Méthodes et outils de test pour microprocesseurs et circuits périphériques

Sylvain Sadier

THESE

présentée à

l' Institut National Polytechnique de Grenoble

pour obtenir le grade de
DOCTEUR INGENIEUR

par

SADIER Sylvain

METHODES ET OUTILS DE TEST POUR MICROPROCESSEURS ET CIRCUITS PERIPHERIQUES.

Thèse soutenue le 7 décembre 1983 devant la commission d'examen.

	L. BOLLIET	Président
	M. LAURENT	Examinateurs
	F. GRILLOT	
	O. MOULIN	
Mmes	C. BELLON	
	G. SAUCIER	

Je tiens à remercier Monsieur Louis BOLLIET, professeur à l'IUT II, de m'avoir fait l'honneur de présider le jury de cette thèse.

Je tiens à exprimer toute ma reconnaissance à Madame Gabrièle SAUCIER, professeur à l'ENSIMAG, de m'avoir permis de mener à bien ce travail dans l'équipe qu'elle dirige.

Que Monsieur François GRILLOT, chef du service Test et Conception Assistée, ainsi que tous les ingénieurs et techniciens de ESD ayant participé à ce projet soient vivement remerciés de leur collaboration et de leur amabilité.

Que Monsieur Olivier MOULIN, professeur à l'ESIEE, soit remercié d'avoir toujours suivi avec intérêt les travaux que j'ai pû mener au sein de cette équipe.

Je tiens à remercier Monsieur Marc LAURENT, ingénieur à MATRA-HARRIS, d'avoir participé à ce jury et d'avoir facilité les échanges avec l'industrie que ce soit avec l'IMAG ou l'ESIEE.

Mademoiselle Catherine BELLON trouvera ici l'expression de toute ma gratitude pour avoir su avec patience diriger ces travaux de recherche et de m'avoir toujours fait part de remarques constructives.

Il serait injuste de ne pas associer à ces remerciements toute l'équipe "Conception et sécurité des systèmes" qui a su créer un environnement humain des plus agréable.
Je remercie Madame Solange ROCHE pour ses précieux conseils et son aide concernant l'utilisation du système de traitement de texte, et Monsieur Daniel IGLESIAS et tout le service de reprographie qui ont assuré la réalisation de qualité de cette thèse.

A MA FAMILLE,

A BRIGITTE.

La tempête arrache les mâts
et la voilure; la brise porte
le bateau sur le bon cap.

AVANT PROPROS

CHAPITRE 1

INTRODUCTION

I - LE TEST - LES METHODES DE TEST

I - 1. Position du problème

1.1.1. Les fabricants de circuits

1.1.2. Les constructeurs de systèmes à base de LSI

1.1.3. Exploitation

I - 2. Niveau de connaissance du circuit

1.2.1. Fabricants de circuits

1.2.2. Constructeurs de systèmes et utilisateurs

1.2.3. Le manuel techniques

I - 3. Les étapes du test

I - 4. Méthodes de test

1.4.1. Généralités

1.4.2. Méthodes d'obtention des vecteurs de test

1.4.3. Moyens d'obtention des vecteurs de test

1.4.4. Observation des résultats

I - 5. Mise en oeuvre du test

1.5.1. Production des vecteurs de test

1.5.2. Production des résultats de la machine juste

1.5.3. Le système de test

I - 6. Définitions

1.6.1. Généralités

1.6.2. Modèle de défaillance

1.6.3. Hypothèses de pannes

1.6.4. Hypothèses d'erreurs

1.6.5. Test et taux de couverture

II - LES CIRCUITS INTEGRES

II - 1. Complexité et densité

2.1.1. Complexité des circuits intégrés

2.1.2. Densité des circuits intégrés

2.1.3. Moyens d'accès aux circuits intégrés

II - 2. Les générations de microprocesseurs

2.2.1. Les microprocesseurs en tranches

2.2.2. Les microprocesseurs à usage général

2.2.3. Les microordinateurs monopuces

2.2.4. Les coprocesseurs

II - 3. Fonctionnement et types de broche des circuits

II - 4. Les circuits périphériques

III - SPECIFICATIONS DU PROJET GAPT

III - 1. Méthode de test usager

III - 2. Le générateur automatique de programmes de test

III - 3. Le système de test

CHAPITRE 2

INTRODUCTION

I - MACHINE SEQUENTIELLE COMPORTEMENTALE

I - 1. Rappels sur la description des automates

1.1.1. Les entrées et les sorties

1.1.2. Les états

1.1.3. La fonction d'état suivant

1.1.4. La fonction de sortie

I - 2. Machine séquentielle comportementale

1.2.1. Ensemble des éléments de mémoire comportementaux

1.2.2. Etat comportemental d'un circuit

1.2.3. Les entrées et les sorties de la machine séquentielle

1.2.4. Les fonctions de la machine séquentielle

I - 3. Description pratique des fonctions d'un microprocesseur

II - METHODE DE TEST

II - 1. Test d'automate par identification

II - 2. Méthode de test pratique

2.2.1. Réduction de l'ensemble des transitions à tester

2.2.2. Méthode de test

CHAPITRE 3

INTRODUCTION

I - MACHINE DE TEST

I - 1. Choix de l'environnement

I - 2. Structure de l'espace mémoire

1.2.1. La mémoire programme

1.2.2. La mémoire signal

I - 3. Présentation générale du testeur

I - 4. Description fonctionnelle du tiroir de test

1.4.1. L'espace mémoire

1.4.2. Le système d'observation des sorties

1.4.3. Les interfaces

1.4.4. Les registres d'activité

II - LE GENERATEUR AUTOMATIQUE DE PROGRAMME DE TEST

II - 1. Introduction

II - 2. Architecture de GAPT

2.2.1. Introduction

2.2.2. Le langage de description de GAPT

III - LE TEST DES SIGNAUX

III - 1. Introduction

III - 2. Mise en oeuvre

III - 3. Description des signaux et des chronogrammes

3.3.1. Définitions

3.3.2. Description des chrongrammes

CONCLUSION

ANNEXE 1

ANNEXE 2

CHAPITRE 4

INTRODUCTION

I - MODELE DE REFERENCE

I - 1. Rappels

I - 2. Hiérarchisation du modèle

I - 3. Graphe de séquencement des macrophases

1.3.1. Ensemble des macrophases

1.3.2. Les transitions

I - 4. Graphe de séquencement des phases

I - 5. Temporisation du modèle

I - 6. Exemple d'application

AVANT PROPOS.

L'utilisation de composants à haute intégration pose de manière accrue le problème du test. La réduction des possibilités d'accès à ces circuits rend les processus de test de plus en plus difficiles.

Parmi les composants des systèmes à microprocesseur certains posent des problèmes importants et pourtant peu abordés: ce sont les circuits périphériques de microprocesseur.

Actuellement les méthodes de test destinées aux microprocesseurs ont été développées en s'orientant de plus en plus vers des solutions fonctionnelles.

Après avoir situé le problème du test dans le chapitre I , nous proposons une méthode de test basée sur une description fonctionnelle du circuit, dans le chapitre II
Le chapitre III est consacré à l'étude du système de test développé en collaboration avec ESD , ainsi qu'au test des signaux asynchrones positionnés en entrée sur le circuit sous test.

L'application à un circuit périphérique est développée dans le chapitre IV. A partir d'un modèle de référence composé de deux niveaux de description et d'un ensemble d'informations concernant le circuit,on donne la structure fine du programme de test du circuit en question. Le test se déroule sur la machine de test décrite au chapitre III.

CHAPITRE 1

INTRODUCTION

Depuis l'introduction sur le marché des circuits intégrés, leur test a toujours présenté des difficultés; la répartition et l'acuité de ces difficultés sont très différentes suivant le type de test (fabricants ou utilisateurs) et suivant le degré d'intégration du circuit concerné (SCR78)(SAU81-2)(SM177).

Dans un premier temps on situera le problème du test du fabricant à l'utilisateur en considérant les niveaux de connaissance des circuits, les méthodes de test et leur mise en oeuvre ainsi que les hypothèses de mauvais fonctionnement.

Dans un deuxième temps on étudie qualitativement l'évolution des circuits intégrés, en complexité et en densité; on aborde ensuite le problème du test des microprocesseurs et de leurs circuits périphériques.

I - LE TEST - LES METHODES DE TEST

I.1 POSITION DU PROBLEME

Les circuits à haute intégration posent des problèmes nouveaux vis-à-vis du test ; ces problèmes sont de natures très diverses et se concrétisent au stade:

- des fabricants de circuits (conception, technologie, fabrication).
- des constructeurs de systèmes à base de circuits LSI.
- des exploitants.

I.1.1 Les fabricants de circuits (ROB78)(DEL77)

Nous résumons dans le tableau I ci-dessous les différents tests, leurs objectifs et les moyens mis en oeuvre.

ETAPE	BUT	MOYEN
CONCEPTION	CONFORMITE AUX SPECIFICATIONS	SIMULATION LOGIQUE SIMULATION ELECTRIQUE
FABRICATION PROTOTYPES PRESERIE ECHANTILLON	CENTRAGE DE LA TECHNOLOGIE VERIFICATION DU CAHIER DES CHARGES AFFINEMENTS DES SPECIFICATIONS	TEST FONCTIONNEL ET TEST DYNAMIQUE ET TEST TECHNOLOGIQUE
SERIE		TEST DE QUALIFICATION TEST DE LOTS TEST RAPIDE

TABLEAU I

I.1.2 Les constructeurs de systèmes à base de LSI

Sauf cas particuliers (constructeurs de systèmes filiales de fabricants de circuits), les connaissances des constructeurs de systèmes à base de LSI sur les circuits intégrés restent très limitées. Nous résumerons les étapes de la conception d'un système dans le tableau II.

ETAPES	BUT	MOYEN
ETUDE	CAHIER DES CHARGES	
	VALIDATION	SIMULATION
	CONCEPTION FONCTIONNELLE	
	VALIDATION	SIMULATION
	CONCEPTION MATERIELLE	
FABRICATION PROTOTYPE	VALIDATION DU CAHIER DES CHARGES	TEST FONCTIONNEL
		TEST PARAMETRIQUE
PRESERIE	AFFINEMENT DES SPECIFICATIONS	TEST LOGIQUE
SERIE	APPROVISIONNEMENT EN MATERIEL	INSPECTION D'ENTREE
	SOUS ENSEMBLE	TEST DE CARTES
	INTEGRATION	BANC DE TEST . RECETTE MISE AU POINT

TABLEAU II

I.1.3 Exploitation

Les problèmes posés aux exploitants sont similaires à ceux rencontrés par les constructeurs de systèmes. Il se greffe sur cet ensemble de contraintes le problème de la disponibilité de leurs équipements. Pour ceci on met en oeuvre des techniques spécifiques:

* Test en ligne: Cette détection peut être continue (insérée dans l'application) ou discontinue (en état d'oisiveté réelle ou forcée du système) (SAU82-2)(COU81)(BEL83-2).

* Test hors ligne: Le test se fait sur un équipement de test après échange de la carte défectueuse.(BEL83-2).

On remarque ainsi la grande diversité des situations dans lesquelles on peut être amener à tester des ensembles (circuits, cartes).

Suivant le même schéma (fabricants de circuits, constructeurs de systèmes, exploitants), on peut situer la qualité des informations exploitables en vue du test.

I - 2. NIVEAU DE CONNAISSANCE DU CIRCUIT (BEL77)

I.2.1 Fabricants de circuits

Ils possèdent naturellement toutes les données possibles concernant le circuit à tester; ils disposent:

- du découpage réel fin en blocs fonctionnels.
- des schéma logiques.
- des paramètres de la technologie.

Le niveau de connaissance s'établit au niveau STRUCTUREL fin.

I.2.2 Constructeurs de systèmes et utilisateurs

Le niveau de connaissance des utilisateurs et des constructeurs de système est en général limité au manuel technique des circuits intégrés.

1.2.3 Le manuel technique (NSC81)(MOTO1)(ZIL83)

Ce document donne toutes les informations nécessaires à la mise en oeuvre du circuit qu'il décrit. On y trouve principalement:

- des données succintes sur l'architecture du circuit.
- les registres et éléments mémoire accessibles.
- la description des modes d'adressage.
- la description du langage d'assemblage.
- la description du fonctionnement du circuit.
- la description des actions des signaux asynchrones.

Le niveau de connaissance du circuit s'établit au niveau ARCHITECTURAL ou FONCTIONNEL.

1 - 3. LES ETAPES DU TEST génération et mise en oeuvre (ABA83)

Il est nécessaire de dissocier dans les étapes du test différentes opérations qui peuvent être soit très fortement imbriquées soit totalement indépendantes.

Toutes ces opérations dépendent de deux paramètres principaux :

- la qualité du test.
- le type de mise en oeuvre.

L'élément central de la chaîne est l'équipement de test. En amont du testeur, on trouve un ensemble d'étapes fortement dépendantes les unes des autres débouchant sur l'envoi des vecteurs de test au circuit par l'intermédiaire du système. En aval du testeur, on dispose des résultats donnés par le circuit sous test. Les résultats de la machine juste sont produits de différentes façons.

L'observation des résultats se fait par comparaison de la séquence des résultats avec une référence juste ; cette séquence de sortie ainsi que la référence peuvent être compactées (registres de signature). (CAR82) (FRU77) (GUR77).

I - 4. METHODES DE TEST

1.4.1 Généralités (ROB79)

Dans le test d'un composant, on peut distinguer selon les buts poursuivis, deux types d'essais :

- les tests paramétriques (SAD81) sont appliqués au circuit, soit en fin de fabrication pour la qualification, soit en inspection d'entrée afin de vérifier son aptitude à subir des contraintes sévères d'utilisation.

- les tests logiques (ROB78)(ROB72) sont appliqués à différents niveaux de la vie d'un composant: en fin de fabrication pour s'assurer de la fonctionnalité du circuit, en intégration dans un sous système, en maintenance.

Nous nous intéressons ici aux méthodes classiques de test concernant les composants intégrés logiques et les évolutions vers le test de circuits logiques à haute intégration (microprocesseurs, circuits périphériques).

1.4.2 Méthodes d'obtention des vecteurs de test (ARE80)

On distingue deux types de méthodes :

- des méthodes de test à vecteurs prédéterminés.
- des méthodes de test à vecteurs aléatoires.

Le choix de l'un ou l'autre type depend du niveau de connaissance du circuit.

1.4.3 Moyens d'obtention des vecteurs de test

a) Vecteurs prédéterminés:(SAU81-1)

Génération déterministe: ces moyens nécessitent la considération soit de la structure soit des fonctions du circuit; elle consiste à partir d'un ensemble de pannes du circuit à chercher un ensemble de vecteurs d'entrée qui détectera ces pannes. Par exemple ces vecteurs d'entrée délivreront des sorties distinctes en présence et en l'absence d'une panne déterminée.

- méthode de propagation des pannes : D-algorithme. (JAI83)(ROT68)
- méthode algébrique : différence booléenne.
- méthode fonctionnelle : vérification de la fonctionnalité.
  identification d'automate.(KOH70).
  distinction entre un fonctionnement juste
  et un fonctionnement erroné.

Génération aléatoire:(AND80) la simulation de circuits sous test en présence de pannes (du circuit juste) permet d'établir à partir de séquences générées aléatoirement les séquences de test détectant les pannes.

b) Vecteurs aléatoires: (DAV81)(DAV79)

Les vecteurs de test sont générés aléatoirement sans étude préalable du circuit au moment du test. Si la structure du circuit est connue, une analyse probabilistique permet d'estimer les longueurs des séquences de test.

1.4.4 Observation des résultats

Quelle que soit l'utilisation des résultats de test (localisation de la panne, CONUGO (SAU81)) on distingue deux principaux types d'observation :

- observation compacte (analyse de signature) (HIT81).
- comparaison directe (VEL82).

1 - 5. MISE EN OEUVRE DU TEST

La mise en oeuvre du test se caractérise par :

- la production des vecteurs de test.
- le système de test.
- la production des résultats correspondant à la machine juste.

1.5.1 Production des vecteurs de test

On dénombre trois principaux moyens :

- des générateurs pseudo-aléatoires de vecteurs.
- des vecteurs de test prédéterminés stockés dans une mémoire de masse.
- des générateurs algorithmiques permettant de générer au cours du test les vecteurs de test prédéterminés, éliminant un stockage trop onéreux en espace mémoire.

1.5.2 Production des résultats de la machine juste

Ces résultats sont :

- prédéterminés ; ils sont fournis par un simulateur ou par le générateur de vecteurs de test.
- calculés référence.

1.5.3 Le système de test

Le test peut etre effectué dans trois conditions différentes :

- le circuit est isolé: les vecteurs de test lui sont envoyés directement par le système de test.
- le circuit est en site réel: les vecteurs de test transitent alors par l'environnement du circuits.
- le circuit est en site artificiel ou en site reproduisant l'environnement réel du circuit.

I - 6. DEFINITIONS

Dans ce paragraphe nous définissons les notions de défaut, de panne et d'erreur ainsi que les hypothèses de mauvais fonctionnement correspondant.

I.6.1 Généralités

- défaut : c'est une imperfection physique soit d'origine, soit accentuée au cours du fonctionnement. On appelle ce type de phénomène une défaillance.
- panne : c'est la manifestation au niveau logique d'un défaut.
- erreur : c'est la manifestation au niveau fonctionnel d'une panne.

Les liens existant entre les notions de défaut, de panne et d'erreur définissent suivant le niveau de description du circuit les hypothèses de mauvais fonctionnement en vue du test.

I.6.2 Modèle de défaillance (ARY82)

Ce modèle s'applique au niveau transistors. De nombreux modèles ont été développés en fonction des technologies utilisées.

I.6.3 Hypothèse de pannes

On appelle panne la manifestation logique de la défaillance physique (EDW80)(collages, perçage...). Ces hypothèses de pannes peuvent être prises en compte tant que le nombre de portes logiques (ou de transistors) reste faible dans l'absolu, et rapporté au nombre de broches. Il est à noter également que cette modélisation logique pose des problèmes pour certaines technologies. Pour les circuits à haute intégration LSI et VLSI les hypothèses de pannes classiques sont inadaptées. (ARY82)(NIC80). De nouveaux modèles sont actuellement proposés pour les technologies MOS. (AKE83) (CHI83) (EDW80) (JAI83) (WAD78)

I.6.4 Hypothèse d'erreurs (tableau III)

On appelle erreur la manifestation fonctionnelle d'une panne, par exemple la conséquence d'une panne sur le déroulement des fonctions du circuit. Il est très difficile de rechercher l'effet des pannes sur les différentes fonctions et de définir des hypothèses d'erreurs correspondant aux pannes. Certains auteurs ont défini, a priori, un ensemble d'hypothèses d'erreurs pour les circuits du type microprocesseurs (THA79). Il est alors difficile d'évaluer la couverture des tests correspondants.

I.6.5 Test et taux de couverture (CAR82)

Pour les circuits peu complexes, l'utilisation de méthodes de génération de vecteurs de test basées sur des hypothèses de panne simples (D-algorithme détectant des collages et des court-circuits sur des schéma logiques) permet d'évaluer un taux de couverture. L'utilisation de méthodes de test fonctionnel par identification ou couverture des fonctions d'un circuit ne permet plus une telle évaluation. On ne peut parler que du taux de couverture par rapport aux spécifications fonctionnelles.

DESCRIPTION	HYPOTHESES	
SCHEMA ELECTRIQUE MASQUES	PANNES DEFAILLANCES	DEFAILLANCES DANS LES TRANSISTORS (AKE83)(ELZ80)(EDW80) (CHI83)(MAL82)
SCHEMA EN PORTES	PANNES	COLLAGES D-ALGORITHME (ROT80) DIFFERENCE BOOLEENNE
FONCTIONNELLE ARCHITECTURALE	ERREUR	HYPOTHESE D'ERREURS FONCTIONNELLES (THA78) (FOU81)(ROB80-1)(SAU81-1))

TABLEAU III

II - LES CIRCUITS INTEGRES (ELE80)

La classe des circuits complexes considérée dans cette étude est composée des microprocesseurs et des circuits périphériques. Nous noterons pour ces circuits :

- l'augmentation de leur échelle d'intégration.
- leur accessibilité réduite (nombre de broches).

L'évolution du test se fera vers des méthodes fonctionnelles utilisant le jeu d'instructions. Nous montrerons que ce test doit être complété par un test des broches non activées par le jeu d'instructions.

II - 1. COMPLEXITE ET DENSITE

II.1.1 Complexité des circuits intégrés TABLEAU IV

On définit la complexité des circuits intégrés en évaluant le nombre de portes logiques équivalent assurant une certaine fonctionnalité.

