.



Déplacement d'un plateau selon le cycle ACDA

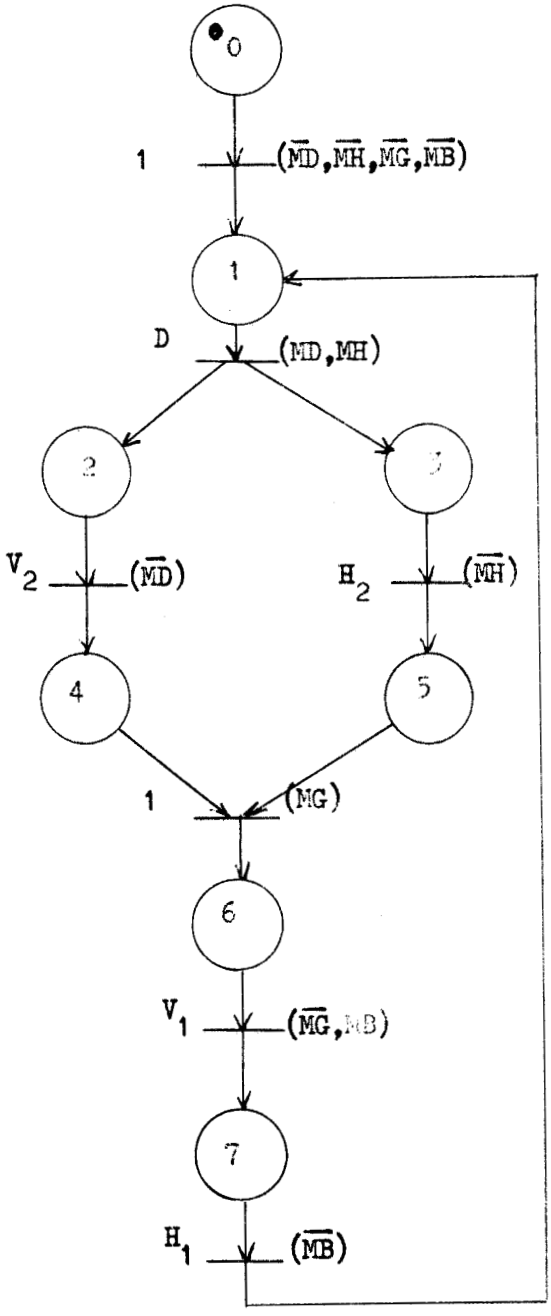
- fig. I.4 -

Le réseau de Pétri type S est donné fig. I.5, celui de type T par la fig. I.6.



Réseau S

- Fig. I.5 -



Réseau T

- Fig. I.6 -

1.2.6 - Réceptivité et sensibilité [8]

Selon son comportement vis à vis des événements nous dirons qu'une machine séquentielle est:

- réceptive, pour un marquage donné, à un événement e, si cet événement est capable de faire évoluer le marquage du réseau.
- sensible, pour un marquage donné, à un événement e, si cet événement, sans modifier le marquage du réseau, est capable de faire évoluer l'état des sorties du système de commande.

Dans un réseau de type T, les modifications de l'état des sorties sont liées au franchissement des transitions, donc obligatoirement à des modifications du marquage. Pour une machine séquentielle, décrite par un réseau de type T, et quelque soit le marquage, il n'existe aucun événement capable de faire évoluer l'état des sorties sans modifier le marquage. Cette machine est donc non sensible pour tout marquage M accessible à partir de M_0.

Ceci est un inconvénient pour les réseaux de type T. Ils ne permettent pas le contrôle de l'état des sorties par des fonctions combinatoires des entrées sans modification du marquage.

Ayant défini:

- un ensemble fini, non vide, d'événements

$$\mathcal{E} = \{ e_1, e_2 \ldots e_n \}$$

- un ensemble fini, non vide, de sorties

$$\mathcal{S} = \{ s_1, s_2 \ldots s_k \}$$

- un ensemble fini, non vide, de marquage du réseau

$$\mathcal{M} = \{ M_1, M_2 \ldots M_i \}$$

nous introduisons également une application multivoque $\varphi_{(M)}$ de $\mathcal{M}$ dans $\mathcal{E}$

A chaque élément M de $\mathcal{M}$, elle fait correspondre un sous ensemble de $\mathcal{E}$ auquel la machine est réceptive. Ce sous ensemble est composé des événements qui font évoluer le marquage. $\Gamma(M)$ est appelée fonction de réceptivité.

Nous introduisons également une application multivoque $\sigma(M)$ de $\mathcal{M}$ dans $\mathcal{E}$. A chaque élément M de $\mathcal{M}$, elle fait correspondre un sous ensemble de $\mathcal{E}$ auquel la machine est sensible. Ce sous ensemble est formé d'événements modifiant la configuration des sorties sans faire évoluer le marquage. $\sigma(M)$ est appelée fonction de sensibilité.

1.2.7 - Représentation matricielle des réseaux de Pétri [8] [20]

Nous définissons:

- Une matrice d'incidence avant notée $|G^-|$. Elle donne les numéros des places antécédentes à chaque transition. Si i et j sont respectivement les numéros de ligne et de colonne de la matrice, les coefficients a_{ij} de cette matrice sont tels que:

  $a_{ij} = 1$ si la place j est antécédente à la transition i

  $a_{ij} = 0$ sinon.

- Une matrice d'incidence arrière notée $|G^+|$. Elle donne les numéros des places subséquentes à chaque transition. Les coefficients a_{ij} de cette matrice sont tels que:

  $a_{ij} = 1$ si la place j est subséquente à la transition i

  $a_{ij} = 0$ sinon.

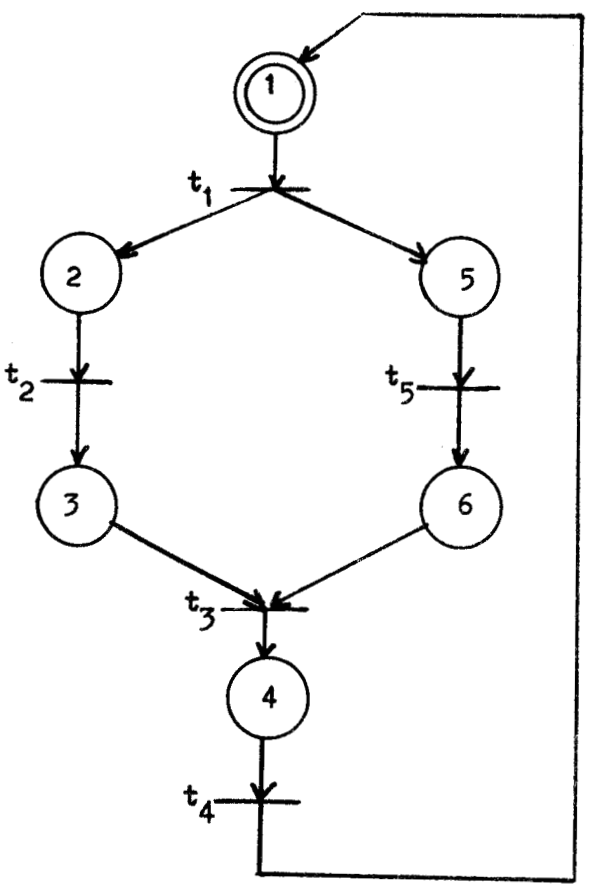
Exemple:

Pour le réseau de Pétri de la fig. I.7 la matrice d'incidence avant est:

$$|G^-| = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{vmatrix}$$

la matrice d'incidence arrière est:

$$|G^+| = \begin{vmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{vmatrix}$$



Exemple de réseau de Pétri

- fig. I.7 -

Dans le cas où aucune place d'un réseau n'est à la fois place antécédente et place subséquente d'une même transition (réseau pur), il est possible de décrire complètement le réseau par la matrice d'incidence globale

$$|G| = |G^+| - |G^-|$$

Ceci donne dans notre exemple:

$$|G| = \begin{vmatrix} -1 & 1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & -1 \\ 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{vmatrix}$$

Le coefficient a_{ij} de la matrice G est tel que:

- $a_{ij} = -1$ si la place j est antécédente à la transition j
- $a_{ij} = 1$ si la place j est subséquente à la transition i
- $a_{ij} = 0$ si la place j n'est pas connectée à la transition i

1.3 - Réalisation d'un système de commande

La description des spécifications fonctionnelles par réseau de Pétri est appelée à se développer. Il est donc souhaitable de choisir, pour la réalisation des dispositifs de commande, des automates utilisant un logiciel spécialisé permettant un passage rapide du réseau de Pétri au programme. Actuellement les réalisations de ce type font appel à des micro ordinateurs non spécialisés. L'adoption d'un langage évolué permettant l'implantation des réseaux de Pétri nécessite l'utilisation d'un programme interpréteur. Nous proposons une structure adaptée à la réalisation de systèmes décrits par réseaux de Pétri, permettant un temps de traitement plus court.

1.3.1 - Choix des microprocesseurs

Le franchissement d'une transition est lié au marquage de ses places antécédentes et à la réponse à un prédicat correspondant à l'événement associé. Pour chaque transition, ceci définit les conditions

d'évolution. Dans un réseau non généralisé sauf, le marquage d'une place peut être représenté par une grandeur "tout ou rien". Il en est de même pour la réponse au test sur l'événement. L'établissement des conditions d'évolution relève donc d'un traitement logique.

Par contre l'événement peut faire appel soit à un traitement logique (grandeurs d'entrée tout ou rien) soit à un traitement numérique (informations digitalisées). La réalisation d'un dispositif de commande nécessite donc un traitement logique et parfois un traitement numérique. Un microprocesseur travaillant sur des mots de 8 ou 16 bits peut réaliser toutes les opérations nécessaires. Nous trouvons, en effet, dans les instructions, des opérations logiques bit à bit. Toutefois pour connaître la valeur d'une variable représentée par un bit dans un mot, il faut:

- sélectionner le mot contenant la valeur de la variable
- l'isoler par marquage
- faire éventuellement un décalage, de façon à placer le bit qui nous interesse à la place choisie dans le mot.

Le calcul en chaîne d'opérations logiques est alors long. Il est donc évident que le traitement logique doit être confié, de préférence, à un microprocesseur monobit.

Par contre, les traitements numériques sur des mots sont longs et de programmation fastidieuse sur un microprocesseur monobit. Partant de ces considérations, il est souvent proposé de choisir entre monobit et multibit en fonction du pourcentage entre traitement logique et numérique. En fait, si nous disposons d'un système de commande multiprocesseur, réalisé autour d'un microprocesseur multibit et d'un microprocesseur monobit, ils seront toujours utilisés de façon optimale.

Ce choix n'est pas utopique car le prix des microprocesseurs et de leurs circuits annexes ne représente pas une part importante dans le coût d'un système de commande. Le prix d'un équipement est surtout lié:

- aux circuits d'interfaçage avec la partie opérative (adaptation des entrées sorties au milieu) ce qui représente une dépense incompressible.
- au temps d'étude, ce qui nous amène à choisir une présentation du cahier des charges efficace et un langage de programmation qui soit adapté à cette technique de représentation.

Nous avons utilisé:

- le microprocesseur monobit M C 14.500 B de Motorola pour sa grande souplesse d'emploi due à un faible taux d'intégration. [15]
- le microprocesseur multibit 8085 de chez Intel. [13]

1.3.2 - Répartition des tâches entre microprocesseurs

Le microprocesseur monobit effectuera tout le traitement logique et en particulier:

- le combinatoire d'entrée
- le combinatoire de sortie
- la gestion du marquage du réseau.

Le microprocesseur multibit effectuera tout le traitement numérique et en particulier:

- les temporisations
- les comptages, décomptages (opération d'incrémentation et de décrémentation)
- les comparaisons entre des grandeurs numériques de consigne et des grandeurs mesurées
- des opérations arithmétiques élémentaires sur ces grandeurs numériques (additions, soustractions).

Compte tenu des impératifs de vitesse à respecter, la solution consistant à faire travailler les processeurs, alternativement, selon la nature de la tâche à exécuter, à partir d'un programme commun a été repoussée. Un travail simultané des deux automates doit en effet améliorer la vitesse de l'ensemble. De plus, l'intégration du compteur ordinal dans les microprocesseurs multibit est telle que les branchements à des adresses ne peuvent se faire autrement que par le multibit lui-même. Ceci interdit la possibilité de gérer des sauts directement à partir du monobit, ce qui ralentit encore le traitement. La solution retenue est donc celle d'un travail simultané, avec dialogue, entre les processeurs par un interface Entrée/ Sortie.

1.3.3 - Définition du dialogue entre les automates numériques et logiques

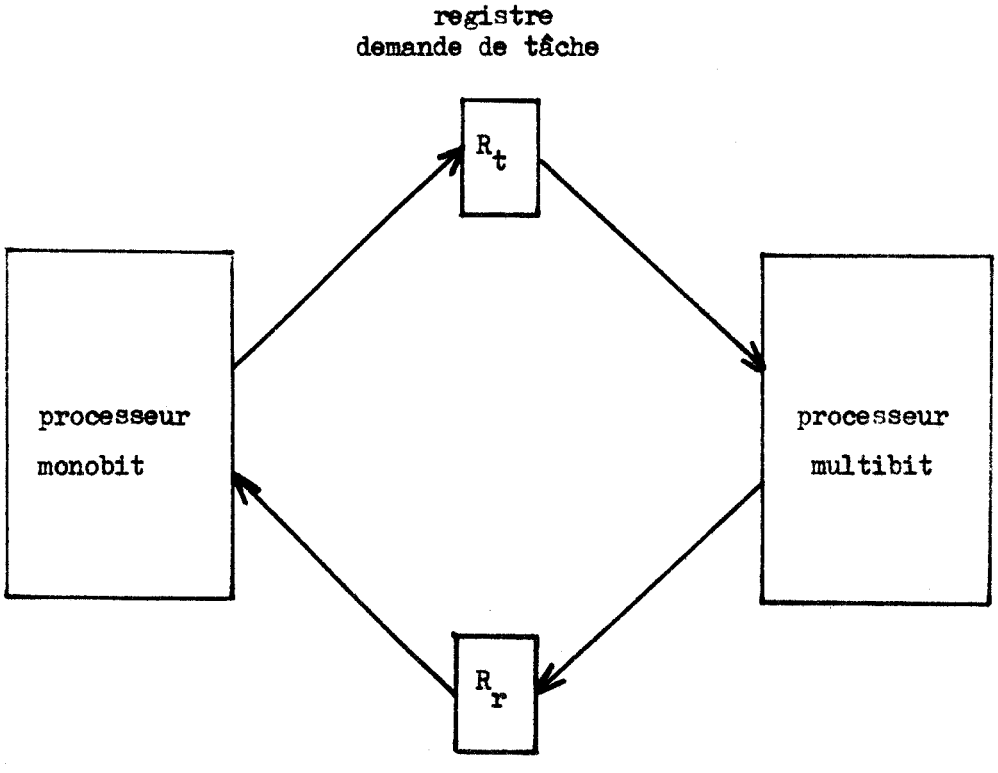
L'automate construit autour du monobit effectuera les traitements booléens et la gestion du marquage du réseau de Pétri. Chaque fois qu'un traitement numérique sera jugé nécessaire par l'automate logique, ce dernier fera une demande de tâche auprès de l'automate numérique. Pendant l'exécution de ce travail, la gestion du réseau sera poursuivie. Ceci nous permet de conserver le contrôle des marquages dans les autres branches en cas de fonctionnement simultané. Les tâches demandées au processeur multibit auront pour effet:

- soit de déterminer une grandeur de sortie. Le résultat du calcul n'a pas d'influence immédiate sur la gestion du réseau.
- soit de permettre l'élaboration d'une condition d'évolution. Dans ce cas, le processeur monobit a besoin de connaître le résultat de la tâche pour prendre ses décisions.

Les deux automates auront accès par leur dispositif respectif d'entrée sortie,

à un registre commun faisant office de répertoire des tâches numériques demandées. Plusieurs tâches numériques doivent pouvoir être traitées en séquence. Le processeur logique ayant seul la possibilité de demander un traitement numérique, écrira dans la case du registre associée à cette tâche. Lorsque le résultat d'une tâche numérique est exploité dans les conditions d'évolution, nous utilisons un registre résultat. Le microprocesseur multibit aura seul la possibilité d'écrire dans ce registre.

Par tâche numérique, nous avons donc un registre demande de tâche et éventuellement un registre résultat. La structure correspondante est donnée fig. I.8.

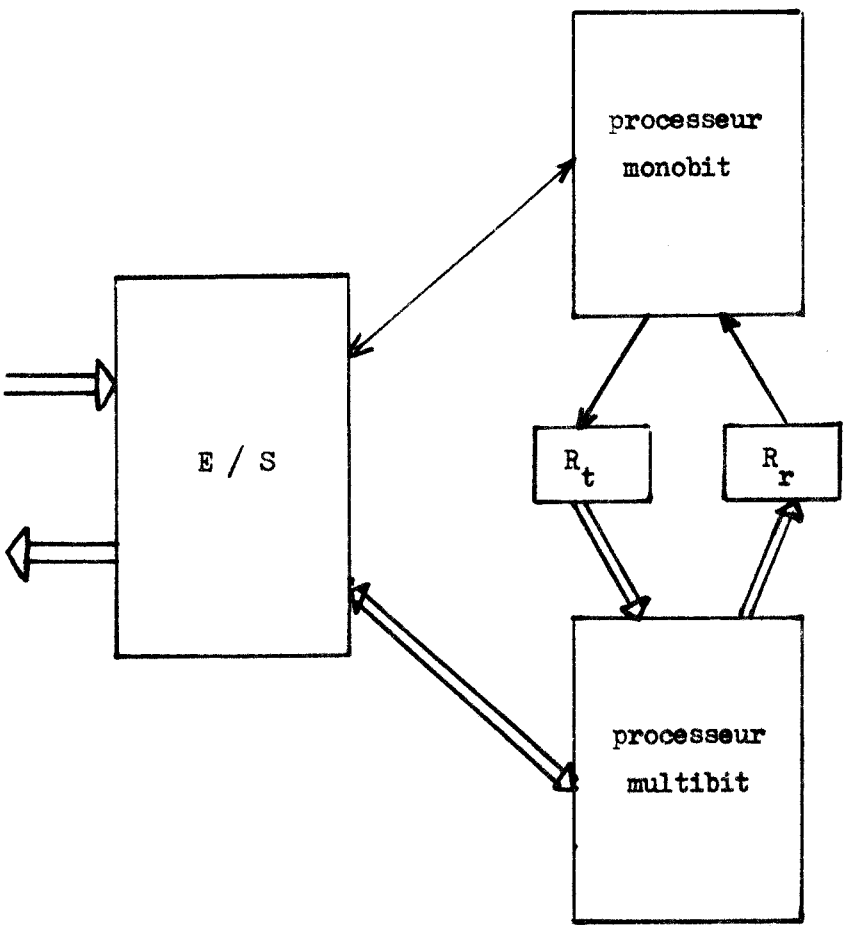


Structure permettant le dialogue
entre les microprocesseurs.

- fig. I.8 -

1.4 - Schéma fonctionnel de l'ensemble

Le schéma fonctionnel de l'ensemble est donné fig I.9



- fig. I.9 -

1.5 Conclusion

Nous avons choisi une structure d'automate à deux micro-processeurs spécialisés dans les traitements logiques ou numériques qui doit permettre de résoudre les problèmes de commande numérique et logique. Nous avons également opté pour un moyen de représentation du cahier des

charges: le réseau de Pétri. Nous en avons rappelé les règles essentielles en mettant l'accent sur le problème de l'affectation des sorties. Ceci nous a conduit à faire la distinction entre les réseaux T et S. Il nous reste à envisager une programmation directe à partir de ces réseaux de Pétri en évitant le retour aux tables de fluence et aux méthodes de synthèse qui s'y rattachent.

C H A P I T R E II

CHOIX D'UNE METHODE DE TRAITEMENT DES RESEAUX DE PETRI

Nous avons défini, dans ses grandes lignes, l'architecture du dispositif de commande logique et numérique. Nous allons choisir un mode de traitement des réseaux de Pétri après avoir passé rapidement en revue quelques unes des solutions possibles.

Nous donnons ensuite la forme sous laquelle le réseau doit être présenté pour être introduit dans le système de commande.

2.1 - Données relatives à un réseau de Pétri

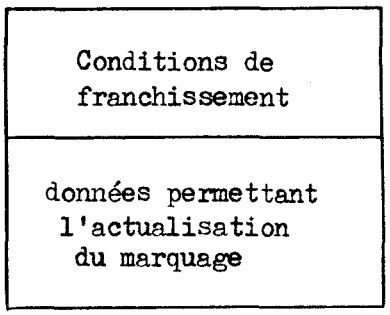
Les différentes données doivent permettre:

- la détermination des conditions d'évolution du marquage ou de franchissement des transitions.
- l'actualisation du marquage après évolution.
- l'affectation des sorties de type T.
- l'affectation des sorties de type S.

L'ensemble des programmes relatifs aux trois premiers points sont généralement regroupés pour chaque transition. Les programmes relatifs à une transition et à l'ensemble du réseau sont illustrés respectivement par les figures II.1 et II.2.

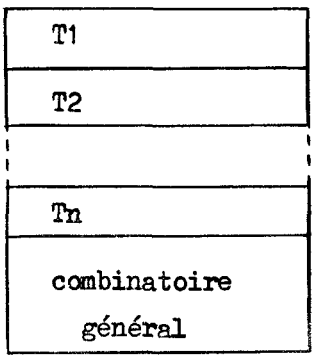
Le combinatoire général regroupe les instructions traitées indépendamment du marquage.

T_1



Représentation d'une transition

- fig. II.1 -



Représentation d'un réseau de Pétri

- fig. II.2 -

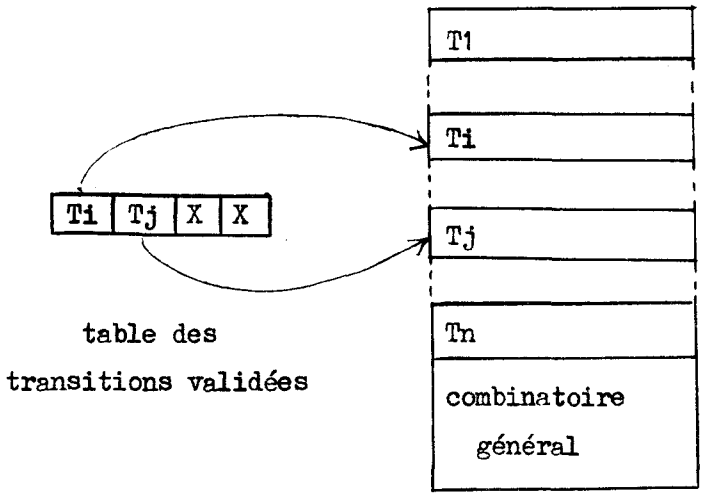
Pour limiter le temps de traitement, nous allons proposer des méthodes permettant l'accès à la partie du programme relative aux seules transitions validées.

2.2 - Traitement d'un graphe d'état [3][4][18]

Un graphe d'état contient au plus une place marquée. Le franchissement d'une transition est alors subordonnée au marquage d'une seule place qui valide seule certaines transitions. Il est possible d'avoir une représentation en mémoire des transitions validées à chaque instant. Cela permet un accès direct à ces transitions et une accélération du traitement.
La figure II.3 illustre cette solution.

A chaque transition est associée la liste des transitions qui seront validées ou invalidées après son franchissement.

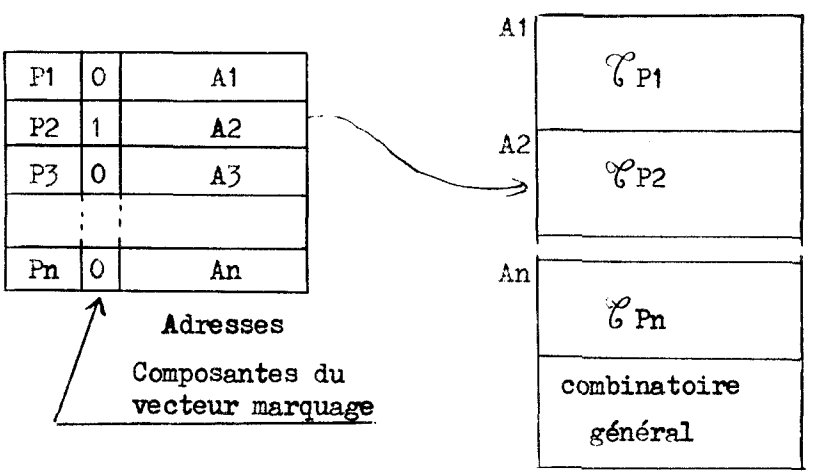
Si toutes les sorties sont impulsionnelles ou considérées comme telles, la représentation mémoire du marquage peut être supprimée.



Utilisation d'une table des transitions validées.

- fig. II.3 -

Une autre solution consiste à regrouper les données relatives à l'ensemble des transitions qui ont une même place antécédente. L'accès à ces transitions se fait alors soit à partir de la représentation en mémoire du marquage de chaque place (Fig. II.4a), soit à partir d'une table des adresses dont l'accès se fait à partir du numéro de la place marquée mis en mémoire (fig. II.4b).

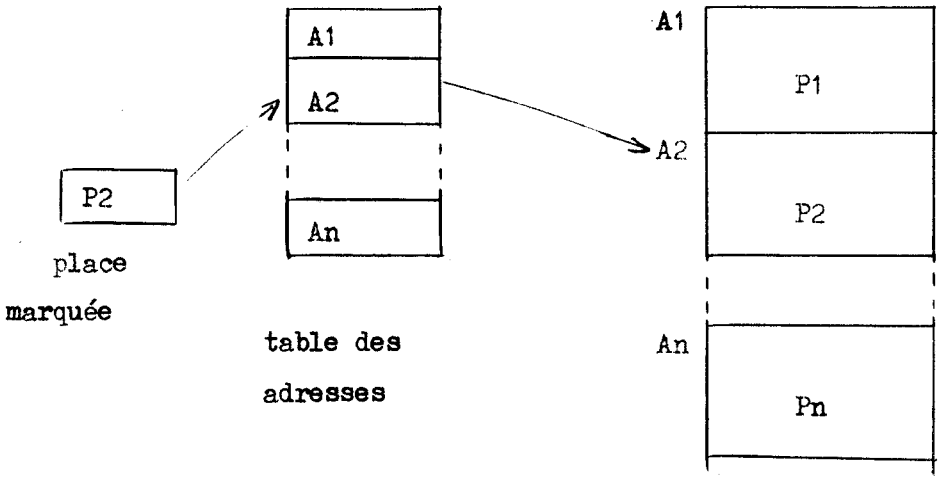


$\mathcal{C}$Pi:ensemble de transitions validées par le marquage de Pi

Ai: adresse de début des données de Pi

vectorisation du marquage

- fig. II.4.a -



Mémorisation de la place marquée

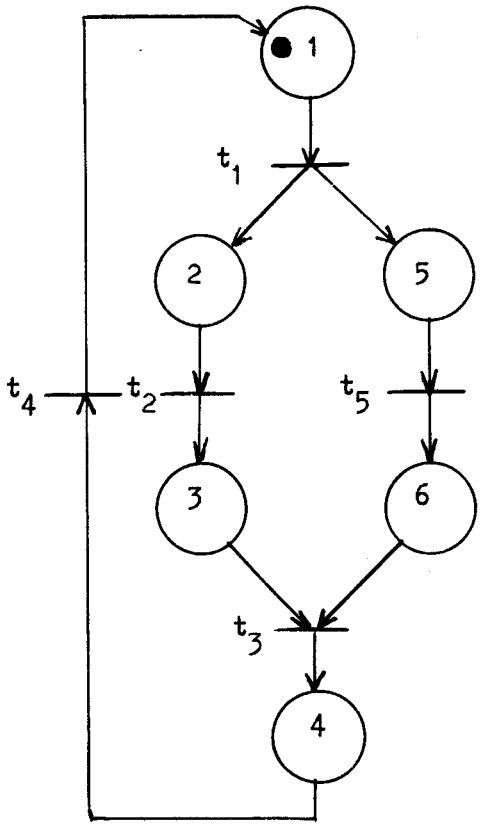
- fig. II.4.b -

2.3 - Traitement d'un réseau de Pétri

Dans le cas de réseaux de Pétri nous pouvons avoir plusieurs places marquées simultanément. Il est possible d'envisager soit une décomposition du réseau en graphes d'état, soit un traitement global.

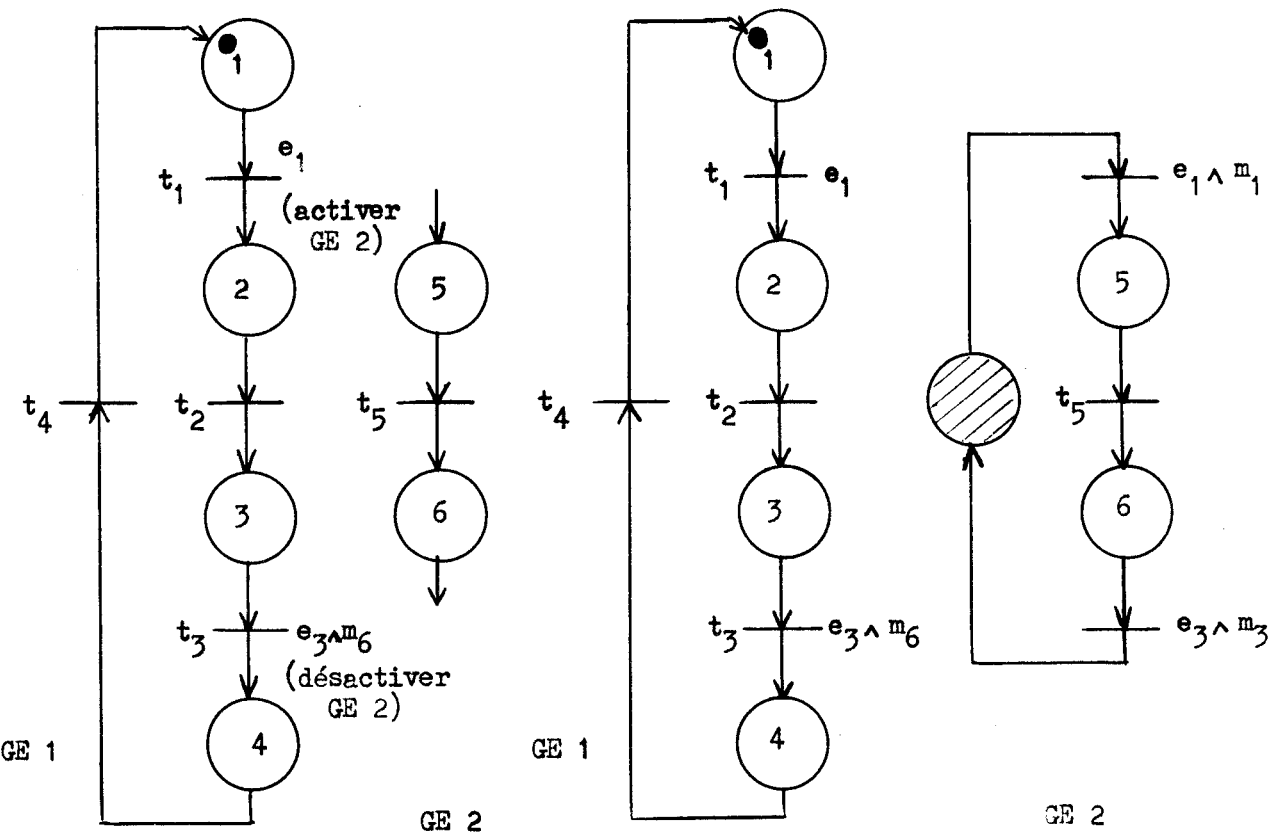
2.3.1 - Décomposition d'un réseau de Pétri en graphes d'état [3][4][21]

Cette décomposition est obtenue en opérant des déconnections au niveau des noeuds ET. Le but recherché est de substituer au réseau de Pétri un ensemble de graphes d'état complétement déconnectés. La synchronisation des évolutions dans les graphes est obtenue en utilisant des variables internes. Si l'on admet qu'un graphe d'état puisse être non marqué (graphe d'état non activé), le réseau de la figure II.5 conduit à la figure II.6a. Le franchissement de la transition t_1 entraîne l'activation du graphe groupant les places 5, 6. La synchronisation est obtenue en validant l'événement associé à t_3 par la variable m_6. Une méthode euristique de décomposition peut être mise en oeuvre. [17]



Réseau de Pétri

- fig. II.5 -



(a) Décomposition en graphes d'états (

- fig. II.6 -

Il est possible aussi de créer des places d'attente sur les graphes d'état ouverts (Fig. II.6b). Dans ce cas le nombre de places marquées est constant et égal au nombre de graphes d'état.

La décomposition peut conduire également à la rupture des noeuds OU. Ceci transforme le réseau de Pétri en séquenceur "pas à pas". Cette solution a été adoptée par la Télémécanique pour son automate TSX 80.

Dans chaque cas il suffit de traiter chaque graphe d'état l'un après l'autre. Les méthodes vues pour la gestion des graphes d'état peuvent être reprises. Dans le cas du traitement illustré par la figure II.4b, le registre de place marquée devient une table de dimension fixe (une case par graphe d'état). Chaque case contient le numéro de la place marquée si le graphe d'état est activé ou un indicateur dans le cas contraire.

2.3.2 - Traitement global d'un réseau de Pétri [12][6][7]

Ces méthodes ne nécessitent pas de décomposition préalable du réseau. Par contre la transposition des méthodes vues pour les graphes d'état n'est pas immédiate.

a) Utilisation d'une table des transitions validées [9][10]

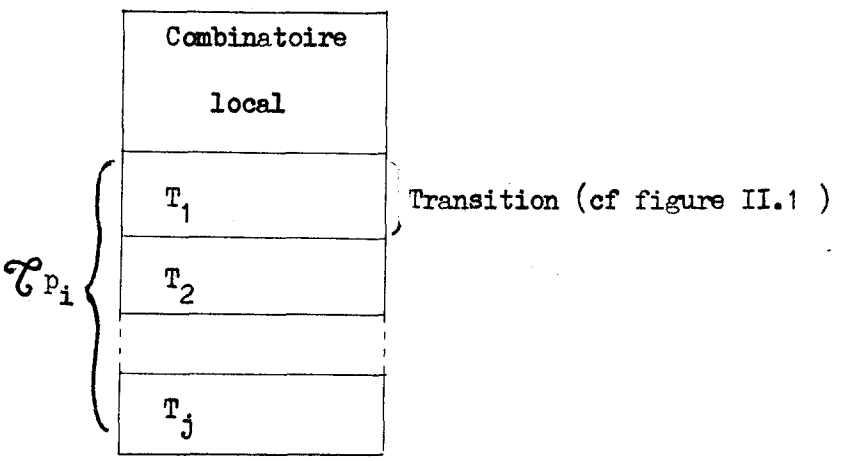
Certaines transitions admettant plusieurs places antécédentes, l'élaboration de cette table nécessite un traitement préalable. Bien que cette méthode permette de limiter le traitement aux seules transitions validées, il semble que cela ne compense pas le temps du traitement préalable.

b) La deuxième solution consiste comme précédemment à regrouper les transitions qui admettent une même place antécédente. Toutefois certaines transitions admettant plusieurs places antécédentes, nous pouvons:

- Tester des transitions non validées (toutes les places antécédentes ne sont pas obligatoirement marquées).

- Représenter et tester plusieurs fois la même transition (autant de fois qu'elle a de places antécédentes).

Pour remédier à ce deuxième défaut, il suffit de décider qu'une transition doit être représentée une fois et une seule. La place P_i donnant accès au seul ensemble de transitions $\mathcal{T} P_i$ qui contient la transition t_j est appelée place clef pour t_j. L'ensemble du programme, dont l'accès est subordonné au marquage de la place P_i forme un pas. Outre l'ensemble relatif à $\mathcal{T} P_i$, il contient éventuellement un combinatoire local exécuté si et seulement si la place P_i est marquée (Fig. II.7).

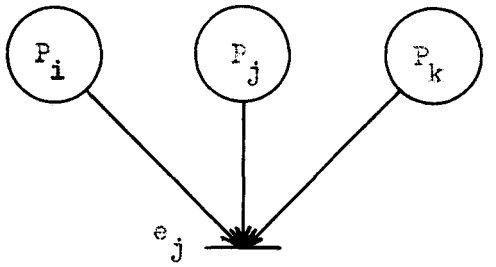


Structure d'un pas

- fig. II.7 -

Les autres places antécédentes à t_j non choisies comme place clef pour t_j sont appelées places de synchronisme. Leur marquage doit apparaître dans les conditions de franchissement de t_j (Fig. II.8).[18]

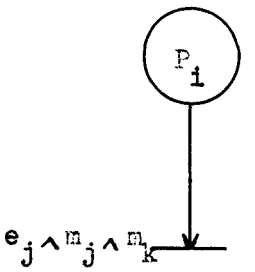
Les places de synchronisme (P_j) sont donc représentées par une variable interne (m_j) qui est mise à 1 quand la place P_j est marquée et à 0 sinon (Fig. II.9).



Noeud ET attributif sur la transition t_j

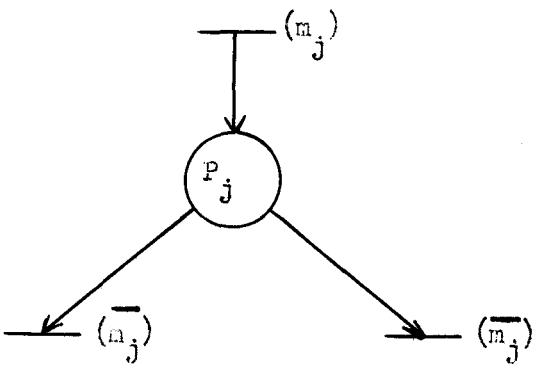
(e_j représente l'événement associé à t_j)

- fig. II.8a -



P_i est la place clef pour la transition t_i. m_j et m_k sont des variables internes représentant le marquage des places P_j et P_k.

- fig. II.8b -

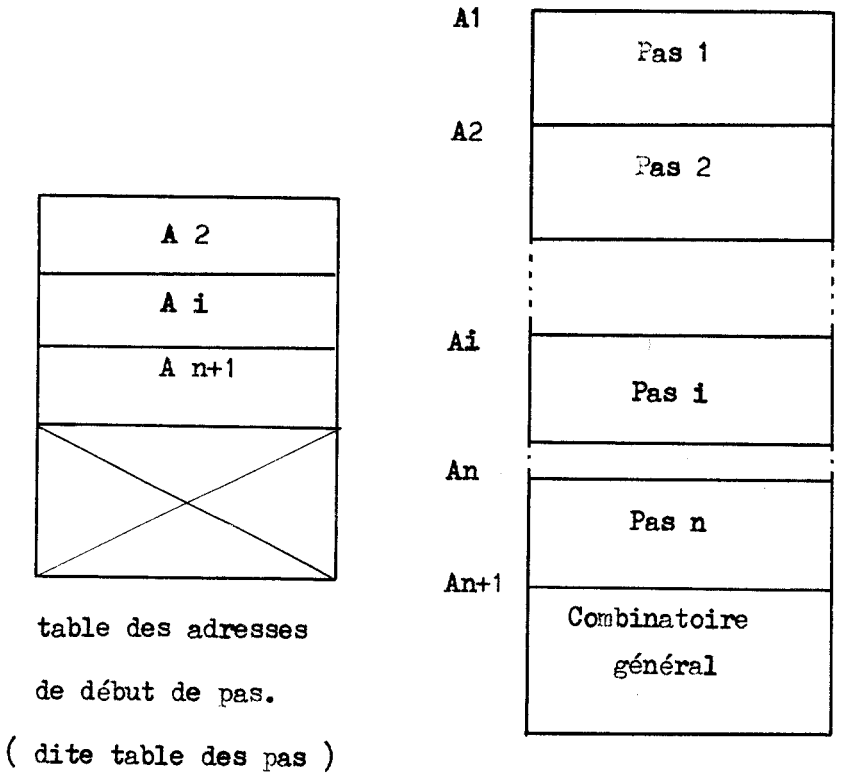


Affectation d'une variable interne associée à une place.

- fig. II.9 -

La solution retenue pour notre implantation est la transposition de celle qui est présentée Fig. II.4b pour laquelle nous créons une table des places clefs marquées. Pour éviter la lecture de la table des adresses nous pouvons mettre dans la table des places clefs marquées l'adresse A_i de début de pas plutôt que P_i (Fig II.10). Cette table prend alors le nom de table des adresses de début de pas (ou en abrégé table des pas).

Cette méthode d'implantation de réseaux de Pétri a déjà été adoptée pour des réalisations sur système câblé à architecture standard [9][10]



Utilisation de la table des pas.

- fig. II.10 -

Il nous reste à définir un critère de sélection des places clefs permettant une représentation optimale. Avant de faire ce choix nous allons préciser le contenu du combinatoire local.

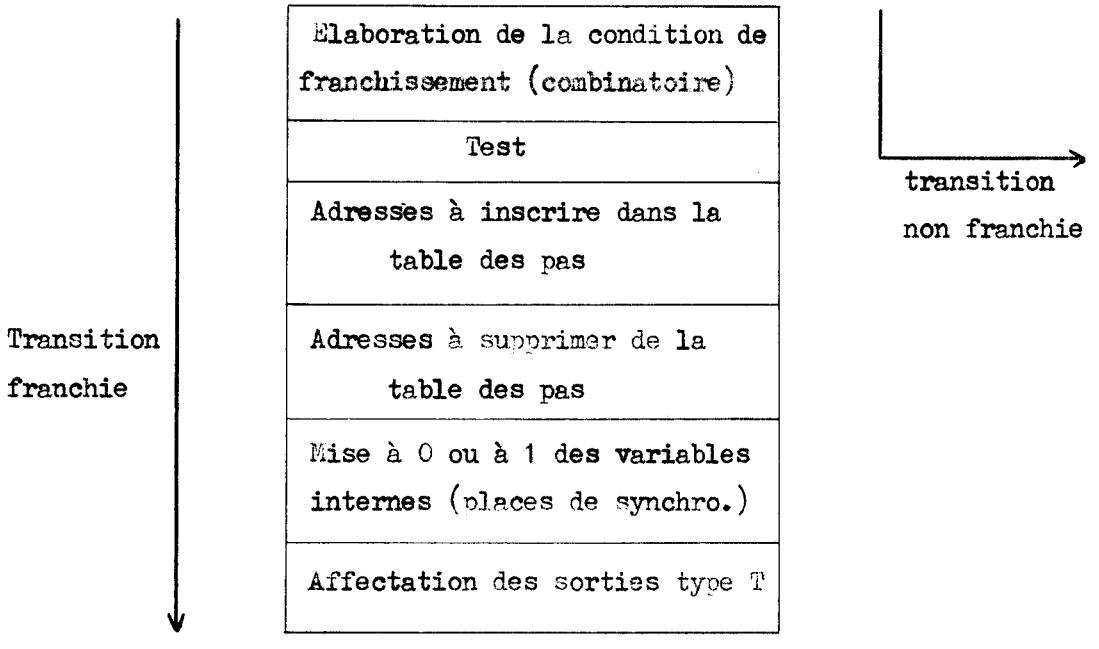
2.4 - Etablissement des sorties

Le programme doit permettre non seulement la gestion du marquage mais aussi l'établissement des sorties. C'est pour l'élaboration de ces sorties qu'un combinatoire peut être nécessaire.

2.4.1 - Sorties de type T

Les sorties sont ici des grandeurs de nature impulsionnelle. Elles permettent le lancement d'actions ayant leur durée propre: lancement de

temporisation, calcul numérique. L'utilisation de dispositifs de sortie bistables permet toutefois d'élaborer des sorties de niveau associées inconditionnellement à une place, en commandant la mise en service ou l'arrêt des actionneurs (§ 1.2.4). La proposition: mettre la sortie Y à 1 (ou à 0), qui permet le lancement(ou l'arrêt) de l'action correspondante, doit être lue uniquement lorsque la transition est franchie. Pour une transition, les instructions permettant la mise à jour du marquage et l'affectation des sorties de type T sont regroupées et localisées après le test sur la condition de franchissement (Fig. II.11).



Configuration du programme relatif à une transition.

- fig. II.11 -

2.4.2 - Sortie de type S

Les sorties de type S correspondent à des variables de niveau dont la valeur est liée directement au marquage des places auxquelles elles sont associées. Ce type de sortie reste obligatoire lorsque nous sommes en présence d'une machine sensible pour laquelle les sorties sont affectées conditionnellement aux places. Le programme relatif à l'affectation de ces sorties peut être localisé dans le combinatoire général. Ceci nécessite une

lecture systématique d'une partie souvent importante de la mémoire.

Une autre solution, permettant une réduction du temps de traitement, consiste à établir les sorties de type S à partir de combinatoires locaux. Chaque place à laquelle est associée une sortie de type S est alors place clef. Une difficulté tient au fait que les sorties, associées à une place, sont mises à 0 quand cette place se démarque. Or, le démarquage de la place, entraîne le non accès au combinatoire local correspondant et donc l'impossibilité de traiter la modification de la sortie.

Nous pouvons envisager des solutions câblées ou programmées pour remédier à cet inconvénient, comme par exemple:

- créer des variables de sortie auxiliaires mises à 0 au début de chaque scrutation du réseau. Le combinatoire local élabore les mises à 1 éventuelles des sorties auxiliaires. En fin de scrutation, les valeurs des sorties auxiliaires sont transférées aux sorties réelles.
  Pour obtenir un temps de scrutation plus court qu'avec le combinatoire général, il faut envisager une réalisation câblée de ces conditions.
- Utiliser des circuits monostables pour la commande des actionneurs. Toute sortie non mise à 1 depuis un certain temps prend automatiquement la valeur 0. Ce temps doit être court vis-à-vis du temps de réponse des actionneurs mais assez long vis-à-vis du temps de traitement maximum du réseau. Des circuits de sortie de ce type sont utilisés sur l'automate programmable Klöckner-Moeller.
- Le démarquage de la place se faisant par franchissement d'une transition, il est possible d'associer la mise à 0 des actions de type S aux transitions qui admettent la

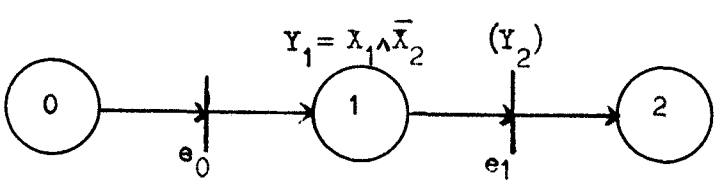
place clef concernée comme place antécédente.

Cette solution purement programmée est adoptée. Elle évite les interfaces différenciées pour les sorties de type S et T.

2.4.3 - Choix de la représentation réseaux S ou réseaux T

La réalisation de réseaux de type S conduit à une élaboration des sorties, même si celles-ci n'évoluent pas. Ceci se traduit par une augmentation du temps de scrutation. Dans un système programmé, effectuant un traitement séquentiel, les sorties de niveau ne peuvent se faire que par un interface bistable. Les conditions nécessaires pour représenter ces grandeurs de sortie par un réseau de type T sont donc satisfaites. L'élaboration des sorties se fait alors uniquement au franchissement des transitions. Lorsque le marquage est stable (donc entre deux évolutions du marquage) le temps de traitement est réduit au minimum. Toutefois, nous avons vu dans le premier chapitre qu'une machine représentable par un réseau de type T est insensible.

En conclusion, pour entrer le réseau de Pétri dans le système de commande, nous donnerons la préférence à la représentation de type T. Nous utiliserons une représentation de type S lorsque l'utilisation, pour un marquage donné, d'une machine sensible permet une simplification de la représentation. Nous adoptons donc, pour entrer un réseau de Pétri dans notre dispositif de commande, une représentation mixte ST en privilégiant la représentation de type T. La fig. II.12 donne un exemple de représentation mixte d'une partie de machine sensible.



Représentation mixte S.T: machine sensible.

- fig. II.12 -

Le combinatoire général, qui permet l'élaboration de grandeur de sorties qui sont indépendantes du marquage, est bien souvent inexistant.

2.5 - Choix des places clefs

La place clef permet de conditionner l'accès à un ensemble de transitions γP_i, à un combinatoire local d'élaboration de sorties de type S, ou à ces deux ensembles de données. Elles sont choisies, parmi l'ensemble $\mathcal{P}$ des places du réseau, en vue de limiter le temps moyen de traitement.

Faute de pouvoir appréhender directement le temps de traitement, nous proposons les critères de sélection suivants:

- Minimiser le nombre de places clefs.

ou

- Prendre le plus grand nombre possible de places clefs.

ou

- Eviter la double représentation (une place peut être place clef pour une transition et place de synchronisme pour une

ou autre).

- Limiter le nombre de variables internes.

Nous examinons les implications de ces critères sur le traitement.

a) Nombre_minimum_de_places_clefs

En limitant le nombre de places clefs, nous augmentons la taille des pas, donc le temps de traitement correspondant. Par contre le nombre de pas traités est limité.

b) Nombre_maximum_de_places_clefs

En augmentant le nombre de places clefs, nous limitons la taille des pas mais nous en augmentons le nombre.

c) Double_représentation_des_places

Le traitement au marquage et au démarquage de cette place risque d'être ralenti. Il y a éventuellement deux traitements distincts à faire pour une même place.

d) Limiter le nombre de variables internes

Ce critère a surtout une influence sur l'occupation mémoire. Il ne sera pas repris.

2.5.1 - Choix du critère

Ce choix est surtout lié au temps nécessaire à l'exécution des différentes tâches élémentaires mises en oeuvre dans le traitement. Nous pouvons toutefois faire les remarques suivantes:

- Le temps de traitement lié au franchissement d'une transition n'est pas pris en compte car cette évolution doit se faire quelque soit le critère choisi. Notre choix sera donc basé sur le temps de scrutation nécessaire lorsqu'aucune transition n'est franchissable.
- Le temps de scrutation réel pour un marquage donné peut être notablement différent du temps moyen de traitement du réseau.
- Notre choix est fait sur des considérations statistiques. Nous supposons que les marques sont distribuées de façon aléatoire. Pour minimiser réellement le temps de traitement il faudrait tenir compte de l'évolution des marqueurs, de l'occurence des événements...

Supposons un réseau de Pétri comportant n transitions et p places. Soit s le nombre moyen de sorties de type S associées à une place et p_c le nombre de places clefs. Le temps moyen de traitement d'un pas dans lequel aucune transition n'est franchie est:

$$t = t_t \frac{n}{p_c} + t_s \frac{s}{p_c}$$

Les coefficients t_t et t_s représentent respectivement :

- le temps moyen de traitement correspondant à l'élaboration

et au test des conditions de franchissement (t_t)

- le temps moyen permettant l'élaboration d'une sortie de type S (t_s).

Le nombre de pas est égal à p_c de par la définition des places clefs. Nous admettons que le nombre moyen de place marquée est $\alpha.p$. Dans la pratique $\alpha \ll 1$ (généralement inférieure même à 0,1). Nous supposons alors que le nombre de places clefs marquées est en moyenne $\alpha.p_c$. Ceci fixe le nombre de pas testés. Le temps moyen de traitement de ces $\alpha.p_c$ pas est alors de:

$$t_1 = (t_t \frac{n}{p_c} + t_s \frac{s}{p_c}) \alpha.p_c$$

$$= \alpha.n.t_t + \alpha.s.t_s$$

A ceci s'ajoute le temps t_2 d'accès au pas: $t_2 = t_a.\alpha.p_c$

t_a est le temps d'accès à un pas; un temps fixe t_3 correspond au temps de traitement du combinatoire général.

Le temps moyen de traitement est:

$$t = t_1 + t_2 + t_3$$

$$= \alpha.n.t_t + \alpha.s.t_s + \alpha.p_c.t_a + t_3$$

Nous retrouvons la remarque du paragraphe 4.3 qui permet de réduire le terme $\alpha.s.t_s$ (voire même à l'annuler dans le cas d'une machine non sensible) en évitant les sorties de type S.

Le terme $\alpha.n.t_t$, dans lequel p_c est absent est indépendant du critère choisi. Le temps moyen de traitement le plus faible est obtenu en minimisant le terme $\alpha.p_c.t_a$, donc en réduisant p_c.

2.5.2 - Méthode euristique de choix des places clefs

Vus les résultats précédents, le critère de choix est donc la réduction du nombre de places clefs. Lorsqu'un choix reste possible après l'application de cepremier critère, nous prendrons la [illegible] clef de façon à

éviter la double représentation. Une méthode de choix optimum des places clefs à partir de ces critères a été proposée [18] . Nous en proposons une variante faite à partir de la matrice d'incidence avant comme table de choix.

Définition:

Place clef essentielle: si une transition admet une seule place antécédente, cette dernière est choisie comme place clef. Nous dirons que nous avons affaire à une place clef essentielle. Lorsqu'un combinatoire local de sortie est associé à une place, cette dernière est place clef essentielle pour ce combinatoire.

La matrice incidence avant est complétée par deux lignes où nous mettons un indicateur X chaque fois qu'une place est choisie comme place clef (première ligne) ou le numéro d'une variable interne lorsque la place est prise comme place de synchronisme (deuxième ligne).

Au départ de la procédure de choix des places clefs, la ligne d'indication des places clefs contient déjà des croix pour toutes les places clefs essentielles pour les combinatoires locaux. Nous faisons apparaître nos choix successifs en cerclant les 1 de la matrice incidence avant qui se trouvent à l'intersection d'une ligne et d'une colonne affectées respectivement à une transition et à sa place clef.

Nous illustrons nos différentes étapes de sélection par l'exemple suivant (Fig. II.13) pour lequel la matrice incidence avant est donnée figure II.14a.

Première étape:

a) Toutes les places clefs essentielles sont mises en évidence.

b) Pour chaque transition ayant au moins une place antécédente parmi l'ensemble des places clefs essentielles, nous prenons l'une quelconque de ces places comme place clef (la plus à gauche dans la table par exemple). Lorsque plusieurs places essentielles sont antécédentes d'une même transition le choix est libre parmi celles-ci. En effet, quelque soit notre choix,

l'ensemble des places clefs n'est pas agrandi (premier critère satisfait) mais la double représentation est inévitable (deuxième critère non satisfait).

c) A toutes les places de synchronisme misent en évidence par cette première étape, nous affectons une variable interne.

d) Nous éliminons les transitions qui ont leur place clef. Nous obtenons un tableau réduit dans lequel toutes les transitions ont au moins deux places antécédentes mais aucune place clef choisie. Si cette table n'est pas vide, nous passons à la deuxième étape.

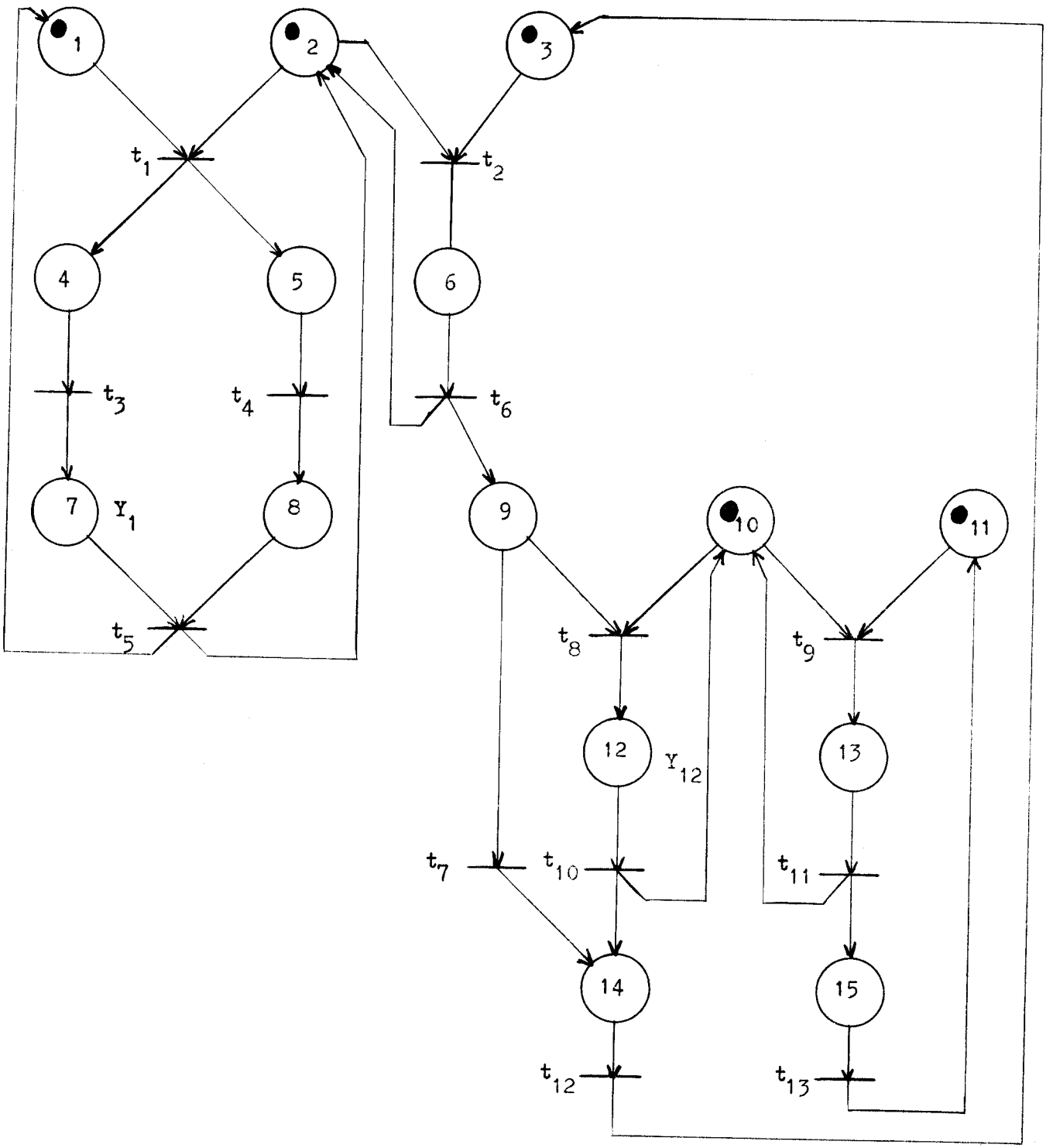
Deuxième étape:

Nous choisissons dans la table réduite la place qui est antécédente au plus grand nombre de transitions possibles et qui n'est pas déjà reconnue place de synchronisme. Cette place est prise comme place clef et nous réduisons la table par élimination des transitions ainsi couvertes. Cette étape est reprise tant que cela est possible. En cas de conflit entre places dans notre choix nous prenons la place la plus à gauche. Si à l'issue de cette étape la table réduite n'est pas vide, nous passons à la troisième étape.

Troisième étape:

Si la table réduite obtenue n'est pas vide, c'est qu'il reste des transitions dont toutes les places antécédentes sont places de synchronisme. La double représentation est inévitable. Nous reprenons la même étude qu'à la deuxième étape mais sans tenir compte des places de synchronisme. Le choix est fini lorsque la table réduite est vide. Toutes les transitions ont alors une place clef.

Les figures II.14b et II.14c donnent les tables réduites obtenues après application des deux premières étapes. A l'issue de la deuxième étape la matrice réduite est vide pour cet exemple. Il n'y a donc aucune double représentation.



Réseau de Pétri interprété utilisé dans l'exemple d'application de l'algorithme d'optimisation (d'après [10]).

- fig. II.13 -

Places	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	transitions
	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	2
	0	0	0	(1)	0	0	0	0	0	0	0	0	0	0	0	3
	0	0	0	0	(1)	0	0	0	0	0	0	0	0	0	0	4
	0	0	0	0	0	0	(1)	1	0	0	0	0	0	0	0	5
	0	0	0	0	0	(1)	0	0	0	0	0	0	0	0	0	6
	0	0	0	0	0	0	0	0	(1)	0	0	0	0	0	0	7
	0	0	0	0	0	0	0	0	(1)	1	0	0	0	0	0	8
	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	9
	0	0	0	0	0	0	0	0	0	0	0	(1)	0	0	0	10
	0	0	0	0	0	0	0	0	0	0	0	0	(1)	0	0	11
	0	0	0	0	0	0	0	0	0	0	0	0	0	(1)	0	12
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	(1)	13
							X					X				Places clefs
																Pl. synchronisme

Début de la première étape: matrice incidence avant

- Fig. II.14a -

Places	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Transitions
	1	(1)	0	0	0	0	0	0	0	0	0	0	0	0	0	1
	0	(1)	1	0	0	0	0	0	0	0	0	0	0	0	0	2
	0	0	0	0	0	0	0	0	0	1	(1)	0	0	0	0	9
				X	X	X	X		X			X	X	X	X	Places clefs
								m_2		m_1						Pl. synchronisme

Début de la deuxième étape

- Fig. II.14b -

Places	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
		X		X	X	X	X		X		X	X	X	X	X	Places clefs
	m_3		m_4					m_2		m_1						Pl. synchronisme

Fin de la deuxième étape

- Fig. II.14c -

En travaillant sur la matrice incidence avant, il est possible de connaître, pour chaque place de synchronisme, la liste des transitions dont le franchissement entraîne la mise à 1 de la variable interne qui lui est associée. De même à partir de la matrice incidence arrière, il est possible de connaître la liste des transitions pour lesquelles ces variables internes sont mises à 0.

2.6 - Aléas liés au traitement asynchrone dans les réalisations programmées

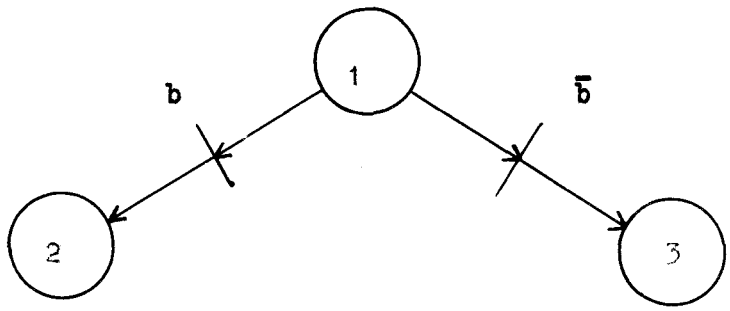
Dans toute réalisation asynchrone il y a risque d'aléas lorsque l'on fonctionne en mode non fondamental. Ceci se produit lorsque des modifications des entrées et du marquage apparaissent en un laps de temps inférieur au temps de scrutation de l'ensemble du réseau.

2.6.1 - Comportement vis-à-vis des noeuds OU distributifs

Un système programmé échantillonne les grandeurs d'entrée. Or les conflits sont éliminés, dans les réseaux interprétés, en associant aux transitions concernées des événements exclusifs. Entre le traitement de deux transitions, il s'écoule un certain temps. Ceci peut permettre le rétablissement du conflit par modification d'une entrée (fig. II.15).

Si le marquage de la place 1 et l'évolution de la variable b se produisent dans un laps de temps inférieur au temps de traitement, et en asynchronisme total avec ce traitement, il est possible que la variable b évolue entre le traitements des deux transitions. Ceci entraîne le franchissement des deux transitions, c'est un aléa. Celui-ci est supprimé si nous prenons synchrone

la scrutation des entrées, c'est à dire si les entrées sont figées pendant tout le traitement.



Noeud OU distributif

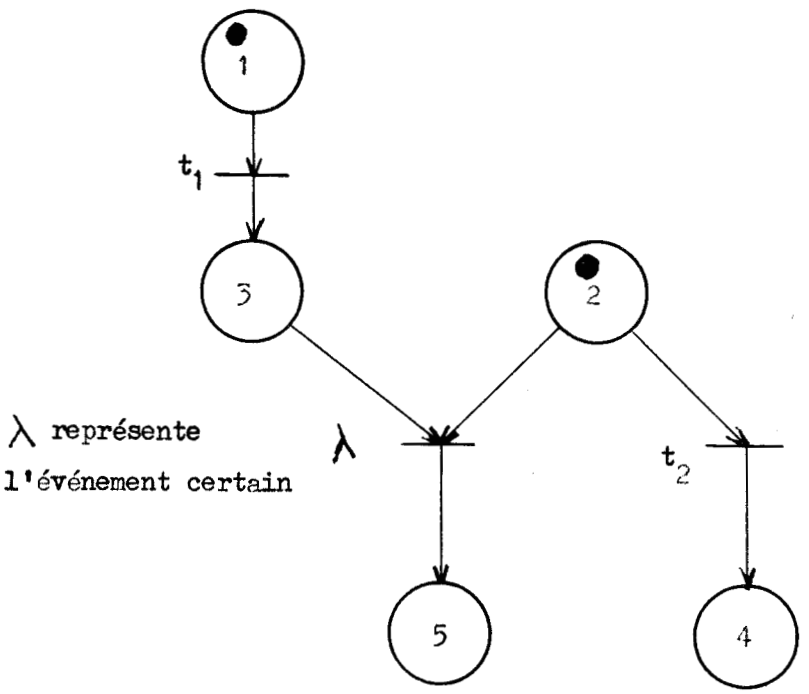
- fig. II.15 -

2.6.2 - Comportement lorsque plusieurs transitions sont franchissables simultanément

Si les conditions d'évolution correspondant aux transitions franchissables simultanément ne sont liées, ni par les entrées, ni par les places marquées, il n'y a pas de probléme. Il suffit alors de traiter l'une aprés l'autre les transitions franchissables. Il y a risque d'aléas si ces conditions ne sont pas satisfaites. L'exemple suivant en est une illustration (fig. II.16a). Théoriquement la premiére transition validée de t_1 ou t_2 entraîne le démarquage de la place 2 et selon le cas le marquage de la place 5 ou 4. Plaçons nous dans le cas où les événements associés à t_1 et t_2 sont vrais et que les places 1 et 2 sont maquées (il suffit par exemple que depuis la derniére scrutation la place 1 se soit marquée et que l'événement associé à t_2 se soit réalisé). Nous constatons alors que nous créons à nouveau un conflit. Selon l'ordre dans lequel nous traitons les transitions, le marquage final est différent (fig. II.16b - fig. II.16c). Ce défaut peut, s'il est bien maitrisé, être une qualité. Pour cela il convient de tenir compte de la hiérarchisation dans le traitement. Une réalisation synchrone ne présente pas les mêmes particularités.