DENSITE	E.P.L.	TYPE DE CIRCUIT
SSI	<12	PORTES LOGIQUES DE BASE PETITS CIRCUITS COMBINATOIRES
MSI	<100	UAL BASCULES MULTIPLEXEURS
LSI	<1000	MEMOIRES CIRCUITS ARITHMETIQUES CIRCUITS PERIPHERIQUES
VLSI	<10000	MICROPROCESSEURS MEMOIRES COPROCESSEUR MICROCONTROLEURS CIRCUITS PERIPHERIQUES
ULSI	>10000	MICROPROCESSEURS MEMOIRES CIRCUITS SPECIALISES COMPLEXES

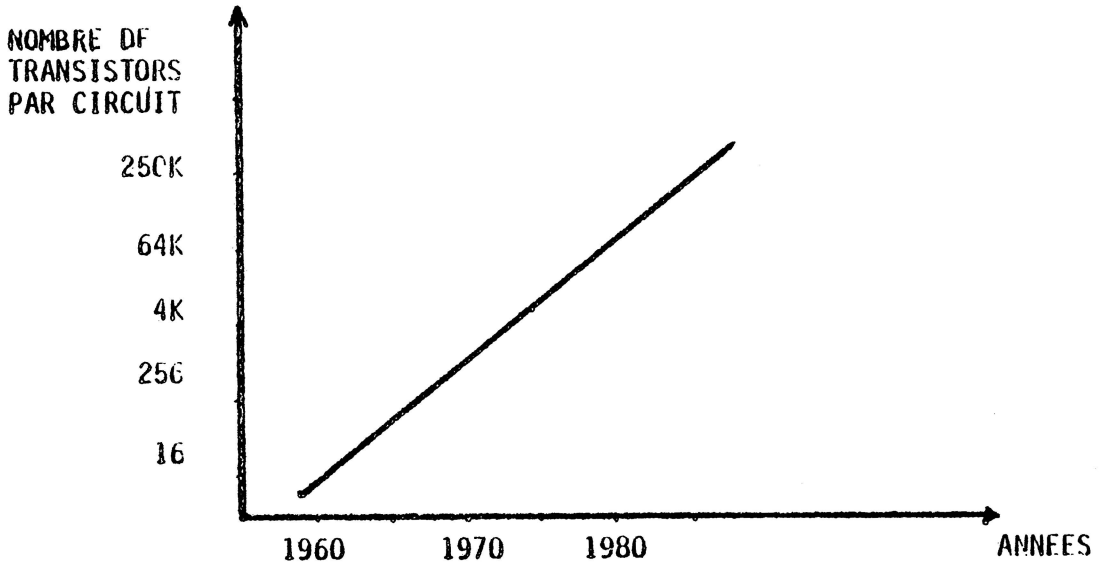
TABLEAU IV

L'augmentation de la complexité des circuits implique un accroissement de la complexité des fonctions réalisées, donc des opérateurs réalisant ces fonctions. Le test classique de ces opérateurs devient très difficile, sinon impossible.

II.1.2 Densité des circuits intégrés

Plusieurs facteurs contribuent à l'accroissement du nombre de transistors par circuits et de la densité d'intégration :

- l'augmentation de la complexité.
- l'évolution de la surface des circuits.
- la réduction continuelle des géométries des transistors.



EVOLUTION DU NOMBRE DE TRANSISTORS
PAR CIRCUIT

FIG 1

EVOLUTION DE LA SURFACE DES TRANSISTORS

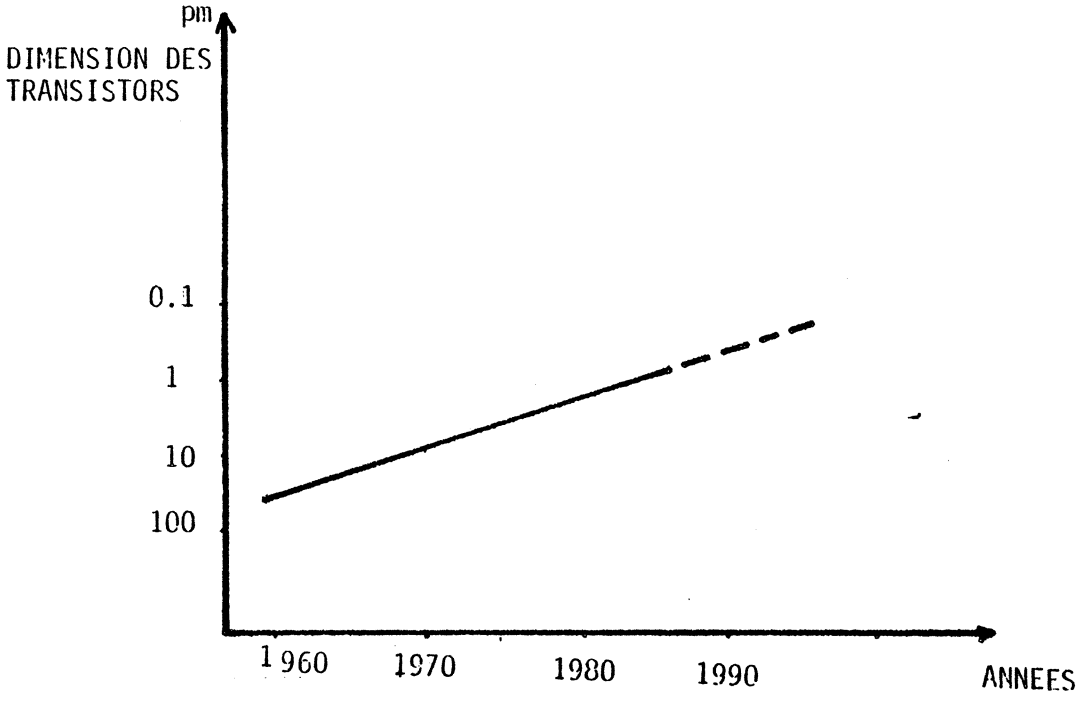
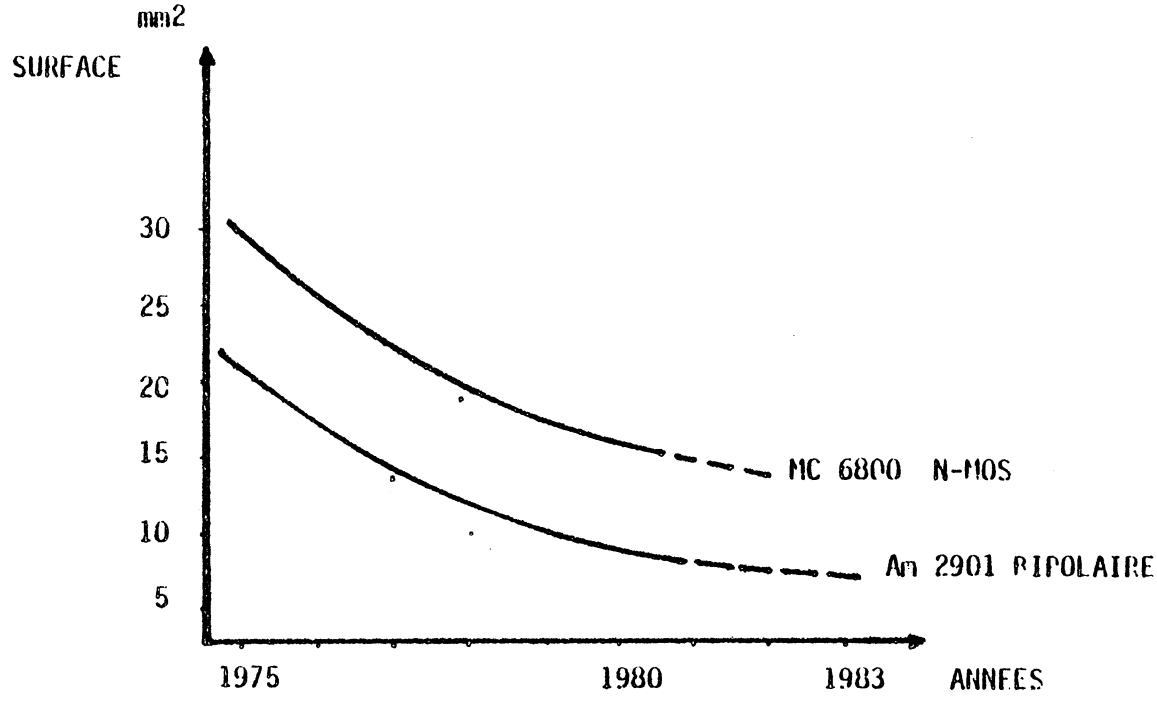
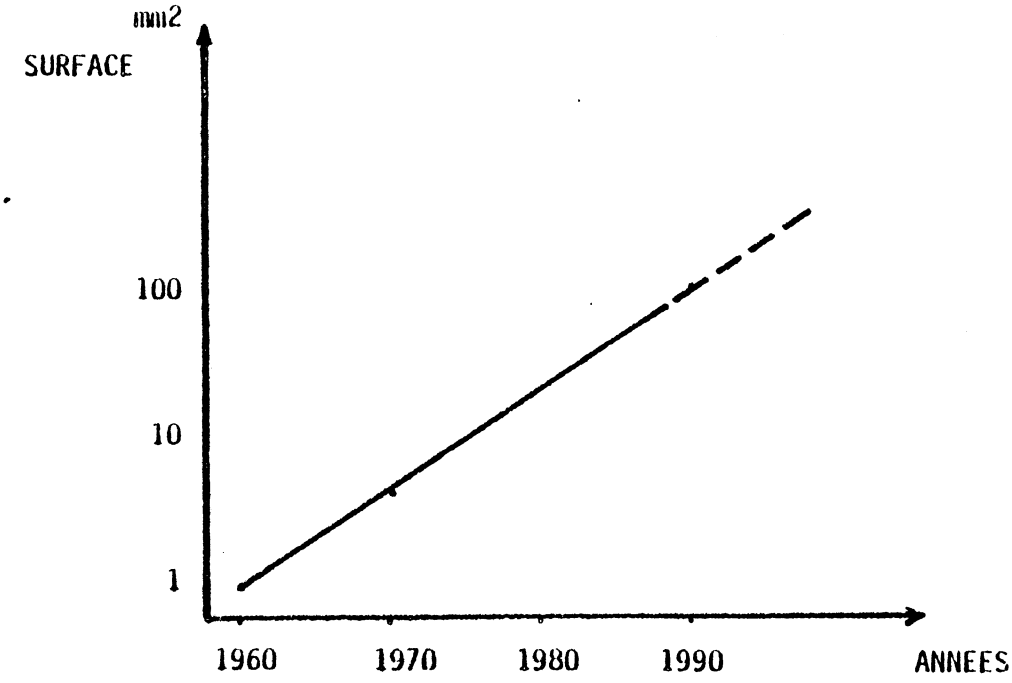


FIG 2



REDUCTION DES GEOMETRIES DES CIRCUITS
MC 6800 ET Am 2901



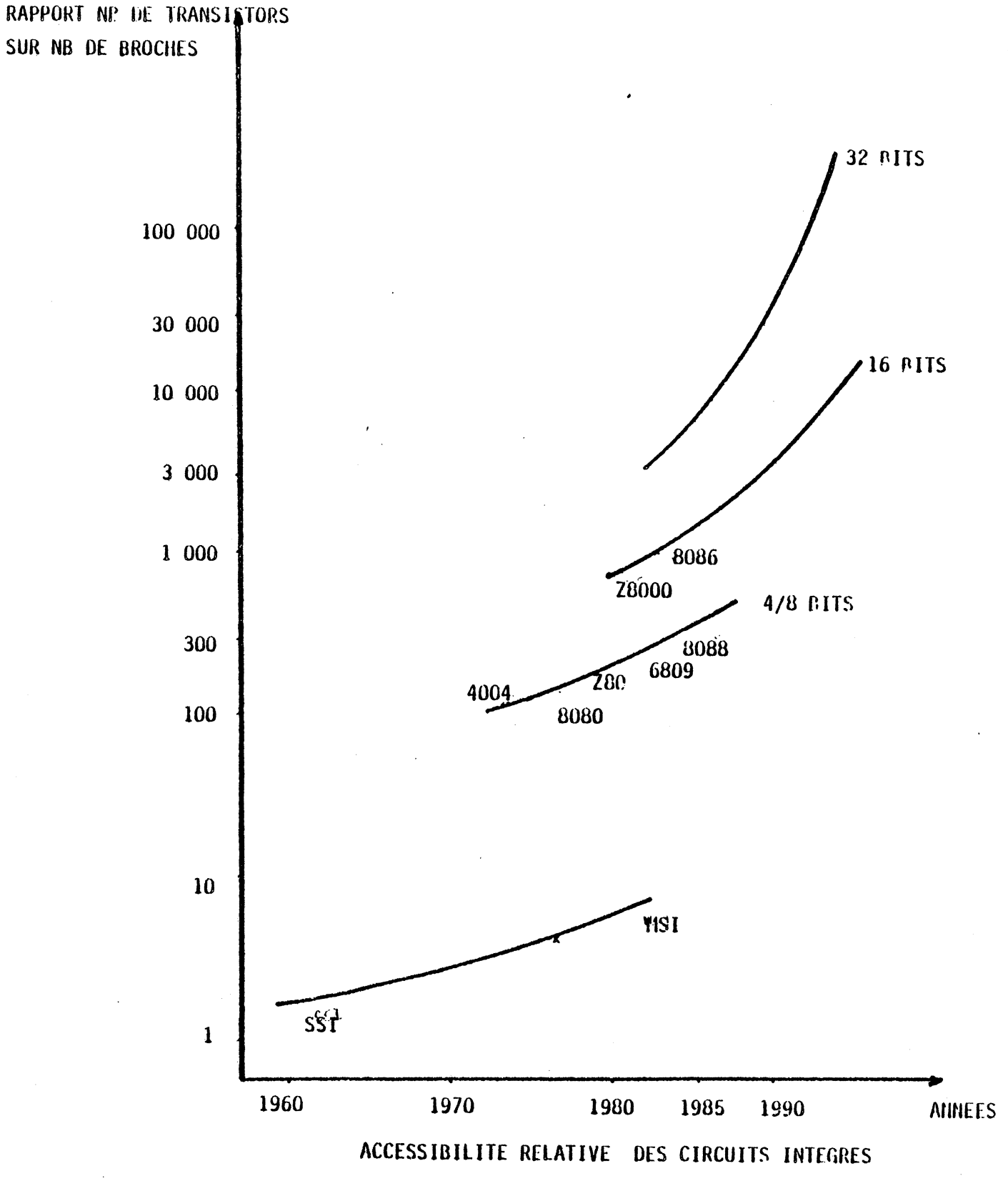
EVOLUTION DE LA SURFACE DES CIRCUITS

FIG 4

Ces courbes montrent l'augmentation continue du nombre de transistors intégrés dans les circuits.

II.1.3 Moyens d'accès aux circuits intégrés

Pour les circuits logiques SSI et MSI, le rapport nombre de transistors sur nombre de broches restait relativement faible, en général inférieur à dix. Pour les circuits de haute complexité ce rapport devient très élevé: de cent à plusieurs milliers. Les circuits intégrés devenant de plus en plus complexes et les moyens d'accès étant en baisse relative, le problème du test devient très aigu.



ACCESSIBILITE RELATIVE DES CIRCUITS INTEGRES

FIGURE 5

II - 2. LES GENERATIONS DE MICROPROCESSEURS

Les circuits de type microprocesseurs peuvent être divisés en plusieurs catégories :

- les microprocesseurs en tranches.
- les microprocesseurs à usage général.
- les microordinateurs.
- les coprocesseurs.

II.2.1 Les microprocesseurs en tranches (GER79)(LIT80)(MIN79)

La machine formée par l'assemblage des différents circuits (UAL, séquenceurs, controleur d'adresse et d'interruption...) peut encore être testée par des méthodes inspirées de celles appliquées aux circuits MSI. Ceci est rendu possible par la connaissance structurelle de la machine.

II.2.2 Les microprocesseurs à usage général (LUC79)(ESI79)

Leur développement a commencé avec l'introduction des 6800 MOTOROLA et 8080 INTEL. Pour leur test on ne disposait pas de nouvelles méthodes destinées à être appliquées à ces circuits. Des essais de test structurel, de test après partitionnement du circuit ont été fait (AKE77), mais dans la plupart des cas ils se sont heurtés à des problèmes de connaissances fines du circuit. Ces problèmes se sont accrus avec l'évolution de ces circuits et l'apparition de circuits de plus en plus complexes comme la famille des 68000 MOTOROLA, les iAPX 80X86 et NS16000. Ceci amène à considérer des méthodes de test fonctionnel.

II.2.3 Les microordinateurs monopuces

Ces circuits intègrent sur une même puce toutes les fonctions nécessaires à la constitution d'un microordinateur. Le dialogue avec l'extérieur se fait par des ports d'échange, ce qui pose d'importants problèmes d'accessibilité au circuit.

II.2.4 Les coprocesseurs

On peut considérer ces circuits comme des macro-fonctions ; en effet les données leur sont fournies par un processeur maître ; après traitement elles sont restituées au processeur maître. Il est quasiment impossible d'accéder aux éléments internes de ces circuits.

II - 3. FONCTIONNEMENT ET TYPES DE BROCHE DES CIRCUITS

Un microprocesseur dispose pour l'exécution des programmes des bus adresses et données et de quelques signaux de contrôle permettant de se synchroniser avec la mémoire ; les dialogues avec la périphérie s'établissent via les bus données et adresses, les signaux de contrôle et des signaux interruptifs.

Lors d'un test on s'intéressera non seulement à l'exécution des programmes pour les microprocesseurs ou aux manipulations de données pour les circuits périphériques mais aussi aux autres signaux permettant des fonctionnements annexes. On remarque que le test d'un microprocesseur limité à son jeu d'instructions ne permet pas l'activation de toutes les broches.

Exemple : Pour le NSC800 (NSC81) le test du jeu d'instructions active 28 broches sur les 40, soit 70% des broches. Le test ne pourra se faire uniquement à travers le jeu d'instructions.

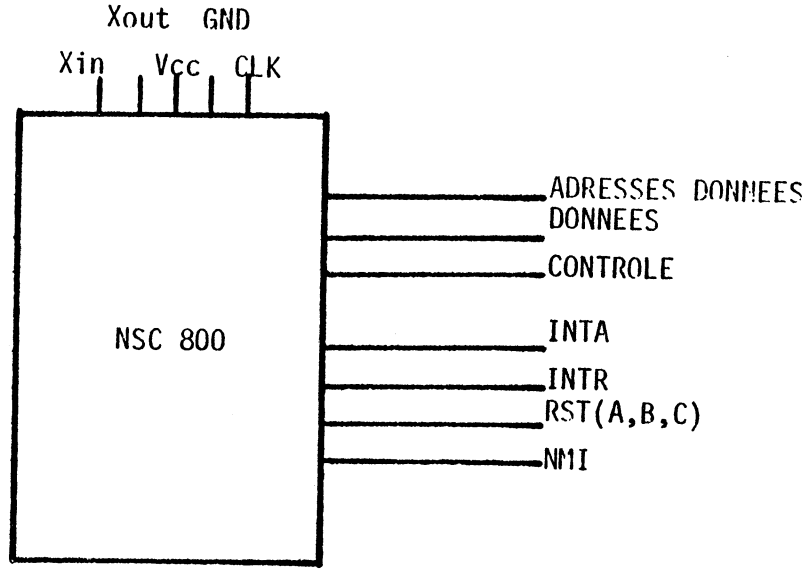

Xout GND
Xin Vcc CLK
NSC 800
ADRESSES DONNEES
DONNEES
CONTROLE
INTA
INTR
RST(A,B,C)
NMI

FIG 6

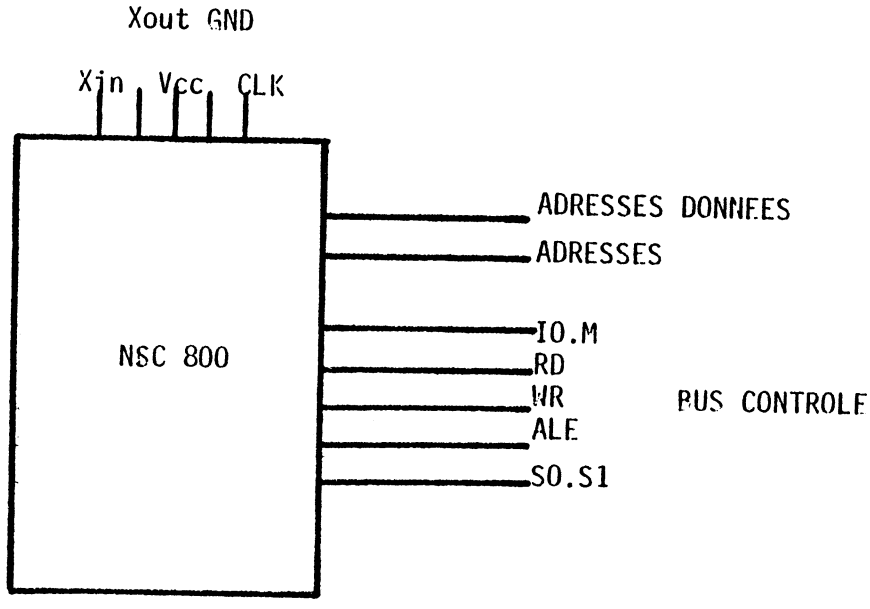

Xout GND
Xin Vcc CLK
NSC 800
ADRESSES DONNEES
ADRESSES
IO.M
RD
WR
ALE
SO.S1
BUS CONTROLE

FIG 7

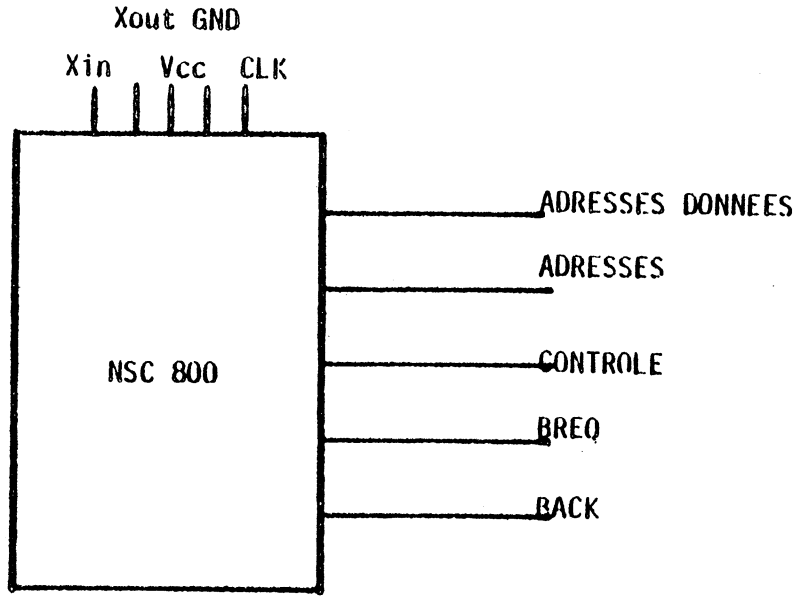


FIG 8

L'étude de ces fonctionnements montre que les bus adresses et données et le bus contrôle (IO.M, RD, WR, ALE, S0, S1) sont toujours utilisés. Par contre certains signaux sont spécifiques de fonctionnements asynchrones comme les interruptions et les demandes de bus. On répartit les broches du circuit en trois catégories :

- les signaux de fonctionnement minimum (horloges, alimentations).
- les bus.
- les signaux asynchrones.