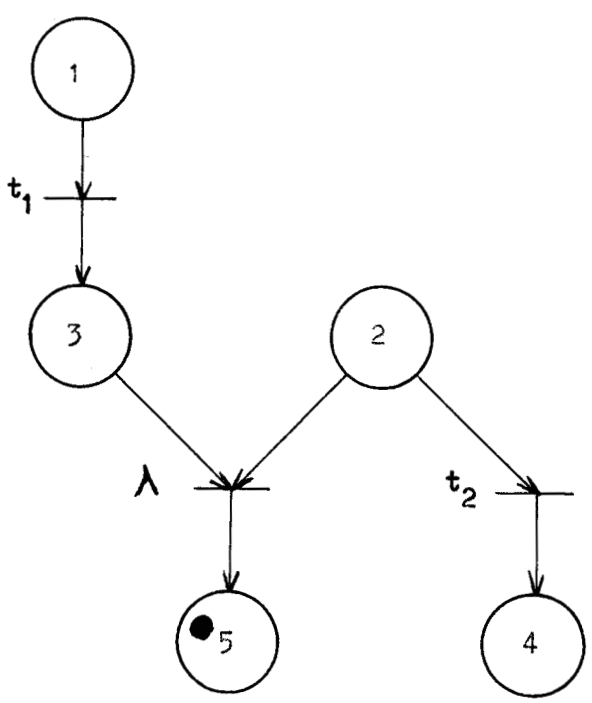
Cette remarque nous parait importante et nous amène à penser qu'il

existe réellement deux modes de représentation, l'un synchrone, l'autre asynchrone et que le choix entre les deux doit être fait dès le départ.



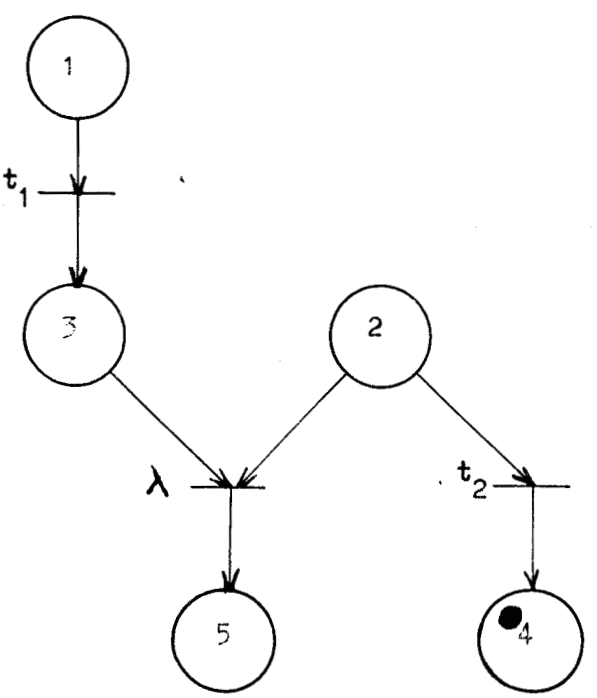
Exemple de conflit au deuxiéme degré.

- fig. II.16a -



Marquage obtenu en traitant les transitions dans l'ordre t_1, t_3, t_2.

- fig. II.16b -



Marquage obtenu en traitant les transitions dans l'ordre t_1, t_2, t_3.

- fig. II.16b -

2.6.3 - Réalisation synchrone

Toutes ces difficultés, dues à un fonctionnement en mode non fondamental, sont d'autant plus graves que nous utilisons un système de traitement programmé donc relativement lent. Les modifications des entrées et des marquages ne sont en effet pas prises en compte immédiatement. Pour éviter les risques d'aléas qui en découlent nous conférons à notre dispositif de commande un comportement synchrone vis-à-vis des conditions d'évolution. Ces conditions d'évolution seront donc maintenues constantes pendant la scrutation complète du réseau, les évolutions des entrées et du marquage n'étant pas prises en compte pendant la scrutation.

2.7 - Conclusion

Dans ce chapitre, nous avons choisi un moyen de présentation des réseaux de Pétri en vue de leurs programmations sur notre système de commande. Nous y avons notamment introduit la notion de place clef qui permet une structuration en sous programmes, appelés pas. Ces derniers sont appelés par un programme fixe, indépendant de l'application, en fonction du marquage. Nous avons également mis en évidence le rôle des places de synchronisme, une variable interne représentant leur marquage. Bien que cela n'interesse pas directement l'utilisateur, nous avons justifié la nécessité de donner au dispositif de commande une structure de machine synchrone vis-à-vis des conditions d'évolution.

C H A P I T R E III

REALISATION DU DISPOSITIF DE COMMANDE

Nous avons défini la présentation du réseau de Pétri en vue de sa programmation. Dans ce chapitre, nous présentons le matériel et le logiciel mis en oeuvre pour réaliser le dispositif de commande. L'aspect multiprocesseur y est tout particulièrement mis en évidence.

3.1 - Architecture du dispositif de commande

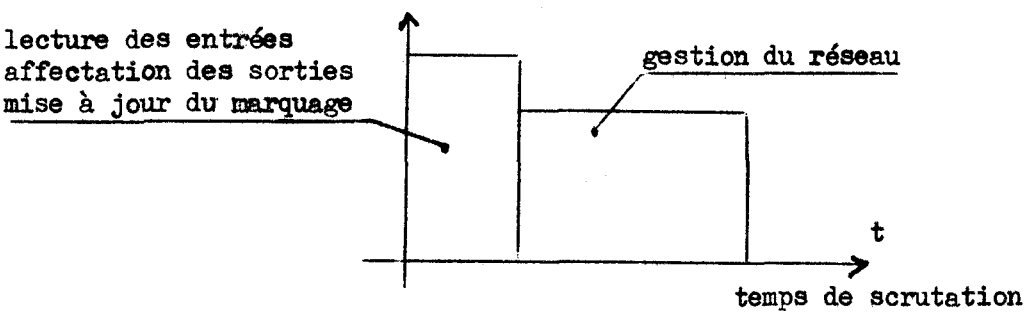
Dans un réseau de Pétri nous pouvons avoir plusieurs places marquées simultanément. Si l'on suppose que plusieurs transitions ne peuvent pas être franchies simultanément, la réalisation peut être asynchrone. Par contre, si nous supposons que plusieurs transitions peuvent être franchissables à un même instant, nous avons un fonctionnement asynchrone non fondamental. Comme nous l'avons vu au chapitre précédent (§2.6), il y a alors risque d'aleas. Nous avons alors choisi de donner un comportement de machine synchrone à notre dispositif de commande.

3.1.1 - Structure synchrone

Pour obtenir un comportement synchrone, nous devons geler les conditions d'évolution pendant tout le temps de scrutation du réseau. Désirant calculer les conditions d'évolution au niveau des seules transitions qui ont leur place clef marquée, nous avons choisi de geler les valeurs des entrées et le marquage pendant le temps de scrutation de l'ensemble du réseau. De même les sorties ne sont transférées vers l'extérieur qu'à la fin du temps de scrutation. Le traitement se décompose alors en trois temps qui sont:

- lecture des niveaux des entrées et des marquages
- détermination de l'évolution des marquages et des sorties (les résultats sont mis en mémoire)
- affectation des sorties.

Le traitement étant cyclique nous pouvons confondre les temps de lecture des entrées et d'affectation des sorties. La fig. III.1 représente sur un chronogramme la succession des opérations proposée .



- fig. III.1 -

3.1.2 - Description des périphériques du microprocesseur monobit

Ces périphériques d'entrée/sortie permettent:

- de lire les entrées (notées X)
- d'affecter les sorties (notées Y)
- d'affecter et de lire les variables internes (notées M) qui représentent le marquage des places de synchronisme
- de faire des demandes de tâches numériques (notées T) vers le microprocesseur multibit et de lire les résultats(notés R)
- d'affecter et de relire des variables intermédiaires(notées Z)

Les variables intermédiaires permettent de stocker les résultats partiels lors de calcul d'expressions booléennes comportant des parenthèses. Ces variables doivent pouvoir être relues à chaque instant. Ce sont les seules grandeurs qui ne sont donc pas gelées pendant le temps de traitement.

Pour obtenir la séparation entre les grandeurs E/S vues de l'extérieur et celles vues par le microprocesseur, nous avons placé sur chaque

entrée ou sortie un registre. Ce registre est rendu transparent au début de chaque cycle par le positionnement du drapeau "Flag RTN".

La structure générale des dispositifs d'entrée/sortie est donnée fig. III.2. Les schémas des cartes correspondantes sont données en annexe I. Les problèmes d'entrée/sortie industrielles (isolement galvanique, antiparasitage, adaptation de puissance) n'ont pas été abordés.

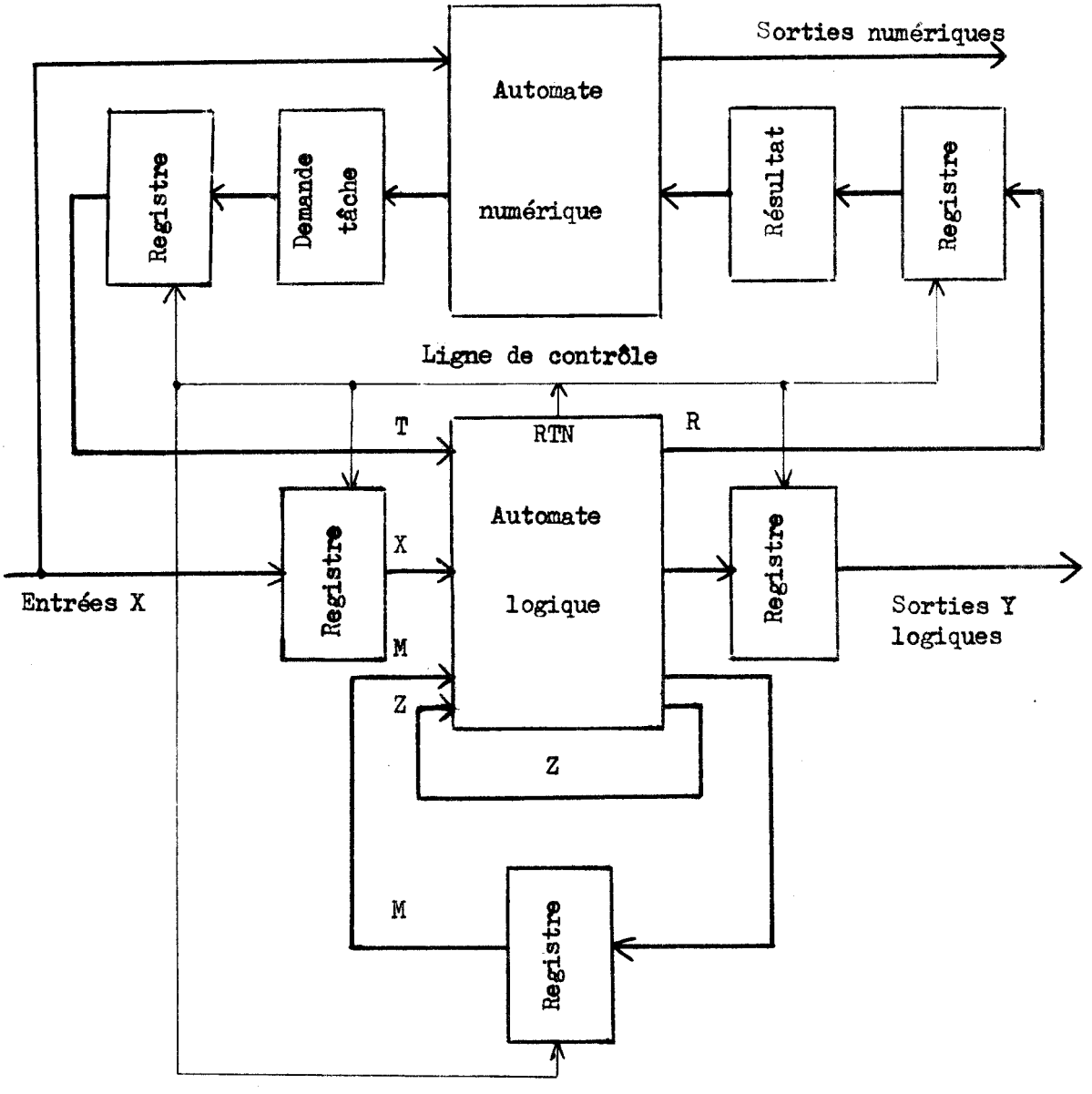


Schéma fonctionnel du dispositif de commande
(systéme synchrone)

- fig. III.2 -

Le probléme du gel des marquages est repris dans le paragraphe suivant.

3.2 - Accés aux pas

Parmi les solutions proposées au chapitre II, nous avons exploités celles qui consistent soit à tester l'ensemble des places clefs (dont le marquage est représenté en mémoire par un vecteur), soit à stocker dans une table les numéros des places clefs maquées.

3.2.1 - Test sur toutes les places clefs

Chaque place clef P_i est représentée par une variable interne M_i mise à 1 si la place est marquée. Si une place clef est marquée, il est fait appel au sous programme représentant le pas correspondant. Après traitement de ce pas, nous reprenons la scrutation des places clefs suivantes. Pour tester chaque place clef il faut trois instructions qui sont en utilisant le Mnémonique du 14 500 B [15] :

LDC Pi	Chargement du complément de la valeur de la variable associée à P_i
SKZ	Si la place est marquée alors
JUMP Ai	Branchement à l'adresse de début du pas i sinon place $P_i + 1$

A la vitesse maximum du microprocesseur utilisé il faut 3 μs par place clef. A ceci s'ajoute évidemment le temps de traitement des pas accessibles. Un tel automate a été développé. Il est présenté en annexe I. Il est acceptable pour la réalisation de réseaux de Pétri donnant peu de places clefs ou lorsque le temps de traitement peut être assez long. Une version plus rapide a été envisagée et mise au point.

3.2.2 - Utilisation d'une table des places clefs marquées (table des pas)

A chaque place clef P_i correspond un pas commençant à l'adresse A_i. Nous nous proposons d'écrire dans une table l'ensemble des

Ai associées aux places clefs marquées. L'accès à un pas se fait alors par un saut à une adresse contenue dans la table.

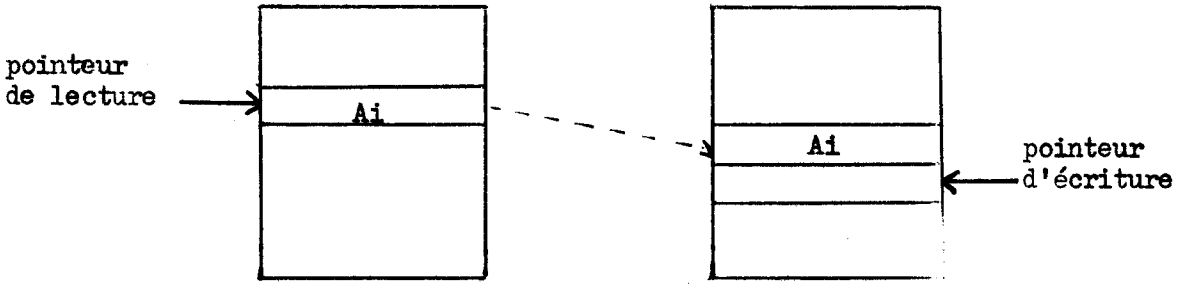
Pour respecter le caractère synchrone du traitement nous devons geler le marquage pendant la scrutation du réseau. Pour obtenir ce résultat nous avons créé deux tables. La première contient l'adresse des pas correspondant aux places clefs marquées au début de la scrutation. La deuxième est remplie au fur et à mesure du traitement en fonction de l'évolution du marquage. Au début de chaque traitement nous échangeons les rôles entre ces deux tables.

Etablissement de la nouvelle table des places clefs marquées:

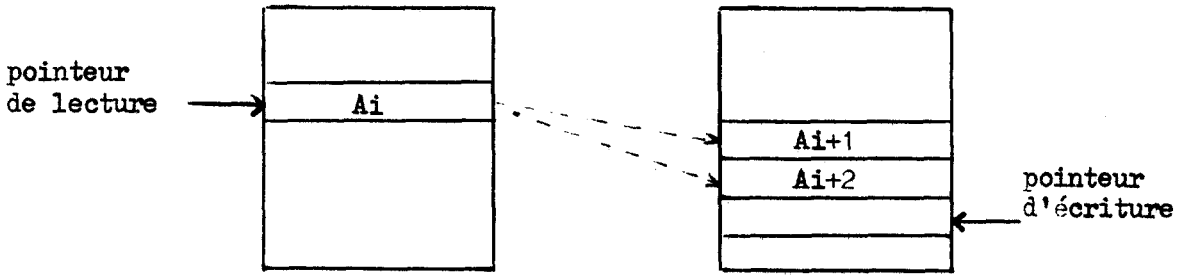
A partir d'un pas dont l'adresse est lue dans la première table, nous avons deux possibilités:

- Si aucune transition du pas n'est franchie, l'adresse de début de pas est recopiée dans la deuxième table.
- Si une transition est franchie, la ou les adresses de début de pas, correspondant aux places clefs subséquentes à la transition qui vient d'être franchie, sont inscrites dans la deuxième table.

Supposons par exemple une transition t_j ayant P_i pour place clef et placée dans le pas i d'adresse A_i. Supposons que le franchissement de t_j entraîne le marquage des places clefs P_{i+1} et P_{i+2} correspondant aux pas d'adresse A_{i+1} et A_{i+2}. Le remplissage de la deuxième table est donné fig. III.3 dans le cas où t_j est non franchie (Fig. III.3a) et dans le cas où t_j est franchie (Fig III.3b).



(a) transition t_j non franchie



(b) transition t_j franchie

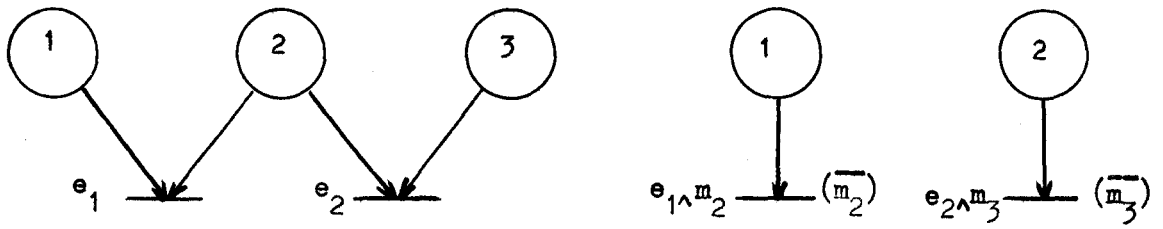
Etablissement de la 2éme table des pas

- fig. III.3 -

Dans la première table, l'adresse de la case contenant l'adresse A_i du pas en cours de traitement est repérée par un pointeur de lecture.

Dans la deuxième table, un pointeur d'écriture pointe la première case vide dans laquelle sera éventuellement écrite l'adresse d'un pas validé.

Une difficulté vient des places qui sont places clefs et qui sont également représentées par une variable interne. Ceci se produit lorsqu'une place est antécédente à, au moins, une transition pour laquelle elle n'est pas place clef bien qu'étant place clef pour d'autres transitions (Fig. III.4).



- fig. III.4 -

Dans cet exemple nous voyons que la transition t_1 est validée si les places P_1 et P_2 sont marquées. Ces deux places sont alors inscrites dans la première table. Si l'adresse A_1 associée à P_1 n'est pas recopiée dans la deuxième table, le problème se pose lorsqu'on traite le pas 2. En effet, la transition t_2

n'étant pas franchissable (pas de conflit), l'adresse A_2 est recopiée. La place P_2 est restée marquée pour la transition t_2. Pour éviter la recherche de A_2 dans la première table nous avons adopté la solution suivante:

> Lorsqu'une place clef est également représentée par une variable interne, nous testons la valeur de cette variable interne en début de pas. Si cette variable interne est à 1, la place est bien marquée, le traitement suit les règles exposées ci-dessus. Si cette variable interne est à 0, c'est que la place clef a été démarquée lors de la scrutation précédente. L'adresse du pas correspondant n'est pas recopié dans la deuxième table, le pas n'est pas traité.

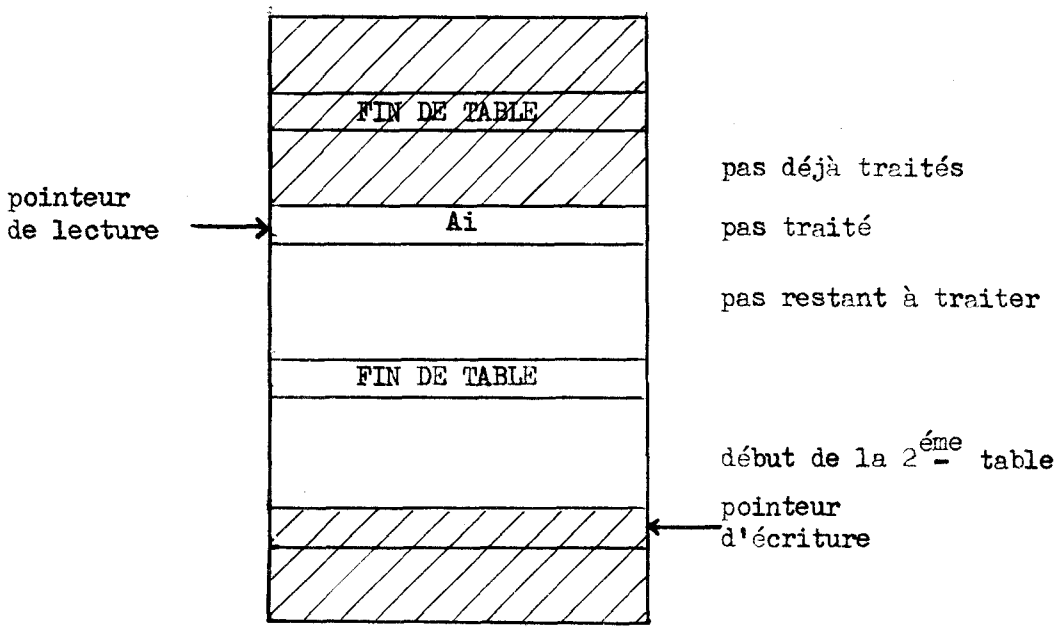
3.3 - Réalisation de l'automate logique

Cet automate bâti à partir du microprocesseur monobit gère le réseau.

3.3.1 - Réalisation des tables des places clefs marquées

A condition de placer un indicateur de fin de table, il est possible de superposer les deux tables dans une même mémoire. Si la première table est placée au-dessus de la deuxième, le pointeur de lecture passera naturellement de la première à la deuxième table assurant ainsi automatiquement l'échange des rôles. Le contenu de la pile ainsi obtenu est alors réparti comme l'indique la fig. III.5.

Lorsque le pointeur de lecture est positionné sur l'indicateur de fin de table, nous recopions cet indicateur dans la case mémoire adressée par le pointeur d'écriture. Nous positionnons également le drapeau "Flag RTN" afin de réaliser les transferts dans les registres d'entrée/sortie des grandeurs correspondantes.



Présentation du contenu de la pile

Le contenu de la partie hachurée est sans importance

- fig. III.5 -

Nous avons choisi pour cette table une mémoire à deux ports d'adresse dont la capacité est de 16 mots (Am 29705 PC de AMD). Ceci nous permet une gestion de réseaux de Pétri ayant jusqu'15 places clefs marquées simultanément.

3.3.2 - Instructions nécessaires à la gestion du réseau

Notre microprocesseur monobit MC 14500 B comprend seize instructions dont neuf sont destinées au calcul de fonctions booléennes. Quatre instructions, dont la notation mnémonique est NOPO, JMP, RTN, NOPF, positionnent les "drapeaux" notés Flag 0, Flag JMP, Flag RTN, Flag F. Le Flag RTN est déjà utilisé comme signal de synchronisation. Il commande les entrées d'horloge des registres de séparation des entrées/sorties. Il nous reste les drapeaux Flag 0, Flag JMP, Flag F pour réaliser les fonctions suivantes:

- incrémenter le pointeur de lecture INL
- incrémenter le pointeur d'écriture INL
- écrire dans la table ETA

- faire un branchement à une adresse donnée JMP adresse

- faire un branchement à une adresse table JTA

Nous avons pris des mots de 16 bits pour nos instructions. Ces mots sont décomposés en champs comme suit:

- incrémentation des pointeurs par utilisation du Flag F

INE | 1 1 1 1 | 1 0 0 0 | 0 0 0 0 | 0 0 0 0 | soit $F800_H$

INL | 1 1 1 1 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | soit $F000_H$

- écrire dans la table des pas par utilisation du Flag 0

ETA donnée | 0 0 0 0 | 1 X X X | X X X X | X X X X |
donnée

ce qui permet l'écriture d'une donnée sur 11 bits dans la table des pas.

- branchement par utilisation du Flag JMP

JTA | 1 1 0 0 | 0 0 0 0 | 0 0 0 0 | 0 0 0 0 | soit $C000_H$

JMP adresse | 1 1 0 0 | 1 X X X | X X X X | X X X X |
adresse

Ceci nous donne une capacité d'adressage de 2K mots qui est suffisante dans la majorité des applications.

Pour l'adressage des entrées/sorties, nous avons choisi de fixer à 0 le bit de poids 12. Ce bit est à 1 lorsque nous avons à adresser une variable interne. Les quatre bits de poids fort restant affectés au code opération. La distinction entre entrée (lire) et sortie (écrire) se fait par le drapeau WRITE qui est positionné par les instructions STO et STOC.
La capacité d'adressage théorique est alors de 2^{12} entrées, 2^{12} sorties, 2^{12} variables internes. Dans la pratique, en utilisant les circuits d'entrée (MC 14512) et de sortie (MC 14099), qui ne décodent que les 3 bits de poids

faible, et en faisant la sélection de boitier avec les 9 bits restant, sans décodage, il reste une capacité d'adressage de:

$2^3 \times 9 = 72$ entrées

72 sorties

72 variables internes.

Une partie de ces variables internes est affectée à l'adressage des registres de demande de tâches et résultats permettant le dialogue entre les deux microprocesseurs. Une partie est également réservée aux variables intermédiaires.

Pour permettre les branchements conditionnels, nous utilisons l'instruction SKZ du microprocesseur (codée $E000_H$) qui permet la non exécution de l'instruction suivante si le contenu du registre résultat est 0.

3.3.3 - Schéma fonctionnel de l'automate logique (fig. III.6)

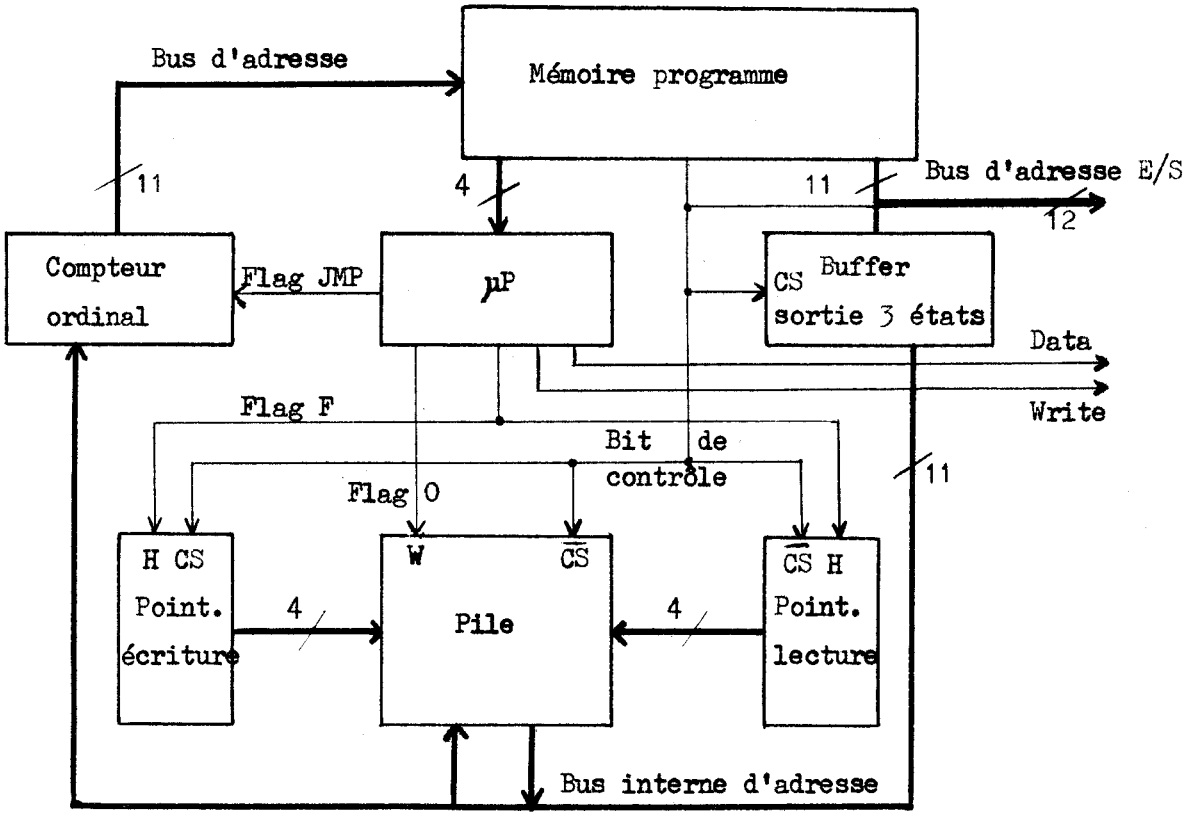


Schéma fonctionnel de l'unité centrale

- fig. III.6 -

Nous avons créé un bus interne d'adresse permettant les échanges de la mémoire programme vers la pile, de la mémoire programme vers les entrées de chargement du compteur ordinal, de la pile vers ces mêmes entrées de chargement. Ce bus est distinct du bus externe qui adresse les circuits d'entrée/sortie. Ceci permet de conserver la capacité d'adressage maximale en entrée/sortie.

3.3.4 - Présentation du logiciel permettant la gestion du réseau

A partir des choix faits au cours des paragraphes précédents, nous allons définir l'algorithme devant permettre la gestion du réseau. Nous pouvons envisager deux solutions pour l'implantation de ce réseau.

a / Le réseau est entré sous forme d'un ensemble de données. Nous élaborons un programme interpréteur permettant à la fois l'interprétation et l'exécution de ces données.

b / Le réseau est entré sous forme d'un programme spécifique à l'application. La traduction peut en être faite par l'utilisateur ou par un compilateur résidant dans la console de programmation.

Nous avons adopté la solution b qui donne une plus grande vitesse de traitement. Toutefois elle nécessite, soit un effort supplémentaire de la part de l'utilisateur, soit une console de programmation plus complexe.Pour permettre la mise au point il est alors utile de conserver en mémoire les programmes source et objet.

3.3.4.1 - Structure générale du programme

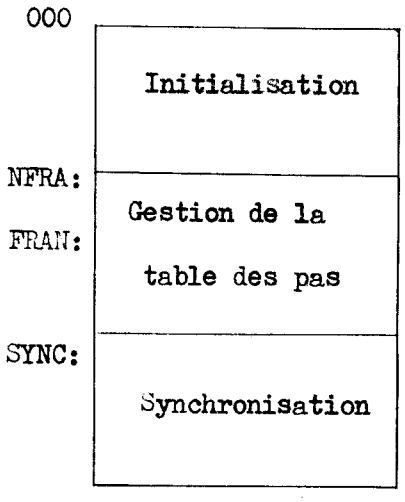
Nous distinguons dans le programme deux parties. La premiére correspond à un programme moniteur valable pour toutes les applications. Il permet: (Fig. III.7a)

- la mise en service de l'installation

- la gestion de la table des pas
- le cadencement des dispositifs d'entrée/sortie (réalisation synchrone).

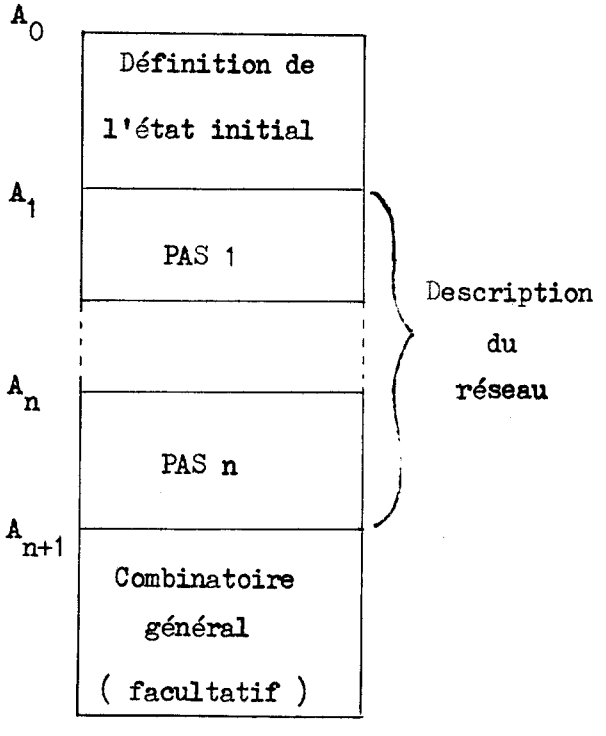
La deuxième, spécifique à l'application, permet la présentation des données nécessaires à (Fig. III.7b):

- la définition de l'état initial
- la description complète du réseau de Pétri
- la description du combinatoire général.



Organisation du programme moniteur

- fig. III.7a -



Organisation du programme spécifique à une application.