TABLEAU V

EXMPLE D'UN MICROPROCESSEUR

-FONCTIONNEMENT MINIMUM

ALIMENTATIONS	Vcc	40		3V U 12V
	Vss	20		Masse
HORLOGES	Xout	10	E	
	Xin	11	E	Connexions oscillateur f.in
	CLK	09	S	Horloge f=0.5 f.in

-BUS

DONNEES	AD0-AD7	12-19	ES	Données et adresses multiplexées
ADRESSES	AD0-AD7	12-19	ES	Données et adresses multiplexées
	A8-A15	1-8	S	Poids forts adresses
CONTROLE	RD	32	S	Lecture
	WR	31	S	Ecriture
	ALE	30	S	Validation adresses poids faibles
	M.IO	34	S	Opération mémoire ou entrée sortie
	S0.S1	29,27	S	Signaux d'état
	RFSH	28	S	Rafraîchissement mémoire
	RESETout	37	S	Signal d'état

-SIGNAUX EXTERNES

BREQ	36	E	Demande du bus, DMA
BACK	35	S	Acquittement
INTR	25	E	Demande d'interruption
INTA	26	S	Acquittement
RSTA,B,C	22-24	E	Interruption
NMI	21	E	Interruption non masquable
PS	39	E	Mise en état de rétention
WAIT	38	E	Synchronisation externe
RESETin	33	E	Réinitialisation

II - 4. LES CIRCUITS PERIPHERIQUES (EHD81)(ELE80)

Ces circuits sont conçus pour fonctionner intégrés à des systèmes basés sur une même famille de circuit. La multiplicité des types de ces circuits est illustrée ci-dessous en proposant un état de la production au début des années 1980.

- processeurs arithmétiques rapides	7 types
- circuits gestion mémoire	17 types
- contrôleur DMA	9 types
- entrée sortie série , contrôle de protocole	59 types
- entrée sortie parallèle	37 types
- contrôleur de disque	23 types
- interface de bus	7 types
- décripteur	7 types
- timer programmable	32 types
- divers	45 types

On regroupe par exemple dans la même rubrique intitulée circuits gestion mémoire tous les circuits dont la fonction principale est la gestion mémoire ; il y en a dix-sept types différents.

Les circuits périphériques peuvent être regroupés suivant quatre catégories:

- les circuits réalisant une fonction non identifiable par le processeur maître (amplificateur de bus...).
- les circuits monofonctions ou ne comprenant qu'une seule fonction principale. Ces circuits ont des fonctions identifiables par le processeur maître.
- les circuits multifonctions.
- les coprocesseurs (on identifie ces circuits aux microprocesseurs).

TABLEAU VI

EXEMPLE D'UN CIRCUIT PERIPHERIQUE INTEL 8255A (INT82)

-FONCTIONNEMENT MINIMUM

ALIMENTATIONS	Vcc	26		U=5Volts
	GND	7		Masse
-BUS				
DONNEES	DO-D7	34-27	ES	
ECHANGES	PAO-PA7	4-1 40-37	ES	Port A
	PBO-PB7	18-25	ES	Port B
	PCO-PC7	14-17 13-10	ES	Port C
CONTROLE	RD	5	E	Lecture
	CS	6	E	Sélection de boîtier
	AO-A1	9-8	E	Sélection du port d'échange
	WR	36	E	Ecriture
-SIGNAUX ASYNCHRONES				
	RESET	35	E	Réinitialisation

III - SPECIFICATIONS DU PROJET G.A.P.T. (*)(BEL82-1)

Le projet global dans lequel s'intégre le travail qui va suivre s'articule autour de trois pôles principaux.

- l'élaboration d'une méthode de test destinée aux utilisateurs de microprocesseurs.(BEL82-3)(BEL82-4)(VEL82)(SAU81-1).
- la conception d'un générateur automatique de programmes de test (BEL82-1)(BEL82-2)(BEL83-1).
- la réalisation d'un système de test dont le développement est étroitement lié aux deux points précédents. (EMD81)(ESD82)(ESD83)

III - 1. METHODE DE TEST USAGER

La méthode envisagée doit être adaptable à tous les microprocesseurs et aux circuits périphériques. Ce test s'adressant aux usagers de circuits VLSI, le niveau de connaissances du circuit se limite aux informations contenues dans le manuel technique.

(*) G.A.P.T.=GENERATEUR AUTOMATIQUE DE PROGRAMMES DE TEST

III - 2. LE GENERATEUR AUTOMATIQUE DE PROGRAMMES DE TEST

La méthode de test et le langage de description des circuits ont été développés avec le souci d'automatiser la génération des programmes de test. Il a été donc développé un outil de génération automatique de programmes de test. (BEL82-2)(EMD81)

III - 3. LE SYSTEME DE TEST

Dans le but de valider toutes les hypothèses qui ont été faites et de dérouler les programmes de test, nous avons conçu en collaboration avec ELECTRONIQUE SERGE DASSAULT un système de test (ESD82).

CHAPITRE 2

INTRODUCTION

La génération de vecteurs de test est effectuée par rapport à une description du circuit à tester. Pour une description structurelle, une méthode de test analytique caractérisée par les hypothèses de pannes sur le schéma structurel peut être utilisée. Le problème consiste à trouver un ensemble de vecteurs de test détectant avec un taux de couverture spécifié, les pannes de l'ensemble donné.

Ces méthodes ont été largement utilisées pour les circuits SSI et MSI, et pour les cartes construites à partir de ces circuits.

Dans le cas des circuits VLSI complexes, ces méthodes sont limitées par :

- le manque de connaissances sur la structure interne du circuit dans certains cas.
- l'inadéquation des hypothèses de pannes (collage à 1 et à 0, au niveau des portes logiques) (ARY82)(NIC80).
- la complexité des circuits rendant le temps de génération prohibitif.

Les descriptions fonctionnelles caractérisent le circuit par l'ensemble des fonctions qu'il réalise. Plusieurs méthodes de test fonctionnel de microprocesseurs ont été proposées en (COU81)(THA78)(BRE80)(THA80). Toutes ces méthodes ont une caractéristique commune : l'utilisation d'hypothèses d'erreurs. Ces erreurs sont du type: une instruction est exécutée à la place d'une autre, un registre est sélectionné à la place d'un autre...

Ces hypothèses d'erreurs sont discutables: une panne effective ne se manifeste que rarement par une erreur de type fonctionnel, c'est à dire que l'on peut formuler en termes de fonction. Une couverture à 100% de telles erreurs n'assure pas forcément une bonne qualité de test.

Nous proposons ici une méthode de test basée sur une description fonctionnelle du circuit, ne s'appuyant pas sur des hypothèses d'erreurs. La méthode de test proposée est en fait une méthode d'identification qui cherche à vérifier la bonne exécution par le circuit des fonctions spécifiées (par opposition aux méthodes de test basées sur des hypothèses d'erreurs sont dites

méthodes de test par distinction). Notons au passage que la notion de couverture de pannes doit être remplacée par la notion de couverture des différents modes de fonctionnement. La méthode de test proposée nécessite une description du fonctionnement du circuit qui soit à la fois :

- compacte étant donnée la complexité du fonctionnement à vérifier.
- complète pour pouvoir déterminer les fonctionnements à vérifier.

Nous avons défini dans ce but la notion de machine séquentielle comportementale MSC qui sert de description de référence et montrons sa relation habituelle avec les descriptions des manuels utilisateur. Nous choisirons ensuite un ensemble de transitions représentatives en vue du test.

I - MACHINE SEQUENTIELLE COMPORTEMENTALE

Les microprocesseurs et la plupart des circuits à haute intégration, comme les coprocesseurs, les circuits périphériques peuvent être décrits comme des machines séquentielles (BEL82-3). Une telle description définit complètement le fonctionnement du circuit ; son établissement et son utilisation sont par contre très complexes que ce soit pour servir en simulation ou en test de production. Nous proposerons ici un modèle de référence formel dit machine séquentielle comportementale et nous comparerons les descriptions usuelles (manuel usager) à ce modèle de référence.

I - 1. RAPPELS SUR LA DESCRIPTION DES AUTOMATES (SIF74)

Soit $A = (E,S,Q,\delta,\lambda)$

Ce quintuplet définit un automate A par les ensembles d'entrée E, de sortie S, d'états Q, par la fonction état suivant δ et la fonction de sortie λ.

I.1.1 Les entrées et les sorties E et S

Les entrées et les sorties sont les broches du microprocesseur. L'ensemble des valeurs des entrées de E et des sorties de S comprend toutes les valeurs que peuvent prendre ces broches en entrée ou en sortie.

1.1.2 Les états Q

On définit l'ensemble des états Q à partir de toutes les valeurs des éléments actifs internes de mémorisation. Cet ensemble comporte un grand nombre d'éléments, dont une partie, qui peut contenir des renseignements primordiaux sur l'état du circuit, est inconnue de l'utilisateur.

1.1.3 La fonction d'état suivant

C'est une application de l'ensemble Q x E sur Q.

$$\delta : Q \times E \longrightarrow Q$$

La fonction δ donnant l'état suivant est très complexe et totalement inconnue lorsque l'on ne dispose pas précisement de la structure interne du circuit.

1.1.4 La fonction de sortie

C'est une application de l'ensemble Q x E sur S.

$$\lambda : Q \times E \longrightarrow S$$

Cette fonction peut etre obtenue à partir d'une description du circuit, mais elle demeure extremement complexe. Etant donné le nombre impressionnant d'états que peut comporter un circuit VLSI et la complexité des différentes fonctions d'état suivant δ et de sortie λ , il est intéressant d'avoir une vision beaucoup plus globale du fonctionnement du circuit: c'est ce que l'on appellera la MACHINE SEQUENTIELLE COMPORTEMENTALE.

I - 2. MACHINE SEQUENTIELLE COMPORTEMENTALE

Soit $\mathcal{A}$ cette machine définie par le quintuplet suivant de la même manière qu'un automate.

$$\mathcal{A} = (\mathcal{E}, \mathcal{Y}, \mathcal{Q}, \Delta, \Lambda)$$

I.2.1 Ensemble des éléments de mémoire comportementaux

DEFINITION : C'est l'ensemble des éléments internes de mémorisation mentionnés dans le manuel technique. Cet ensemble réunit la plupart du temps les registres d'usage général, les registres d'adresses, le compteur programme, le registre d'état et éventuellement quelques bascules supplémentaires d'indicateurs.

I.2.2 Etat comportemental d'un circuit

DEFINITION : Cet état est défini par les valeurs des éléments de mémoire comportementaux ; soit Q l'ensemble des états comportementaux, ces états ne sont pas toujours définis, en particulier si pendant l'exécution d'une instruction ou la prise en compte de signaux asynchrones les éléments de mémoire comportementaux prennent des états inconnus de l'utilisateur. De ce fait on s'intéressera à l'état comportemental d'un circuit pour les VALEURS FINALES résultant de l'exécution d'une instruction et/ou d'un traitement de signal. L'état d'une machine comportementale est donc défini à des instants successifs macroscopiques par rapport à la machine séquentielle réelle.

I.2.3 Les entrées et les sorties de la machine séquentielle

DEFINITION : Une entrée $\mathcal{E}$ est une suite de configurations envoyées sur les broches d'entrée du circuit permettant le passage d'un état comportemental à un autre état comportemental.

Fonctionnellement ces configurations sont de deux types :

- des codes qui, après traitement par le circuit, sont interprétés comme des codes opération d'instruction ou des opérandes.
- des signaux permettant la synchronisation et le dialogue avec l'environnement.

On compose ces deux types pour obtenir une configuration d'entrée (codes et signaux). Les signaux sont actifs ou inactifs suivant les nécessités du fonctionnement.

EXEMPLE :

* Instruction d'addition

Soit la description au niveau transfert de registre de l'instruction :

ADD R1 , (ADR) R1+(ADR)----->R1

On ne tient pas compte des échanges sur le bus contrôle dans la description de l'instruction.

1er CAS: l'instruction se déroule sans rupture de séquence. L'entrée $\mathcal{E}$ est composée de deux configurations binaires: l'une décodée donne l'adresse d'un opérande l'autre le code opération de l'instruction.

2ème CAS: il y a rupture de séquence pendant le déroulement de l'instruction par une interruption. L'entrée "demande d'interruption" est active dans l'entrée $\mathcal{E}'$ alors que dans $\mathcal{E}$ elle était inactive.

DEFINITION : Une transition de la machine séquentielle comportementale est définie par un couple (état comportemental q,entrée e) avec $q \in Q$ et $e \in \mathcal{E}$.

DEFINITION : Une sortie $s \in \mathcal{S}$ de la machine séquentielle comportementale est définie par la séquence de valeurs prises par les broches de sortie d'un circuit pendant une transition de la machine séquentielle comportementale.

1.2.4 Les fonctions de la machine séquentielle comportementale

DEFINITION : Δ la fonction d'état suivant est définie par une application de $\mathcal{Q} \times \mathcal{E}$ -----> $\mathcal{Q}$

DEFINITION : Λ la fonction de sortie est définie par une

une application de $\mathcal{Q} \times \mathcal{E}$ -----> $\mathcal{Y}$

En résumé la machine séquentielle comportementale est définie comme une machine séquentielle classique.

$\mathcal{Q}$: ensemble des états comportementaux.

$\mathcal{E}$: ensemble des entrées. Une entrée est une séquence de valeurs en entrée permettant le passage d'un état comportemental à un autre état comportemental.

$\mathcal{Y}$: ensemble des sorties. Une sortie est une séquence de valeurs en sortie accompagnant le passage d'un état comportemental à un autre état comportemental.

Δ : fonction état suivant

Λ : fonction de sortie

I - 3. DESCRIPTION PRATIQUE DES FONCTIONS D'UN MICROPROCESSEUR

Transitions et instructions

Nous avons vu qu'une description de référence rigoureuse d'un microprocesseur est donnée par la liste des transitions.

état comportemental initial x entrée ⟶ état compotemental final

⟶ sortie

Il est clair qu'une telle description est purement formelle et doit servir de référence abstraite pour évaluer la complétude d'un test pratique. Dans un manuel utilisateur, une description très compactée par rapport à la définition de référence est donnée par instruction ou par famille d'instructions. Une instruction spécifie les parties de l'état initial utilisées ou affectées par une instruction: ce sont les registres puits et sources de l'instruction (VEL82). Les entrées précisent les données, l'opération à effectuer et dans certains cas les prédicats de condition associés à l'exécution de l'instruction. On retrouve dans une telle description les primitives de transfert de registres et les expressions de condition des langages transfert de registre habituels. Une instruction simple modélise un grand nombre de transitions du modèle formel.

EXEMPLE : Soit l'instruction d'addition des deux registres D1 et D2

ADD D2 , D1

Cette instruction ajoute le contenu de D1 et de D2 et le range dans D1; l'instruction n'est pas conditionnelle.

(D1)+(D2) ------> D1

A partir d'un état bien défini du microprocesseur on obtient le nombre de transitions Nt suivant dans le modèle de la machine séquentielle comportementale :

- pour des opérandes de 8 bits Nt=2**16
- pour des opérandes de 16 bits Nt=2**32
- pour des opérandes de 32 bits Nt=2**64

Dans le manuel technique d'utililisation, les instructions sont regroupées par famille. Ces familles d'instructions sont développées par adjonction des modes d'adressage, des types de conditions, des registres affectés, de la taille des opérandes...

EXEMPLE : Instruction d'addition du MOTOROLA 68000 (MOT81)

ADD <ea> to Dn (Dn)+(<ea>) -----> Dn

ADD Dn to <ea> (<ea>)+(Dn) -----> <ea>

Dn est un représentant d'un ensemble de huit registres D0 à D7. <ea> représente l'adresse réelle de l'opérande en mémoire.

Le nombre de modes d'adressage permis dépend de la taille de l'opérande mémoire et du sens de l'instruction.

ADD <ea> to Dn

- pour un double mot	32 bits	12 modes sont permis
- pour un mot	16 bits	12 modes sont permis
- pour un octet	8 bits	11 modes sont permis

ADD Dn to <ea>

- pour tout type d'opérande	9 modes sont permis

Dans certains modes d'adressage interviennent des ensembles de registres autres que ceux désignés dans l'instruction (ex : les registres adresses A0 à A7), des données relatives à des déplacements en mémoire ou les tailles des opérandes. L'instruction d'addition n'est pas conditionnelle, son déroulement est donc linéaire.

Il existe environ 8000 instructions différentes constituant la famille "addition binaire".

Les descriptions des instructions sont compactées au niveau de leur représentation mnémonique. (ZIL85)

EXEMPLES :

- Transfert d'un mot mémoire et/ou d'un registre vers un mot mémoire et/ou un registre.

LD r , r' (r)------>r'

r et r' représentent des ensembles de registres.

r = r' = A,B,C,D,E,H,L

On décrit ainsi une famille d'instructions : voir exemple précédent.

LD A,B (A)------>B
LD A,C (A)------>C
⋮ ⋮
LD L,H (L)------>H

- Opération sur un mot mémoire ou un registre.

ADD A , r (A)+(r)------>A

r = A,B,C,D,E,H,L

Les instructions ainsi décrites sont :

ADD A , A (A)+(A)------>A
⋮ ⋮
ADD A , H (A)+(H)------>A

- Instruction conditionnelle

JP cc , nn

cc représente les conditions.
nn est une adresse sur 16 bits.

L'instruction représente deux exécutions distinctes :

* si cc est vraie alors nn------>PC
* si cc est fausse alors (PC)+1-->PC

- Instruction répétitive

INIR

L'exécution de l'instruction est répétée jusqu'à ce que (B)=0 ; le corps de l'instruction est :

(C) ------> (HL)
B-1 ------> B
HL-1------> HL

II - METHODE DE TEST

On peut distinguer deux approches de test (VEL82) :

- Les méthodes de test " par distinction "
- Les méthodes de test " par identification "

Les méthodes de test par distinction sont caractérisées par :

- des hypothèses d'erreurs sur un modèle du circuit à tester ; ce modèle peut être de n'importe quel niveau.
- la recherche de séquences de test permettant de distinguer le comportement de la machine juste de celui de l'ensemble des machines fausses, défini à partir des hypothèses d'erreurs.

Les méthodes de test par identification visent à vérifier le fonctionnement correct du circuit; c'est à dire à identifier le circuit sous test à ses spécifications. Une telle méthode ne repose pas sur des hypothèses d'erreurs.

Dans le cas du test de circuits complexes dont la structure est inconnue, on ne dispose que d'une description comportementale ; les hypothèses d'erreurs

qui peuvent être établies sur ce modèle ne correspondent pas à des pannes réelles : Une METHODE DE TEST PAR IDENTIFICATION S'IMPOSE !

Après un rappel des méthodes de test d'automates par identification, une extension aux machines séquentielles comportementales est proposée.

II - 1. TEST D'AUTOMATE PAR IDENTIFICATION

Une méthode de test d'automates par identification a été proposée initialement en (KOH70). Le modèle utilisé est le tableau d'états de l'automate; le but du test est de parcourir toutes les transitions du tableau d'états.

EXEMPLE : Soit un automate à trois états et à deux entrées.

x y / ETAT	0 0	0 1	1 1	1 0
a	a	a	b	b
b	c	a	b	a
c	c	b	b	c

TABLEAU DES ETATS

TABLEAU 1

Le tableau 1 des états définit 12 transitions. Le test de cet automate consistera à les parcourir. Pour un tableau de n états et m entrées la séquence de test a une longueur supérieure à n*(2**m). En (KOH70), les problèmes que l'auteur cherche à résoudre sont :

- le positionnement de l'automate dans un état connu. On recherche une séquence de positionnement ou à défaut de synchronisation)
- le parcours des transitions, en vérifiant à chaque pas l'état atteint par observation des sorties. On recherche des séquences de distinction.

Ces problèmes sont difficiles à résoudre dans le cas étudié en (KOH70) ; il s'agit en effet d'automates :

- sans signal de type RESET.
- ayant peu d'entrées et peu de sorties pour un nombre important d'états, ce qui pose des problèmes de commandes et d'observation.

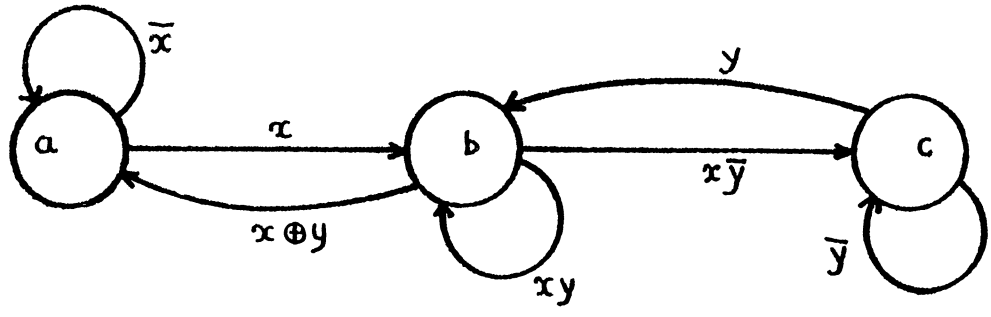
Dans le cas d'automates complexes, les problèmes sont différents : on dispose en général d'un signal RESET, et le nombre d'entrées et de sorties est important. En conséquence :

- d'une part le positionnement dans un état est facilité ; l'identification de l'état atteint peut être immédiate si les valeurs en sortie sont caractéristiques de l'état.
- d'autre part la complexité de l'automate interdit le parcours exhaustif des transitions.

Un automate complexe peut être décrit par un graphe de contrôle, avec les entrées sous forme compactée.

EXEMPLE : L'automate dont le tableau d'état est donné en TABLEAU-I peut être décrit par le graphe de la FIGURE 1.

FIGURE 1



L'arc a----->b correspond à deux transitions du tableau d'état : a*10 et a*11.

Le test par identification d'un automate complexe peut se ramener au parcours de tous les arcs du graphe. Pour chacun des arcs, on ne prend qu'une des configurations de valeurs des entrées permettant de parcourir cet arc. Le problème du test est alors ramené au problème du parcours de tous les arcs d'un graphe.

EXEMPLE : Pour le graphe de la FIGURE 1 , on suppose que le signal RESET positionne l'automate dans l'état a et que une configuration de valeurs des sorties différente est associée à chaque état, permettant d'identifier immédiatement un état.

Une séquence optimale pour le test est de longueur 7.

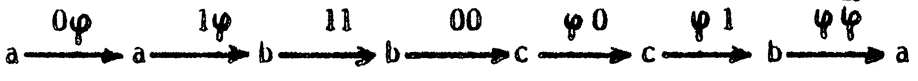
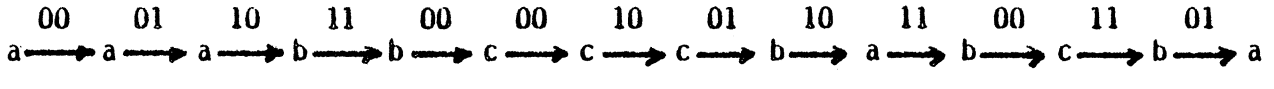


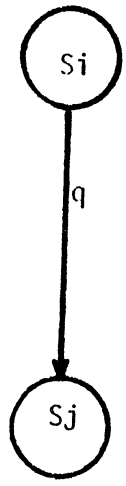
FIGURE 2

Dans le cas où l'on aurait cherché à parcourir toutes les transitions du tableau d'états, on aurait obtenu la séquence de longueur 13.

FIGURE 3



Dans le cas général, considérons un automate complexe à m entrées ; une transition du type suivant (voir FIGURE 4)



TRANSITION

GENERALE

FIGURE 4

représente 2**(m-1) transitions ; on testera cet ensemble de transitions à l'aide d'un seul représentant de l'ensemble.

II - 2. METHODE DE TEST PRATIQUE

II.2.1 Réduction de l'ensemble des transitions à tester

La méthode de test proposée s'inspire de la méthode de test d'automates complexes par couverture des arcs. Il s'agit ici de couvrir les transitions de la machine comportementale ; on s'affranchit en effet du test de séquence de transitions en supposant que ces transitions sont indépendantes. Le nombre important des entrées interdit bien entendu un test exhaustif. La réduction de ces domaines d'entrée s'effectue en deux temps.

- réduction du domaine des entrées servant à l'élaboration des primitives de contrôle suivant les remarques du paragraphe précédent.

- réduction du domaine d'entrée des opérateurs : Ce problème n'étant pas soluble au niveau fonctionnel, on suppose l'existance d'un test préalable appelé TEST DE BALAYAGE qui à partir de considérations structurelles envoie un ensemble de vecteurs de test aux différents opérateurs et garantit leur intégrité. Cet envoi se fait en employant les "plus simples" instructions activant les opérateurs.

a - ELABORATION DE CONDITION

Une condition exprimée par une composition de prédicats sera testée exhaustivement par rapport à ces prédicats et non par rapport aux données source de ce prédicat.

EXEMPLE : (MOT01)

- Instruction de branchement : La condition de branchement correspond au bit Z du mot d'état.

BNE ea ea : adresse de branchement

si Z=0 ea est chargée dans PC
si Z><0 PC est incrémenté

La partie significative de l'entrée définit DEUX tests.

Dans l'état initial on positionne - Z=0
- Z><0

- Instruction privilégiée : Certains microprocesseurs possèdent deux modes de fonctionnement ; en général SUPERVISEUR et UTILISATEUR. Une sous partie du jeu d'instructions n'est accessible qu'en mode superviseur ; cette sous partie forme l'ensemble des instructions privilégiées. L'indicateur S reflète le mode de fonctionnement imposé au circuit.

  si S=1 mode superviseur. Les instructions privilégiées sont exécutées.
  si S=0 mode utilisateur. L'exécution d'une instruction privilégiée entraîne une trappe logicielle.

Pour chaque instruction privilégiée on testera ces DEUX MODES.

- Instruction CHK: CHK ea

Cette instruction compare le contenu du registre D0 aux valeurs de deux bornes : l'une est égale à 0 et l'autre est modifiable et est contenue dans la position mémoire ea. Si le contenu de D0 ne dépasse pas les bornes fixées le programme continue en séquence ; sinon il y a rupture de séquence et exécution d'un programme correspondant. On définit trois situations de test :

si 0< (D0) <= (ea) le contenu de D0 est compris entre les bornes.
si (D0) <= 0
(D0) > (ea) le contenu de D0 dépasse les bornes fixées.

- Erreurs d'adressage : Certains microprocesseurs vérifient la position mémoire des mots de 16 bits et de 32 bits. L'adresse de ce type de mot doit être paire.

  adresse paire : l'exécution de l'instruction ou la prise en compte de la donnée se fait normalement.
  adresse impaire: il y a exécution d'une trappe logicielle consécutive à l'erreur d'adressage.

On définit ainsi DEUX configurations de test.

b - PRIMITIVES PLUS COMPLEXES

Pour les primitives plus complexes, une couverture identique à celle obtenue par des méthodes de test de logiciel (exécution symbolique) est recherchée.

EXEMPLE :

SI <prédicat> alors <.....> SINON <.....>

Si le prédicat est composé de n prédicats élémentaires alors 2**n classes d'équivalence sont définies et 2**n transitions peuvent être activées, Si un représentant de chaque classe peut être construit.

REPEAT <.....> UNTIL <prédicat>

Une activation permet de vérifier le CORPS de la boucle, la CAPACITE du bouclage et la SORTIE de la boucle. De la même manière si le prédicat est composé de n prédicats élémentaires alors AU PLUS 2**n activations sont nécessaires. Ce nombre peut cependant être réduit.

WHILE <prédicat> THEN <.....>

Au moins deux activations sont nécessaires: une pour vérifier la non-exécution de la boucle si le prédicat est faux en entrée, une autre pour vérifier le corps de la boucle, la boucle et la sortie de la boucle. On exécutera au moins deux fois le corps de la boucle. Si le prédicat est composite au moins 2**n activations sont nécessaires.

EXEMPLE : microprocesseur ZILOG Z80

Soit l'instruction CPIR exécutant comparaison, incrémentation et répétition jusqu'à.

Soit la représentation symbolique de l'exécution.

Corps de la boucle C

(A)-(HL)
(HL)+1 ---> HL
(BC)-1 ---> BC

REPEAT <C> UNTIL <P>

Prédicat P

(A)=(HL) ou
(BC)=0

Le prédicat P composite définit quatre cas. (A)><(HL) et (BC)><0 sont des valeurs initiales telles que la boucle s'exécute au moins DEUX fois.

c - SIGNAUX EXTERIEURS

Les signaux extérieurs apparaîssent en fait comme des conditions et sont traîtés comme dans -a-.

EXEMPLE : Exécution d'une instruction I et envoi du signal d'interruption.

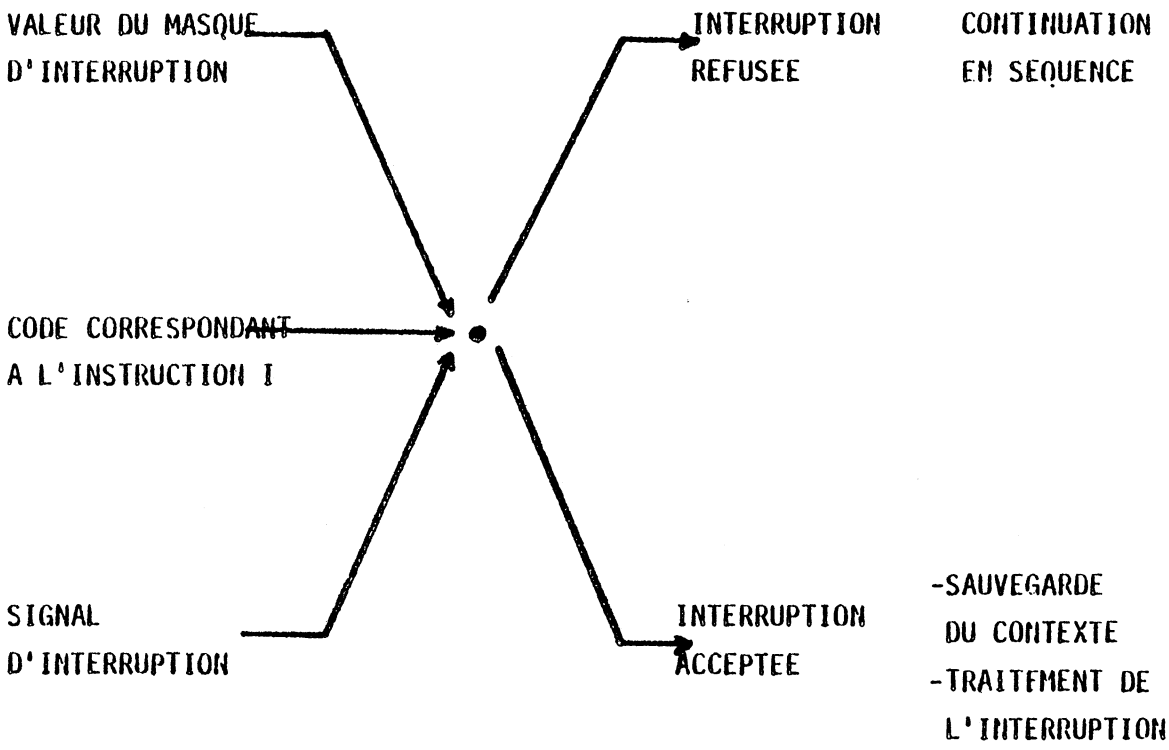


FIGURE 5

On définit ainsi DEUX configurations de test pour la même entrée suivant la valeur de la bascule d'autorisation des interruptions.

d - SEGMENTS COMMUNS A PLUSIEURS INSTRUCTIONS

Une autre simplification consiste à ne tester qu'une seule fois des segments communs à plusieurs instructions.

EXEMPLE : Un traitement d'interruption peut être considéré comme un segment commun à tout le jeu d'instructions. On peut restreindre le nombre d'activations de la manière suivante: on teste les n instructions du jeu d'instructions puis l'une des instructions avec envoi du signal d'interruption. On obtient ainsi n+1 transitions à tester au lieu de 2*n.

EXEMPLE : Le traitement d'erreur d'adressage du microprocesseur MC68000 peut être considéré comme un segment commun à toute instruction utilisant le même mode d'adressage dans le même but (une lecture opérande, une écriture opérande ou une recherche d'instruction (FETCH)).

II.2.2 Méthode de test (VEL82)

Le test se déroule en deux étapes: le test de conformité et le test de balayage

b - LE TEST DE BALAYAGE

Ce test garantit l'intégrité des modules matériels prédéfinis lors de l'étude du circuit à tester (AKE77). Ces modules sont activés mais non testés au cours du test de conformité.

EXEMPLE : L'étude de l'architecture du Z80 permet par exemple de définir les modules matériels suivants :

- unité arithmétique et logique.
- ensemble des registres dédiés.
- logique de contrôle.
- logique d'adressage.
- décodeur instructions.
- logique d'interruption.

Nous disposons ainsi d'un ensemble de modules matériels à tester. Le but est d'envoyer une donnée de test au module ou à une partie de module matériel, puis d'activer ce module matériel et d'observer le résultat obtenu. Nous choisissons un ensemble d'instructions simples permettant ces diverses opérations. Les modules les plus courant ainsi que les types d'instruction servant à leur balayage sont donnés en figure 6.

Les différents modules matériels peuvent être testés soit exhaustivement soit par une méthode classique soit à l'aide de vecteurs aléatoires. (VEL82)(AUR81)

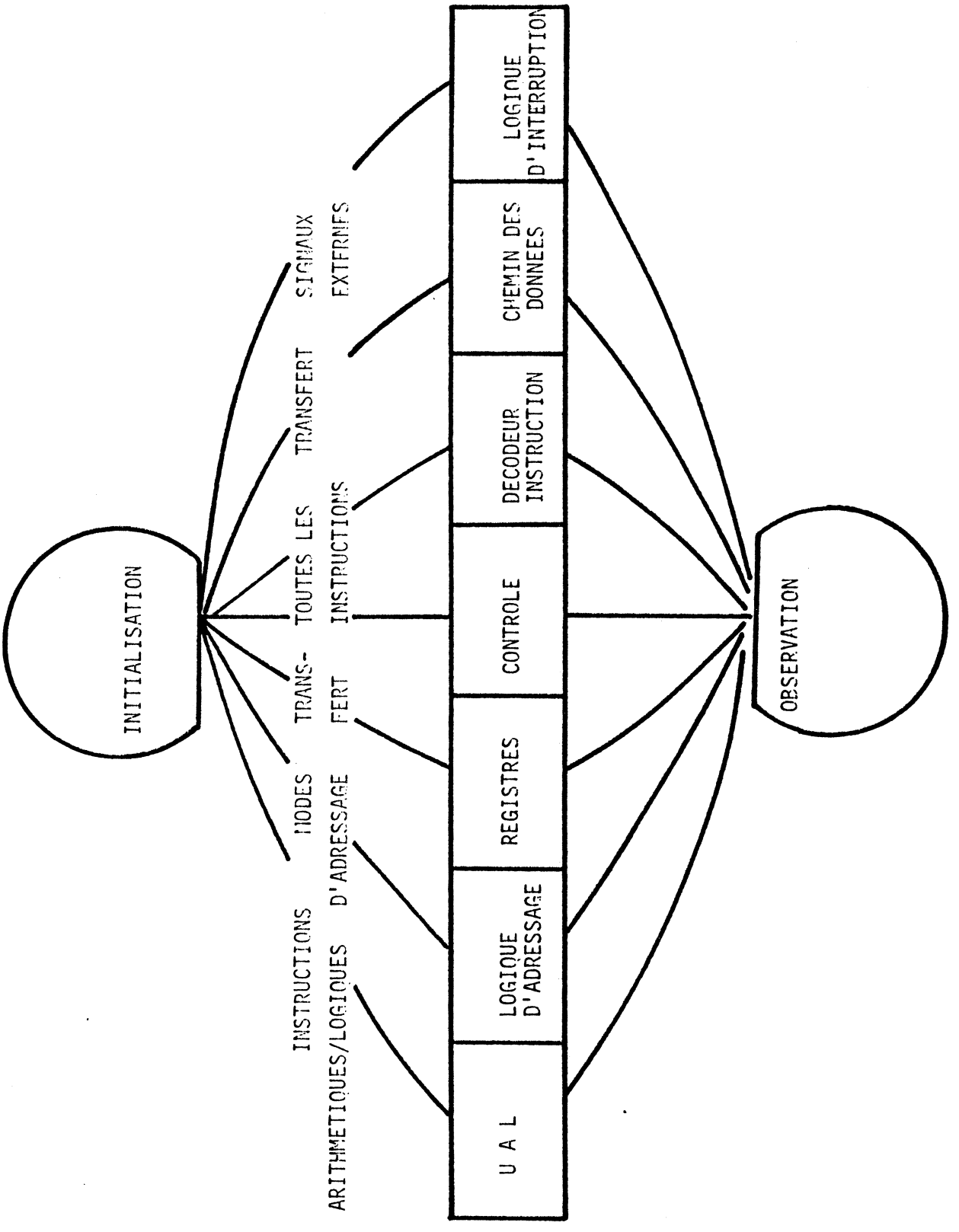

INITIALISATION
INSTRUCTIONS ARITHMETIQUES/LOGIQUES
MODES D'ADRESSAGE
TRANS-FERT
TOUTES LES INSTRUCTIONS
TRANSFERT
SIGNAUX EXTERNES
U A L
LOGIQUE D'ADRESSAGE
REGISTRES
CONTROLE
DECODEUR INSTRUCTION
CHEMIN DES DONNEES
LOGIQUE D'INTERRUPTION
OBSERVATION

FIGURE 6

EXEMPLE : On considère individuellement les registres du MC6800 de MOTOROLA. On suppose que l'on teste les registre par une méthode dérivée du WALKING 1/0.

Soient les vecteurs de test pour les registres de huit bits.

$$R8 = \{ 80, 40, 20, 10, 08, 04, 02, 01 \}$$

Soient les vecteurs de test pour les registres de seize bits.

$$R16 = \{ 8000, 4000, 2000, 1000, 0800, \ldots\ldots, 0008, 0004, 0002, 0001 \}$$

Registre A : ACCUMULATEUR

```
LDAA  . 80
STAA    ADR
LDAA  . 40

LDAA  . 01
STAA    ADR
```

Registre X : INDEX

```
LDX   . 8000
STX     ADR
LDX   . 4000
STX     ADR

LDX   . 0001
STX     ADR
```

Registre CC : CODE CONDITION

```
LDAA . 80
TAP
TPA
STAA . ADR
```

```
LDAA   01
TAP
TPA
STAA   ADR
```

a - LE TEST DE CONFORMITE

Il s'agit de déterminer un ensemble de transitions représentatives. Ces transitions doivent :

- activer une seule fois chaque opérateur.
- activer exhaustivement les configurations de conditions (externes ou internes).

Cet ensemble de transitions de transition peut être réduit par les règles en II-2-1.

Pour tester un représentant de l'ensemble des transitions on applique la séquence suivante :

- initialisation de l'état comportemental.
- activation de la transition.
- observation de l'état comportemental.

L'initialisation de l'état comportemental se fait au moyen d'instructions de chargement en respectant les règles suivantes :

- les éléments de mémorisation activés par la transition sont initialisés à des valeurs telles que chacun d'entre eux est modifié par l'activation de la transition.
- le compteur programme n'est pas initialisé. Il est observé en sortie
- les éléments de mémorisation non utilisés sont initialisés à des valeurs neutres différentes des précédentes permettant de vérifier la réelle activation d'une transition.

CHAPITRE 3

1 - LA MACHINE DE TEST

Afin de valider la méthode de test définie au CHAPITRE 2 et d'utiser les programmes de test générés automatiquement par G.A.P.T. une machine de test adaptée à été conçue.

1 - 1. CHOIX DE L'ENVIRONNEMENT (VEL82)(SAD81)

Les tests classiques s'effectuent au moyen d'équipements automatiques de test ; ces machines envoient les vecteurs de test séquentiellement. Deux raisons s'opposent à l'utilisation de ce type de machine pour le test usager que nous envisageons :

- chaque instruction doit être analysée au niveau de son chronogramme pour déterminer les entrées du circuit correspondant à chaque front d'horloge.
- l'architecture de certains microprocesseurs est de type "pipe-line"; la lecture de la mémoire peut ne pas se faire dans un ordre logique ou fonctionnel.

Si l'on admet que le microprocesseur sous test reste maître de l'exécution des programmes de test, les problèmes précédents disparaîssent. Cette stratégie permet l'utilisation d'un système classique de développement de logiciel de microprocesseurs.