- fig. III.7b -

3.3.4.2 - Présentation du moniteur

Ce programme indépendant de l'application a trois fonctions que nous développons. Ces programmes sont donnés dans l'annexe II.

a) Mise en service de l'installation

Ce programme nommé initialisation débute à l'adresse 0. Il permet:

- La mise à 1 des registres IEN et OEN du microprocesseur. [15]

- Le transfert des valeurs des entrées dans les registres d'entrée.
- L'inscription dans la table des pas de l'adresse A_0 et de l'indicateur fin de table.

Nous rappelons que cet indicateur sert de séparation entre les deux tables (§ 3.3.1).

b) Gestion de la table des pas

Ce programme nommé programme principal est accessible par les adresses repérées par les labels non franchie (NFRA) et franchie (FRAN). Il provoque l'incrémentation du pointeur d'écriture lorsqu'aucune transition du pas n'a été franchie. Ceci valide l'inscription de l'adresse A_i qui est faite au début du pas qui vient d'être testé dans la nouvelle table (§ 3.3.4.3).
Dans tous les cas, après incrémentation du pointeur de lecture, il force le compteur ordinal à l'adresse lue dans la table (pas suivant). Ce programme est heureusement très court. C'est le temps d'exécution de celui-ci qui détermine le temps d'accès t_a à un pas. Il est pour chaque pas de:

- 4 μs si aucune transition n'est franchie
- 2 μs si une transition est franchie

à la vitesse maximale de travail du microprocesseur.

c) Cadencement des entrées/sorties

Ce programme de synchronisation (label SYNC) permet le transfert dans les registres d'entrée et de sortie des grandeurs correspondantes.

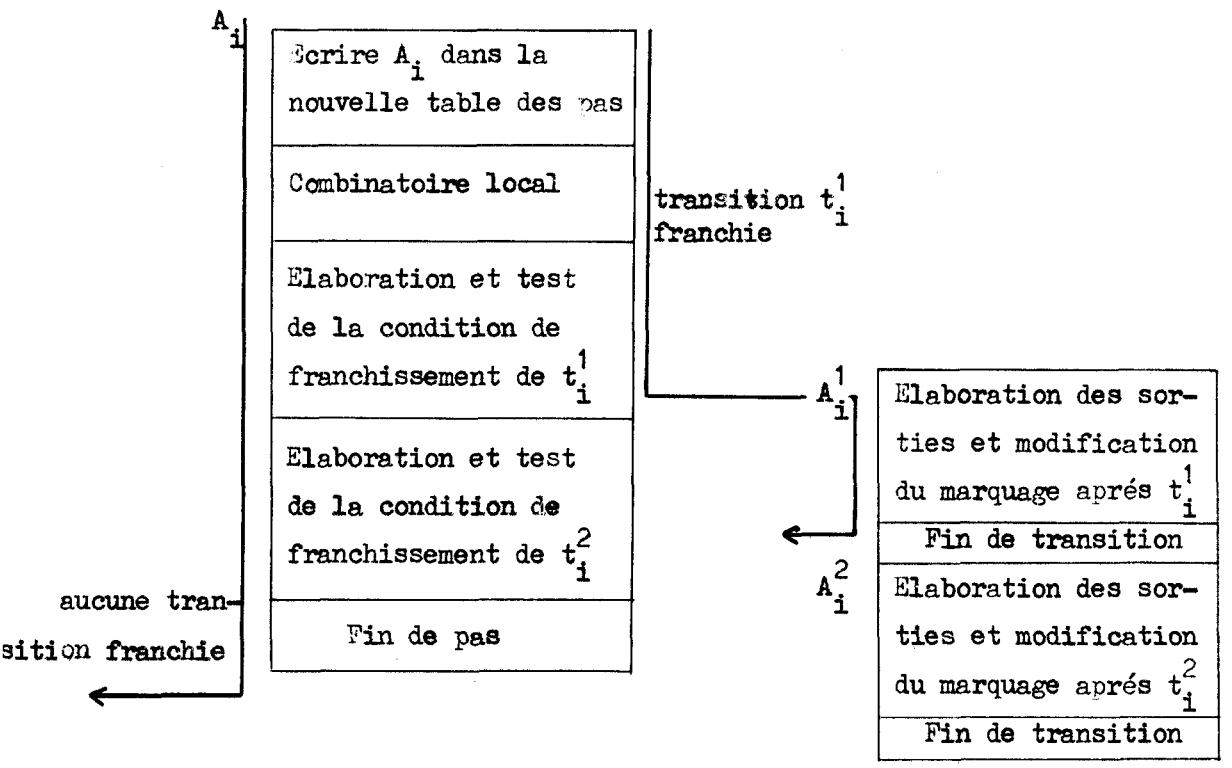
Ce programme est conçu comme un pas commençant à l'adresse repérée par le label SYNC. Il est naturellement appelé par le programme principal lorsque le pointeur de lecture est positionné sur une case contenant son adresse. Ce programme commence par l'instruction permettant d'écrire son adresse de début. Il se termine par l'indicateur fin de pas. Il contient les instructions permettant le positionnement du signal de synchronisation (flag RTN).

3.3.4.3 - Présentation du programme spécifique à une application

Chacune des n + 2 parties de ce programme est traitée si et seulement si son adresse de début A_i est contenue dans la table des pas. (fig. III.7b)

- Programme constitutif d'un pas:

Dans un réseau de Pétri interprété, il n'y a pas de conflit. Nous trouvons donc au plus une transition franchissable par pas. Il est inutile de traiter les autres transitions d'un pas après y avoir détecté une transition franchissable. Le programme relatif à un pas prend la forme suivante (fig.III.8).



Représentation d'un pas contenant deux transitions

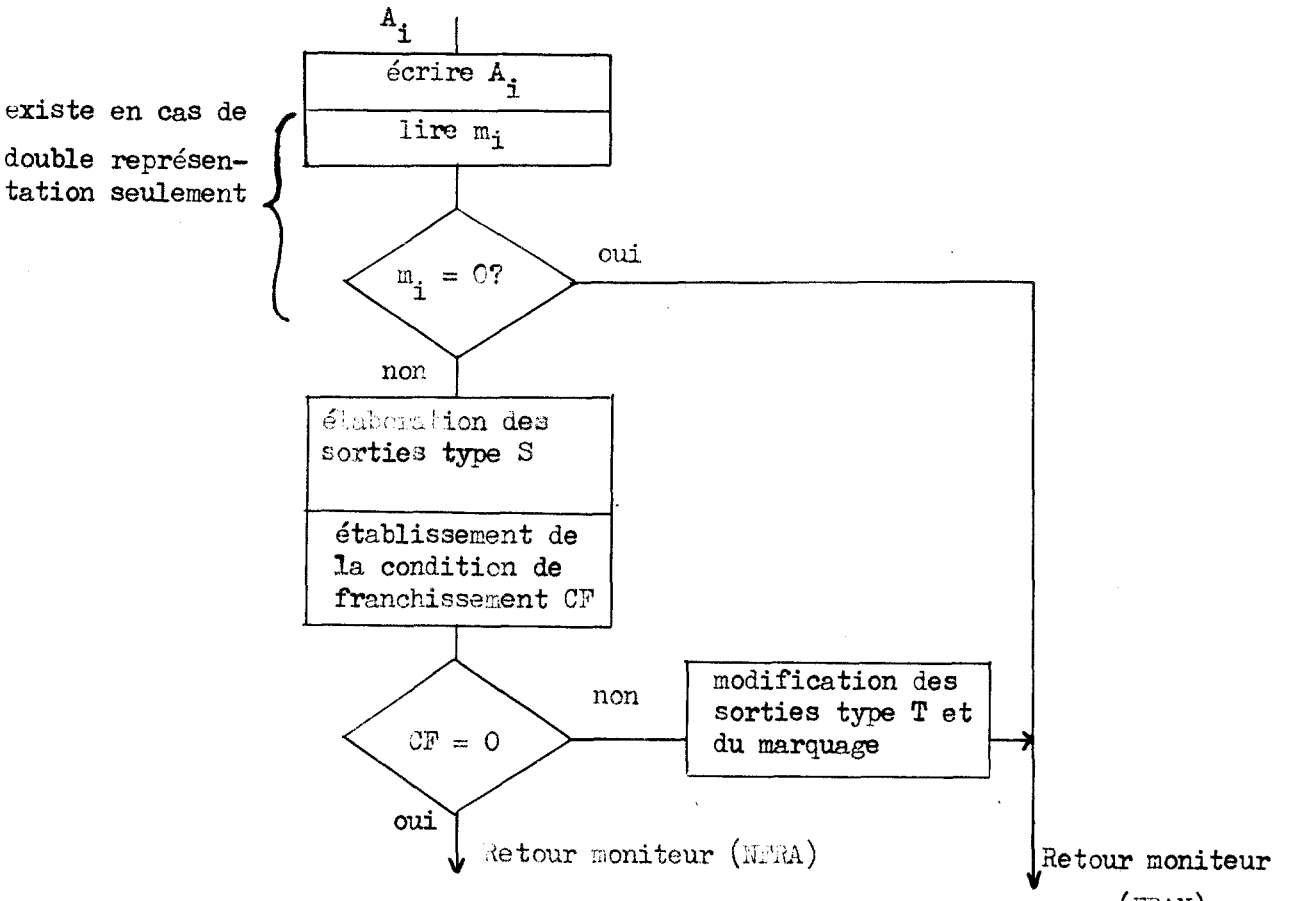
- fig. III.8 -

L'indicateur de fin de transition est rencontré si et seulement si une transition est franchie. Il provoque le retour au moniteur à l'adresse FRAN.

L'indicateur de fin de pas est rencontré si et seulement si aucune transition du pas n'est franchie. Il provoque le retour au moniteur à l'adresse NFRA.

Le test sur la condition de franchissement relatif à une transition permet le branchement conditionnel à la zone du programme qui contient les instructions nécessaires à la modification des sorties et du marquage. Pour permettre la gestion de la table des pas, la première instruction du pas provoque l'inscription de l'adresse de début de ce pas dans la nouvelle table des pas. Cette adresse sera "écrasée" si une transition du pas est franchie.

REMARQUE: Au paragraphe 3.2.2 nous avons justifié la nécessité de tester, en début de pas, l'état de la variable interne qui est associée à la place clef lorsque celle-ci est aussi place de synchronisme (double représentation). Dans ce cas, après avoir inscrit l'adresse de début de pas dans la nouvelle table, nous lisons et testons la valeur de la variable interne (fig. III.9). Si cette variable interne est à 0 (place démarquée), nous faisons un retour au moniteur à l'adresse FRAN. Ceci permet de supprimer l'adresse de début de pas de la nouvelle table (cf. exemple en annexe II et figure III.11a).



Organigramme relatif à un pas contenant une transition

- fig. III.9 -

- Programme relatif à l'initialisation:

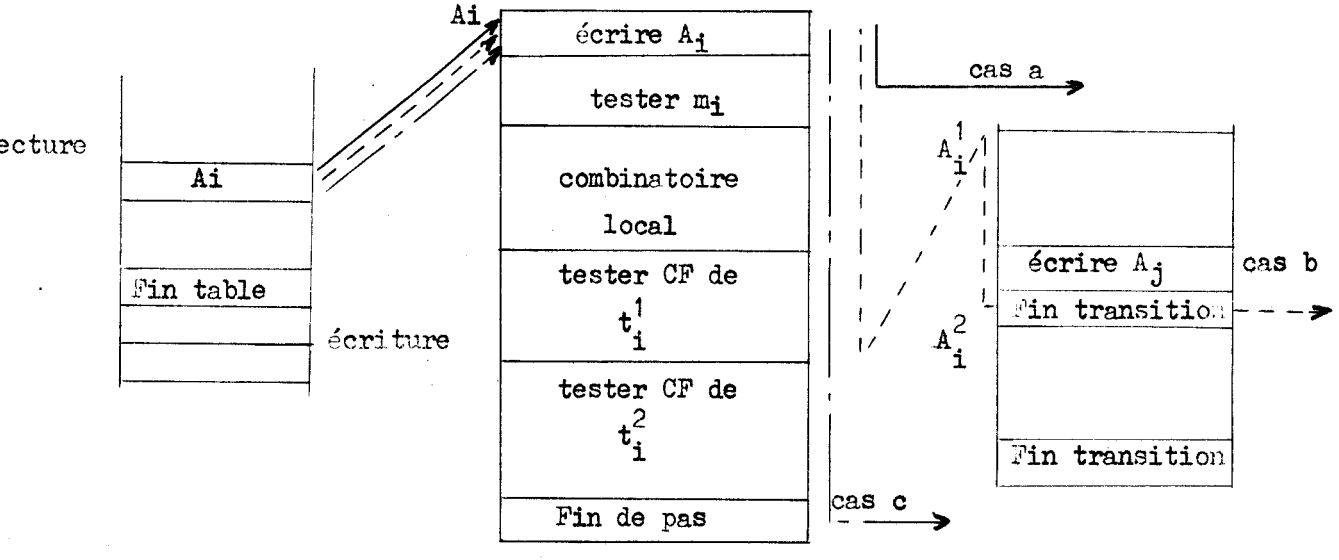
C'est en fait le programme correspondant au pas qui est associé à la place marquée initiale unique (toujours place clef) qui a éventuellement été ajoutée (§ 1.2.4.3).

Sachant qu'à la mise en service toutes les sorties sont mises à 0, seules les sorties qui doivent être mises à 1 sont à définir. Ce programme commence par les instructions permettant l'inscription de A_{n+1} (combinatoire général) puis A_0 dans la table. Il est ensuite constitué comme un pas banal.

- Programme relatif au combinatoire général:

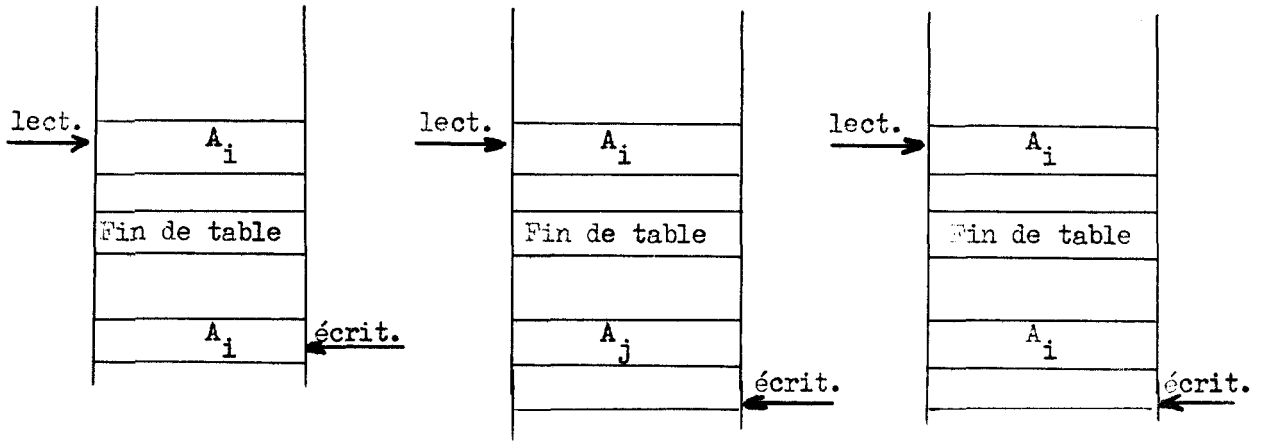
Il commence par l'instruction permettant l'inscription de A_{n+1} dans la table. Il se termine par l'indicateur fin de pas.

Les figures III.11a, III.11b, III.11c montrent diverses évolutions possibles du contenu de la table des pas à partir de la situation définie par la figure III.10.



Contenu initial de la table des pas dans l'exemple de gestion de cette table.

- fig. III.10 -



Place clef démarquée (double représentation) (a)

Transition t_i^1 franchie (b)

Aucune transition franchie (c)

Contenu de la table des pas après scrutation du pas

- fig. III.11 -

3.4 - Automate numérique

Cet automate bâti autour d'un microprocesseur multibit effectue des traitements opérations numériques simples sur des opérandes qui sont:

- des constantes
- des nombres contenus en mémoire
- des valeurs numériques se rapportant à un port d'entrée/sortie.

Les opérations envisagées sont: l'addition, la soustraction, la comparaison entre deux opérandes ($=; <; >$), le transfert d'un résultat vers un port de sortie ou une case mémoire. Cet automate gère également les temporisations.

3.4.1 - Dialogue entre les microprocesseurs

A chaque tâche numérique d'ordre i est associée un registre de demande de tâche appelé Ti et un registre résultat Ri [illegible] si le registre résultat correspondant est inutilisé.

3.4.1.1 -Demandes de tâches numériques autres que les temporisations

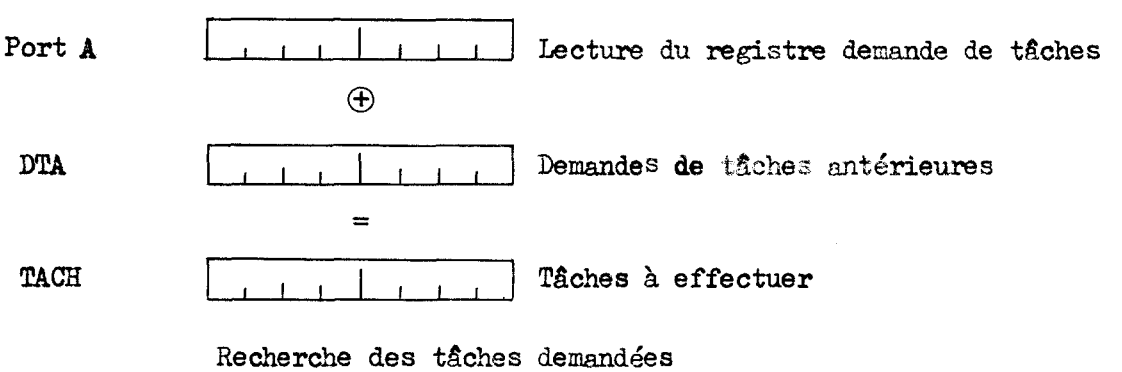
Si nous nous interessons aux traitements numériques, autres que les temporisations, nous constatons que toutes les demandes de tâches peuvent être de nature impulsionnelle. Les opérations correspondantes ont en effet une durée propre. En conséquence, nous avons choisi d'effectuer les demandes de tâches par modification de la valeur affectée à la variable Ti. Ceci nous permet d'éviter les signaux d'acquittement généralement utilisés dans ce genre de dialogue. Cette solution est acceptable car la périodicité des demandes de tâches est faible vis-à-vis du temps de traitement. Il n'y a donc jamais de microprocesseur à arrêter.

Pour faire une demande de tâche, l'automate logique complémente la valeur de Ti. Les instructions sont:

LD Ti $(Ti) \rightarrow RR$

STOC Ti $(\overline{RR}) \rightarrow Ti$

Pour détecter les demandes de tâche, l'automate numérique compare les valeurs des variables Ti à celles qu'elles avaient lors de la scrutation précédente. Pour cela il garde en mémoire les valeurs antérieures des registres Ti dans un mot noté DTA (Demande de Tâches Antérieures). Nous élaborons alors un mot (notation TACH) dans lequel les bits à 1 indiquent les tâches à effectuer (fig. III.12).



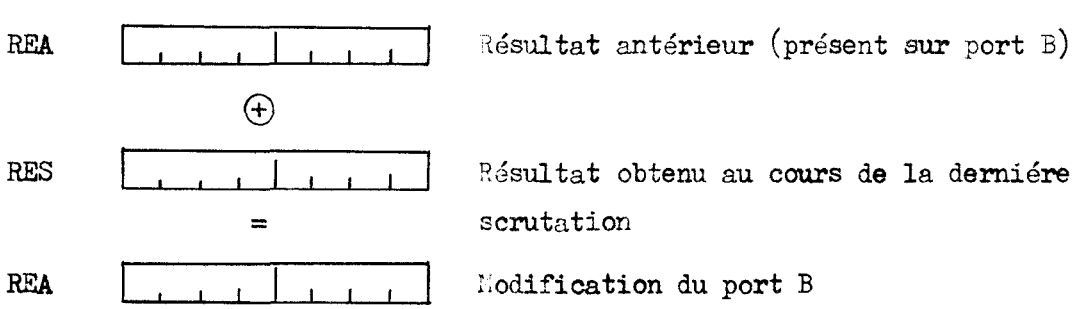
- fig. III.12 -

Une table des tâches donne les adresses de début des programmes correspondant aux diverses tâches. Le programme de gestion des demandes de tâche en gère aussi le pointeur.

Remarque: Nous avons fait ici les demandes de tâche en utilisant un port de l'automate numérique (huit tâches demandées). Il est possible de prévoir extérieurement un registre à décalage de demande de tâche et d'utiliser l'entrée série de donnée. Ceci libère les ports d'entrées et permet une extension du nombre de tâches possibles. La seule limite au nombre de tâches est alors le temps de traitement et éventuellement la taille mémoire.

3.4.1.2 - Résultat des tâches numériques autres que les temporisations

Les résultats des tâches numériques affectent un port de sortie (noté port B) qui sert de registre résultat. Pour une tâche de rang i, la valeur du bit de même rang de ce port est modifié si le résultat attendu est obtenu. Il ne l'est pas dans le cas contraire ou si la tâche correspondante n'a pas été demandée. Le mot ainsi obtenu est noté REA (résultats antérieurs). Il est déterminé à partir du mot résultat (noté RES) dans lequel seuls les bits correspondants aux tâches dont le résultat vient d'être obtenu, sont à 1.



Affectation des registres résultat

- fig. III.13 -

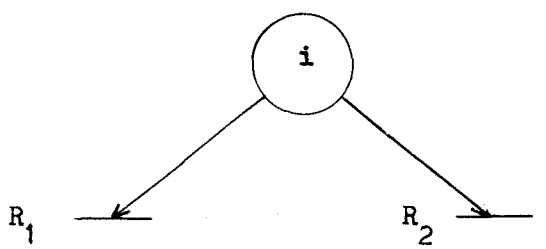
Remarque: Pratiquement, lorsqu'une tâche mettant en oeuvre une comparaison est demandée, c'est que le résultat de cette comparaison est attendu pour permettre l'évolution du marquage du réseau. Le fait que le résultat attendu ne soit pas obtenu n'est pas une information en soi car cela regroupe les cas suivants:

- la tâche n'a pas été demandée (normalement dans ce cas le résultat n'est pas lu)
- la tâche n'a pas encore été exécutée
- la tâche a été exécutée et le résultat de la comparaison est négatif.

Exemple:

$R_1 = 1$ si $A = B$

$R_2 = 1$ si $A > B$



Influence des tâches numériques dans l'évolution du marquage du réseau

- fig. III.14 -

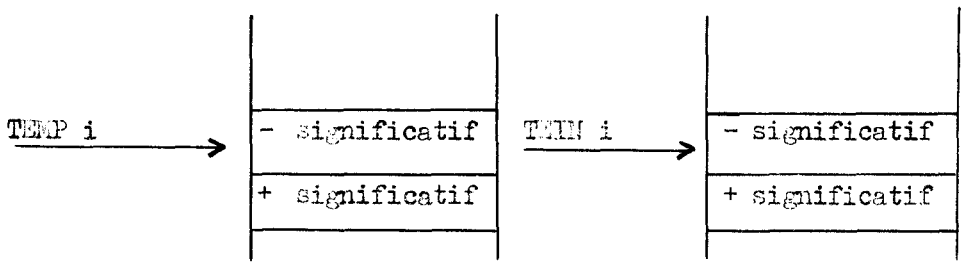
Supposons que l'on aie deux évolutions distinctes selon que les expressions $A = B$ ou $A > B$ sont vérifiées (Fig. III.14). Il faut alors utiliser deux tâches T_1 et T_2 distinctes. Le marquage de la place i est maintenu tant que les résultats R_1 ou R_2 ne sont pas présents.

Pour la lecture du registre résultat par l'automate logique, nous avons adopté une solution câblée permettant de détecter les variations de niveau sur une grandeur d'entrée. Ce dispositif d'entrée est présenté dans la deuxième partie (chapitre V) car il est utilisé également pour la détection de défauts. L'organigramme permettant la gestion de l'accès aux tâches numériques et l'élaboration du résultat est donné en annexe II.

3.4.2 - Temporisations

Désirant pouvoir armer et arrêter les temporisations, nous avons choisi de faire les demandes de temporisations par des variables de niveau. La temporisation reste active tant que la demande de temporisation correspondante est maintenue à 1.

A chaque temporisation i, nous affectons deux mots (de 16 bits dans notre réalisation) qui sont repérés par TEMP i (valeur de la temporisation) et TEIN i (valeur intermédiaire). Les mots sont inscrits en mémoire (fig. III.15).



Représentation mémoire d'une temporisation

- fig. III.15 -

A chaque front d'un signal d'horloge, la valeur de TEIN i est décrémentée si la temporisation i est active. Le registre résultat Ri correspondant est mis à 0 si TEIN i est à 0. La valeur de TEMP i peut être fixée au moment de la programmation ou à la suite d'une tâche numérique décrite par ailleurs.

Lorsque la temporisation i est activée, nous entamons une procédure d'initialisation qui consiste à donner à TEIN i la valeur de TEMP i. Les demandes de temporisation sont présentes sur un port d'entrée (noté Port C) et les résultats sont transférés sur un port de sortie (noté Port D). Pour la gestion des temporisations le programme correspondant élabore les mots suivants:

TEAN qui représente les temporisations actives dans la scrutation précédente. Il mémorise le contenu du port C entre deux scrutations.

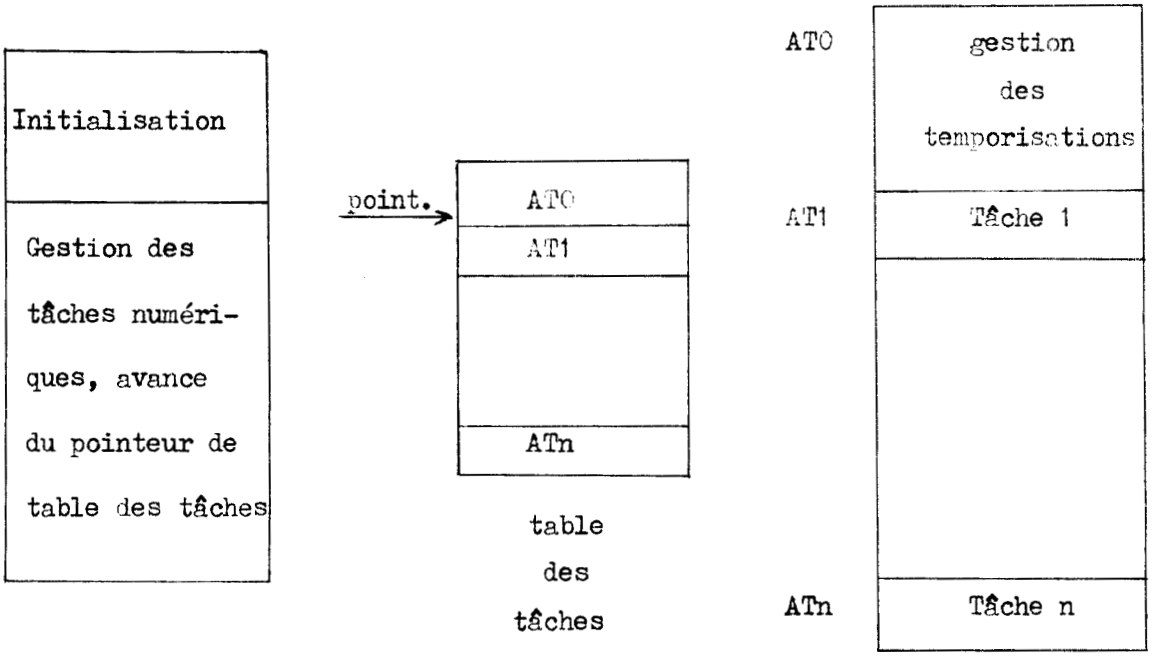
TINI qui représente les temporisations devant être initialisées. TINI est obtenu en faisant (Port C) + TEAN

FTEM qui représente les fins de temporisation. Après élaboration ce mot est transmis au Port D.

3.4.3 - Organisation du programme de l'automate numérique

Le signal d'horloge utilisé comme base de temps pour les temporisations est raccordé à l'une des entrées de demande de tâche définie précédemment.

L'organisation générale du programme de l'automate numérique est donnée figure III.16.



Organisation du programme de l'automate numérique

- fig. III.16 -

Les programmes de gestion des tâches et des temporisations sont valables pour toutes les applications. Seules les adresses contenues dans la table des tâches et les valeurs des temporisations sont modifiables. La localisation dans la mémoire de ces tables peut être fixée.

Dans notre dispositif de commande expérimental, nous nous sommes limité à huit temporisations et huit tâches numériques. Les organigrammes et les programmes sont donnés en annexe II.

3.5 - Conclusion

Dans ce chapitre nous avons présenté l'architecture du dispositif de commande ainsi que le logiciel permettant à chacun de nos automates d'effectuer les tâches qui lui sont dévolues. Nous avons été amené à étendre le code opération du microprocesseur monobit utilisé. Il reste à définir un langage utilisateur permettant d'entrer les données et d'assurer un développement aisé du programme relatif à une application. Ces aspects permettant la mise en oeuvre d'une console de programmation n'ont pas été abordés.

C H A P I T R E IV

AMELIORATION DE LA SURETE DE FONCTIONNEMENT D'UNE MACHINE SEQUENTIELLE

Les causes de mauvais fonctionnement d'un systéme contrôlé par un dispositif de commande logique et numérique sont nombreuses. De ce fait une sûreté de fonctionnement totale est utopique. La lutte contre ces défauts doit être évidemment préventive. Néanmoins un dépistage précoce des défaillances du matériel doit permettre un arrêt (ou une mise en sauvegarde) de la machine avant que des ordres, non attendus normalement, puissent être donnés.

Dans ce chapitre nous proposons une surveillance accrue de la partie opérative au travers des comptes rendus qu'elle fournit à la commande.

4.1 - Causes et effets des pannes d'un systéme à commande numérique et logique

Nous nous intéressons ici à des systémes éprouvés, ou supposés tels, fonctionnant donc conformément au cahier des charges en l'absence de défauts. Les mauvais fonctionnements de tels dispositifs sont alors dus:

- soit à des pannes physiques sur les capteurs, les actionneurs, leurs liaisons avec le dispositif de commande, le dispositif de commande lui même;
- soit aux parasites industriels.

Dans tous les cas cela se traduit, au bout d'un temps plus ou moins long:

- soit par la création d'un marquage impossible du réseau de

Pétri (parasitage de l'unité de traitement elle même);

- soit par l'arrivée d'un compte rendu non conforme à l'état réel de la partie opérative (parasites sur les entrées, défauts de capteur ou de liaison);
- soit par une mauvaise interprétation, par la partie opérative, des ordres reçus (défauts sur les actionneurs, leurs liaisons ou les interfaces).

Toutes les précautions doivent être prises pour limiter l'influence des parasites sur le comportement de la partie opérative. Nous trouvons entre autres la séparation galvanique entre, les entrées et sorties de l'unité centrale et les capteurs et actionneurs. Les photocoupleurs sont généralement utilisés dans ce but. L'utilisation de ces circuits de l'optoélectronique, nécessitant une énergie de commande relativement grande et un temps de réponse assez long, ont un effet antiparasite propre non négligeable.

4.2 - Réseaux de Pétri pondérés [14]

- Poids d'une place: nous obtenons un réseau de Pétri pondéré en attribuant à chaque place P_i un nombre entier strictement positif p_i appelé poids de la place P_i.

La valeur de p_i est choisie de façon telle que, pour toute transition t_j du réseau, la somme des poids des places antécédentes soit égale à la somme des poids des places subséquentes. Cette définition n'est applicable qu'à des réseaux de Pétri pour lesquels chaque transition a au moins une place antécédente et une place subséquente.

Si le coefficient a_{ij} de la matrice G est -1 et si le coefficient a_{lj} de la matrice G est +1, alors $\sum_i p_i = \sum_l p_l$.

- Poids total marqué d'un réseau: la quantité $\sum_i m_i p_i$, où m_i est le nombre de marqueurs de la place P_i de poids p_i, est appelée poids total marqué du réseau.

Un réseau est dit sauf si, pour le marquage initial M_o et pour tout marquage du réseau accessible à partir de M_o, aucune place ne peut posséder plus d'un marqueur.

Nous nous interessons uniquement au cas des réseaux de Pétri interprétés saufs. Pour nous, m_i est au plus égal à 1.

Pour tout réseau de Pétri pondéré, le poids total marqué reste constant à partir du marquage initial pour toute séquence de franchissements de transitions. La règle d'évolution des marquages des réseaux est telle que, au franchissement d'une transition t_j, la diminution du poids total marqué, liée au démarquage des places antécédentes à t_j est:

$$\sum_i m_i p_i - \sum_i (m_i - 1) p_i = \sum_i p_i$$

$\forall$ i tel que le coefficient a_{ij} de la matrice G soit -1

De même l'augmentation du poids total marqué liée au marquage des places subséquentes à t_j est:

$$\sum_l (m_l + 1) p_l - \sum_l m_l p_l = \sum_l p_l$$

$\forall$ l tel que le coefficient a_{lj} de la matrice G soit +1

L'augmentation du poids total marqué après franchissement de la transition j est: $\sum_l p_l - \sum_i p_i = 0$

Cette quantité est toujours nulle par définition des réseaux pondérés.