On se heurte de nouveau à une série de problèmes :

- les écritures mémoire du microprocesseur sous test sont uniformément réparties dans l'espace mémoire ce qui implique une gestion délicate de l'implantation de ces programmes dans la mémoire.
- Le test concerne le jeu d'instruction mais aussi les signaux asynchrones positionnés en entrée du circuit sous test. Les systèmes classiques de développement ne permettent pas ce type d'excitation de signaux.

On propose donc une architecture originale de machine de test. elle permet le déroulement de programmes testant le jeu d'instructions et l'envoi de signaux externes au circuit. La mémoire de test n'est pas validée pendant les écritures du microprocesseur sauf dans des cas particuliers comme les sauvegardes de contexte.

Toutes les broches en sortie sont observées à l'aide de registres de signature. La signature est programmée de manière logicielle et matérielle.

L'architecture proposée est la suivante :

ARCHITECTURE DU TIROIR DE TEST

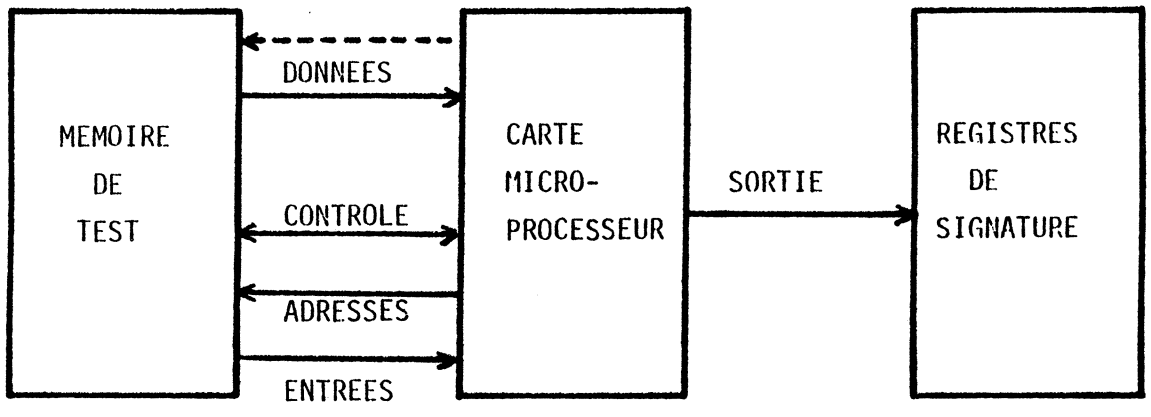


FIGURE 1

La taille des programmes de test est parfois supérieure à celle de la mémoire programme ; ceci implique un découpage du programme. Le programme complet sera stocké dans une mémoire de masse, un processeur de contrôle aura la maîtrise de tous les transferts. La nécessité du test du jeu d'instructions et des signaux externes implique pour le stockage des informations une structure particulière de la mémoire.

I - 2. STRUCTURE DE L'ESPACE MEMOIRE (ESD82)

L'espace mémoire est séparé en deux parties :

- une mémoire programme.
- une mémoire signal contenant toutes les informations nécessaires à l'envoi de signaux externes au microprocesseur.

I.2.1 La mémoire programme

Cette mémoire contient le programme ou le segment de programme de test que le circuit aura à exécuter. En phase de test, la mémoire de test n'est jamais validée en écriture sauf pour une zone réservée d'une page.

I.2.2 La mémoire signal

Cette mémoire contient toutes les informations nécessaires à la génération de chronogrammes à envoyer au circuit sous test. Les informations sont de trois types :

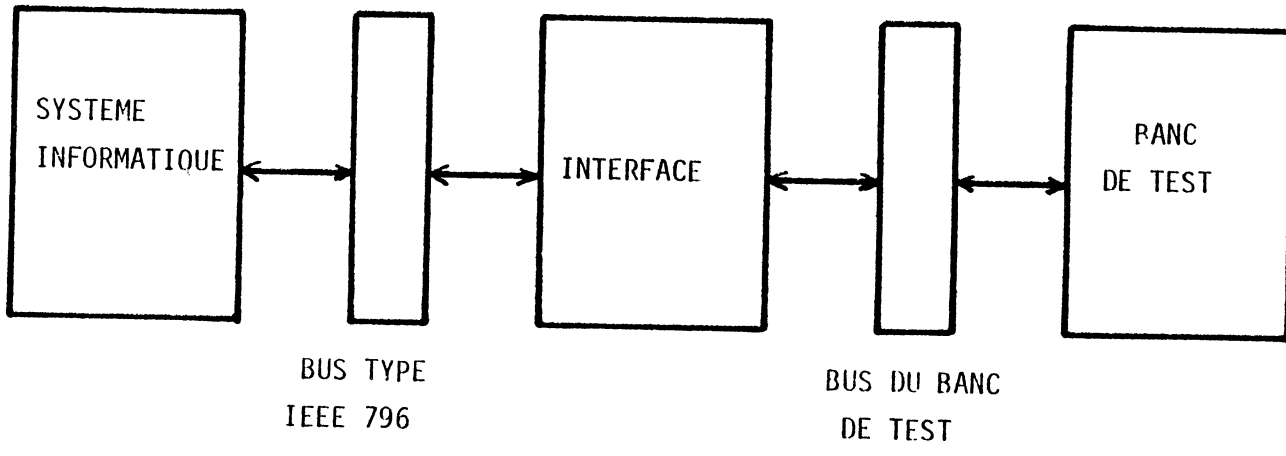
- une donnée correspondant à la configuration de signaux à appliquer.
- le temps de maintient avant validation sur le bus de la configuration à appliquer.
- une donnée de gestion de la mémoire.

I - 3. PRESENTATION GENERALE DU TESTEUR (EMD81)(ESD82)

Le système de test est destiné à mener à bien différentes opérations :

- rangement des programmes de test dans une mémoire de masse.
- tansfert des programmes vers les mémoires de test.
- initialisation du système de test.
- lancement des programmes de test.
- rangement des résultats de test pour exploitation.

Nous avons retenu l'architecture suivante :



ARCHITECTURE DU TESTEUR

FIGURE 2

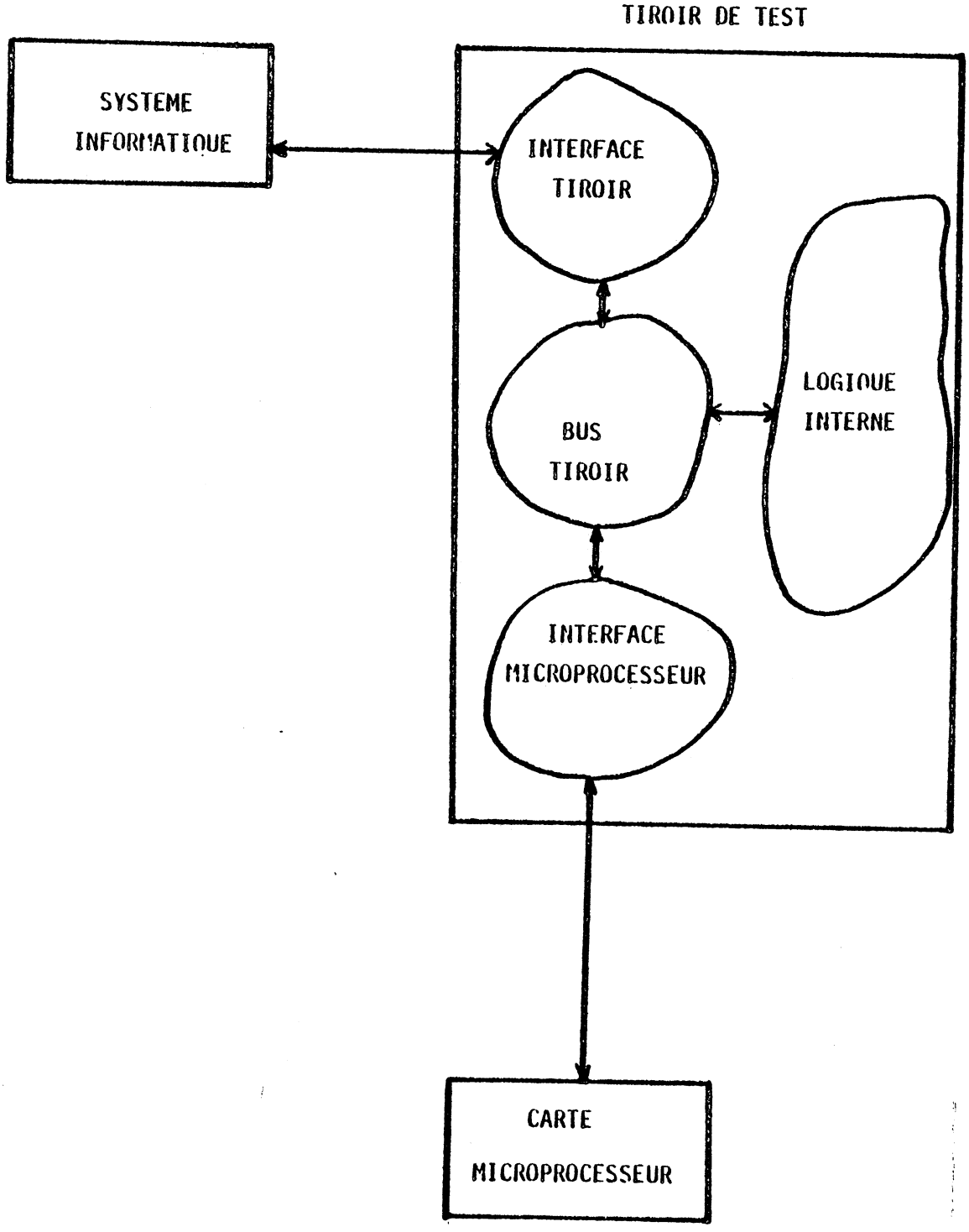

TIROIR DE TEST
SYSTEME
INFORMATIQUE
INTERFACE
TIROIR
BUS
TIROIR
LOGIQUE
INTERNE
INTERFACE
MICROPROCESSEUR
CARTE
MICROPROCESSEUR

FIGURE 3

Le système informatique a deux fonctions essentielles :

- la gestion des programmes et des signatures de test.
- la gestion des activités du tiroir de test.

Les différentes phases d'activité du tiroir de test sont les suivantes :

- le transfert des programmes de test (phase DMA).
- l'initialisation du tiroir de test (phase initialisation).
- le lancement du programme de test (phase de test).
- le rangement des signatures de test (phase observation).

A travers l'interface tiroir de test, le système informatique gère le banc de test. Il accède ainsi aux bus internes du banc de test, soit :

- le bus signaux de gestion du tiroir de test.
- le bus signaux de gestion mémoire.
- le bus données.
- le bus adresses.

Le circuit sous test est relié au banc de test par une carte d'interfaçage.

Pendant la phase de test le circuit devient le maître de toutes les ressources du banc de test. Il exécute le programme contenu enmémoire et reçoit des signaux externes provenant de la mémoire signal. La fin de l'exécution du programme de test rend la main au système informatique.

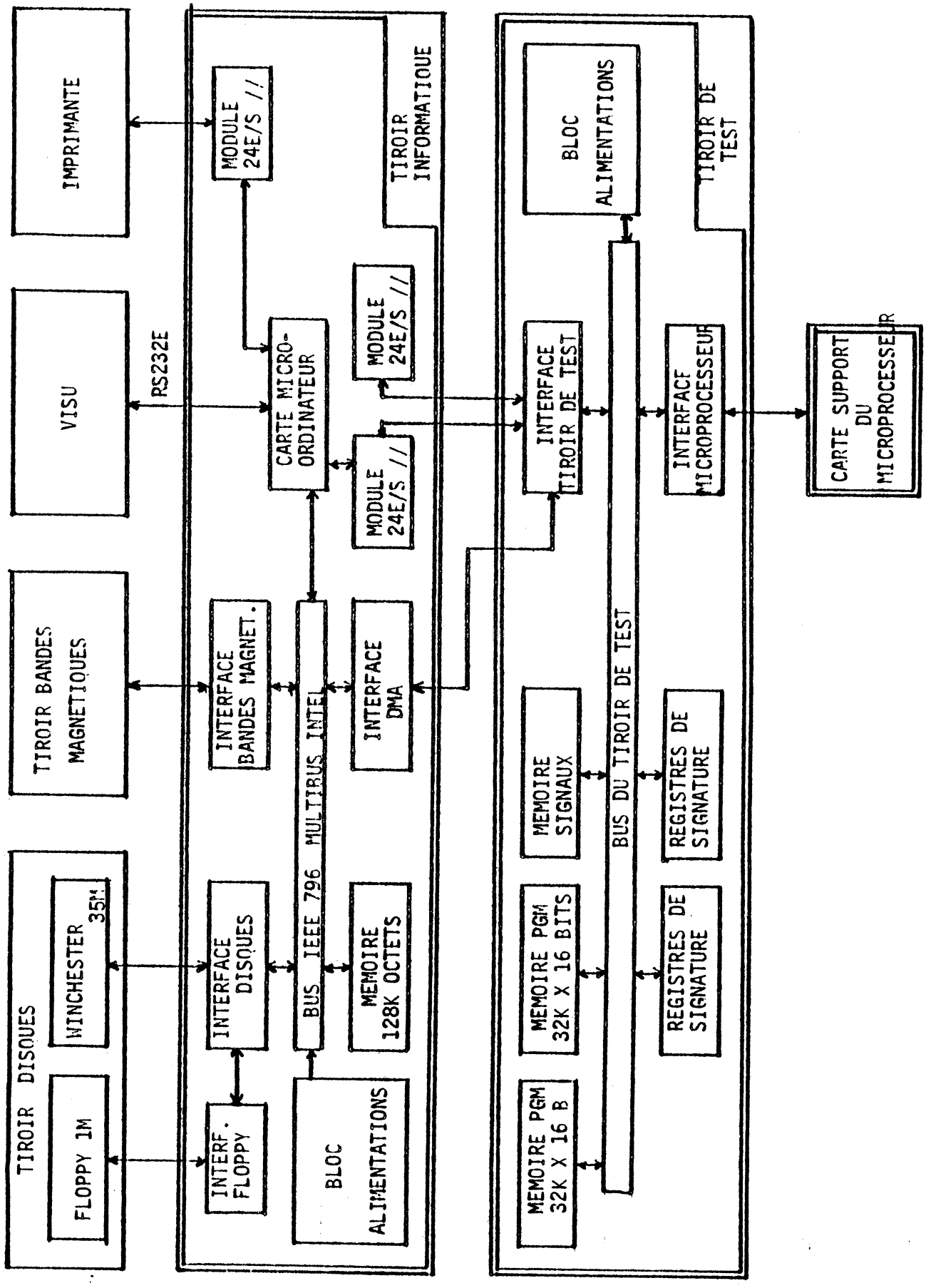

IMPRIMANTE
VISU
RS232E
TIROIR BANDES MAGNETIQUES
TIROIR DISQUES
FLOPPY 1M
WINCHESTER 35M
INTERF. FLOPPY
INTERFACE DISQUES
BLOC ALIMENTATIONS
BUS IEEE 796 MULTIBUS INTEL
MEMOIRE 128K OCTETS
INTERFACE BANDES MAGNET.
INTERFACE DMA
MODULE 24E/S //
CARTE MICRO-ORDINATEUR
MODULE 24E/S //
MODULE 24E/S //
TIROIR INFORMATIQUE
BLOC ALIMENTATIONS
TIROIR DE TEST
INTERFACE TIROIR DE TEST
INTERFACE MICROPROCESSEUR
CARTE SUPPORT DU MICROPROCESSEUR
MEMOIRE SIGNAUX
MEMOIRE PGM 32K X 16 BITS
MEMOIRE PGM 32K X 16 B
BUS DU TIROIR DE TEST
REGISTRES DE SIGNATURE
REGISTRES DE SIGNATURE

FIGURE 4

1 - 4. DESCRIPTION FONCTIONNELLE DU TIROIR DE TEST (ESD83)

Dans cette partie nous allons décrire les principales parties du tiroir de test. Ce sont :

- l'espace mémoire.
- le système d'observation des sorties.
- les interfaces.
- les registres d'activation.

1.4.1 L'espace mémoire

- La mémoire programme : de dimension 64K X 16bits, cette RAM contient le programme à exécuter par le microprocesseur. En phase TEST, cette mémoire n'est jamais validée en écriture sauf si ces écritures s'adressent à une zone réservée de 256 mots.

- La mémoire signal : cette RAM de 1K X 32bits est chargée en phase DMA et est lue en phase TEST. En phase TEST, la mémoire signal génère les chronogrammes des signaux d'entrée du circuit sous test. L'adresse de la mémoire signal est unique et sert à sa validation. La gestion est assurée par une logique indépendante.

Cette mémoire est partagée en trois champs :

- un champ de 24 bits contenant les configurations des signaux d'entrée du microprocesseur sous test. (Epp.n)
- un champ de 6 bits (nombre de 0 à 63). Ce nombre correspond à la valeur de temporisation Tn à appliquer avant que Epp.n ne soit envoyé au microprocesseur sous test.
- un champ de 2 bits permettant la gestion de la mémoire. (Sn,Vn) Sn est le signal autorisant le transfert du mot mémoire suivant Epp.n+1 après décomptage de la temporisation Tn+1, vers le microprocesseur sous test.

Sn=1 Epp.n+1 est validé.

Sn=0 L'activité de la mémoire est suspendue jusqu'à l'accès suivant.

Vn est l'indicateur de validité du test.

Vn=1 Après la phase d'initialisation du microprocesseur.

Vn=0 A la fin du test.

Pendant la phase d'initialisation du microprocesseur.

La procédure d'utlisation de la mémoire signal est la suivante :

- les adresses sont fournies par un compteur binaire.
- Epp.n est connecté au bus allant vers le microprocesseur sous test.
- Vn est connecté au signal de validation du test.
- Tn est connecté à l'entrée d'un temporisateur.
- Sn est connecté à la logique de séquencement.
- la mémoire est validée en lecture.

La séquence de transfert d'un mot Epp.n est déclenchée par une écriture du microprocesseur sous test à l'adresse unique de la mémoire signal. La reconnaissance de cette adresse déclenche le processus de temporisation. La fin de la temporisation provoque les actions suivantes :

- les mots Epp.n et Vn sont transférés vers le bus microprocesseur et le signal de validation du test.
- les mots Epp.n+1 et Vn+1 sont écrits dans les tampons intermédiaires.
- la temporisation Tn+1 est chargée.
- le compteur d'adresses est incrémenté.
- le test de fin de séquence est effectué sur Sn.

Soit la structure du contenu de la mémoire signal :

Eμp	T	S	V
BLOCAGE DU μp	ø	1	0
PROCEDURE START INITIALISATION	Ti	1 1	0 0
		1	0
Eμp		1	1
SEQUENCE N°1		1 0	1 1
SEQUENCE N°2		1 0	1 1
PROCEDURE STOP		1 1	0 0

FIGURE 5

- Registre de données périphériques :

Ce registre permet au microprocesseur sous test d'effectuer des lectures et des écritures en dehors de la mémoire programme; la lecture est toujours autorisée.

1.4.2 Le système d'observation des sorties (ESD82)

L'observation des sorties se fait avec des registres "feed-back" inspirés de la technique Bilbo (DAV80) dont le schéma de principe est donné ci-après :

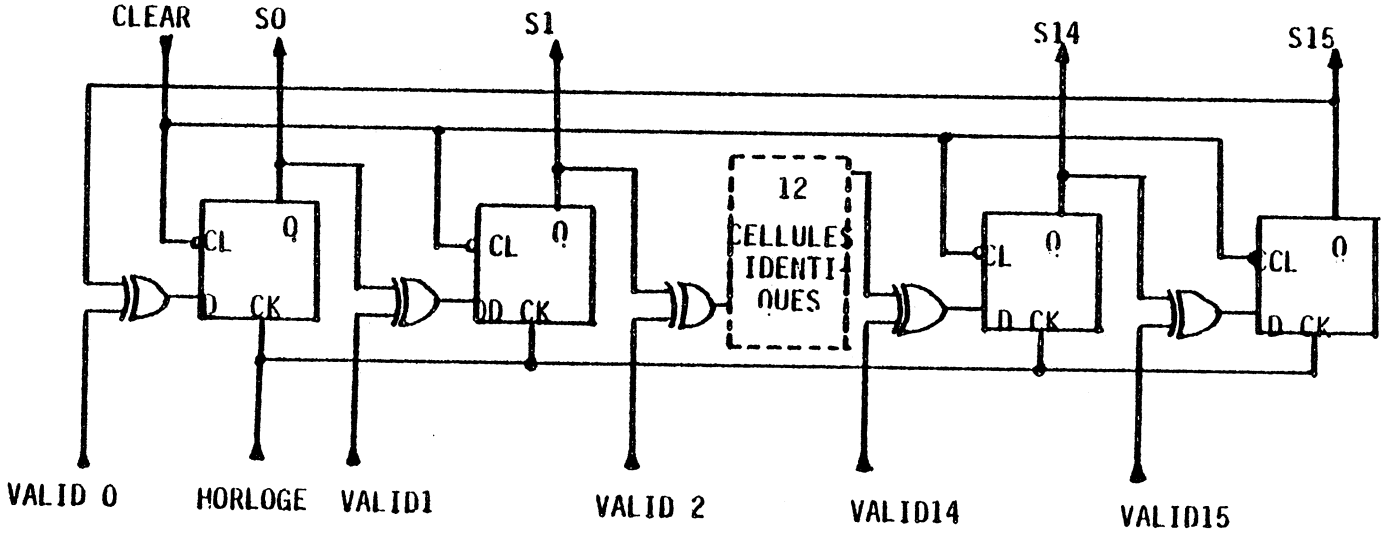


FIGURE 6

En fait on signe 16 signaux par registre. En pratique un registre de signature est affecté à chacun des bus D,Ad,S,Z. Le système de signature n'est validé que pendant la phase de TEST.

Chaque signal peut être masqué de manière logicielle ou matérielle ; de plus il lui est associé un signal de validation de signature. Tous ces signaux sont générés à partir de registres.

A chaque entité formant un octet est associé un signal de validation de signature ; il est généré sur la carte microprocesseur.

A l'ensemble des bus D,Ad,S,Z est associé un signal de validation de masquage ; on autorise ainsi le masquage sur les 64 signaux à signer. Ce masquage est validé par une écriture à une adresse unique spécialisée. La fin d'une procédure de masquage entraîne la remise à zéro des registres de masquage logiciel. Les registres de signature sont remis à zéro en début de test et sont activés par une horloge validée uniquement pendant le test et générée sur la carte microprocesseur.

1.4.3 Les interfaces (ESD82)(ESD83)

Le tiroir de test dispose de deux interfaces, l'une vers le système informatique et l'autre vers le microprocesseur sous test.

- Interface tiroir de test : (système informatique)

Cet interface est, en principe, anonyme vis à vis du système informatique. Le tiroir de test devrait pouvoir être géré par n'importe quel système. Cet interface assure la mise en forme des signaux de gestion du tiroir de test et des mémoires. Il assure également le multiplexage et la mise en forme des signaux d'entrée-sortie sur les bus D et Ad.

- Interface microprocesseur :

La carte interface microprocesseur présente deux aspects :

- l'interconnection des bus du microprocesseur avec ceux du tiroir de test et la logique de validation associée à cette connection. Cette fonction est indépendante du microprocesseur.

- la génération des signaux spécifiques pour un microprocesseur donné. Ces fonctions sont réalisées à partir d'une carte spécifique à chaque circuit. Cette carte sera en général aussi le support du microprocesseur sous test.

1.4.4 Les registres d'activité

Ils contiennent toutes les conditions spécifiques du fonctionnement du tiroir de test. Ces conditions sont relatives au microprocesseur sous test. Les registres d'activité sont répartis en trois groupes.

- Registres d'adresses :

Ils contiennent des valeurs d'adresses dont les apparitions sur le bus correspondant déclenchent des actions spécifiques.

- Registres de masquage :

Il est parfois nécessaire de masquer des parties d'information. Ces registres contiennent des valeurs de masquage relatives aux bus données D, adresses Ad, signature S et Z. Ces masquages peuvent être soit matériels soit logiciels.

- Registres de validation des signatures :

Ils contiennent les validations de signature pour chacun des bus D,Ad,S et Z.

II - LE GENERATEUR AUTOMATIQUE DE PROGRAMME DE TEST (BELGO-1)

II- 1. INTRODUCTION

La génération manuelle des programmes de test devient fastidieuse voire impossible étant donnée leur taille. Des langages de haut niveau sont utilisés; ils permettent la description des jeux de test; la traduction automatique en données de test finales est ensuite possible: PASCAL (KIZ80), PLT développé par IBM (PAS80). Le langage robin proposé dans (ROB81) utilise des macroinstructions de test. Ces macroinstructions spécifient le type, les registres source et puits, le format... des instructions. Après assemblage une suite d'instructions mettant en oeuvre ces registres, les différents formats... sont générés automatiquement.

Dans notre approche un langage de description du microprocesseur est proposé ; à partir de ce langage le programme de test est généré automatiquement. Cette génération s'appuie sur le caractère itératif des programmes de test ; chaque pas est constitué de :

- l'initialisation du circuit.
- la stimulation du circuit.
- l'observation du circuit.

Les itérations se font sur des domaines comme :

- les modes d'adressage.
- les tailles des opérandes.
- le contexte du circuit (superviseur ou utilisateur).
- les registres utilisés.

On illustre l'utilisation de ce langage de description par un exemple: l'instruction d'addition binaire du MC68000 de MOTOROLA.

EXEMPLE : ADD <ea>,Dn

Dn est un ensemble de registres: D0 à D7
ea est l'adresse réelle du mot mémoire.

L'instruction effectue Dn + (<ea>) ----> Dn

L'instruction est décrite de la manière suivante :

' ADD.' size ea ',' Dn

Les parties entre apostrophes sont fixes et forment le noyau de la syntaxe assembleur. Les parties variables décrites ci-après donnent sa forme définitive à l'instruction générée.

size = B,W,L la taille des opérandes peut être: B=8 bits, W=16bits, L=32 bits.

ea on précise les modes d'adressage autorisés pour l'instruction décrite.

Dn est défini par l'ensemble des registres D0 à D7.

L'algorithme de génération du programme de test permet d'obtenir la séquence suivante :

```
INITIALISATION
ADD.B     premier mode d'adressage , D0
OBSERVATION
INITIALISATION
ADD.W     premier mode d'adressage , D0
OBSERVATION


INITIALISATION
ADD.W     dernier mode d'adressage , D7
OBSERVATION
INITIALISATION
ADD.L     dernier mode d'adressage , D7
OBSERVATION
```

Le mnémonique ADD <ea>,Dn représente environ 8000 instructions différentes. L'option maximum génère l'ensemble de ces 8000 instructions ; des options réduites permettent de ne générer qu'une partie du programme en n'itérant que sur les tailles d'opérandes D0 sur les modes d'adressage D0 sur des ensembles de registres.

II - 2. ARCHITECTURE DE G.A.P.T. (BEL82-3)(BEL82-4)

A partir de la description des circuits et des opérandes de test prédéterminés, le système génère les modules du programme de test en langage assembleur du microprocesseur testé ou du microprocesseur associé au circuit périphérique testé.

LE SYSTEME GAPT

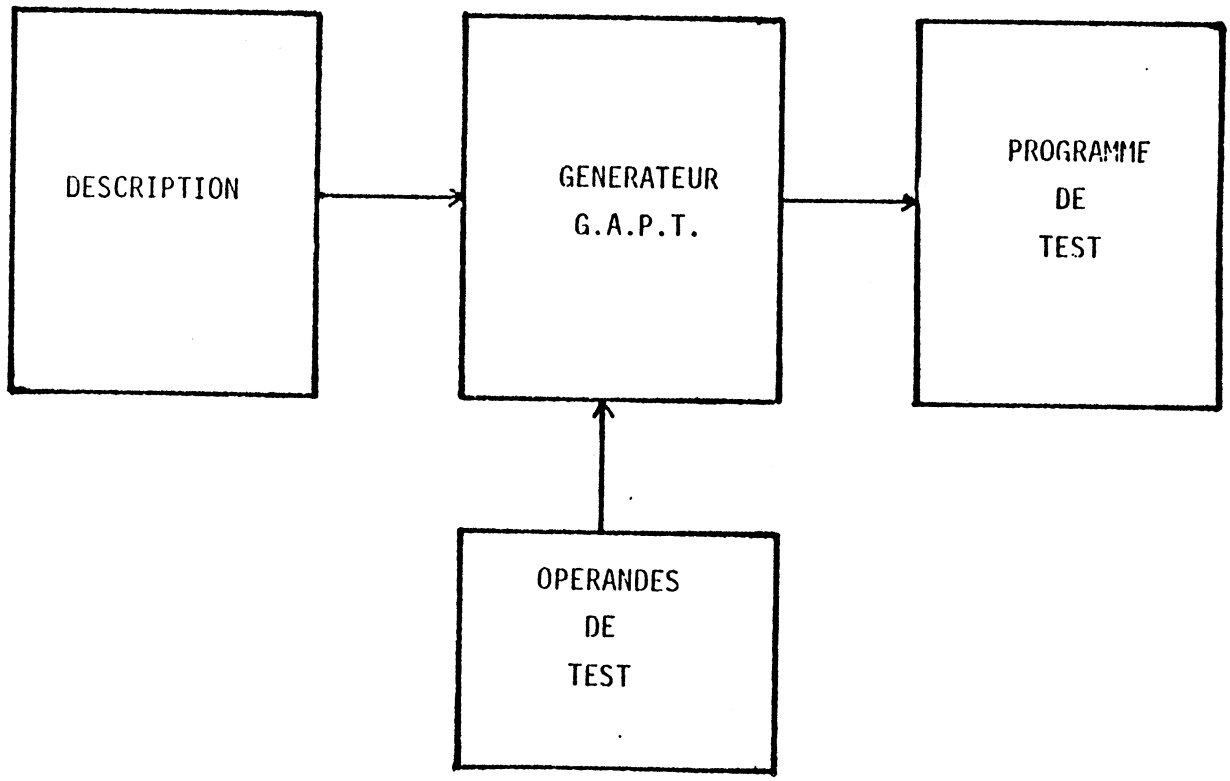


FIGURE 7

II.2.1 Introduction

Le système G.A.P.T. peut être considéré comme un compilateur particulier. Il est composé comme tout système du même type de quatre parties :

- l'analyse lexicographique est l'interface entre le programme source, dans notre cas la description du microprocesseur, et le système G.A.P.T. Elle permet l'acquisition des éléments du langage.
- l'analyse syntaxique vérifie la syntaxe de la description du circuit sous test et réalise la récupération des erreurs.
- l'analyse sémentique effectue des vérifications en contexte : déclarations, types, validité des types dans le contexte...
- le générateur de code, dans notre cas produit un programme de test en assembleur.

II.2.2 Le langage de description de G.A.P.T. (BEL83-1)(ESD83)

Les informations nécessaires à la génération du programme de test sont regroupées sous trois item :

- une description normalisée du microprocesseur: ces informations concernent les éléments de mémorisation interne, les valeurs spécifiques de ces éléments de mémorisation comme le registre d'état, les modes d'adressage, le jeu d'instructions...
- génération des programme de test: on regroupe sous cet item les directives d'assemblage nécessaires au générateur (réservation de données, définition de constantes...), les informations nécessaires à la gestion de l'espace mémoire et les utilitaires (contenu des zones réservées, programmes de traitement des exceptions, macroinstructions assembleur, macroinstructions signaux...)
- description des chronogrammes: les informations concernant les signaux et leurs chronogrammes sont mises en forme dans cette rubrique ; on précise également la structure de la mémoire signal et la procédure de déclenchement de celle-ci. Les tests des signaux, les macroinstructions signaux, les transformations de chronogrammes, la gestion de la mémoire signal ... sont aussi définis dans cette rubrique.

III - LE TEST DES SIGNAUX

III - 1. INTRODUCTION

Les broches affectées aux signaux de fonctionnement minimum sont vérifiés implicitement pendant toute la durée du test; les bus adresses et données et contrôle le sont lors du test du jeu d'instruction; pour compléter le test il est nécessaire de vérifier la fonctionnalité des broches positionnées en entrée concernant les signaux asynchrones.

III - 2. MISE EN OEUVRE (ESD82)(ESD83)

Le système de test qui a été développé permet le positionnement des broches d'entrée correspondant aux signaux asynchrones à l'aide de la mémoire signal. Cette mémoire contient toutes les séquences de combinaisons de valeurs nécessaires à l'excitation des différentes broches. Toutes les séquences sont possibles car une combinaison peut être envoyée à chaque phase d'horloge. Les informations concernant les signaux sont rangées en mémoire sous forme condensée; de ce fait on diminue l'occupation mémoire des programmes de test des signaux. Le déclenchement de cette mémoire se fait par une écriture à une adresse fixe. A partir de ce moment les signaux sont envoyés au microprocesseur dans un délai dont le domaine de variation est prédéfini.

III - 3. DESCRIPTION DES SIGNAUX ET DES CHRONOGRAMMES

III.3.1 Définitions

On définit trois notions de base concernant les signaux :

- un signal.
- un chronogramme élémentaire.
- un chronogramme.

DEFINITION :

Un signal est affecté à une broche d'un circuit ; il est défini par un nom, une position dans la mémoire signal et sa valeur initiale.

DEFINITION :

Un chronogramme élémentaire est associé à un signal ; on peut associer plusieurs chronogrammes élémentaires à un même signal. Le chronogramme élémentaire représente les variations d'un signal pour une activation donnée.

DEFINITION :

Un chronogramme est composé de plusieurs chronogrammes élémentaires pouvant porter sur le même signal ou sur des signaux différents.

III.3.2 Description des chronogrammes

On utilise pour la description des chronogrammes et des signaux un formalisme approprié dérivé des travaux sur l'algèbre des événements de (HAL82) (CAS82) (BOU82).

Ce formalisme permet une description aisée et très lisible des variations d'un signal ; de plus il est très souple d'utilisation et d'écriture au niveau des ajouts et des compositions des chronogrammes.

Toutes les entités ainsi définies et traduites sont rangées en mémoire signal.

Quel que soit le type de test envisagé, on définit deux chronogrammes de base :

- un chronogramme permettant l'initialisation du circuit avant le test. Ce chronogramme dit START est placé au début de la mémoire signal.
- un chronogramme permettant la clôture du test. Ce chronogramme STOP est placé en dernière position dans la mémoire signal.

Préalablement on définit deux grandeurs :

- le retard à l'envoi du chronogramme.
- les intervalles de variation des signaux.

- Retard à l'envoi du signal ou du chronogramme: on le note RET ; cette grandeur symbolise l'intervalle de temps minimal compris entre la validation de la mémoire signal et l'envoi réel du premier vecteur ACTIF au microprocesseur. On peut faire varier le retard RET à l'intérieur d'un ensemble de valeurs D.RET.

- Temporisation entre deux variations: pour un chronogramme portant sur un seul signal, l'intervalle de temps i entre deux variations du signal définit la temporisation à stocker dans la mémoire signal.

Remarque : Toutes les variations se font par rapport à l'état initial du signal. Le sens de variation importe peu.

Le symbolisme utilisé est illustré par une série d'exemples significatifs.

EXEMPLES :

- <u>description d'un chronogramme élémentaire.</u>

$\mathcal{G}$ est un chronogramme associé au signal S.

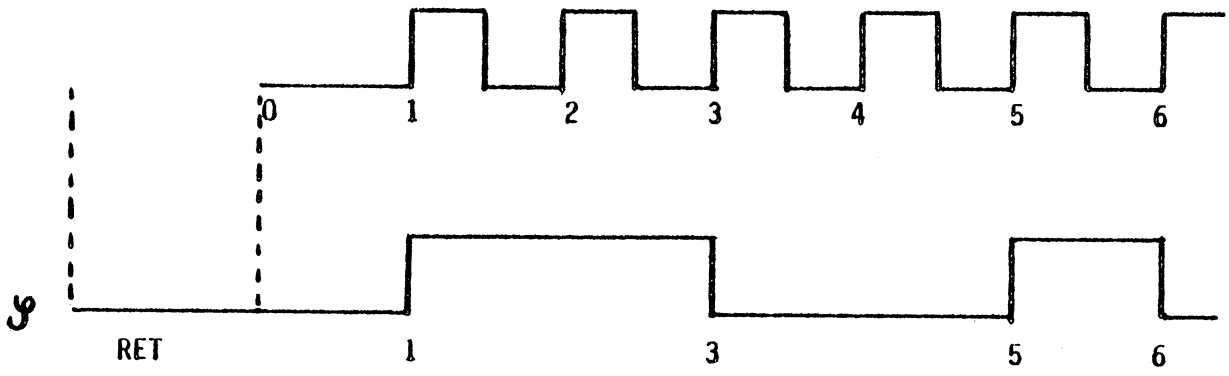


FIGURE 8

$$\mathcal{G} = (R\ast\ast RET)\ast S(R\ast\ast 1+R\ast\ast 3+R\ast\ast 5+R\ast\ast 6)$$

$$\mathcal{G} = S(R\ast\ast(RET+1)+R\ast\ast(RET+3)+R\ast\ast(RET+5)+R\ast\ast(RET+6))$$

- addition de deux chronogrammes élémentaires portant sur le même signal S.

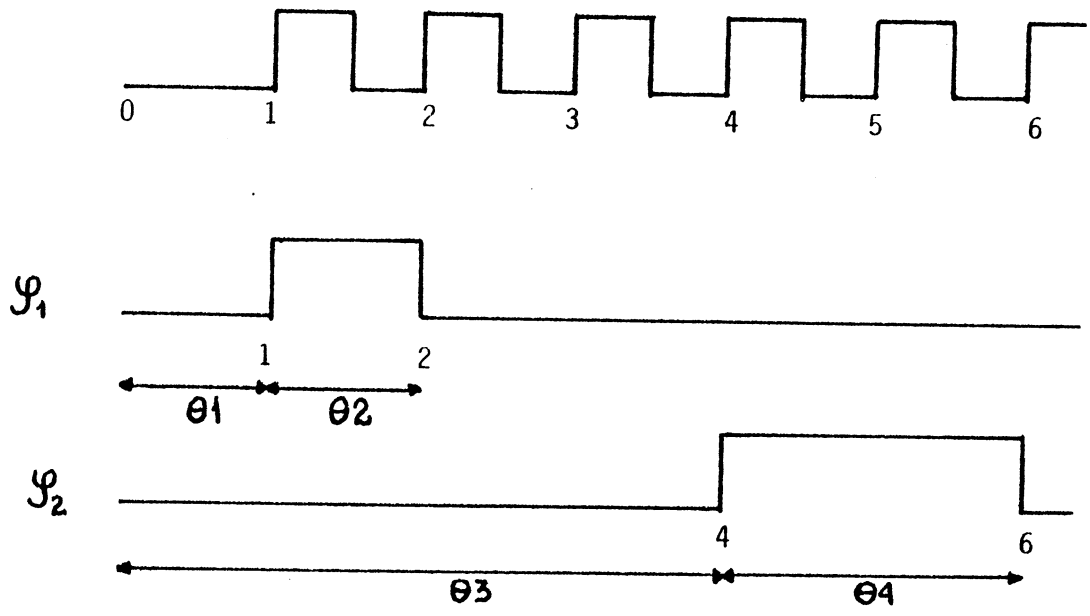


FIGURE 9

$\varphi_1 = S(R\ast\ast 1+R\ast\ast 2)$

$\varphi_2 = S(R\ast\ast 4+R\ast\ast 6)$

$\varphi = \varphi_1 + \varphi_2 = S(R\ast\ast 1+R\ast\ast 2+R\ast\ast 4+R\ast\ast 6)$

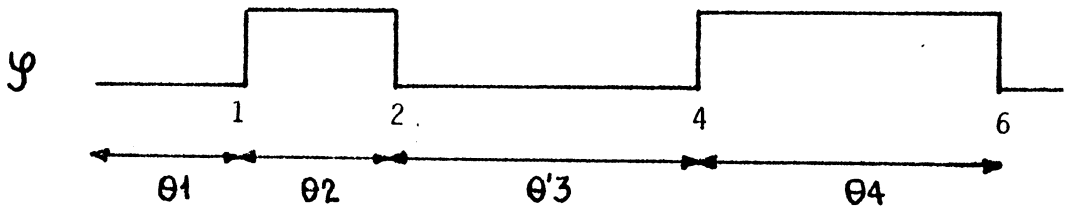


FIGURE 10

$\theta'3 = \theta 3 - (\theta 1 + \theta 2)$

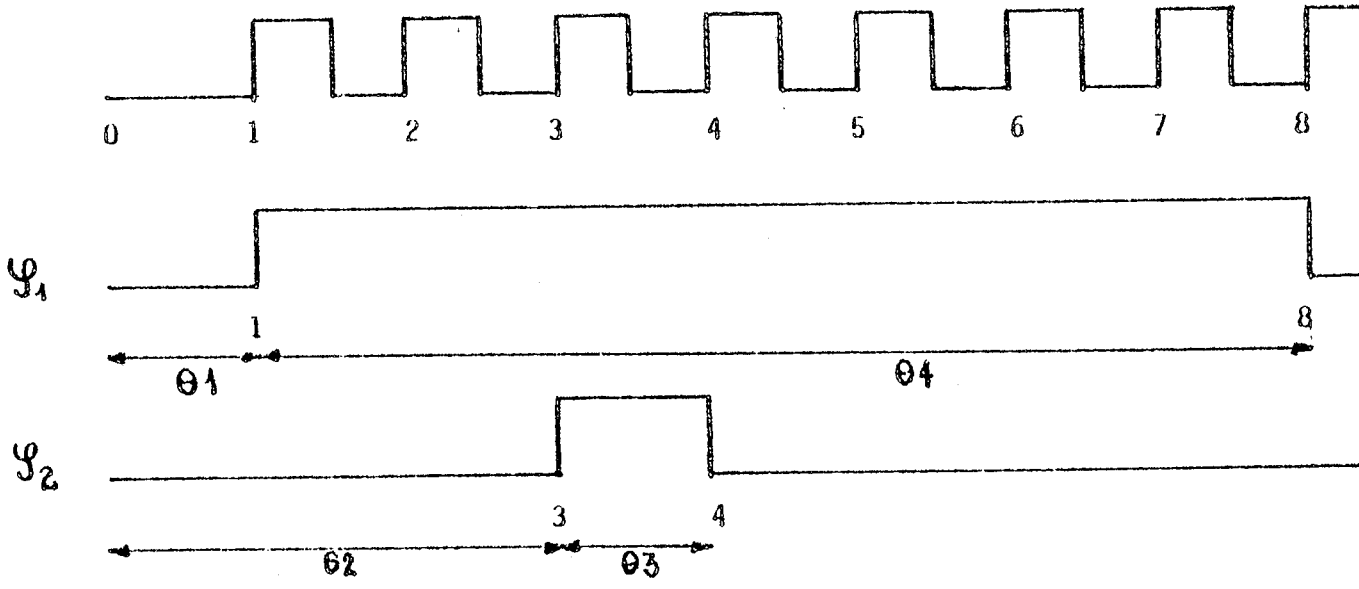


FIGURE 11

$$\varphi_1 = S(R*1+R*8)$$
$$\varphi_2 = S(R*3+R*4)$$

$$\varphi = \varphi_1 * \varphi_2 = S(R*1+R*3+R*4+R*8)$$

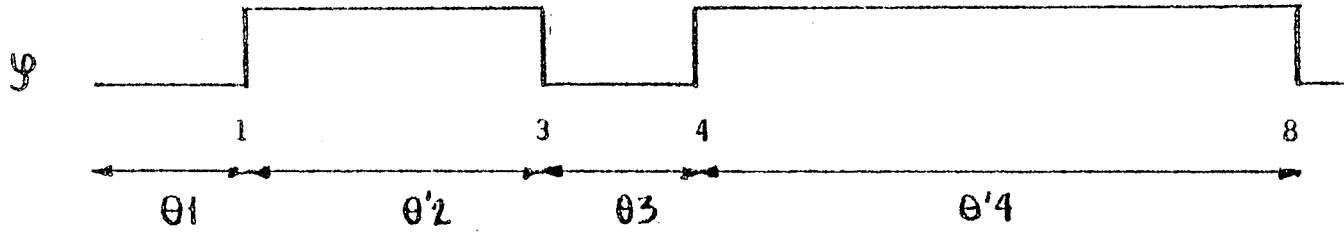


FIGURE 12

$$\theta'2 = \theta 2 - \theta 1$$
$$\theta'4 = \theta 4 - (\theta'2 + \theta 3)$$

Cet exemple illustre le fait que l'on considère uniquement l'instant de variation et non le sens de cette variation.