Plusieurs approches sont possibles pour trouver la pondération du réseau [14] .L'utilisation des réseaux pondérés permet de détecter des marquages non valables. Dans les réalisations câblées, cela permet de détecter une déficience des mémoires représentant les places. En particulier, au franchissement d'une transition, nous verrons si une mémoire

subséquente ne reste pas accrochée ou si une mémoire antécédente reste marquée. Dans une réalisation programmée éprouvée (dont le programme est donc réputé répondre au cahier des charges), ce genre d'incident est peu courant. Une telle panne de l'unité centrale est généralement plus globale et la question est de savoir alors si ce système en panne est encore capable d'effectuer un traitement. Par contre, l'utilisation des réseaux de Pétri pondérés doit permettre de détecter des modifications du marquage dues à l'action directe de parasites. A ce titre, les réseaux pondérés nous semblent interessants. Toutefois le système de commande mis en oeuvre n'exploite pas cette possibilité. En effet les fonctionnements défectueux sont essentiellement liés à la réception de comptes rendus, issus de la partie opérative, erronés soit à cause de défauts sur les capteurs ou leurs liaisons, soit par parasites.

4.3- Détections des pannes sur les capteurs et leurs liaisons

Une réalisation permettant le test en ligne est possible. [14] Pour la partie purement séquentielle les possibilités des réseaux pondérés ont été exploitées. Pour la partie combinatoire correspondant à l'établissement des conditions d'évolution et des sorties, la méthode de Diaz [10] a été utilisée. Elle conduit à des circuits combinatoires totalement autotestables.

4.3.1 - Circuits combinatoires autotestables

Pour obtenir ce résultat, chaque information binaire est en fait donnée par un couple de variables binaires dont les valeurs appartiennent en fonctionnement normal au code un parmi deux. Chaque fonction

binaire est également représentée par un couple de fonctions binaires prenant leurs valeurs dans (0,1) ou (1,0).

Lorsqu'un câble de liaison est coupé, ou qu'un contact ne se ferme plus, un couple de variables peut prendre ses valeurs dans (0,0). De même le collage d'un contact peut, si l'autre contact reste mobile, donner un couple de variables prenant ses valeurs dans (1,1). Dans les deux cas le couple de fonctions prendra ses valeurs également dans (0,0) ou (1,1) ce qui permet de détecter une panne.

Cette méthode ne permet de détecter que certains défauts. Elle est applicable avec des capteurs entrainants des contacts électriques. Elle ne l'est plus avec les capteurs électroniques (détecteur de proximité). Elle est également inexploitable lorsque l'événement prend la forme d'un prédicat nécessitant par exemple un calcul numérique. De plus en augmentant le nombre de contacts et de liaisons, la fiabilité de l'ensemble diminue.

Nous envisageons donc une autre solution qui doit permettre, en évitant au maximum d'augmenter le nombre des capteurs, de détecter un défaut sur ceux-ci.

4.3.2 - Description d'un système de commande à surveillance accrue

4.3.2.1 - Réceptivité et sensibilité d'une machine séquentielle

Dès la description de niveau I [22] traduisant les spécifications fonctionnelles du cahier des charges, les informations utilisées sont, en général, réduites au minimum. En définitive, parmi l'ensemble des comptes rendus fournis par la partie opérative et des ordres reçus de l'environnement, une partie seulement de ces informations est exploitée par le dispositif de commande.

Nous rappelons les définitions de la réceptivité et de la sensibilité vues au paragraphe 1.2.6. A chaque transition du réseau de Pétri est associé un événement formé d'un prédicat qui sera traduit ultérieurement par des expressions logiques. Selon son comportement vis-à-vis de ces événements nous dirons qu'une machine séquentielle est:

- <u>réceptive</u>, pour un marquage donné, à un événement e, si cet événement est capable de faire évoluer le marquage du réseau.
- <u>sensible</u>, pour un marquage donné, à un événement e, si cet événement, sans modifier le marquage du réseau, est capable de faire évoluer l'état des sorties du système de commande.

Pour éviter le combinatoire de sortie, en vertu des principes présentés dans les chapitres précédents, nous représentons les spécifications fonctionnelles par un réseau de Pétri tel que l'ensemble des événements, qui rendent la machine séquentielle sensible, est vide quelque soit le marquage.

Ayant défini:

- un ensemble fini, non vide, d'événements

$$\mathcal{E} = \{ e_1, e_2, \dots e_n \}$$

- un ensemble fini, non vide, de sorties

$$\mathcal{S} = \{ s_1, s_2, \dots s_k \}$$

- un ensemble fini, non vide, de marquages du réseau

$$\mathcal{M} = \{ M_1, M_2, \dots M_i \}$$

nous introduisons une application multivoque $\varphi_{(M)}$ de $\mathcal{M}$ dans $\mathcal{E}$. A chaque élément M de $\mathcal{M}$, elle fait correspondre un sous-ensemble de $\mathcal{E}$ auquel la machine est réceptive. Ce sous-ensemble est composé des événements qui

font évoluer le marquage. $\varphi_{(M)}$ est appelée fonction de réceptivité. Nous introduisons également une application multivoque $\sigma_{(M)}$ de $\mathcal{M}$ dans $\mathcal{E}$. A chaque élément M de $\mathcal{M}$, elle fait correspondre un sous ensemble de $\mathcal{E}$ auquel la machine est sensible. Ce sous ensemble est formé d'événements modifiant la configuration des sorties sans faire évoluer le marquage. $\sigma_{(M)}$ est appelée fonction de sensibilité.

Pour un marquage donné notre machine séquentielle est non réceptive à un certain nombre d'événements. En effet la fonction de réceptivité $\varphi_{(M)}$ pour le marquage M est définie sur une partie de l'ensemble des événements. Il n'est pas question de rendre le système de commande réceptif à toute modification d'une variable, ce qui nous ramènerait au graphe de fluence. Toutefois il est possible d'imaginer que notre machine séquentielle, pour un marquage donné, soit réceptive non seulement aux événements qui entrainent une modification des sorties (ou éventuellement un comptage d'événements) mais également à ceux qui, pour le marquage en cours, ont une probabilité de réalisation certaine au bout d'un temps plus ou moins long.

4.3.2.2 - Passivité d'une machine séquentielle

Parmi les événements qui ne participent pas, pour un marquage donné, à la réceptivité ou à la sensibilité, nous distinguons:

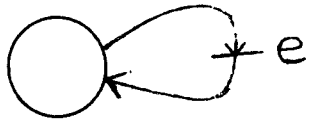
- les événements dont la réalisation est possible
- les événements dont la probabilité de réalisation est nulle en fonctionnement normal.

Nous dirons qu'une machine séquentielle est passive, pour un marquage M donné, vis à vis d'un événement e, si cet événement, ayant une probabilité de réalisation non nulle en fonctionnement normal, ne participe

ni à la fonction de réceptivité $\rho_{(M)}$, ni à la fonction de sensibilité $\sigma_{(M)}$.

Tout événement, reçu à un moment donné par un dispositif de commande séquentiel, qui ne participe ni à la réceptivité, ni à la sensibilité, ni à la passivité est une anomalie de fonctionnement. Il doit provoquer un arrêt de la machine.

Un événement e pour lequel la machine séquentielle est passive pour une place marquée donnée sera associé à une transition admettant cette place à la fois comme place antécédente et subséquente (Fig. IV .1).



Représentation de la fonction de passivité.

- fig. IV.1 -

4.3.3 - Description d'une machine séquentielle à partir des variations de niveaux logiques des entrées

Pour obtenir le degré de surveillance visé des comptes rendus à partir des niveaux logiques des entrées, nous sommes obligés de dresser, pour chaque marquage, la liste des variables d'entrées qui peuvent prendre indifféremment la valeur "1" ou "0".

Une méthode équivalente consiste alors à décrire le fonctionnement de la machine séquentielle, chaque fois que cela est possible, à partir des variations de niveau logique des variables d'entrées. Aux variables d'entrée nous associons alors une variable dérivée qui prend la valeur "1" lorsque le niveau logique de la variable d'entrée est modifié.

Elle reprend la valeur "0" après prise en compte par le système de commande. La variable dérivée associée à une entrée X est notée $\dot{X}$.

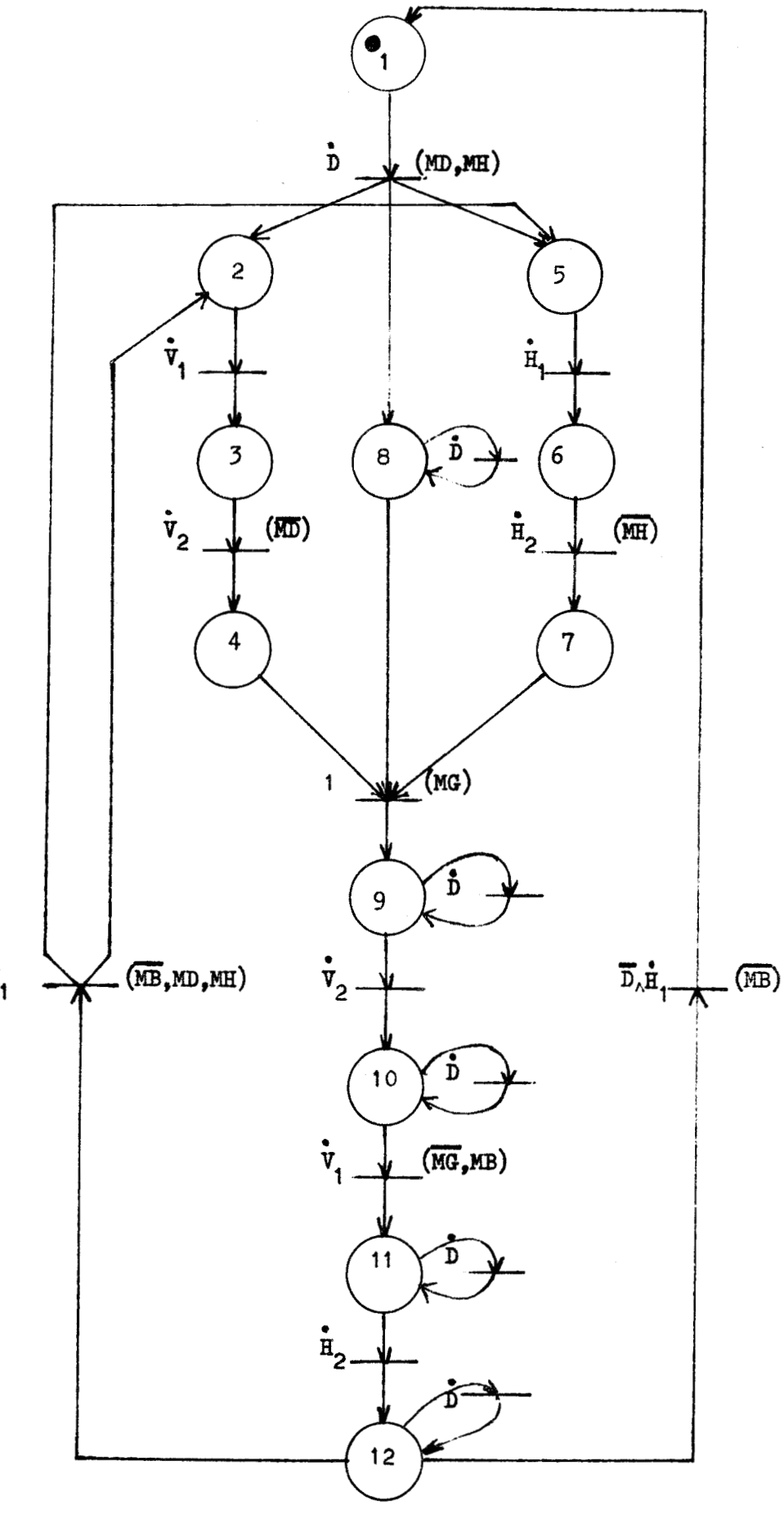
Il est parfois indispensable de prendre en référence les niveaux de certaines variables d'entrée pour permettre, par exemple, une sélection de cycle. Cette notion sera reprise à partir d'une classification des variables d'entrée. Nous illustrons cette description à partie des variables dérivées par les exemples suivants.

Exemple I:

Reprenons le dispositif décrit par la fig. I.4. Le réseau de Pétri de la fig.IV.2 permet une description du fonctionnement satisfaisante.

La variable "D" donnant l'information de départ est ici représentée en sélection de cycle. Si la valeur de D est "0" en fin de cycle, le plateau est arrêté jusqu'à ce que D passe à "1", sinon le cycle est repris.

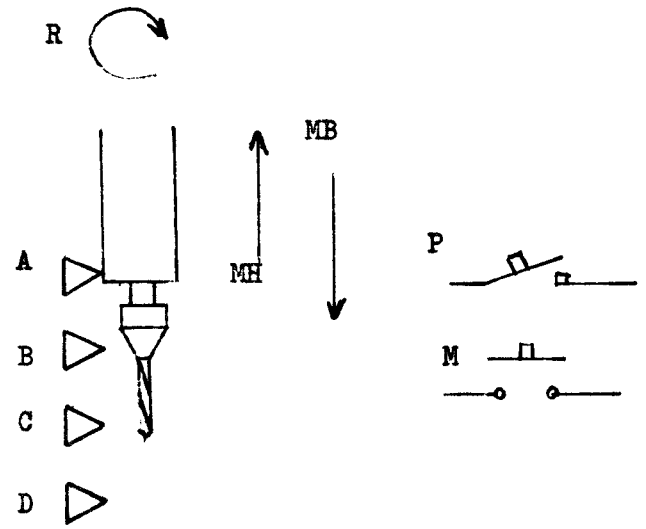
Nous remarquons que la variation de D est autorisée quelque soit le marquage. En effet, D fait partie, pour tout marquage, de la fonction de passivité ou de réceptivité. Dans ce cas, le contrôle des variations de D ne permet plus de détecter les défauts de fonctionnement du dispositif technologique correspondant (bouton poussoir par exemple). Pour alléger le graphisme nous supprimons dans ce cas les parties de réseau correspondantes.



Réseau utilisant les variables dérivées associées aux entrées.

- fig. IV .2 -

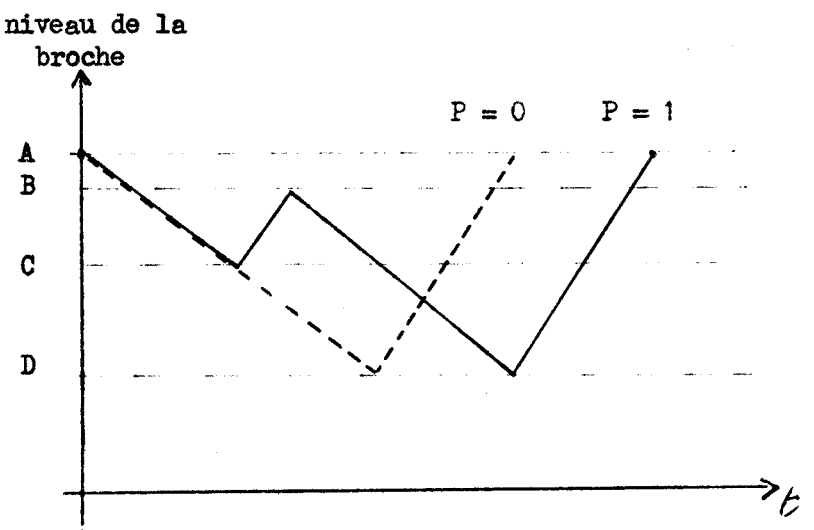
Exemple II: Perçeuse avec ou sans débourrage (Fig. IV .3) [17]



P = 1 perçage avec débourrage

P = 0 perçage sans débourrage

a) Dispositif mécanique

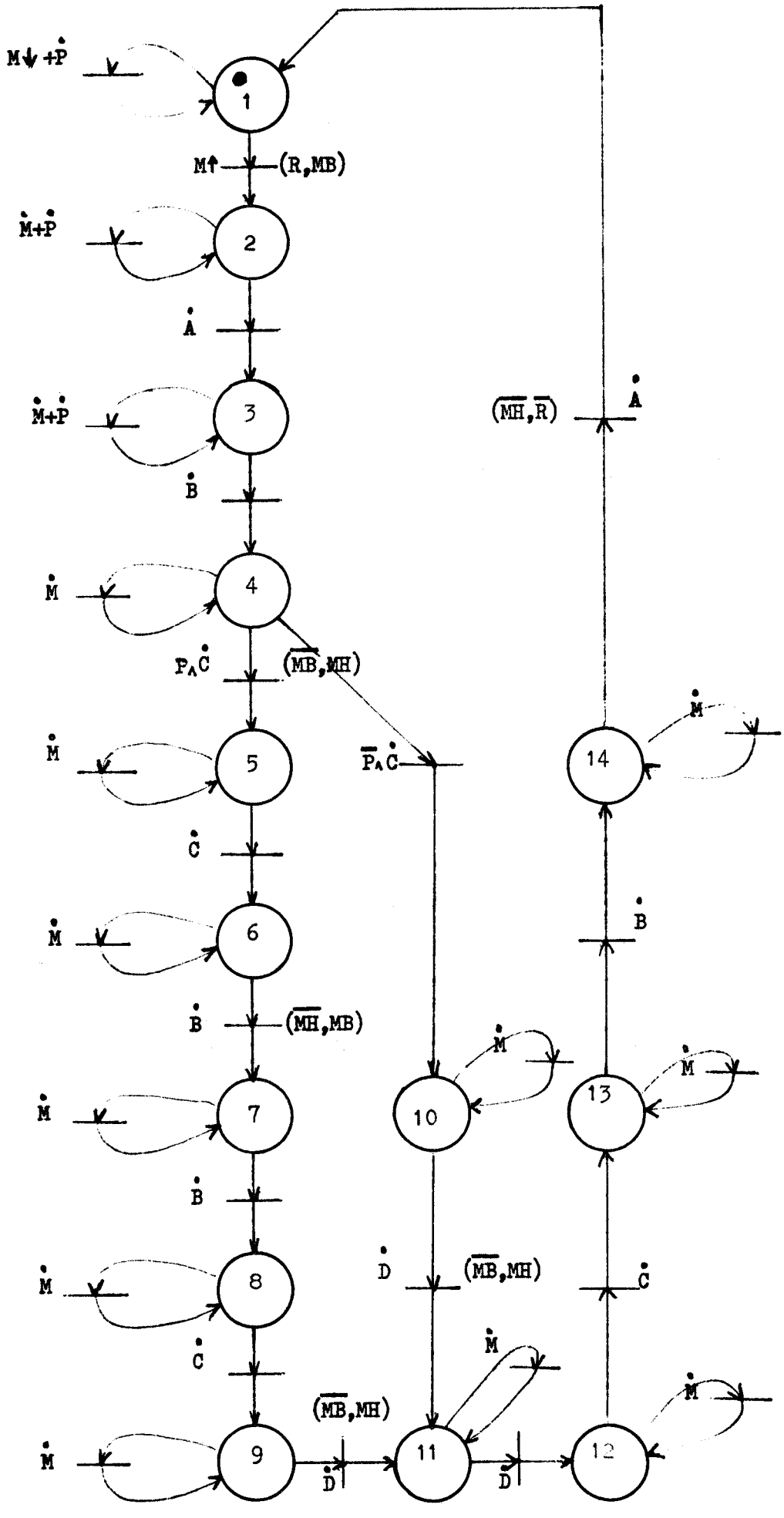


b) Cycle de perçage

Perçeuse avec ou sans débourrage

(Fig. IV.3)

L'action de M est telle qu'elle permet le départ du perçage sur un front montant de M (fonctionnement cycle par cycle). L'action sur P est interdite depuis le moment où le capteur B est atteint jusqu'à la fin du perçage. Le cycle de perçage avec débourrage est donné fig. IV .3b. Le réseau de Pétri correspondant est celui de la fig. IV.4.



-fig. IV.4- Réseau de Pétri de la perçeuse à surveillance accrue.

A partir du marquage de la place 4 jusqu'à la fin du cycle, la variation de P (choix du cycle) est impossible. Ceci illustre parfaitement le cahier des charges proposé par Naslin dont est extrait cet exemple. Le cahier des charges restant muet quant à la décision à prendre si la commande P est modifiée après avoir atteint le niveau de B, nous avons choisi d'arrêter alors le perçage.

Détecter un front sur une variable X, c'est constater une variation de niveau de l'entrée correspondante conduisant à une valeur finale 0 (front descendant) ou 1 (front montant). Un front montant sur la variable X peut alors s'exprimer par $\dot{X} \wedge X$, un front descendant par $\dot{X} \wedge \bar{X}$. Ceci peut être appliqué dans notre exemple à la variable M

($M\uparrow \equiv \dot{M} \wedge M$ et $M\downarrow \equiv \dot{M} \wedge \bar{M}$)

Nous remarquons alors que le contrôle des variations de M n'est pas utile car toujours admises.

4.3.4 - Classification des variables primaires

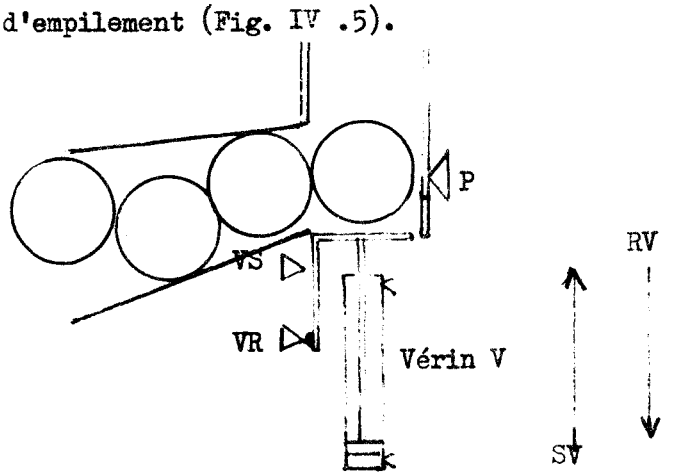
Parmi les variables primaires du système de commande nous rencontrons des grandeurs dont l'évolution est liée au marquage. Elles indiquent la fin d'une action en cours (type fin de course) en général. Le compte rendu issu de la partie opérative est obtenu à partir des valeurs de ces grandeurs d'entrée. A ce titre nous dirons que nous sommes en présence des variables primaires de contrôle. En général, ce sont les variations de niveau de ces variables qui interviennent dans les fonctions de réceptivité et de passivité.

Il existe également des variables primaires dont la valeur est fixée indépendamment des actions en cours (commandes manuelles, détection de surcharge, présence pièce lorsque l'approvisionnement n'est pas contrôlé par la machine elle-même...). Ces grandeurs ont pour rôle d'opérer à des sélections ou des départs de cycle. A ce titre ces grandeurs seront appelées variables primaires de commande. De ce fait il est généralement

leur niveau logique qui est pris en compte dans la description du système de commande. Par définition leurs modifications de niveau peuvent intervenir à tout instant donc pour tout marquage. Le système séquentiel est donc passif ou éventuellement réceptif (sélection de cycle par front) aux variables dérivées qui leur sont associées, quelque soit le marquage du réseau. La surveillance des variations ne permet pas alors de détecter de défaut sur les organes technologiques concernés. Ceci a été signalé dans les exemples du paragraphe § 3. Cette surveillance est alors inutile.
En définitive, seules les variables dérivées associées aux variables primaires de contrôle feront l'objet d'une surveillance permettant une détection de panne sur les dispositifs technologiques et leur liaison.

Dans l'exemple de la perçeuse (§ 3.3), les variables de contrôle sont: A-B-C-D-P; la variable de commande est M. La variable P est une grandeur de contrôle car son évolution est impossible lorsque la place 4 est marquée. Elle sert pourtant à une sélection de cycle. Ceci représente un cas limite de la classification. Une autre illustration de ca cas limite se retrouve dans l'exemple suivant:

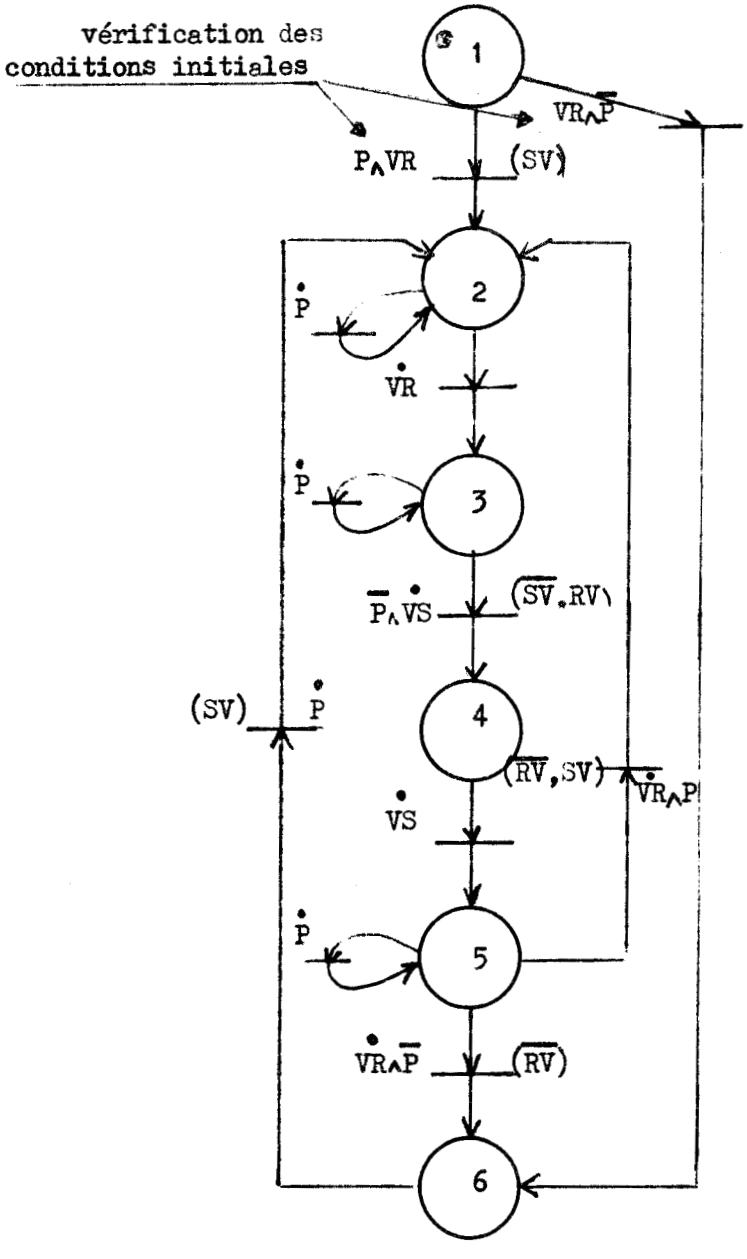
Un vérin déplace des objets arrivant par une goulotte d'alimentation. Un dispositif de détection de présence pièce "P" déclenche le cycle d'empilement (Fig. IV .5).



Systéme d'empilement d'objets.

- fig. IV.5 -

Les capteurs VR et VS détectent les positions extrêmes entrée ou sortie du vérin. Le réseau de Pétri peut être le suivant (Fig. IV.6).



Réseau de Pétri d'un systéme d'empilement d'objets.

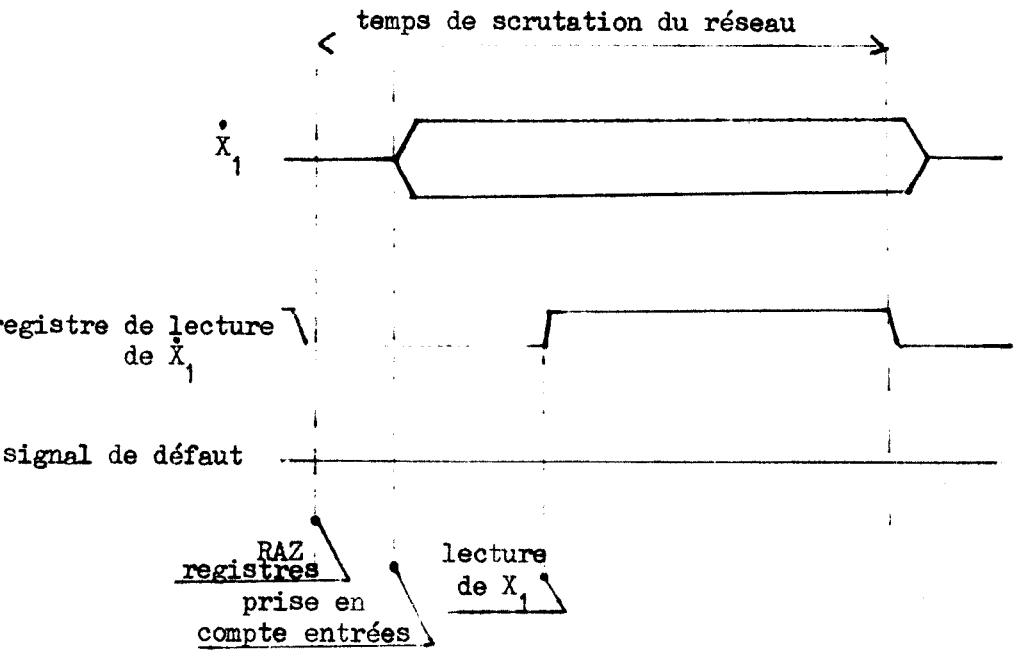
- fig. IV.6 -

La difficulté vient ici de ce que l'action "sortir le vérin (SV)" a un effet direct: faire tomber le fin de course "vérin rentré (VR)", et un effet secondaire: supprimer la présence pièce.

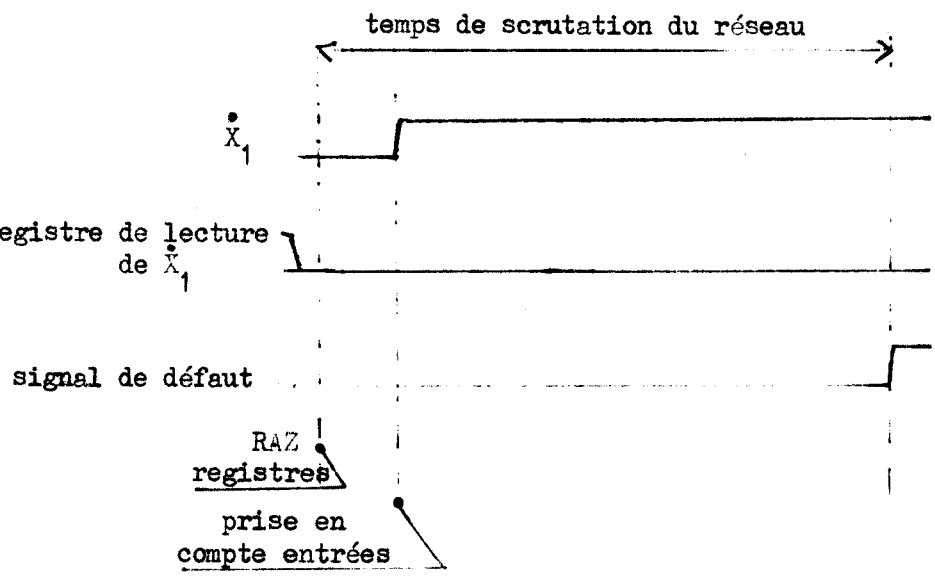
Une même action a donc deux effets qui peuvent être simultanés (avec un traitement synchrone) ou différés dans un ordre non connu. Dans ce cas l'effet principal participe à la réceptivité (passage des places 3 à 4 par $\dot{VR}$), les effets secondaires sont rejetés dans la fonction de passivité. Toutefois, à la fin du mouvement, nous considérons que les effets secondaires ont eu lieu en vérifiant les niveaux des entrées correspondantes. Ici cela se traduit par le produit logique $\bar{P}_{\wedge}\dot{VS}$ pris comme événement associé à la transition permettant le passage de P3 vers P4.

4.3.5 - Surveillance des variations des entrées: détection de pannes.

Chaque fois que la variation de niveau d'une entrée est normale, ou acceptable pour le marquage en cours, la variable dérivée associée est prise en compte dans les fonctions de réceptivité ou de passivité. A chaque scrutation du réseau, l'ensemble des variables dérivées, susceptibles de prendre la valeur "1" pour le marquage en cours, sont donc lues. Nous affectons pour chaque variable dérivée associée à une variable de contrôle un registre qui, mis à "0" au début de chaque scrutation du réseau, sera mis à "1" si la sortie correspondante du détecteur de variation de niveau des entrées a été lue. En fin de scrutation toute variable dérivée qui a pris la valeur "1" a du être lue, sinon il y a défaut. La détection de variations intempestives de niveau sur les entrées se fait en fin de traitement du réseau en comparant l'état des sorties du détecteur de variation et les sorties des registres de lecture. Un signal de défaut est élaboré. Ceci est illustré par les chronogrammes suivants (FIG. IV .7a et Fig. IV.7b).



a/ Chronogramme de surveillance de X_1 en fonctionnement normal



b/ Chronogramme de surveillance de X_1, détection d'un défaut.

- fig. IV.7 -

4.3.6 - Niveau de surveillance atteint

Nous passons en revue un certain nombre de cas de panne parmi les plus fréquents et nous les analysons. Nous savons déjà que seuls les défauts relatifs aux variables primaires de contrôle sont décelables. Le comportement du dispositif de commande vis à vis d'un défaut dépend du type d'incident et de l'instant où il se produit.