- composition de chronogrammes.

Cette opération permet de former à partir de plusieurs chronogrammes élémentaires portant chacun sur un signal, une entité constituant un chronogramme.

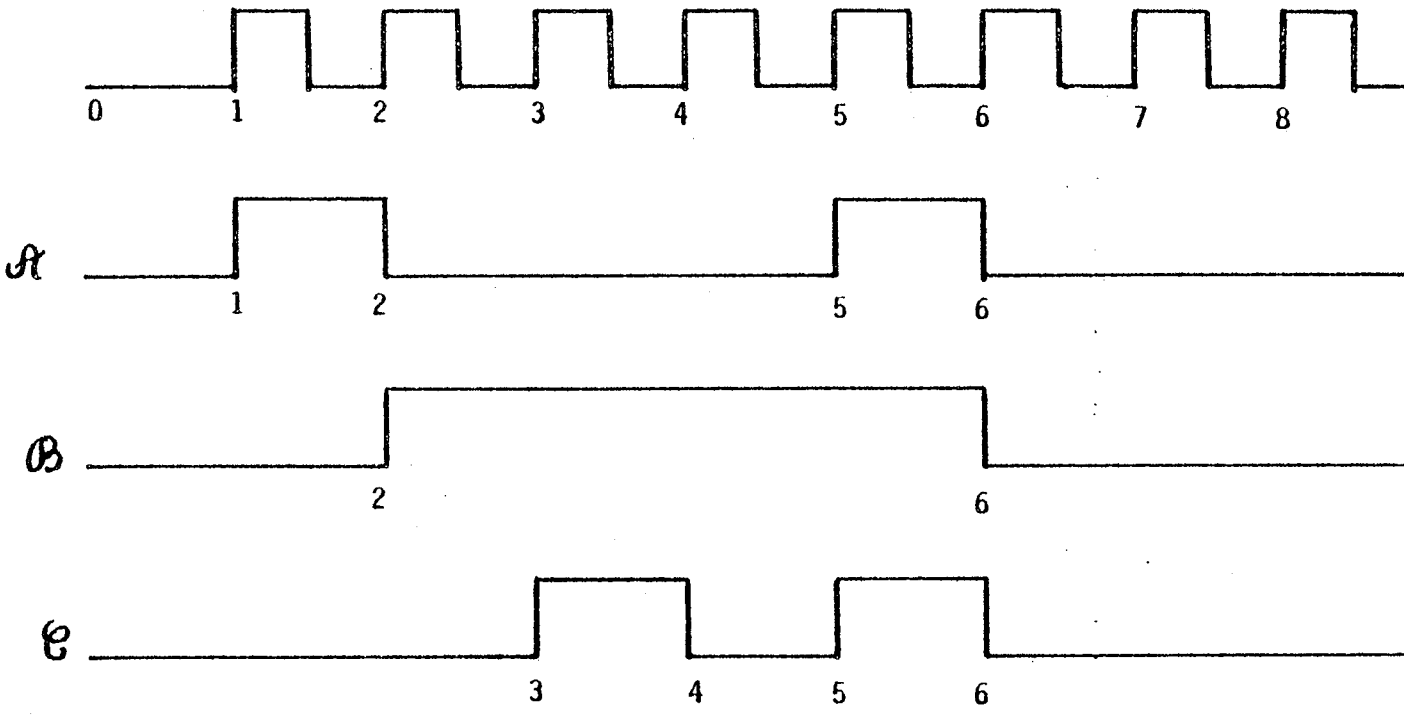


FIGURE 13

A = A(R**1+R**2+R**5+R**6)
B = B(R**2+R**6)
C = C(R**3+R**4+R**5+R**6)

On définit le chronogramme $\mathcal{C}$ résultat de la composition des chronogrammes élémentaires $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$.

$$\mathcal{C} = \mathcal{A} + \mathcal{B} + \mathcal{C}$$

$$\mathcal{C} = A(R_{*}1+R_{**}2+R_{**}5+R_{**}6)+B(R_{**}2+R_{**}6)+C(R_{*}3+R_{**}4+R_{**}5+R_{**}6)$$

Cette expression peut se mettre sous une forme faisant apparaître les instants de variations et les signaux affectés par cette variation.

$$\mathcal{C} = AR_{*}1+(A+B)R_{**}2+CR_{**}3+CR_{**}4+(A+C)R_{**}5+(A+B+C)R_{**}6$$

- addition de chronogrammes.

Le procédé d'addition des chronogrammes est identique à celui des chronogrammes élémentaires. Les chronogrammes peuvent ne pas être composés de chronogrammes concernant les mêmes signaux. On additionne alors les chronogrammes élémentaires concernant les mêmes signaux, puis on compose au chronogramme ainsi obtenu les chronogrammes élémentaires restant.

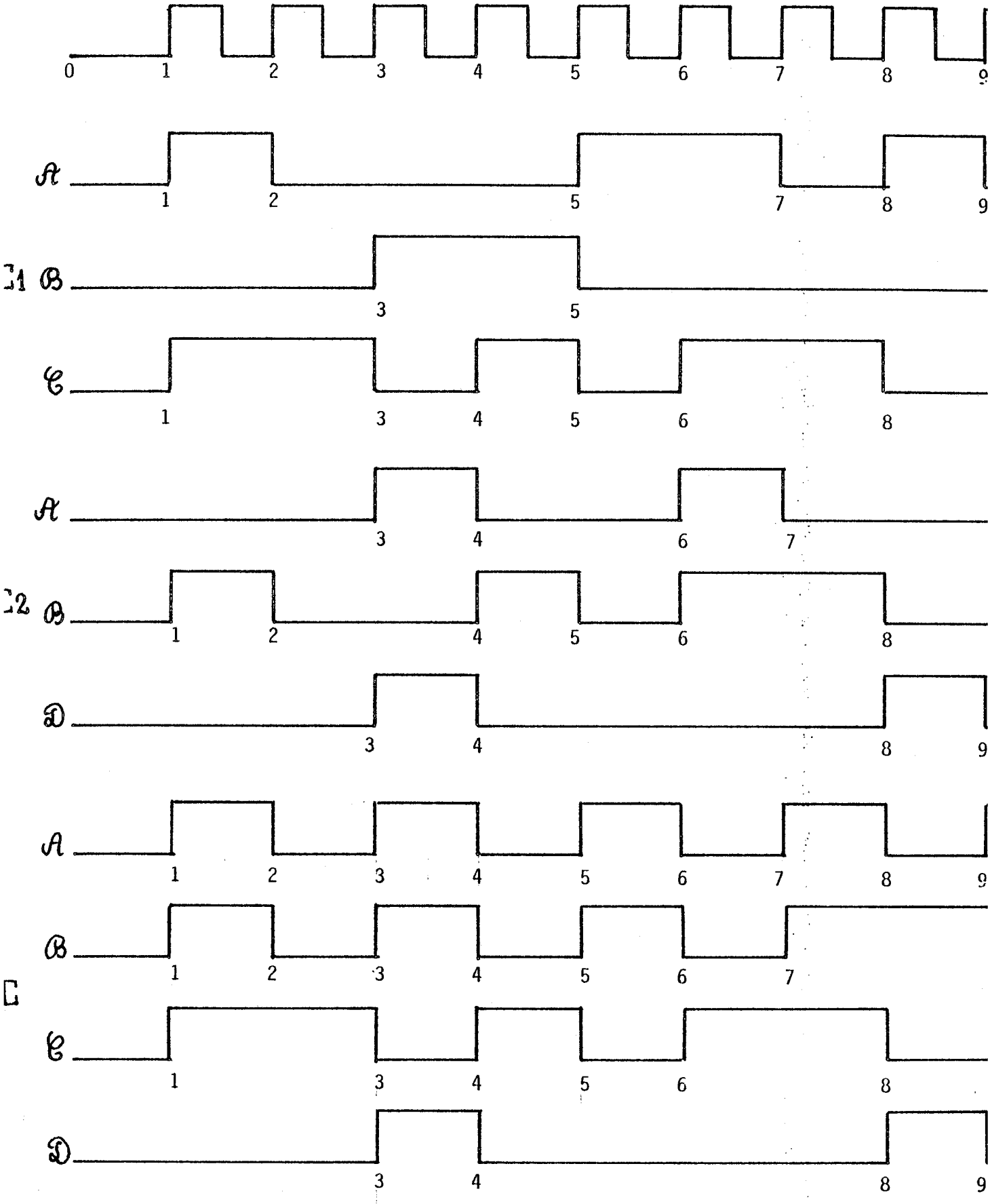

0 1 2 3 4 5 6 7 8 9
A
1 2 5 7 8 9
B
3 5
C
1 3 4 5 6 8
A
3 4 6 7
B
1 2 4 5 6 8
D
3 4 8 9
A
1 2 3 4 5 6 7 8 9
B
1 2 3 4 5 6 7
C
1 3 4 5 6 8
D
3 4 8 9

FIGURE 14

- description d'une famille de chronogrammes.

Pour décrire une famille de chronogrammes dont les instants de variations sont compris entre deux bornes, on utilise le formalisme suivant.

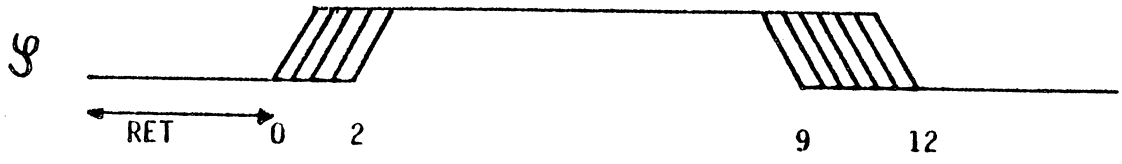


FIGURE 15

$$\mathcal{G} = (R \ast\ast RET) \ast S(R \ast\ast \tau 1 + R \ast\ast \tau 2)$$

$$RET = (5,7)$$

$$\tau 1 = (0,1,2)$$

$$\tau 2 = (9,10,11,12)$$

A partir de ce chronogramme paramétré par RET, $\tau 1$ et $\tau 2$, GAPT génèrera tous les chronogrammes possibles avec les 24 triplets de valeurs (RET, $\tau 1$, $\tau 2$).

Ce même paramétrage peut être effectué sur un chronogramme complexe portant sur plusieurs signaux.

- remplissage de la mémoire signal.

Soit le chronogramme $\mathcal{C}$ (retard RET=2)

$$\mathcal{C} = R \ast\ast 2(AR \ast\ast 1 + (A+B)R \ast\ast 2 + CR \ast\ast 3 + CR \ast\ast 4 + (A+C)R \ast\ast 5 + (A+B+C)R \ast\ast 6 + AR \ast\ast 9 + AR \ast\ast 10$$

Eμp.0 = (0,0,0)

	SIGNAUX			TEMPORISATION		
	A	B	C		T	S	V
DEBUT	0	0	0	11	0	1
	1	0	0	01	1	1
	0	1	0	01	1	1
	0	1	1	01	1	1
	0	1	0	01	1	1
	1	1	1	01	1	1
	0	0	0	11	1	1
	1	0	0	01	1	1
	0	0	0	T CHRONO SUIVANT	0	1

TABLEAU II

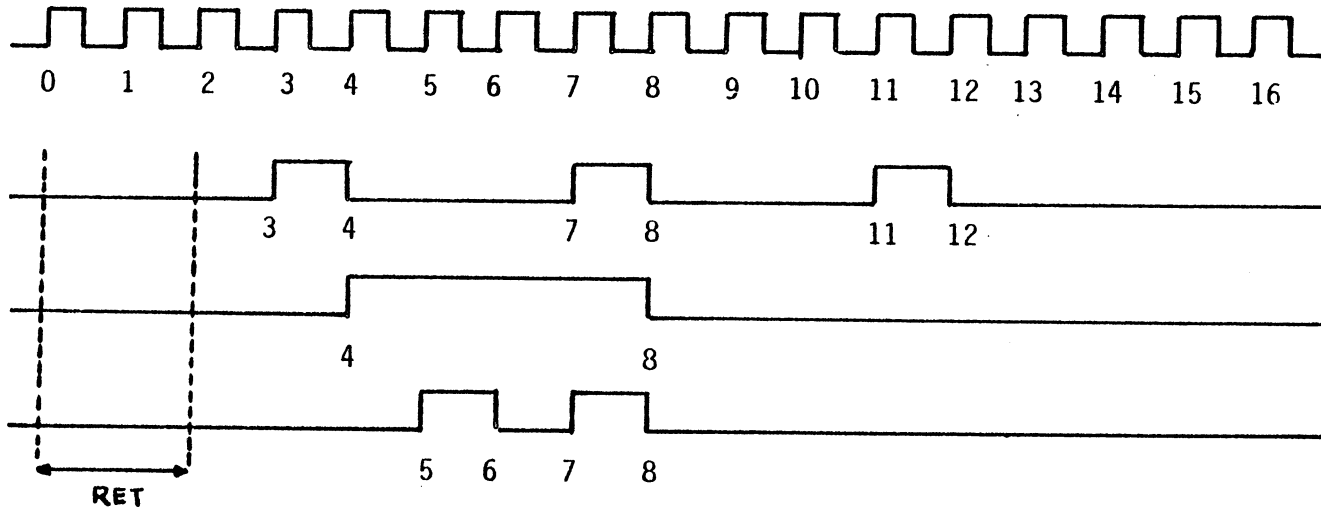


FIGURE 16

La description des signaux et des chronogrammes permet différentes opérations ; on peut combiner et ajouter :

- des chronogrammes élémentaires pour former des chronogrammes.
- des chrogrammes pour former d'autres chronogrammes, ceux-ci pouvant être paramétrés.

On précise dans une rubrique de description les instructions (donc le programme à exécuter) pendant lesquels on déclenche la mémoire signal ; on fait appel à différentes combinaisons de chronogrammes en donnant éventuellement les listes de valeurs de paramétrage. Pour un chronogramme paramétré GAPT générera autant de séquences de test qu'il y a de combinaisons de valeurs des parametres.

CONCLUSION

Le système de test proposé dans ce chapitre est un ensemble cohérent permettant le test de circuits au niveau logiciel et matériel.

Le système est extensible au test des circuits périphériques, voir le chapitre IV.

Cet ensemble forme un tout: générateur de programme de test, et machine de test. Il reste cependant possible de déterminer les séquences de test d'un circuit pour les appliquer par l'intermédiaire d'un testeur classique.

ANNEXE 1 AU CHAPITRE 3

(ESD82)

NOMENCLATURE DU TESTEUR

SCHEMA BLOC DU TESTEUR

NOMENCLATURE DU SYSTEME DE TEST

- Système informatique

carte microordinateur	iSBC 86/05
carte mémoire	iSBC 028A
carte contrôleur disque dur	iSBC 215
disque dur, disque souple	MTD P3450
carte contrôleur disque souple	iSBX 218
carte 24 entrées/sorties	2 x iSBX 350
logiciel	iRMX 86
console de visualisation	
imprimante	