4.3.6.1 - Rupture ou mise à la masse d'une liaison avec un capteur entrainant une modification de niveau logique.

Le défaut est détecté s'il apparaît à un moment où le dispositif de commande n'est ni réceptif ni passif pour les variations de niveau de la grandeur concernée. Si à l'apparition du défaut le système est passif pour cette entrée, nous pouvons encore espérer un blocage de l'évolution du marquage. Il faut alors que le dispositif devienne réceptif, pour un marquage ultérieur, à la variation de cette variable (variation devenue impossible).

4.3.6.2 - Rupture ou mise à la masse d'une liaison avec un capteur n'entraînant pas de modification de niveau logique.

Ceci conduit à un blocage de l'évolution du marquage car la variation de niveau logique, sur cette variable, ne sera plus jamais obtenue.

4.3.6.3 - Blocage d'un capteur

Ce cas est très semblable aux précédents. Les défauts de ce type apparaissent souvent au moment de l'ouverture (ou de la fermeture) des contacts du capteur et conduisent alors à des défauts sans modification du niveau logique.

4.3.6.4 - Défaut sur un actionneur ou sa liaison au système de commande

Tout mouvement contrôlé nécessite une information de fin de tâche. Lorsque l'action est commandée, le capteur correspondant à la position de départ doit être relâché. Si l'actionneur n'exécute pas les ordres fournis par le système de commande, nous aboutirons à un blocage de l'évolution du marquage. Ceci est dû au fait que le système de commande a été rendu réceptif à la variable dérivée associée au capteur de position initiale.

Nous constatons que la majorité des défauts ont un effet, soit de mise en sauvegarde, soit de blocage du système séquentiel. Dans ce dernier cas, il faut prévoir de stopper également les actions correspondantes en provoquant une mise en sauvegarde. Ce résultat est obtenu en utilisant des réseaux de Pétri temporisés.

4.3.7 - Réseaux de Pétri temporisés

Chaque fois qu'une action est lancée, nous enclenchons une temporisation. Elle est réglée à une valeur légèrement supérieure au temps maximum nécessaire pour terminer, en fonctionnement normal, l'action correspondante. Lorsque l'action est arrêtée nous désactivons en même temps la temporisation. En fonctionnement normal, la sortie de la temporisation ne doit donc jamais évoluer. Tout blocage de l'évolution du marquage sur une branche du réseau sera donc détecté et entraînera la mise en sauvegarde de la machine séquentielle et le déclenchement d'une alarme.

4.4 - Conclusion

De nombreux défauts sur les capteurs ou les actionneurs sont détectés. Toute séquence aberrante est rendue pratiquement impossible

ce qui facilite le dépannage. La visualisation des défauts détectés est envisageable lors de la mise en sauvegarde de la machine. Toutefois pour des machines séquentielles dont l'arrêt est impossible (industrie nucléaire, aviation...), même en cas de défaut, il faut avoir recours à des dispositifs redondants de prise de l'information. L'information prise en compte est alors celle qui est majoritaire (en général, il est adopté la solution 2 parmi 3). [11]

La détection des variations est une solution interessante dans un système programmé car il est possible de prendre en compte des variations "durables". Dans les systèmes câblés, il faut éviter d'utiliser cette méthode en ayant recours à des circuits sensibles à des fronts, car la sensibilité aux parasites et aux rebondissements des contacts est trop importante.

C H A P I T R E V

ADAPTATION DU SYSTEME DE COMMANDE

Nous avons vu que la sûreté de fonctionnement peut être augmentée par une description plus détaillée de l'évolution de la partie opérative. Dans ce chapitre nous examinons des comportements possibles en cas de détection de fonctionnements défectueux. Nous adaptons à cette mission le matériel et le logiciel de notre dispositif de commande. Un exemple d'application à la commande d'une presse est enfin développé.

5.1 - Comportement en cas de défaut

Ayant détecté un fonctionnement défectueux, nous employons une procédure d'arrêt qui s'identifie, en général, à celle de l'arrêt d'urgence. Nous rencontrons:

- l'arrêt immédiat de l'installation;
- l'arrêt après déroulement d'un cycle spécial. Cette procédure d'arrêt peut être liée au marquage du réseau au moment de la détection du défaut.

De même les conditions de reprise sont à envisager. Nous avons:

- la reprise aux conditions initiales;
- la reprise du cycle normal (qui n'est envisageable qu'après un arrêt immédiat avec blocage des actionneurs);
- la reprise après un cycle spécifique.

Un programme, accessible uniquement après détection d'un défaut, doit alors modifier les sorties. En cas de reprise du cycle normal, l'état des sorties

et le marquage avant apparition du défaut sont mémorisés. Les valeurs des sorties sont alors transférées à des variables internes avant modification. Pour le marquage il suffit de conserver le contenu de la table des pas, la position des pointeurs et la valeur des variables internes représentant le marquage des places de synchronisme. Dans tous les autres cas le contenu de la table des pas doit être modifié pour permettre la réinitialisation du système ou la gestion d'un réseau spécial, distinct du précédent, décrivant la procédure d'arrêt ou de reprise.

Nous allons maintenant envisager les modifications à apporter au matériel et au logiciel pour adapter notre automate à cette mission.

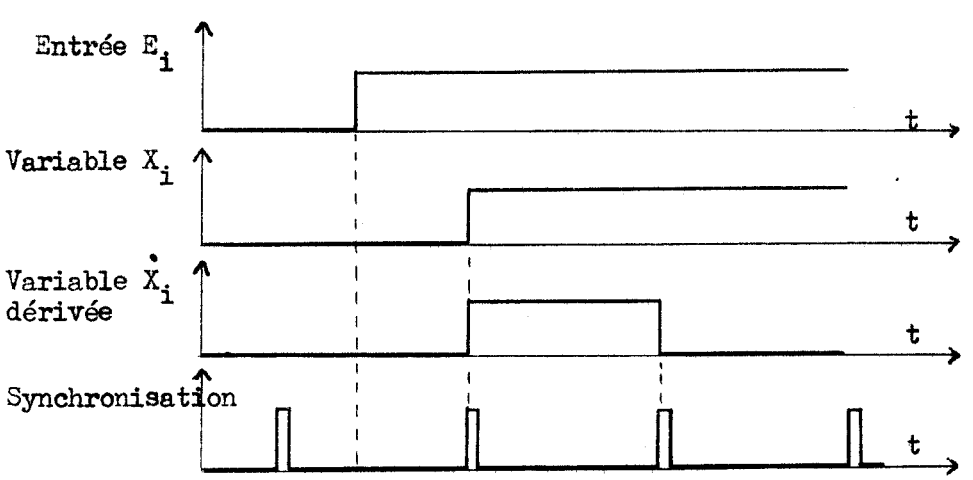
5.2 - Adaptation du matériel

5.2.1 - Automate d'entrée: détecteur de variations

Ce dispositif remplit deux fonctions:

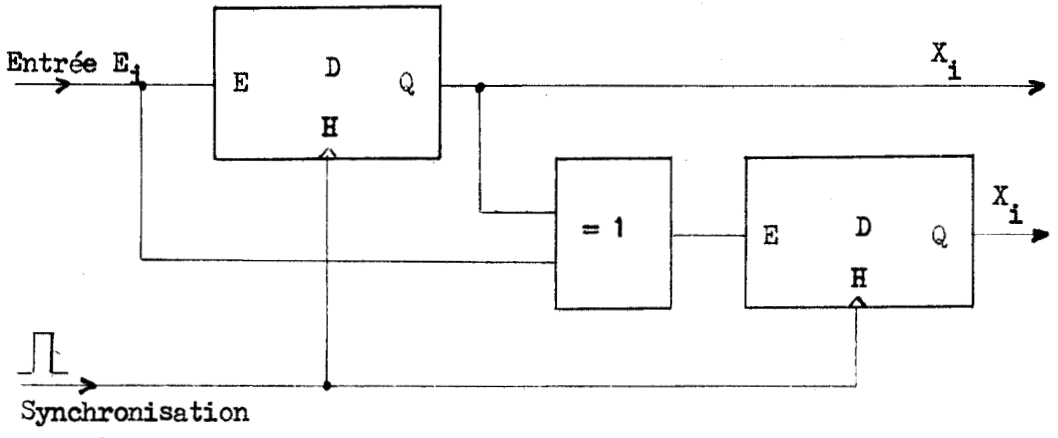
- Il permet le gel des grandeurs d'entrée pendant le temps de scrutation du réseau.
- Il élabore la valeur des variables dérivées qui sont également maintenues constantes pendant le traitement.

Sur le chronogramme de la figure V.1 nous indiquons l'évolution des grandeurs X_i et $\dot{X}_i$ lorsque l'entrée X_i correspondante est modifiée.



- Fig. V.1 -

Ce résultat est obtenu en comparant à la fin du traitement la valeur de chaque entrée à celle qu'elle avait lors de la scrutation précédente. Le schéma pour une entrée est donné figure V.2. Il met en oeuvre des bascules type D flip-flop.



- Fig. V.2 -

Nous remarquons sur le chronogramme de la figure V.1 qu'un front montant sur une entrée E_i se détecte en faisant un ET entre la grandeur X_i et sa grandeur dérivée $\dot{X}_i$. Pour un front descendant, il suffit de prendre la grandeur X_i sous sa forme niée.

$$E_i \uparrow \equiv X_i \wedge \dot{X}_i$$
$$E_i \downarrow \equiv \bar{X}_i \wedge \dot{X}_i$$

Le signal de synchronisation est élaboré lors du traitement du programme SYNC comme pour le dispositif d'entrée vu en première partie. La sélection entre une variable d'entrée et sa grandeur dérivée est faite par un bit du champ de données. Nous avons choisi dans notre réalisation le bit de poids quatre.

5.2.2 - Détection de défaut par modification non attendue d'une entrée

Nous avons vu au chapitre précédent que toute variation d'une entrée, qui n'est pas prise en compte dans les fonctions de réceptivité,

de sensibilité ou de passivité, est le signe d'un fonctionnement défectueux. Au cours du traitement, le dispositif de commande vient lire la valeur de toutes les variables dérivées dont la variation est attendue ou acceptable. En fin de traitement, si nous trouvons une variable dérivée qui a la valeur 1 et qui n'a pas été lue, nous sommes en présence d'un défaut.

A chaque variable dérivée $\dot{X}_i$ nous associons un registre de lecture. Sa sortie (notée L_i) est mise à 1 lorsque la valeur de $\dot{X}_i$ est lue par l'automate logique. Tous ces registres sont remis périodiquement à 0 lors du transfert des grandeurs d'entrée après avoir tester les expressions de la forme $\dot{X}_i \wedge \overline{L}_i$. Si l'un quelconque de ces produits a la valeur 1, il y a un défaut. Ce dernier porte sur l'entrée E_i ou sur l'un des actionneurs dont le capteur élaborant E_i surveille le travail. Une visualisation de l'entrée sur laquelle le défaut a été détecté est possible pour apporter une aide à la maintenance. Il est très facile d'élaborer et de tester les expressions $\dot{X}_i \wedge \overline{L}_i$ avec l'automate logique. Toutefois, lorsque le nombre d'entrées est important, le temps nécessaire peut devenir prohibitif. Ici encore nous avons préféré une solution câblée.

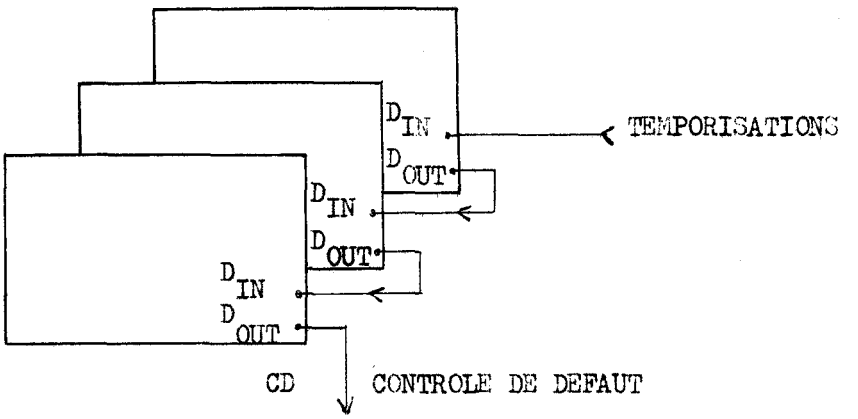
Sur chaque carte d'entrée est élaboré un signal de défaut obtenu en faisant $D_{OUT} = \sum_{i=0}^{7} \dot{X}_i \wedge \overline{L}_i + D_{IN}$

L'entrée D_{IN} permet d'associer plusieurs cartes d'entrée (fig. V.3). La sortie D_{OUT} de la dernière carte d'entrée est lue par le microprocesseur monobit à la fin de chaque scrutation. En fonctionnement normal cette sortie doit être à 0 lorsqu'elle est lue.

Remarque:

Le défaut dû à une variation intempestive d'une entrée est lié au positionnement de la variable dérivée correspondante. La sortie "Défaut" des cartes d'entrée doit donc être lue avant l'envoi du signal de synchronisation. De ce fait, cette sortie est raccordée directement à l'une des voies d'un multiplexeur d'entrée (entrée non gelée).

Le schéma d'une carte d'entrée est donné en annexe I.



La sortie contrôle défaut (CD) est lue par le microprocesseur à la fin de chaque scrutation.

- Fig. V.3 -

5.2.3 - Sélections des variables de commande et de contrôle

Au paragraphe IV.3.4 nous avons classé les grandeurs d'entrée en variables primaires de contrôle ou de commande selon leur rôle. Seules les variables de contrôle, dont l'évolution est liée à l'état de la partie opérative, sont l'objet d'une détection de défaut par modification non attendue. Pour éviter l'arrêt du système lors d'une variation sur une grandeur de commande, nous pouvons:

- Lire, au cours du traitement, l'ensemble des grandeurs de commande, ce qui positionne le registre de lecture correspondant.
- Raccorder ces entrées sur des cartes spécialisées dépourvues de la partie détection de défaut.
- Utiliser pour chaque entrée une information d'inhibition qui mette à 1 le registre lecture correspondant. Un mot d'état placé dans un registre permet alors la distinction entre les grandeurs de contrôle et de commande.

La dernière solution, bien que séduisante, n'a pas été exploitée. Nous avons retenu la première méthode car dans notre exemple le nombre de variables de commande est limité.

5.2.4 - Réseaux temporisés

Nous avons vu au chapitre IV que certains défauts provoquent un blocage de l'évolution du marquage. Nous avons donc choisi d'arrêter toutes les actions contrôlées au bout d'un temps légèrement supérieur à celui qui est nécessaire pour la réaliser en fonctionnement normal. Ce résultat est obtenu en armant une temporisation lorsqu'une action est lancée. Cette temporisation est arrêtée en même temps que l'action à laquelle elle est associée. Elle est réalisée par l'automate numérique au même titre que les temporisations introduites par le cahier des charges. Il n'est toutefois pas indispensable de disposer de l'ensemble des fins de temporisation qui ne doivent pas évoluer en fonctionnement normal. Un seul bit, disponible sur un port de sortie, sera alors positionné à 1 si une de ces temporisations vient à échéance. Cette ligne de sortie est raccordée à l'entrée D_{IN} de la première carte d'entrée (fig. V.3).

Dans la pratique, il arrive très souvent que la durée d'une action soit constante. Il est alors possible de confondre le registre de sortie de l'automate logique avec le registre de demande de tâche affecté aux temporisations.

5.3 - Adaptation du logiciel

La quantité d'informations à traiter est notablement augmentée par rapport à la description minimale habituelle. Toutefois, le principe de choix des places clefs et de représentation en mémoire du réseau reste valable en tout point. Le moniteur subit toutefois quelques modifications.

5.3.1 - Description de la procédure d'arrêt

Une zone mémoire, commençant à l'adresse B_0 est réservée à la description de cette procédure d'arrêt. Selon que le marquage et les sorties sont sauvegardées ou non, nous distinguons deux modes opératoires.

Marquage sauvegardé:

A la détection d'un défaut nous effectuons un branchement à l'adresse B_0. Le programme qui y est implanté permet l'affectation des sorties après sauvegarde des valeurs de ces sorties et des variables associées aux places de synchronisme avant défaut, dans des variables internes. A la reprise, un retour au moniteur à l'adresse FRAN après restitution des sorties permet un redémarrage du cycle normal.

Marquage non sauvegardé:

La table des pas est maintenant réutilisée, ce qui entraîne le positionnement en début de table des pointeurs, puis une initialisation du contenu de celle-ci grâce à un programme débutant à l'adresse B_0. Le réseau décrivant les procédures d'arrêt et de reprise est ensuite implanté normalement. Si la gestion est classique, nous éliminons toutefois la détection de défaut durant le cycle d'arrêt. Ceci est obtenu par modification du moniteur.

5.3.2 - Modifications du moniteur

Initialisation:

Le positionnement des pointeurs d'écriture et de lecture de la table des pas se fait maintenant par programme par utilisation de l'instruction RET. La lecture d'une entrée ayant le niveau 1 à la mise en marche fait apparaître un niveau 1 sur l'entrée dérivée correspondante après le premier transfert dans les registres d'entrée. Ceci risque d'être pris en compte comme un défaut. Deux transferts successifs sont alors nécessaires.

Synchronisation:

Comme dans la première réalisation, le rôle de ce sous-programme est de

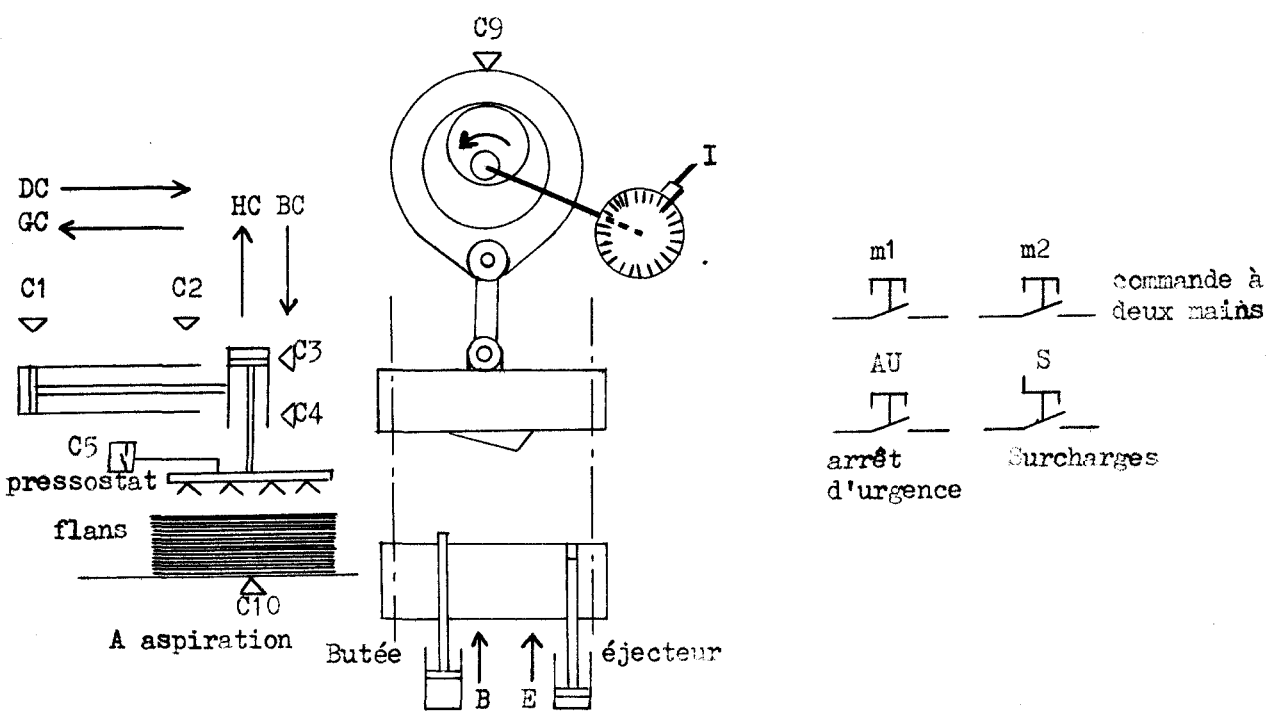
réinscrire l'indicateur de fin de table (JMP SYNC) et de générer le signal de synchronisation. Toutefois, avant de commander le transfert dans les registres d'entrée et de sortie, nous devons tester la ligne de contrôle défaut (CD). Lorsqu'un défaut est détecté, nous mémorisons cette information par une variable interne notée ZD. Un branchement à l'adresse B_0 permettant d'initialiser la procédure d'arrêt est faite si et seulement si nous avons CD ZD = 1. Le masquage de CD permet l'utilisation du programme de synchronisation devant la procédure d'arrêt, sans prendre en compte les nouveaux défauts qui peuvent se produire du fait que la demande d'arrêt est faite ici de façon aléatoire.

Le programme du moniteur modifié est également donné en annexe.

A la reprise il est nécessaire de prévoir la remise à 0 de ZD pour rendre au système sa capacité de détection de défaut.

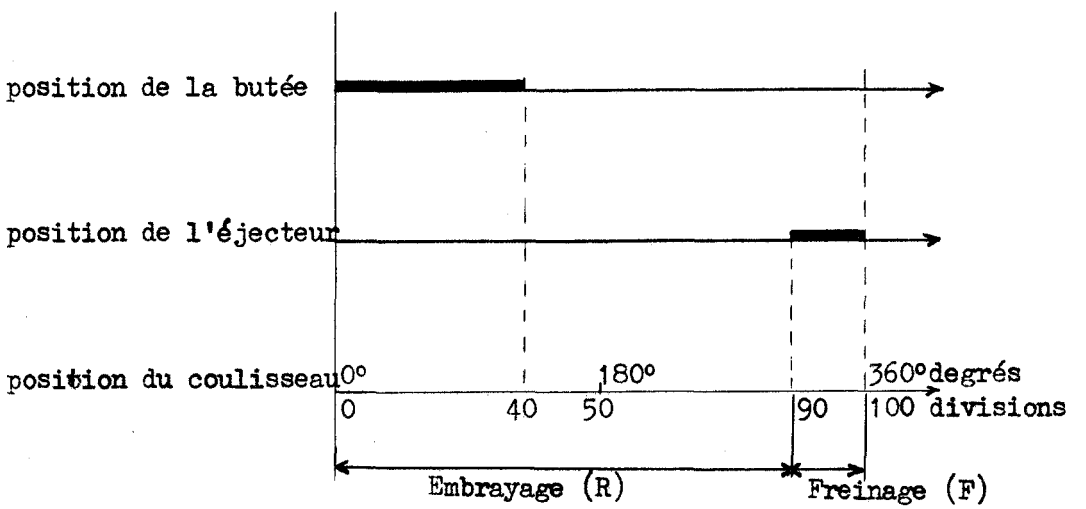
5.4 - Commande d'une presse

Nous envisageons la commande d'une presse et du dispositif de chargement (fig. V.4).



- Fig. V.4 -

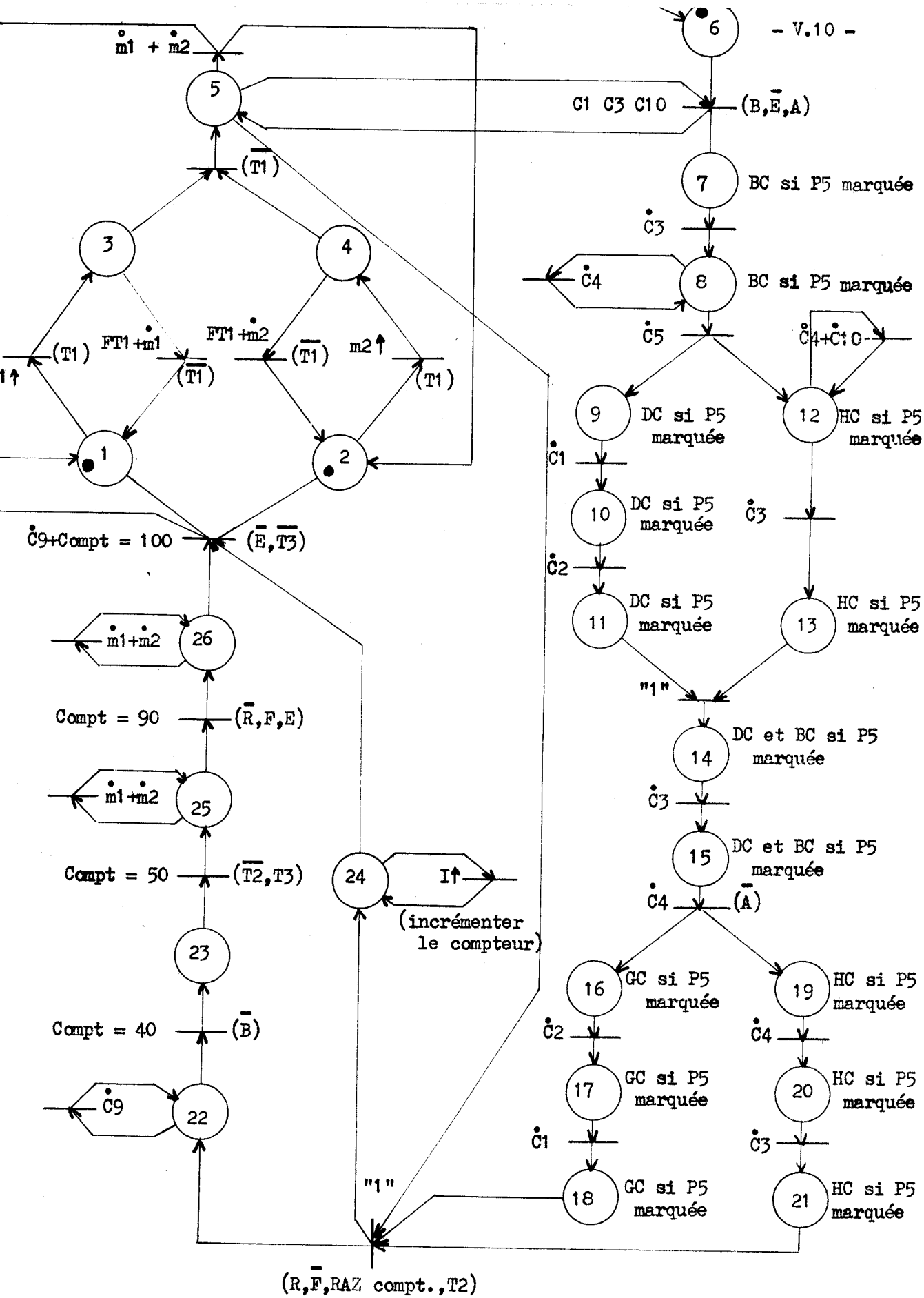
La position du coulisseau est repérée par codeur (cent points par cycle) de façon à faciliter l'adaptation d'outils divers. Le pupitre de commande comprend deux boutons poussoirs de mise en marche qui doivent être actionnés dans un intervalle de temps limité et maintenus appuyés jusqu'au passage par le point mort bas du coulisseau. Toutefois, tant que la presse n'est pas démarrée, un relachement de l'un au moins de ces boutons provoque le blocage du dispositif chargeur, mais la remise en route reste possible par M_1 et M_2. Les distributeurs qui commandent les déplacements du chargeur sont à trois positions et double pilotage, ceux qui alimentent les vérins de la butée (B) et de l'éjecteur (E) à simple pilotage.Le cycle de la presse en fonction de la position du coulisseau est donné figure V.5.



Cycle de la presse

- FIG. V.5 -

En cas d'arrêt d'urgence, le chargeur est arrêté, la butée et l'éjecteur sont escamotés, l'aspiration reste en état. Si le coulisseau est en mouvement, il est freiné s'il n'a pas passé le point mort bas, sinon il reste entraîné jusqu'au passage dans la zone de freinage ou au plus tard en arrivant sur le capteur C_6 (point mort haut).


m1 + m2
5
(T1)
3
4
(T1)
FT1+m1
FT1+m2
(T1)
m2↑
(T1)
1
2
Ċ9+Compt = 100
(E, T3)
m1+m2
26
Compt = 90
(R, F, E)
m1+m2
25
Compt = 50
(T2, T3)
24
I↑
(incrémenter le compteur)
23
Compt = 40
(B)
Ċ9
22
"1"
(R, F, RAZ compt., T2)
6
C1 C3 C10
(B, E, A)
7
BC si P5 marquée
Ċ3
Ċ4
8
BC si P5 marquée
Ċ5
Ċ4+Ċ10
9
DC si P5 marquée
12
HC si P5 marquée
Ċ1
10
DC si P5 marquée
Ċ3
Ċ2
11
DC si P5 marquée
13
HC si P5 marquée
"1"
14
DC et BC si P5 marquée
Ċ3
15
DC et BC si P5 marquée
Ċ4
(A)
16
GC si P5 marquée
19
HC si P5 marquée
Ċ2
Ċ4
17
GC si P5 marquée
20
HC si P5 marquée
Ċ1
Ċ3
18
GC si P5 marquée
21
HC si P5 marquée

- Fig. V.6 -

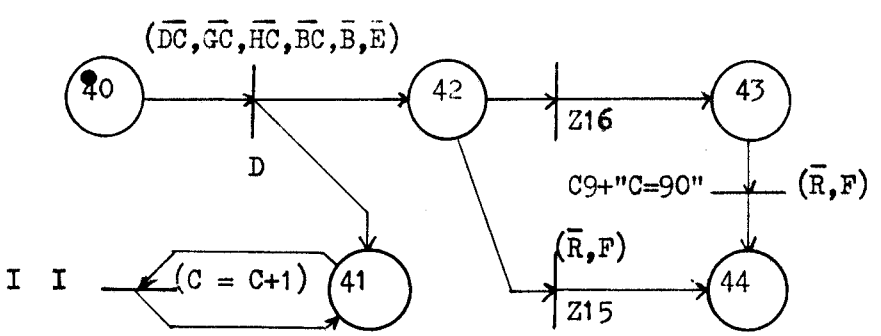
Le mouvement du coulisseau est obtenu par la commande d'un embrayage (E) qui solidarise la manivelle à l'arbre moteur toujours en rotation. Un frein (F) permet le blocage du coulisseau. Les flans sont aspirés par le chargeur par un dispositif pneumatique mis en pression par la commande (A).

Un réseau décrivant le fonctionnement normal de la presse est donné figure V.6.

En observant le réseau proposé, nous remarquons que l'utilisation de sorties type S a permis une simplification de la représentation par l'introduction du marquage de la place M_5 dans la détermination des commandes du chargeur.

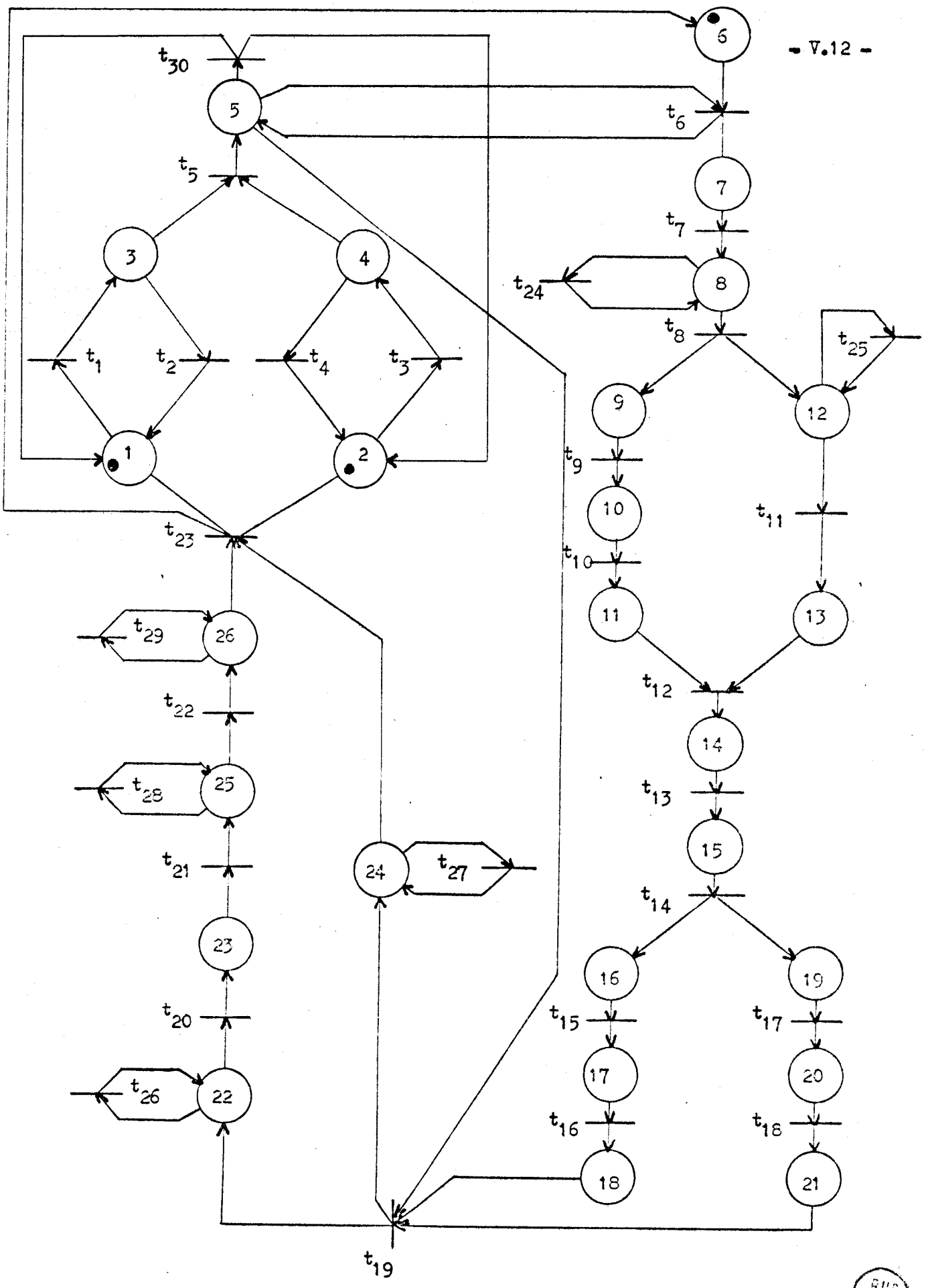
La notion de passivité est illustrée par l'introduction des transitions sur les places 8, 12, 25 et 26 associées aux événements C_4; C_4+C_{10}; M_1+M_2. La variation de C_4 est liée à la position du sommet de la pile d'approvisionnement par rapport à la hauteur de dépose des flans sur l'outil. Celle de C_{10} est obtenue lorsque le poste de chargement est vide. Les temporisations T_2 et T_3 sont ajoutées pour contrôler les temps limites du déplacement du coulisseau qui ne doivent jamais être atteints (réseau temporisé). Par contre, dans la première partie du cycle (chargement), il n'est pas possible de temporiser le réseau car l'évolution peut être bloquée volontairement par le conducteur de la presse.

La procédure d'arrêt est décrite par un réseau spécial (fig. V.7).



Procédure d'arrêt de la presse

- Fig. V.7 -



- Fig. V.8 -

La reprise se fait par une remise aux conditions initiales de l'unité centrale après éventuellement une marche manuelle (non représentée ici).

Recherche des places clefs.

La numérotation des places et des temporisations est donnée par le réseau de la figure V.8.

Compte tenu des sorties S associées aux places 7 à 21, seules les places 5 et 6 ne sont pas places clefs essentielles.

Nous choisissons arbitrairement

- la place 3 comme place clef pour t_5
- la place 11 comme place clef pour t_{12}
- la place 18 comme place clef pour t_{19}
- la place 26 comme place clef pour t_{23}

La matrice réduite ainsi obtenue après cette première étape est:

P_5	P_6	
1	1	t_6
		place clef choisie
⊠		place de synchronisme

La place P_5 est déjà place de synchronisme pour la transition t_6, donc P_6 est place clef pour t_6 par application de la deuxième étape.

Les places 4,5,13,21 et 24 ont une double représentation. Nous pouvons écrire maintenant le programme relatif à cet exemple, il est donné en annexe.

5.5 - Conclusion

L'adaptation du dispositif de commande porte essentiellement sur le matériel. Cette option a été prise afin de respecter les contraintes de temps qui nous sont imposées. Aucun élément nouveau n'est apporté au niveau de l'implantation en mémoire du réseau. Seule la pro-

cédure d'arrêt, qui doit être décrite, peut s'avérer plus délicate. Le moniteur n'est pratiquement pas modifié, ceci permet un temps de scrutation comparable à celui obtenu pour une commande sans surveillance accrue.

L'exemple proposé met en oeuvre l'ensemble des propriétés qui ont été mises en évidence dans les chapitres précédents tant au niveau de la description que de l'implantation. En se reportant au programme présenté en annexe nous trouvons notamment les procédures d'affectation des sorties S et T, de détection des places clefs démarquées dans la table des pas en cas de double représentation, de demande de tâches numériques et de temporisations.

C O N C L U S I O N

La méthode d'implantation développée dans la première partie permet de réduire le temps de traitement grâce à une limitation du nombre d'instructions traitées à chaque scrutation du réseau. La mise en oeuvre d'un automate spécialisé confiant la gestion du marquage à un microprocesseur monobit permet un temps de cycle de l'ordre de 200us pour un réseau de 100 places clefs dont 10 sont supposées marquées simultanément. Ce temps est pratiquement réduit dans un rapport 10 en comparaison d'une implantation similaire sur un système non spécialisé.

La structure biprocesseur est telle que toutes les tâches numériques (temporisations, comptages, comparaisons) sont confiées à un microprocesseur multibit mieux adapté. Lorsque ces traitements sont conséquents, il est possible d'augmenter le nombre de ces processeurs tout en conservant le même principe de dialogue avec l'automate logique. Pour rendre ce dispositif de commande opérationnel sur le plan industriel, il reste à développer une console de programmation assurant l'aide au développement.

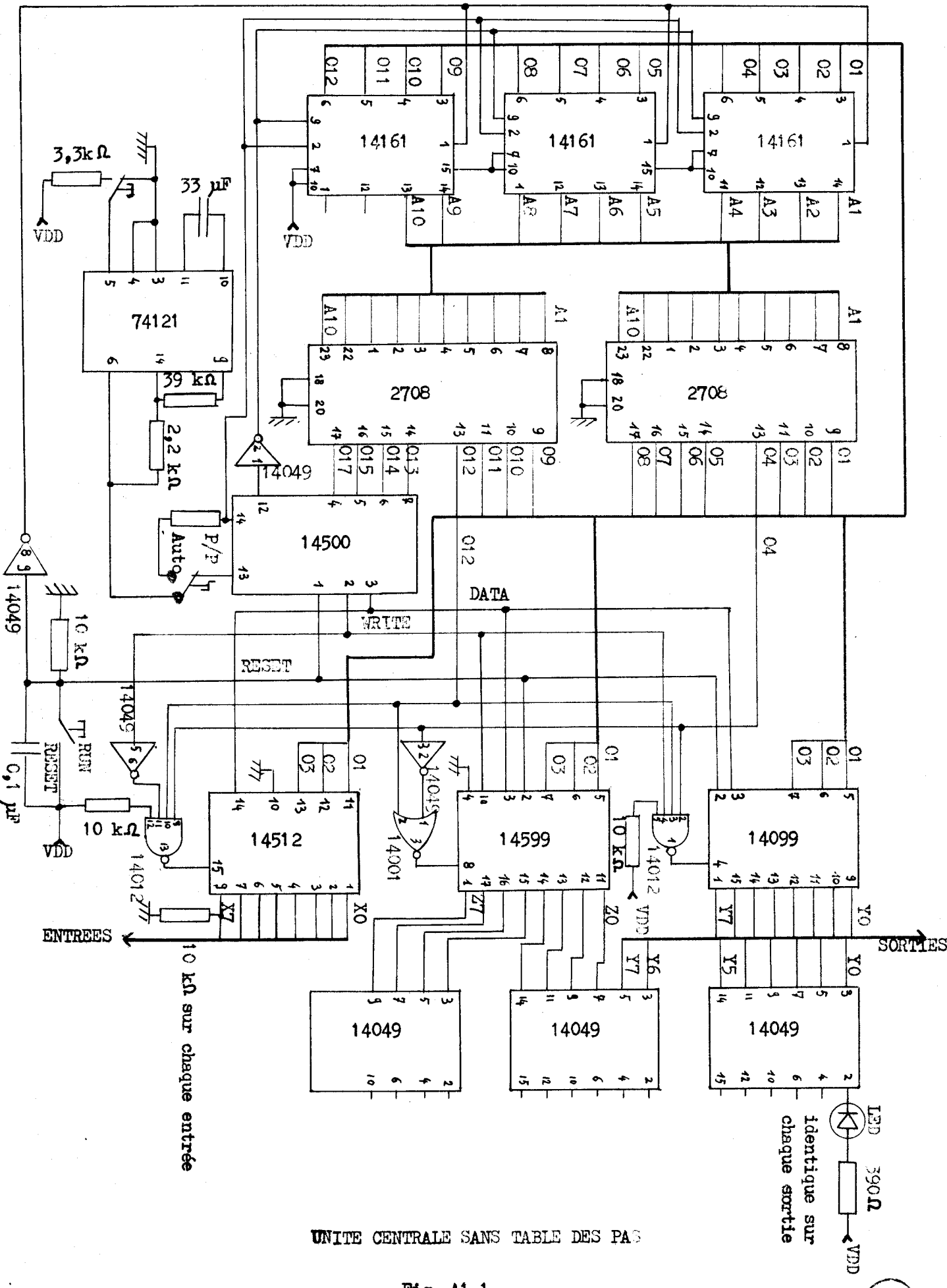
Dans la deuxième partie, nous avons mis l'accent sur la sûreté de fonctionnement. La méthode développée conduit à la détection des défauts sur les capteurs qui contrôlent une action de durée limitée, les actionneurs correspondants et leurs liaisons. Cette surveillance accrue permet d'éviter les évolutions aberrantes liées aux pannes les plus fréquentes en milieu industriel. La technique proposée peut être mise en oeuvre conjointement avec d'autres méthodes et en particulier celles des réseaux pondérés ou des systèmes de commande simplement redondants ou à vote majoritaire. Des études dans cette voie restent à développer.

A N N E X E I

MATERIEL MIS EN OEUVRE

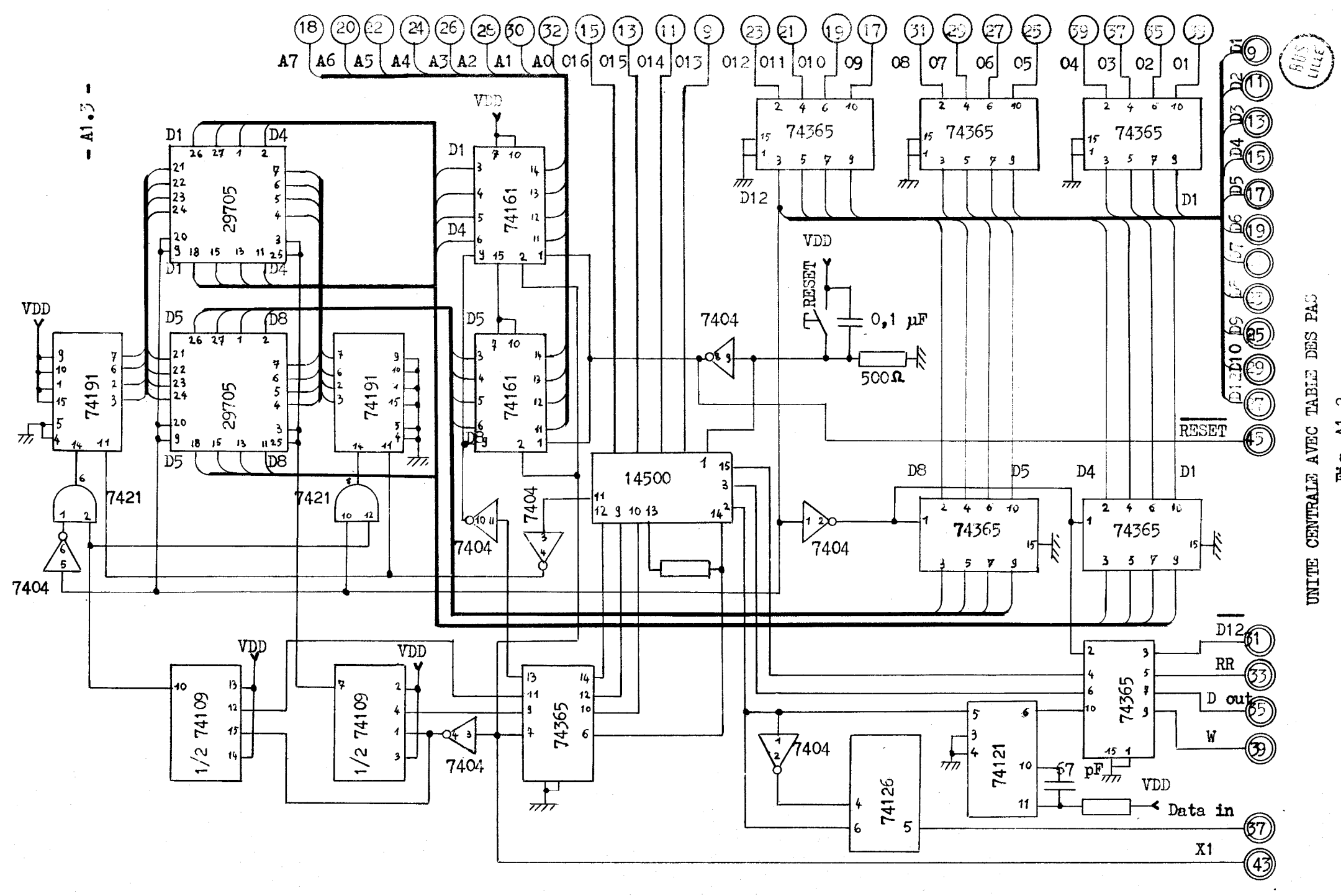
Dans cette annexe nous présentons les schémas des différentes cartes réalisées. La première planche (Fig. A1.1) correspond à un automate sans table des pas. Le test de toutes les places clefs est donc indispensable. Bien que représenté avec ses dispositifs d'entrées/sorties, il admet éventuellement les périphériques de la version plus élaborée. Il permet donc la détection de défauts. Son utilisation est réservée à des applications où la vitesse de traitement n'est pas critique.

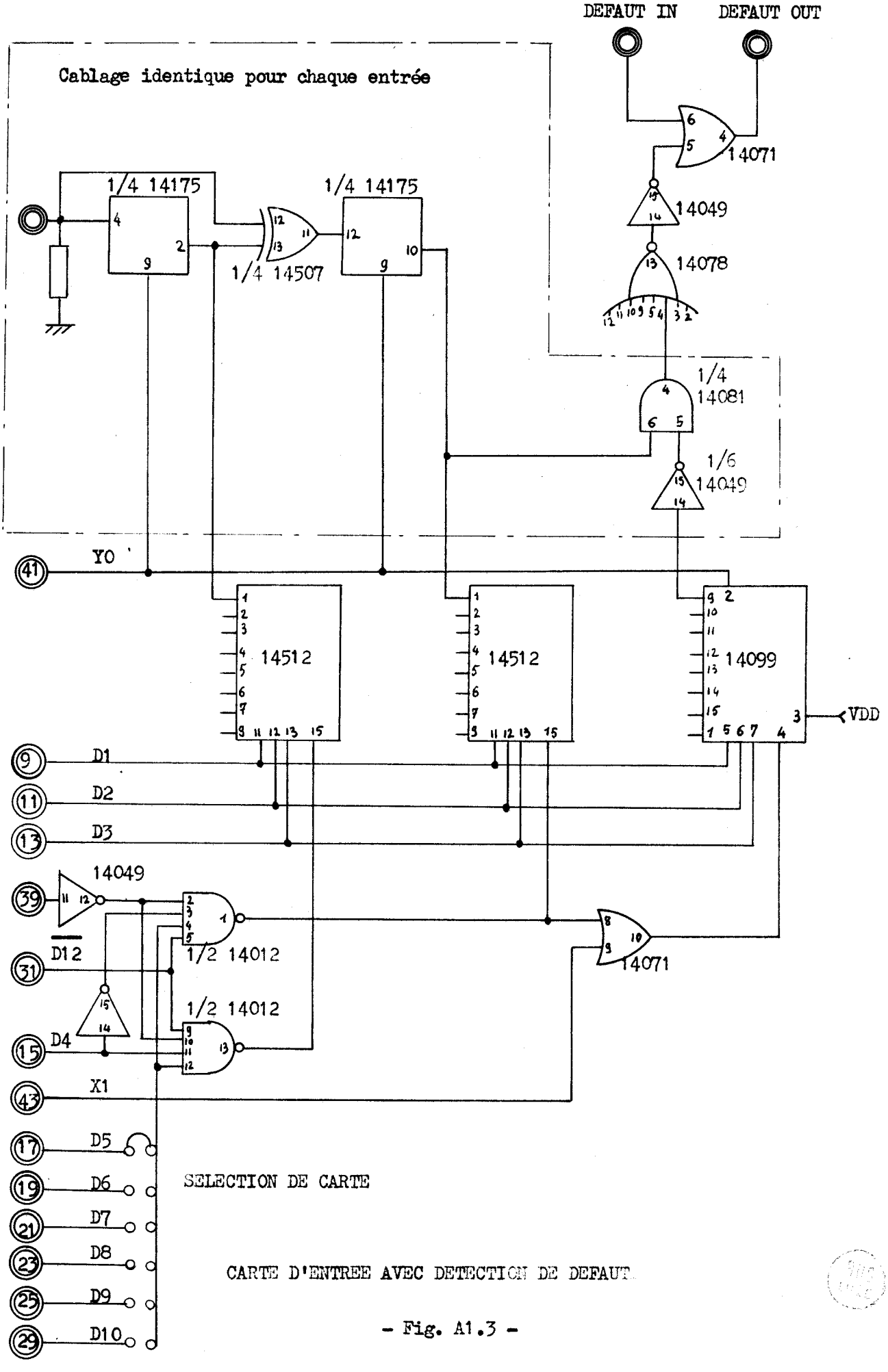
La planche suivante (Fig. A1.2) correspond à la version élaborée de l'unité centrale permettant la gestion des réseaux avec utilisation d'une table des pas.Les figures A1.3 et A1.4 représentent les schémas des cartes d'entrées et de sorties. La carte de sortie est utilisée pour les variables internes en ajoutant le multiplexeur 14512 qui y est dessiné, la sélection de boitier se fait alors par D12 au lieu de $\overline{D12}$. La sélection de carte se fait dans tous les cas par les bits D5 à D10.



UNITE CENTRALE SANS TABLE DES PA

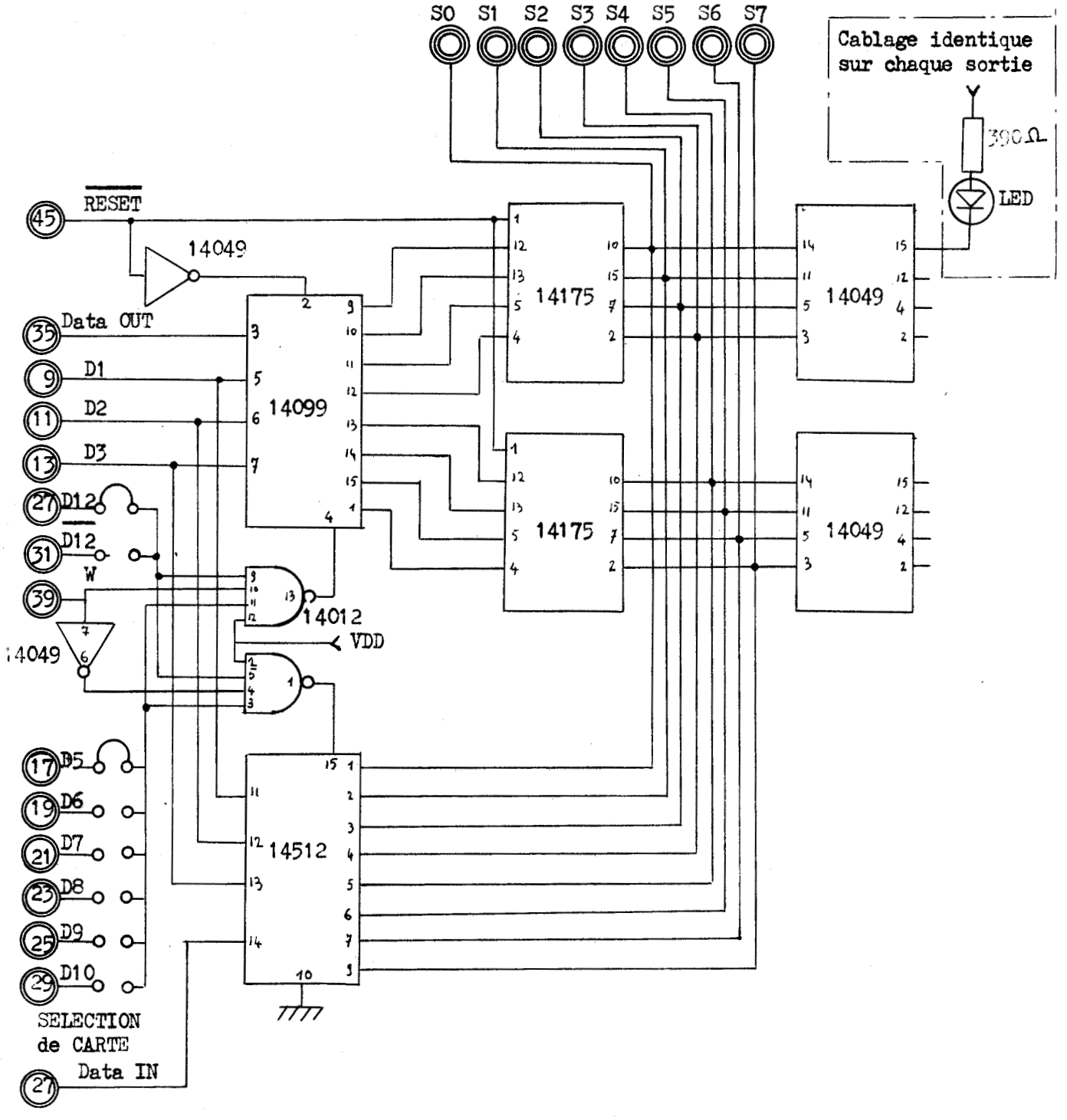
- Fig. A1.1 -


A7 A6 A5 A4 A3 A2 A1 A0 016 015 014 013 012 011 010 09 08 07 06 05 04 03 02 01
74365
74365
74365
D12
D1
VDD
RESET
0,1 µF
500Ω
7404
29705
74161
74191
14500
7421
D1 D4
D5 D8
74365
74365
74365
RESET
D12
RR
D out
W
Data in
X1
1/2 74109
1/2 74109
74365
74126
74121
67 pF
UNITE CENTRALE AVEC TABLE DES PAS



CARTE D'ENTREE AVEC DETECTION DE DEFAUT

- Fig. A1.3 -



Nota: pour les cartes de sortie ne pas cabler le multiplexeur 14512 et adresser la carte par $\overline{D12}$ au lieu de D12.

CARTE DE SORTIE OU DE VARIABLE INTERNE

A N N E X E II

LOGICIEL DU DISPOSITIF DE COMMANDE

Dans cette annexe nous présentons l'ensemble des programmes mis en oeuvre dans le dispositif de commande proposé. Nous y trouvons trois parties, la première correspond au séquenceur logique sans surveillance accrue, la deuxième à l'automate numérique, la derniére au processeur logique avec détection de défauts.

A2.1 - Logiciel de l'automate logique sans détection de défauts

A2.1.1 - Moniteur

Dans cette réalisation le registre résultat du microprocesseur (RR) est connecté à l'entrée d'adresse 820, le signal de synchronisation est fourni par le FLAG RTN.

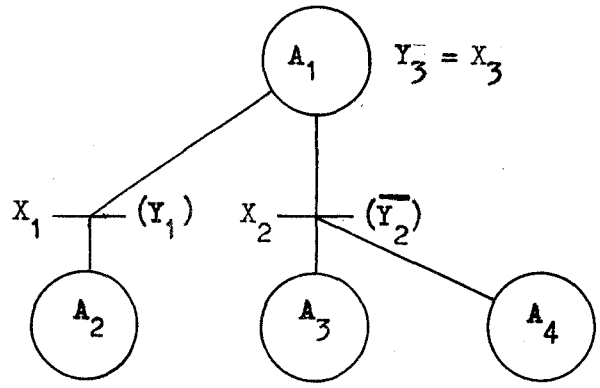
```
000  68 20   INIT: ORC  RR     ;mise à 1 de RR
001  A8 20         IEN  RR     ;mise à 1 du registre IEN interne au uP
002  B8 20         OEN  RR     ;mise à 1 du registre OEN interne au uP
003  08 10         ETA  A1     ;écrire adresse 1er pas dans table
004  F8 00         INE         ;incrémenter pointeur d'écriture
005  08 05   SYNC: ETA  SYNC   ;écrire indicateur fin de table
006  D0 00         RTN         ;signal de synchronisation
007  18 20         LD   RR     ;instruction non traitée aprés RTN
008  F8 00   NFRA: INE
009  18 20         LD   RR     ;faire tomber flag F
00A  F0 00   FRAN: INL         ;incrémenter pointeur de lecture
00B  C0 00         JTA         ;branchement au pas lu dans la table
```

A2.1.2 - Exemple de sous programme relatif à un pas

Le sous programme relatif à un pas regr[illegible]es instructions qui permettent l'élaboration des sorties S et les test[illegible]ur les conditions

de franchissement d'une part, celles qui fixent les valeurs des sorties de type T et effectuent la mise à jour du marquage d'autre part.

Le sous programme présenté correspond au pas défini par la figure A2.1 ci-dessous.



Exemple de pas

- fig. A2.1 -

Le pas A2 commence à l'adresse 020

Le pas A3 commence à l'adresse 030

Le pas A4 commence à l'adresse 040

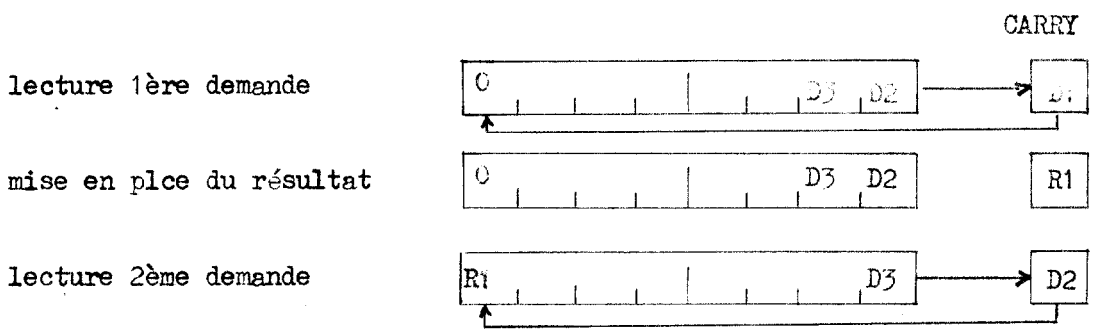
```
010  08 10   A1:   ETA  A1     ;écrire A1 dans la table
011  10 13         LD   X3     ;élaboration d'une sortie S
012  80 13         STO  Y3
013  10 10         LD   X1     ;événement associé à 1ère transition
014  E0 00         SKZ       ;si cet événement est vrai  alors
015  C8 50         JMP  B1     ;traiter les modifications des sorties
                               ;et des marquages localisées en B1
016  10 12         LD   X2     ;événement associé à 2ème transition
017  E0 00         SKZ       ;si cet événement est vrai alors
018  C8 60         JMP  B2     ;sorties et marquages en B2
019  C8 08         JMP  NFRA   ;indicateur fin de pas, provoque le
                               ;retour au moniteur à l'adresse NFRA
                               ;ce qui laisse A1 dans la table


                               ;élaboration des sorties et des modifi-
                               ;cations de marquage. A l'entrée dans ces
                               ;sous programme nous avons (RR) = 1

050  80 11   B1:   STO  Y1     ;mise à 1 de Y1
051  90 13         STOC Y3     ;mise à 0 de la sortie de type S notée Y3
052  08 20         ETA  A2     ;écrire l'adresse de la clef A2
```

```
053  F8 00          INE
054  C8 0A          JMP  FRAN  ;retour moniteur en FRAN ce qui élimine
                               ;A1 de la table (indicateur fin de tran-
                               ;sition)

060  90 12  B2:     STOC Y2    ;mise à 0 de Y2
061  90 13          STOC Y3    ;mise à 0 de la sortie S notée Y3
062  08 30          ETA  A3    ;écrire place clef A3
063  F8 00          INE
064  08 40          ETA  A4    ;écrire place clef A4
065  F8 00          INE
066  C8 09          JMP  FRAN  ;indicateur fin de transition: retour
                               ;moniteur
```

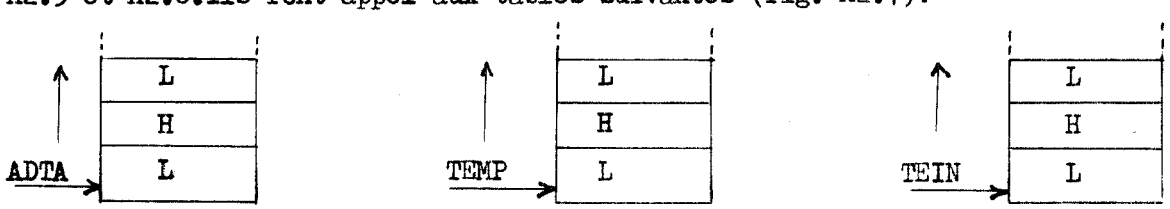
A2.2 - Logiciel de l'automate numérique

Les programmes présentés ici permettent la recherche des tâches demandées et le branchement aux sous programmes correspondants, ils gérent également les temporisations.

La lecture des demandes et l'élaboration des résultats se fait par rotation à travers carry d'un même mot de la façon suivante.

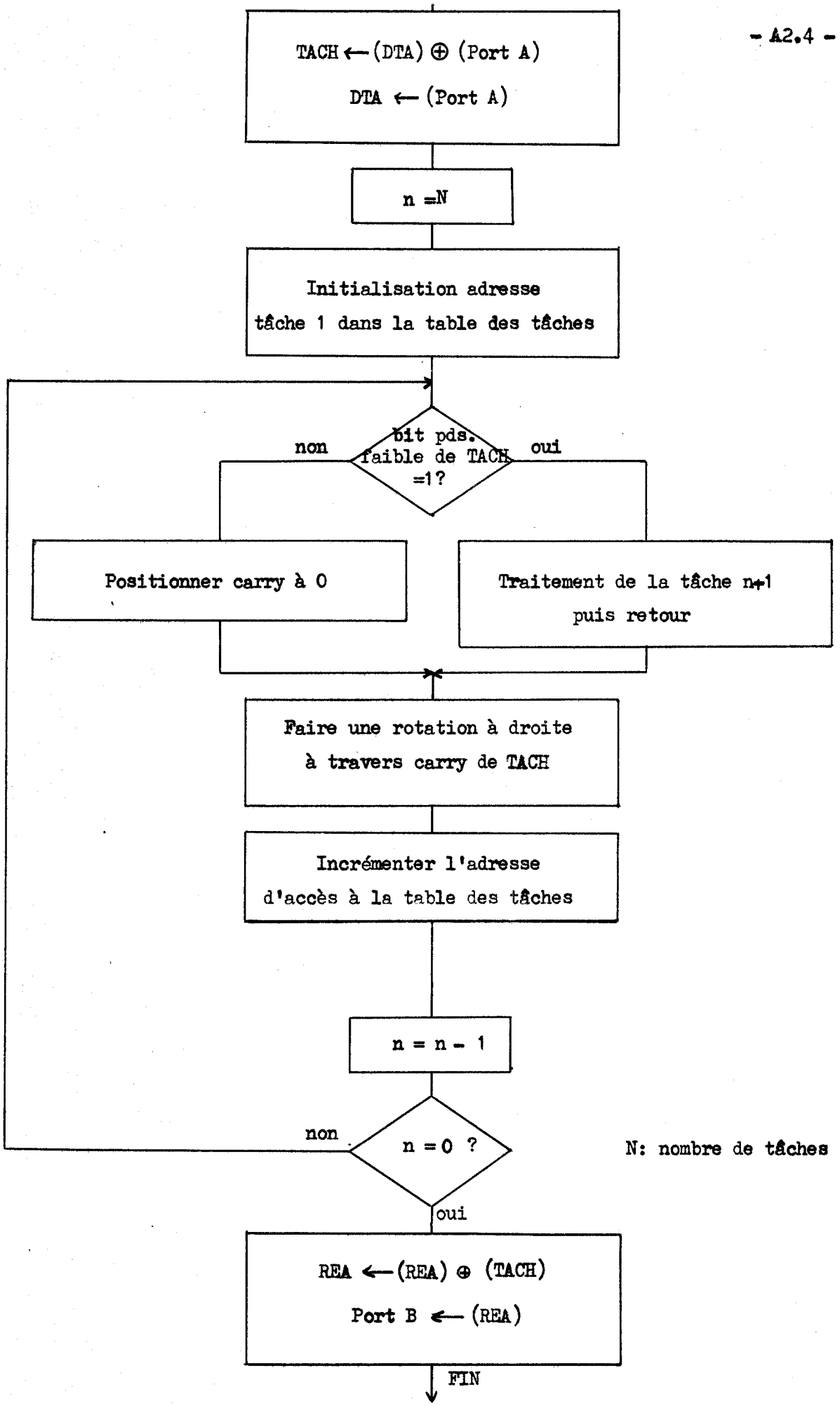


Les programmes de description des taches numériques sont à définir pour chaque application. L'organigramme relatif à la recherche des tâches demandées est donné fig. A2.2, le programme par la fig. A2.3. Ceux qui se rapportent aux temporisations sont donnés par les figures A2.4, A2.5 et A2.6.Ils font appel aux tables suivantes (fig. A2.7):



Adresses début des tâches Valeurs des temporisations Temps intermédiaires

- Fig. A2.7 -



- FIG. A2.2 - G E S T I O N D E S T A C H E S

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0 MODULE PAGE 1

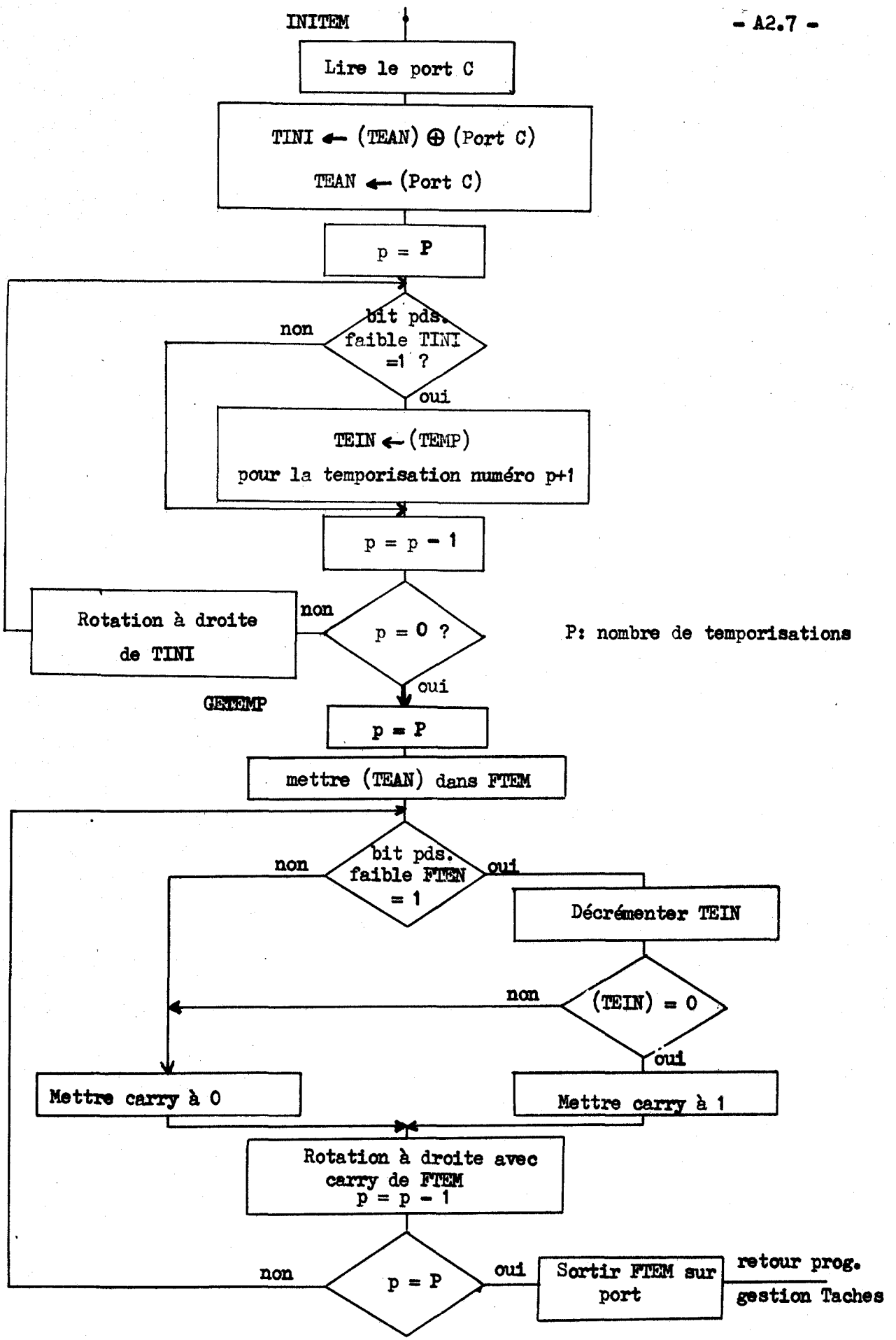
```
LOC  OBJ          SEQ          SOURCE STATEMENT

2840                1          ORG     2840H
28FF                2          SOMP    EQU     28FFH
2800                3          ADTA    EQU     2800H
2830                4          DTA     EQU     2830H
2831                5          TACH    EQU     2831H
2832                6          TFAN    EQU     2832H
2833                7          TINI    EQU     2833H
2834                8          FTFM    EQU     2834H
28A1                9          INITFM  EQU     28A1H
                   10
2840 3EC3          11          MVI     A,0C3H  ;INIT. HORLOGE
2842 D32D          12          OUT     2DH
2844 3EFF          13          MVI     A,0FFH
2846 D32C          14          OUT     2CH
2848 3EC2          15          MVI     A,0C2H  ;INIT.PORT TEMPO.
284A D328          16          OUT     28H
284C 3E02          17          MVI     A,02H   ;INIT. PORT TACHE
284E D320          18          OUT     20H
2850 31FF28        19          LXI     SP,SOMP
2853 219128        20          LXI     H,ADRET ;ADRES RETOUR PILE
2856 E5            21          PUSH    H
2857 213028        22          LXI     H,DTA   ;INIT DEM TACHE ANTER.
285A 3600          23          MVI     M,00H
285C 213228        24          LXI     H,TFAN  ;INIT DEM TEMPO ANTER.
285E 3600          25          MVI     M,00H
2861 DB28          26 DEBUT:   IN      28H     ;LIRE C/S REGISTER
2863 E640          27          ANI     40H     ;TEST TIMER BIT
2865 C2A128        28          JNZ     INITFM  ;SI PAS 0 TEMPO
2868 DB21          29          IN      21H     ;LIRE DEMANDE TACHE REC.
286A 47            30          MOV     B,A
286B 3A3028        31          LDA     DTA     ;DEM. TAC. ANTERIEURE
286E A8            32          XRA     B       ;ELABO. TACH
286F 323128        33          STA     TACH
2872 78            34          MOV     A,B     ;RAP. REC.DEM.TACHE
2873 323028        35          STA     DTA
2876 210028        36          LXI     H,ADTA  ;INIT.POINT.TAB.TACH
2879 0E09          37          MVI     C,9     ;INIT.COMPTEUR
287B AF            38          XRA     A       ;CARRY = 0
287C 3A3128        39 REPRI:   LDA     TACH    ;BIT P. FAIBLE DE
287F 1F            40          RAR             ;TACH DANS CARRY
2880 323128        41          STA     TACH
2883 D28C28        42          JNC     SUIT    ;SI CAR. =0 TACH NON DEM
2886 5E            43          MOV     E,M     ;SINON RECHERCHE ADRESSE
2887 23            44          INX     H
2888 56            45          MOV     D,M
2889 23            46          INX     H
288A EB            47          XCHG
288B F9            48          PCHL            ;SAUT SP TACHE
288C 23            49 SUIT:    INX     H       ;TACHE NON DEMANDEE
288D 23            50          INX     H
288E C36128        51          JMP     TEST
2891 3B            52 ADRET:   DCX     SP      ;REINIT. POINT. PILE
```

```
LOC  OBJ          SEQ          SOURCE STATEMENT

2892 3B             53         DCX     SP
2893 EB             54         XCHG
2894 0D             55 TEST:   DCR     C        ;DERNIERE TACHE ?
2895 C27C28         56         JNZ     DEPRI
2898 47             57         MOV     B,A      ;SINON SORTIR RESULTAT
2899 DB16           58         IN      22       ;LECT. RESULTAT PRECEDENT
289B A8             59         XRA     B
289C D316           60         OUT     22       ;SORTIR RESULTAT
289F C36128         61         JMP     DEBUT
                    62         END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
ADRET  A 2891    ADTA   A 2800    DEBUT  A 2861    DTA    A 2830
SOMP   A 28FF    SUIT   A 288C    TACH   A 2831    TRAV   A 2832

ASSEMBLY COMPLETE,  NO ERRORS
```

Initialisation du traitement numérique et

recherche des tâches demandées

- Fig. A2.3 -



T E M P O R I S A T I O N S

- Fig. A2.4 -

```
ASM80 INITFM.SCR

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0          MODULE   PAGE    1


  LOC  OBJ            SEQ          SOURCE STATEMENT

  28A1                  1            ORG     28A1H
  2832                  2            TFAN    EQU     2832H
  2833                  3            TINI    EQU     2833H
  2810                  4            TFMP    EQU     2810H
  2820                  5            TFIN    EQU     2820H
                        6                            ;PROGRAMME INITIALISATIO
                        7                            ;DES TEMPORISATIONS
  28A1 DB29             8  INITFM:    IN      29H     ;LIRE DEMANDE TEMPO
  28A3 4F               9            MOV     C,A
  28A4 3A3228          10            LDA     TFAN    ;FLAG. TEMPO A INITIAL.
  28A7 A9              11            XRA     C
  28A8 47              12            MOV     B,A
  28A9 79              13            MOV     A,C
  28AA 323228          14            STA     TFAN
  28AD 0E09            15            MVI     C,9H    ;INIT.COMPTEUR
  28AF 111028          16            LXI     D,TFMP  ;INIT POINTEUR TABLE
                       17                            ;DES VALEURS TEMPO
  28B2 212028          18            LXI     H,TFIN  ;INIT POINTEUR TABLE
                       19                            ;VALEURS INTERMEDIAIRES
  28B5 78              20            MOV     A,B
  28B6 323328          21            STA     TINI
  28B9 3A3328          22  BOU:      LDA     TINI    ;ISOLER BIT POIDS FAIBLE
  28BC 1F              23            RAR             ;DANS CARRY
  28BD 323328          24            STA     TINI
  28C0 DAC828          25            JC      INIT    ;C=1 TEMPO A INIT
  28C3 13              26            INX     D       ;SINON TEMPO SUIVANTE
  28C4 23              27            INX     H
  28C5 C3CF28          28            JMP     TEST
  28C8 1A              29  INIT:     LDAX    D       ;VAL.FAIBLE DE TEMPO
  28C9 77              30            MOV     M,A
  28CA 13              31            INX     D
  28CB 23              32            INX     H
  28CC 1A              33            LDAX    D       ;VAL.FORTE DE TEMPO
  28CD 77              34            MOV     M,A
  28CF 13              35  TEST:     INX     D
  28CF 23              36            INX     H
  28D0 0D              37            DCR     C
  28D1 C2B928          38            JNZ     BOU     ;DERNIERE TEMPO ?
                       39            END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
BOU    A 28B9    INIT   A 28C8    INITFM A 28A1    TFAN   A 2832    TFI
TINI   A 2833

ASSEMBLY COMPLETE,  NO ERRORS
```


- Fig. A2.5 - Initialisation des temporisations

```
ASM80 CFTEMP.SCR

ISIS-II 8080/8085 MACRO ASSEMBLER, V2.0          MODULE   PAGE    1

  LOC  OBJ            SEQ          SOURCE STATEMENT

  28D4                  1            ORG     28D4H
  2861                  2            DEBUT   EQU     2861H
  2832                  3            TFAN    EQU     2832H
  2834                  4            FTFM    EQU     2834H
  2820                  5            TFIN    EQU     2820H
                        6                            ;RECHERCHE ET DECREMEN-
                        7                            ;TATION DES TEMPORISA-
                        8                            ;TIONS ACTIVES
  28D4 AF               9 CFTEMP:    XRA     A       ;GESTION TEMPO (C=0)
  28D5 0E09            10            MVI     C,9H
  28D7 3A3228          11            LDA     TFAN    ;TRANSFERT DE TEMP. ANTE
  28DA 323428          12            STA     FTFM    ;DANS FIN TEMPO
  28DD 212028          13            LXI     H,TFIN  ;INIT POINTEUR  TEMP INE
  28E0 3A3428          14 BOUC:      LDA     FTFM    ;BIT POIDS FAIBLE DANS
  28E3 1F              15            RAR             ;CARRY
  28E4 323428          16            STA     FTFM
  28E7 D2E028          17            JNC     TDN     ;SI C=0 TEMPO NON ACTIVE
  28EA 7E              18            MOV     A,M     ;SINON VAL. FAIB. EN A
  28EB C699            19            ADI     99H     ;AJOUTER 99 EN DECIMAL
  28ED 27              20            DAA
  28EE 77              21            MOV     M,A
  28EF 47              22            MOV     B,A
  28F0 23              23            INX     H
  28F1 DAF928          24            JC      CONT    ;SI C=1 DECREM.TERMINEE
  28F4 7E              25            MOV     A,M     ;SINON IDEM P. FORTS
  28F5 C699            26            ADI     99H
  28F7 27              27            DAA
  28F8 77              28            MOV     M,A
  28F9 23              29 CONT:      INX     H
  28FA B0              30            ORA     B       ;TEMPO A 0 ?
  28FB C20429          31            JNZ     FIN     ;SI Z=0 C RESTE A 0
  28FE 3F              32            CMC             ;SINON C=1
  28FF C30429          33            JMP     FIN
  2902 23              34 TDN:       INX     H       ;TEMPO SUIVANTE
  2903 23              35            INX     H
  2904 0D              36 FIN:       DCR     C       ;DERNIERE TEMPO ?
  2905 C2E028          37            JNZ     BOUC
  2908 D32A            38            OUT     2AH     ;SORTIR FIN TEMPO
  290A C36128          39            JMP     DEBUT   ;REPRISE DU TRAITEMENT
                       40                            ;NUMERIQUE
                       41            END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS
BOUC   A 28E0    CONT   A 28F9    DEBUT  A 2861    FIN    A 2904    FTF
TFAN   A 2832    TFIN   A 2820

ASSEMBLY COMPLETE,  NO ERRORS
```

Décrémentation des temporisations actives

- Fig. A2.6 -

A2.3 - Logiciel de l'automate logique avec détection de défauts

Le programme moniteur est modifié comme suit:

Entrées REGISTRE RESULTAT ..... RR adresse 820

CONTROLE DEFAUT ..... CD adresse 821

Sorties SYCHRONISATION ..... YO adresse 824

MEMORISATION DEFAUT ..... ZD adresse 825

Ces entrées sorties de service ne sont pas vues par le microprocesseur à travers des registres.

```
000  68 20   INIT: ORC  RR    ;mise à 1 des registres IEN et OEN
001  A8 20         IEN  RR
002  B8 20         OEN  RR
003  88 24         STO  YO    ;1ère impulsion de synchronisation
004  98 24         STOC YO
005  88 24         STO  YO    ;2ème impulsion de synchronisation
006  98 24         STOC YO
007  98 25         STOC ZD    ;mise à 0 mémoire défaut
008  D0 00         RTN        ;initialisation pointeurs table des pas
009  18 20         LD   RR    ;instruction non traitée après RTN
00A  08 20         ETA  A1    ;adresse 1er pas dans la table
00B  F8 00         INE
00C  08 0C   SYNC: ETA  SYNC  ;indicateur fin de table
00D  18 21         LD   CD    ;lecture ligne défaut
00E  48 25         ANDC ZD    ;ET mémoire défaut complémentée
00F  E0 00         SKZ        ;si CD∧ZD̄ = 1 alors
010  C8 18         JMP  B0    ;branchement à B0 pour initialiser la
                              ;procédure d'arrêt
011  68 20         ORC  RR    ;impulsion de synchronisation
012  88 24         STO  YO
013  98 24         STOC YO
014  F8 00   NFRA: INE        ;voir moniteur sans défaut
015  18 20         LD   RR
016  F0 00   FRAN: INL
017  C0 00         JTA
018  68 20   B0:   ORC  RR    ;mise à 1 de la mémoire défaut ZD
019  88 25         STO  ZD
                              ;zone réservée à l'initialisation ou
                              ;à la description de la prcédure d'arrêt
                              ;cette zone peut être étendue par un saut.
```

A N N E X E III

PROGRAMME DE COMMANDE D'UNE PRESSE AVEC DETECTION DE DEFAUT

Le cahier des charges de cet équipement est donné sous forme d'un réseau de Pétri par la figure V.6. Ce réseau est repris ici en portant les adresses des entrées et des sorties cablées (Fig. A3.1).

Le résultat R1 est atteint si le compteur est à 40

Le résultat R2 est atteint si le compteur est à 50

Le résultat R3 est atteint si le compteur est à 90

Le résultat R4 est atteint si le compteur est à 100

Le réseau est temporisé par les temporisations notées T2 et T3.L'initialisation du compteur et sa décrémentation sont demandées respectivement par DT1 et DT2.

Les entrées et leurs variations sont repérées comme suit:

ENTREE	NIVEAU	VARIATION
m1	X20	X28
m2	X21	X29
C1	X10	X18
C2	X11	X19
C3	X12	X1A
C4	X13	X1B
C5	X14	X1C
C9	X15	X1D
C10	X16	X1E
I	X17	X1F

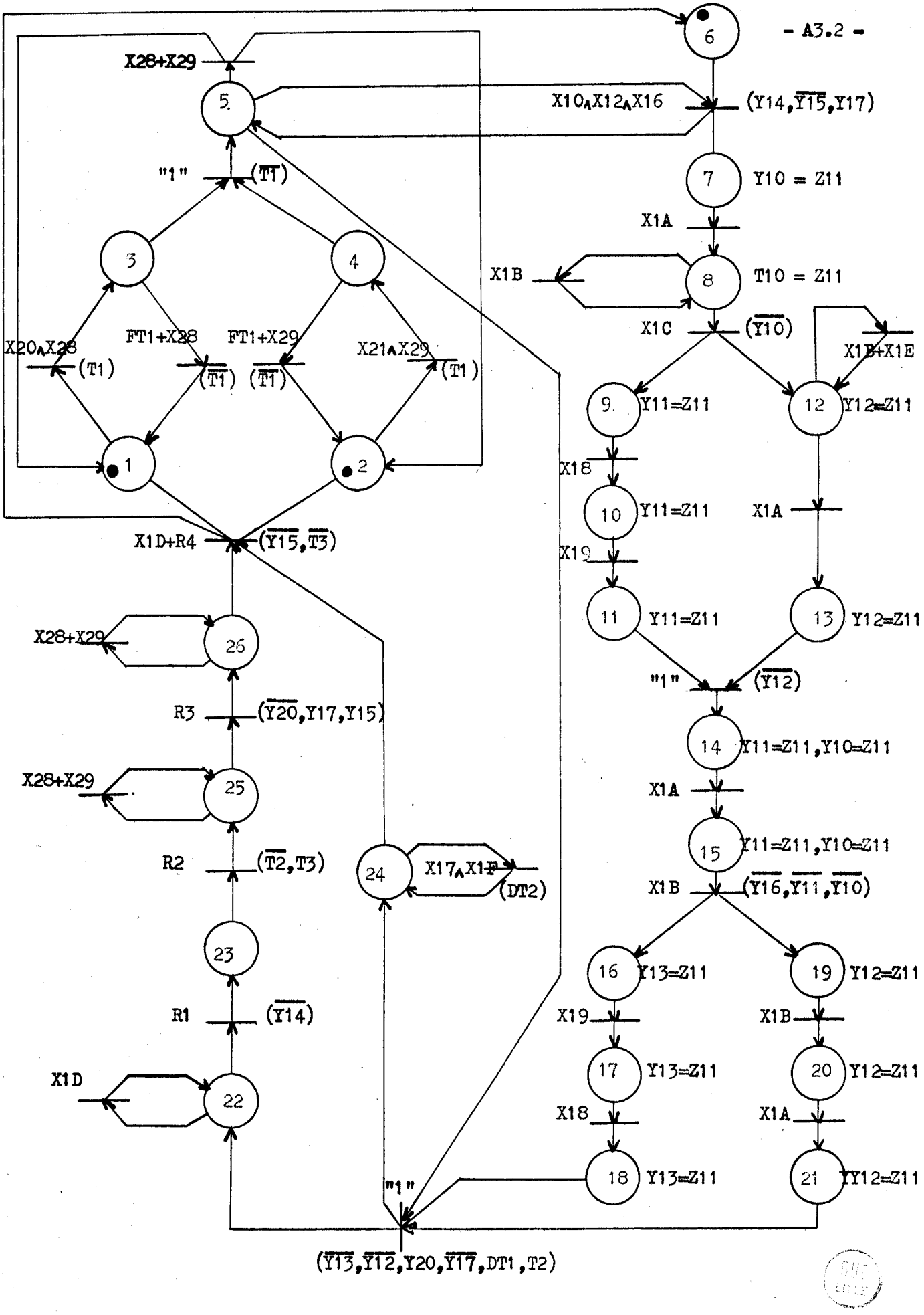
Les variables internes représentant le marquage des places de synchronismes sont repérées par:

M5 = Z11, M4 = Z10, M13 =Z12, M21 = Z13, M24 = Z14;

Les sorties sont repérées par:

BC = Y10, DC = Y11, HC = Y12, GC = Y13, B = Y14, E = Y15, A = Y16,

F = Y17, R = Y20.


6
X28+X29
5
X10∧X12∧X16
(Y14,Y15,Y17)
"1"
(T1)
7
Y10 = Z11
X1A
3
4
X1B
8
T10 = Z11
X1C
(Y10)
X1B+X1E
X20∧X28
(T1)
FT1+X28
(T1)
FT1+X29
(T1)
X21∧X29
(T1)
9
Y11=Z11
12
Y12=Z11
1
2
X18
10
Y11=Z11
X1A
X19
X1D+R4
(Y15,T3)
11
Y11=Z11
13
Y12=Z11
X28+X29
26
"1"
(Y12)
R3
(Y20,Y17,Y15)
14
Y11=Z11,Y10=Z11
X28+X29
25
X1A
15
Y11=Z11,Y10=Z11
R2
(T2,T3)
24
X17∧X1F
(DT2)
X1B
(Y16,Y11,Y10)
23
16
Y13=Z11
19
Y12=Z11
R1
(Y14)
X19
X1B
17
Y13=Z11
20
Y12=Z11
X1D
22
X18
X1A
18
Y13=Z11
21
Y12=Z11
"1"
(Y13,Y12,Y20,Y17,DT1,T2)

- Fig. A3.1 -

PROGRAMME DE L'AUTOMATE LOGIQUE DANS LE CADRE DE LA COMMANDE D'UNE PRESSE.

```
A0   ETA  A1
     INE
     ETA  A2
     INE
     ETA  A6
     INE
     JMP  FRAN

A1   ETA  A1
     LD   X20
     AND  X28
     SKZ
     JMP  B1
     JMP  NFRA

A2   ETA  A2
     LD   X21
     AND  X29
     SKZ
     JMP  B2
     JMP  NFRA

A3   ETA  A3
     LD   FT1
     OR   X28
     SKZ
     JMP  B31
     LD   Z10
     SKZ
     JMP  B32
     JMP  NFRA

A4   ETA  A4
     LDC  Z10
     SKZ
     JMP  FRAN
     LD   FT1
     OR   X28
     SKZ
     JMP  B4
     JMP  NFRA

A6   ETA  A6
     LD   Z11
     AND  X10
     AND  X12
     AND  X16
     SKZ
     JMP  B6
     JMP  NFRA

A7   ETA  A7
     LD   Z11
     STO  Y10
     LD   X1A
     SKZ
     JMP  B7
     JMP  NFRA

A8   ETA  A8
     LD   Z11
     STO  Y10
     LD   X1B
     LD   X1C
     SKZ
     JMP  B8
     JMP  NFRA

A9   ETA  A9
     LD   Z11
     STO  Y11
     LD   X18
     SKZ
     JMP  B9
     JMP  NFRA

A10  ETA  A10
     LD   Z11
     STO  Y11
     LD   X19
     SKZ
     JMP  B10
     JMP  NFRA

A11  ETA  A11
     LD   Z11
     STO  Y11
     LD   Z12
     SKZ
     JMP  B11
     JMP  NFRA

A12  ETA  A12
     LD   Z11
     STO  Y12
     LD   X1B
     OR   X1E
     LD   X1A
     SKZ
     JMP  B12
     JMP  NFRA

A5   ETA  A5
     LD   Z11
     SKZ
     JMP  FRAN
     LD   X28
     OR   X29
     SKZ
     JMP  B5
     JMP  NFRA
```

```
A13   ETA   A13
      LDC   Z12
      SKZ
      JMP   FRAN
      LD    Z11
      STO   Y12
      JMP   NFRA

A14   ETA   A14
      LD    Z11
      STO   Y11
      STO   Y10
      LD    X1A
      SKZ
      JMP   B14
      JMP   NFRA

A15   ETA   A15
      LD    Z11
      STO   Y11
      STO   Y10
      LD    X1B
      SKZ
      JMP   B15
      JMP   NFRA

A16   ETA   A16
      LD    Z11
      STO   Y13
      LD    X19
      SKZ
      JMP   B16
      JMP   NFRA

A17   ETA   A17
      LD    Z11
      STO   Y13
      LD    X18
      SKZ
      JMP   B17
      JMP   NFRA

A18   ETA   A18
      LD    Z11
      STO   Y13
      LD    Z13
      SKZ
      JMP   B18
      JMP   NFRA

A19   ETA   A19
      LD    Z11
      STO   Y12
      LD    X1B
      SKZ
      JMP   B19
      JMP   NFRA

A20   ETA   A20
      LD    Z11
      STO   Y12
      LD    X1A
      SKZ
      JMP   B20
      JMP   NFRA

A21   ETA   A21
      LDC   Z13
      SKZ
      JMP   FRAN
      LD    Z11
      STO   Y12
      JMP   NFRA

A22   ETA   A22
      LD    X10
      LD    R1
      SKZ
      JMP   B22
      JMP   NFRA

A23   ETA   A23
      LD    R2
      SKZ
      JMP   B23
      JMP   NFRA

A24   ETA   A24
      LDC   Z14
      SKZ
      JMP   FRAN
      LD    X17
AND   AND   X1F
      SKZ
      JMP   B24
      JMP   NFRA

A25   ETA   A25
      LD    X28
      OR    X29
      LD    R3
      SKZ
      JMP   B25
      JMP   NFRA

A26   ETA   A26
      LD    X10
      OR    R4
      AND   Z14
      SKZ
      JMP   B26
      JMP   NFRA
```

```
B1      ETA     A3
        INE
        STO     T1
        JMP     FRAN

B2      STO     T1
        STO     Z10
        JMP     FRAN

B31     ETA     A1
        INE
        STOC    T1
        JMP     FRAN

B32     STOC    T1
        STOC    Z10
        STO     Z11
        JMP     FRAN

B4      ETA     A2
        INE
        STOC    Z10
        STOC    T1
        JMP     FRAN

B6      ETA     A7
        INE
        STO     Y14
        STOC    Y15
        STO     Y16
        JMP     FRAN

B7      ETA     A8
        INE
        JMP     FRAN

B8      ETA     A9
        INE
        ETA     A12
        INE
        STOC    Y10
        JMP     FRAN

B9      ETA     A10
        INE
        JMP     FRAN

B10     ETA     A11
        INE
        JMP     FRAN

B11     ETA     A14
        INE
        STOC    Z12
        STOC    Y12
        JMP     FRAN

B12     ETA     A13
        INE
        STO     Z12
        JMP     FRAN

B14     ETA     A15
        INE
        STO     Z12
        JMP     FRAN

B14     ETA     A15
        INE
        JMP     FRAN

B15     ETA     A16
        INE
        ETA     A19
        INE
        STOC    Y16
        STOC    Y10
        STOC    Y11
        JMP     FRAN

B16     ETA     A17
        INE
        JMP     FRAN

B17     ETA     A18
        INE
        JMP     FRAN

B18     ETA     A22
        INE
        ETA     A24
        INE
        STOC    Z11
        STOC    Z13
        STOC    Y13
        STOC    Y12
        STOC    Y17
        STO     Y20
        STO     T2
        STO     Z15
        LD      DT1
        STOC    DT1
        JMP     FRAN

B19     ETA     A20
        INE
        JMP     FRAN

B20     ETA     A21
        INE
        STO     Z13
        JMP     FRAN
```

```
B22   ETA   A23
      INE
      STOC  Y14
      JMP   FRAN

B23   ETA   A25
      INE
      STOC  T2
      STO   T3
      STOC  Z15
      STO   Z16
      JMP   FRAN
B24   ETA   A24
      INE
      LD    DT2
      STOC  DT2
      JMP   FRAN
B25   ETA   A26
      INE
      STO   Y15
      STOC  Y20
      STO   Y17
      JMP   FRAN

B26   ETA   A6
      INE
      ETA   A1
      INE
      ETA   A2
      INE
      STOC  Z16
      STOC  Y15
      STOC  T3
      STOC  Z14
      JMP   FRAN

B5    ETA   A1
      INE
      ETA   A2
      INE
      STOC  Z11
      JMP   FRAN
```

PROCEDURE D'ARRET D'URGENCE EN CAS DE DEFAUT

```
B0      RTN
        LD      RR
        ETA     A41
        INE
        ETA     A42
        INE
        ETA     SYNC
        INE
        ORC     RR
        STOC    Y10
        STOC    Y11
        STOC    Y12
        STOC    Y13
        STOC    Y14
        STOC    Y15
        STO     ZD
        JMP     FRAN

A41     ETA     A41
        LD      X17
        AND     X1F
        SKZ
        JMP     B41
        JMP     NFRA

A42     ETA     A42
        LD      Z15
        SKZ
        JMP     B42
        LD      Z16
        SKZ
        JMP     B46
        JMP     NFRA

A43     ETA     A43
        LD      X1D
        OR      R3
        SKZ
        JMP     B43
        JMP     NFRA

B41     ETA     A41
        INE
        LD      DT2
        STOC    DT2
        JMP     FRAN

B42     STOC    Y20
        STO     Y17
        JMP     FRAN

B46     ETA     A43
        INE
        JMP     FRAN

B43     STOC    Y20
        STO     Y17
        JMP     FRAN
```

B I B L I O G R A P H I E

[1] AFCET. Groupe de travail Systèmes Logiques. "Pour une représentation normalisée du cahier des charges d'un automatisme logique". Revue Automatique et Informatique Industrielle n°61-62. Novembre/Décembre 1977.

[2] AFCET. Rapport de la commission Systèmes Logiques présenté par M. BLANCHARD. "Normalisation du cahier des charges d'un automatisme logique". Revue Automatisme tome XXIII n°3-4 Mars/Avril 1978.

[3] C. ANDRE. "Sur une méthode de conception assistée par ordinateur des systèmes logiques à évolutions simultanées". Thèse Docteur 3ème cycle, Nice, juin 1975.

[4] C. ANDRE, F. BOERI, J. MARIN. "Synthèse et réalisation des systèmes logiques à évolutions simultanées". Revue Française d'Automatique, Informatique et Recherche Opérationnelle (RAIRO) Vol. 10 n°4, avril 1976.

[5] C. BEOUNES. "Automate sûr et modulaire adapté aux régulations avioniques". Thèse Docteur Ingénieur. Toulouse, Univ. Paul Sabatier 1977.

[6] M. BLANCHARD, J.C. CAVARROC, V. FUERTES, J. GILLON, G. THUILLIER. "Conception modulaire d'automatismes séquentiels asynchrones". Rapport DGRST n°71.7.2912.01, DERA-Télémécanique Electrique, janvier 1976.

[7] M. BLANCHARD, V. FUERTES, J. GILLON, G. THUILLIER. "Un automate programmable par réseau de Pétri". Colloque AFCET/ADEPA Automatismes logiques: recherche et applications industrielles. Paris, décembre 1976.

[8] E. DACLIN, M. BLANCHARD. "Synthèse des systèmes logiques". Editions CEPADUES, coll. SUP'AERO, 1976.

[9] G. DELORY. "Automate programmé par réseaux de Pétri". Mémoire d'Ingénieur CNAM, Lille, mai 1978.

[10] M. DIAZ. "Conception de systèmes totalement auto-testables et à pannes non dangereuses". Thèse de Docteur ès Sciences, Univ. Paul Sabatier, Toulouse, juin 1974.

[11] P. ESQUISSAUD, G. GUESNIER. "Systèmes logiques de commande dans les centrales électriques: fonctionnement et sûreté". Journées d'études AFCET/SEE février 1978.

[12] J. GILLON. "Une nouvelle approche de la synthèse des systèmes logiques". Journées d'études AFCET/SEE février 1978.

[13] INTEL "MCS 85 User's Manual". Septembre 1978.

[14] J. MARIN. "Sur le test en ligne des machines séquentielles réalisées à partir des réseaux de Pétri". Thèse de Docteur 3ème cycle Nice, décembre 1975.

[15] MOTOROLA "MC 14500B-Industrial Control Unit Handbook". 1977

[17] P. NASLIN. "Circuits logiques et automatismes à séquences". Editions Dunod, 1970.

[17] J.P. PARSY. "Méthode de décomposition des réseaux de Pétri interprétés en vue de leur réalisation". DEA Automatique, Lille, juin 1978.

[18] M. SILVA SUAREZ. "Contribution à la synthèse programmée des automates logiques". Thèse INPG, Grenoble 1978.

[19] B. TACONET, B. CHOLLOT. "Programmation de Grafcet sur automate à langage logique, à relais ou booléen". Revue Automatisme tome XXIV n° 1-2, janvier/février 1979.

[20] S. THELLIEZ. "Pratique séquentielle et réseaux de Pétri". Editions Eyrolles, coll. E.E.A.

[21] J.M. TOULOTTE. "Réseaux de Pétri et automates programmables" Revue Automatisme, tome XXIII, n° 6-7, juillet/août 1978.

[23] R. VALETTE, J.G. GEFFROY, M. COURVOISIER. "Modélisation des systèmes de commande numérique et leur analyse". Congrès AFCET, Toulouse, novembre 1975.

[24] R. VALETTE. "Sur la description, l'analyse et la validation des systèmes de commande parallèles". Thèse Doctorat ès Sciences, Univ. Paul Sabatier, Toulouse, novembre 1976.