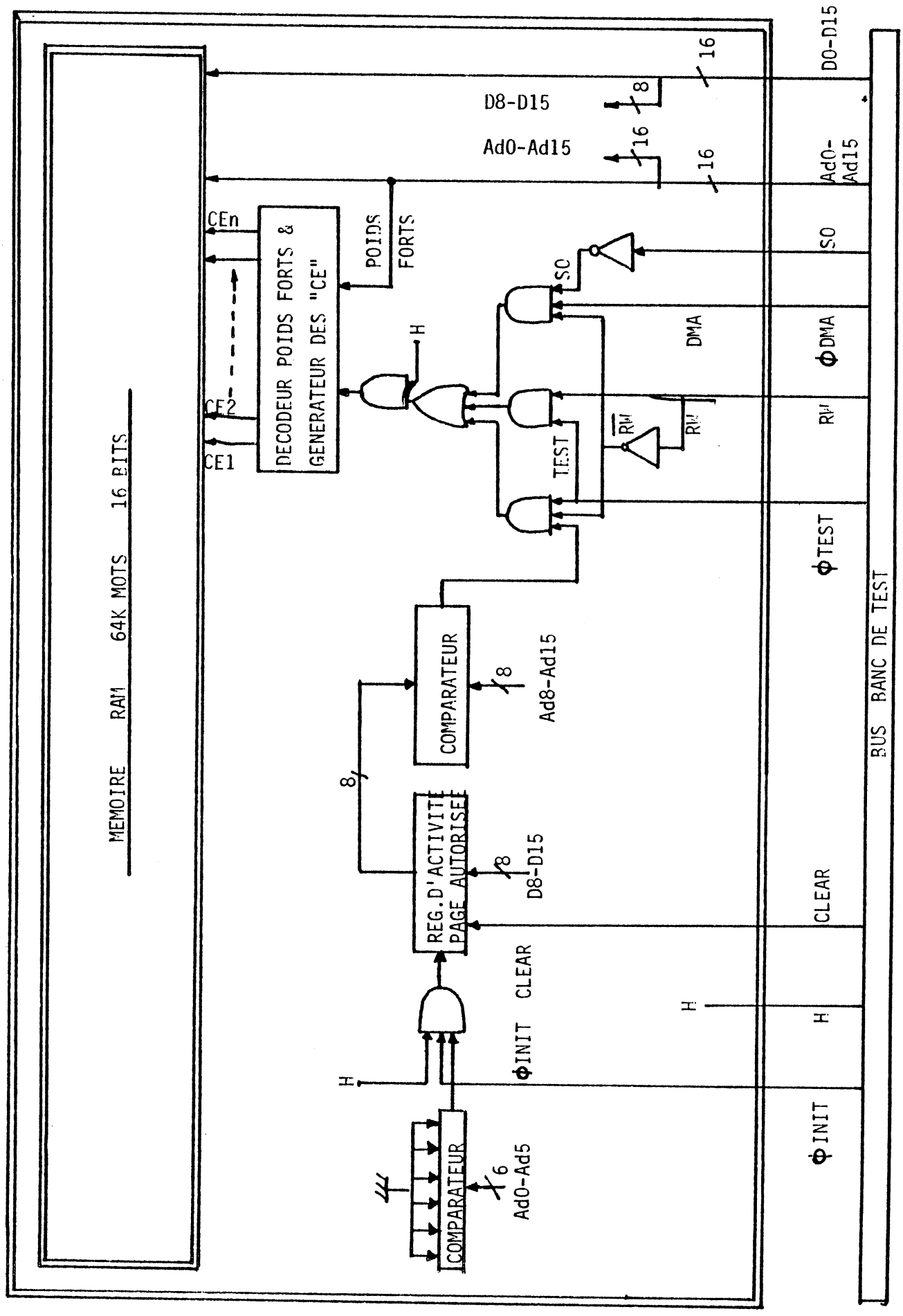
- Interface

carte interface gestion
carte interface DMA

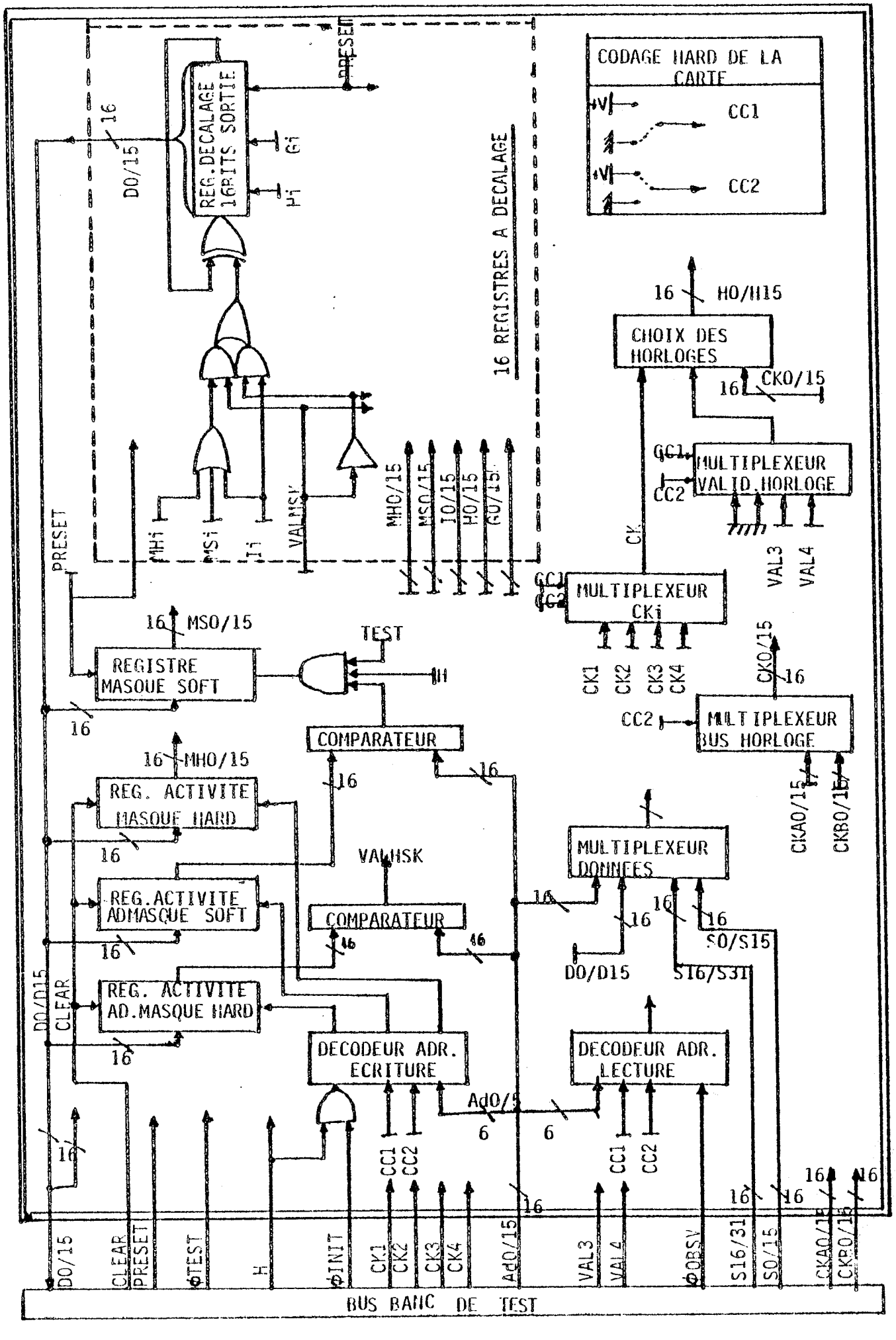
- Banc de test

carte mémoire programme	64K x 16 bits
carte mémoire signal	1K x 32 bits
carte de registre de signature	
carte du circuit sous test	

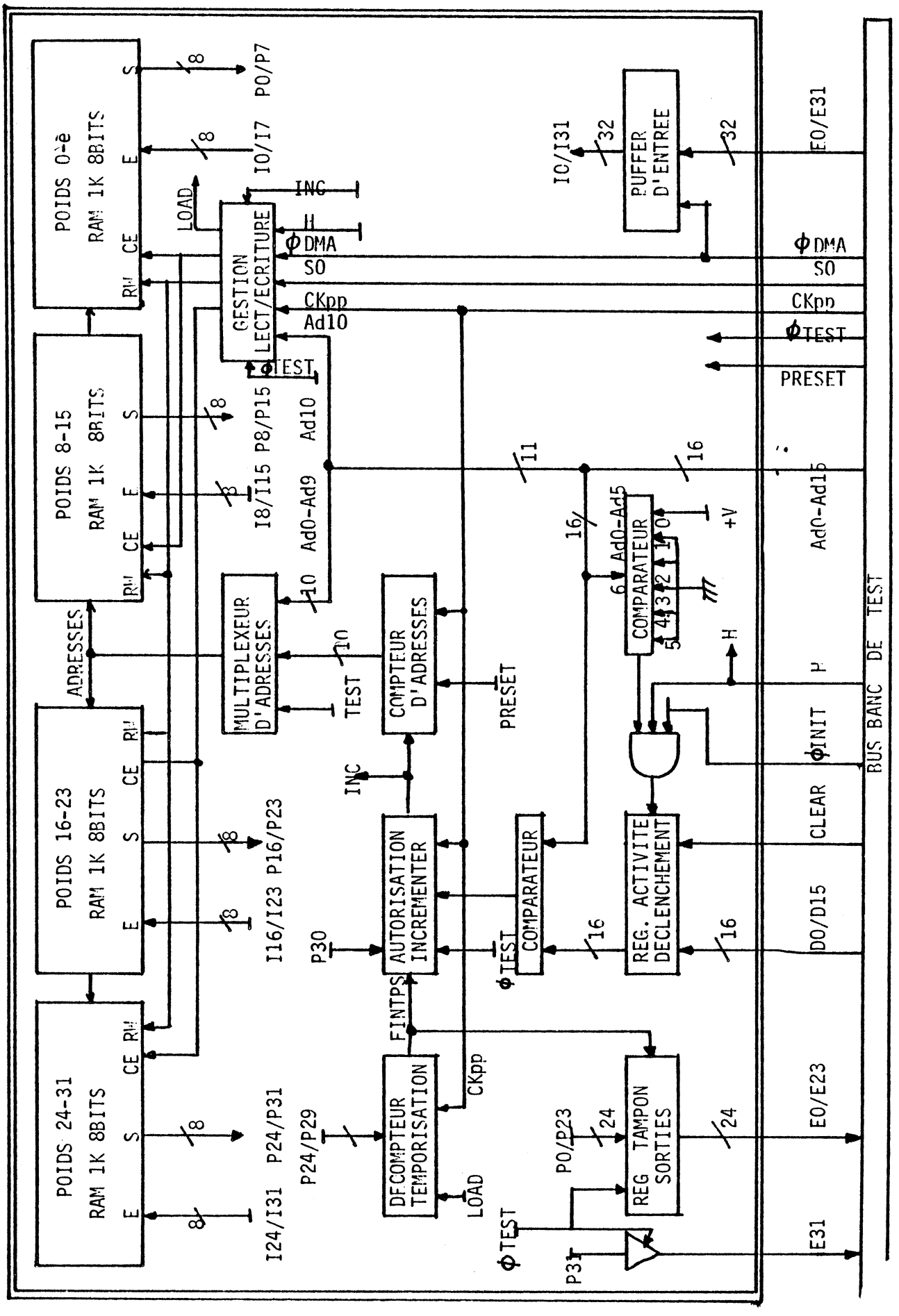
CARTE MEMOIRE PROGRAMME



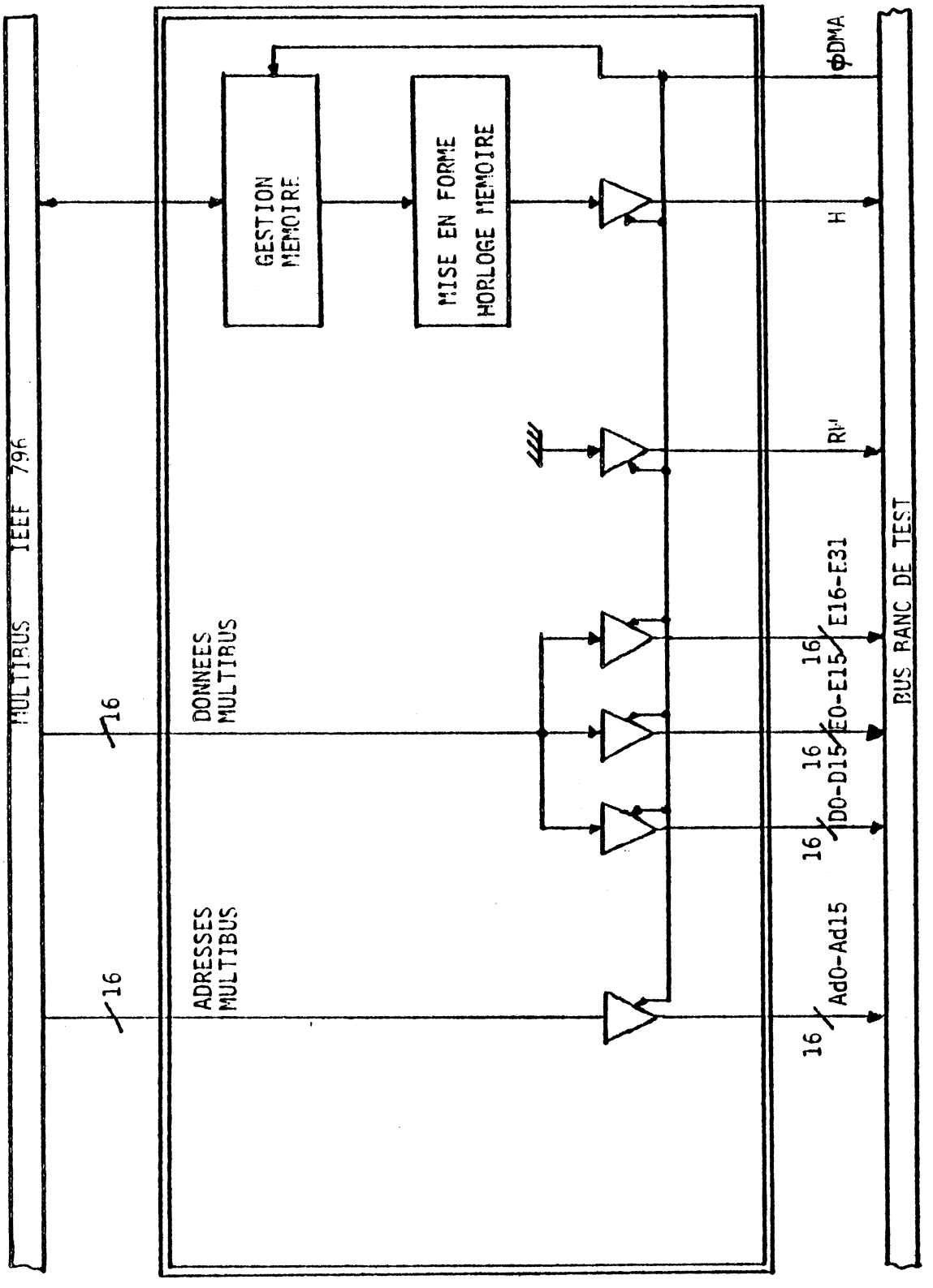
CARTE REGISTRES DE SIGNATURE



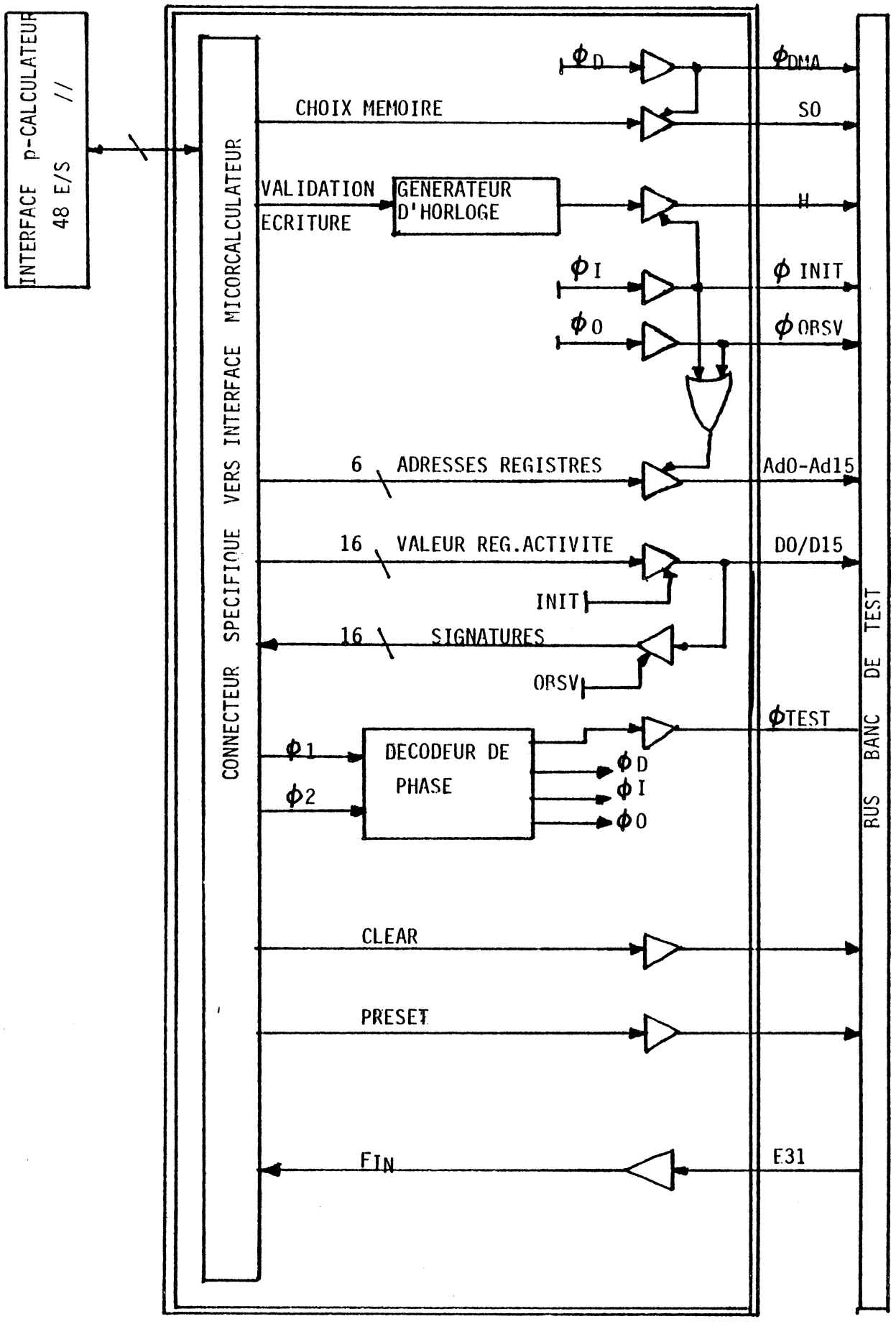
CARTE MEMOIRE SIGNAUX



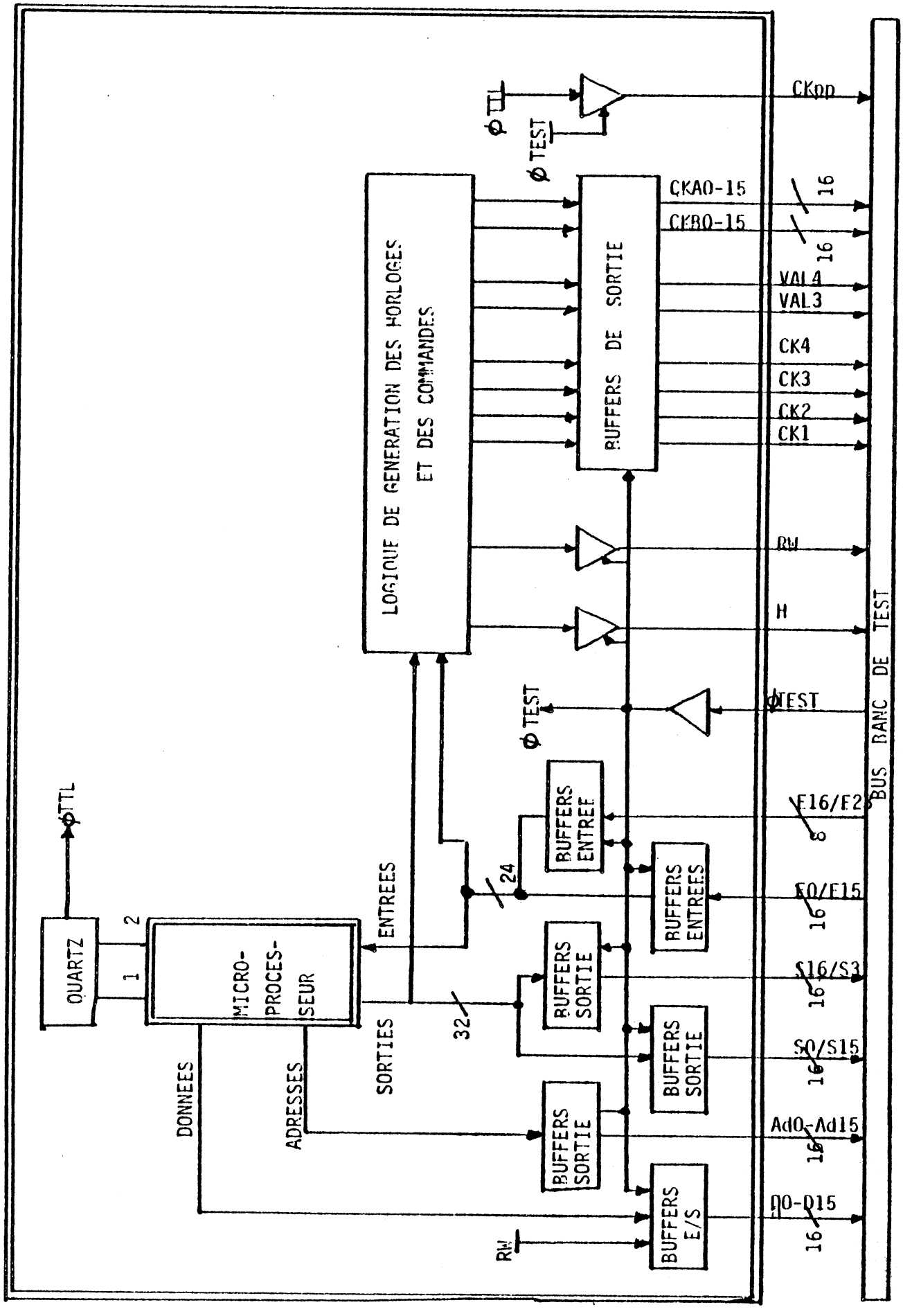
CARTE INTERFACE DMA



CARTE INTERFACE GESTION



CARTE SUPPORT DU MICROPROCESSEUR SOUS TEST



ANNEXE 2 DU CHAPITRE 3

LES SIGNAUX DU 68000 MOTOROLA

$\overline{DTACK}$: 1 : En général ce signal indique la fin d'un transfert de données

$\overline{BR}$: 1 : Ce signal indique qu'un circuit de l'environnement désire se rendre maître du bus.

$\overline{BGACK}$: 1 : C'est l'indicateur de prise du bus par un autre circuit.

IPL012 : 0 : Priorité du circuit interrompant.

$\overline{BERR}$: 1 : Indicateur d'erreur sur le bus.

$\overline{RES}$: 1 : Réinitialisation du processeur.

$\overline{HALT}$: 1 : Arrêt du processeur à la fin du cycle en cours.

$\overline{VPA}$: 1 : Signal de synchronisation avec les périphériques de la famille MC6800.

LES CHRONOGRAMMES ELEMENTAIRES

NOTATION: Ci-nom du signal:
i: numéro d'ordre du chronogramme i=1,...,n.

-BR-

C1-BR	BR(R*1+R*5)	FIGURE 23
C2-BR	BR(R*5+R*9)	FIGURE 24
C3-BR	BR(R*0+R*6)	FIGURE 25

-BGACK-

C1-BGACK	BGACK(R*4+R*7)	FIGURE 23
C2-BGACK	BGACK(R*6+R*11)	FIGURE 24
C3-BGACK	BGACK(R*5+R*8)	FIGURE 25

-BERR-

C1-BERR	BERR(R**4+R*7)	FIGURE 26
C2-BERR	BERR(R**2+R**5)	FIGURE 27

-HALT-

C1-HALT	HALT(R**2+R**9)	FIGURE 27
C2-HALT	HALT(R**2+R**7)	FIGURE 28

-IPL012-

C1-IPL	IPL(R** 1+R** τ)	τ VARIABLE
C2-IPL	IPL(1)	

-VPA-

C1-VPA	VPA(R**2+R** τ)	$10 < \tau < 18$
C2-VPA	VPA(R**13+R**19)	FIGURE 29

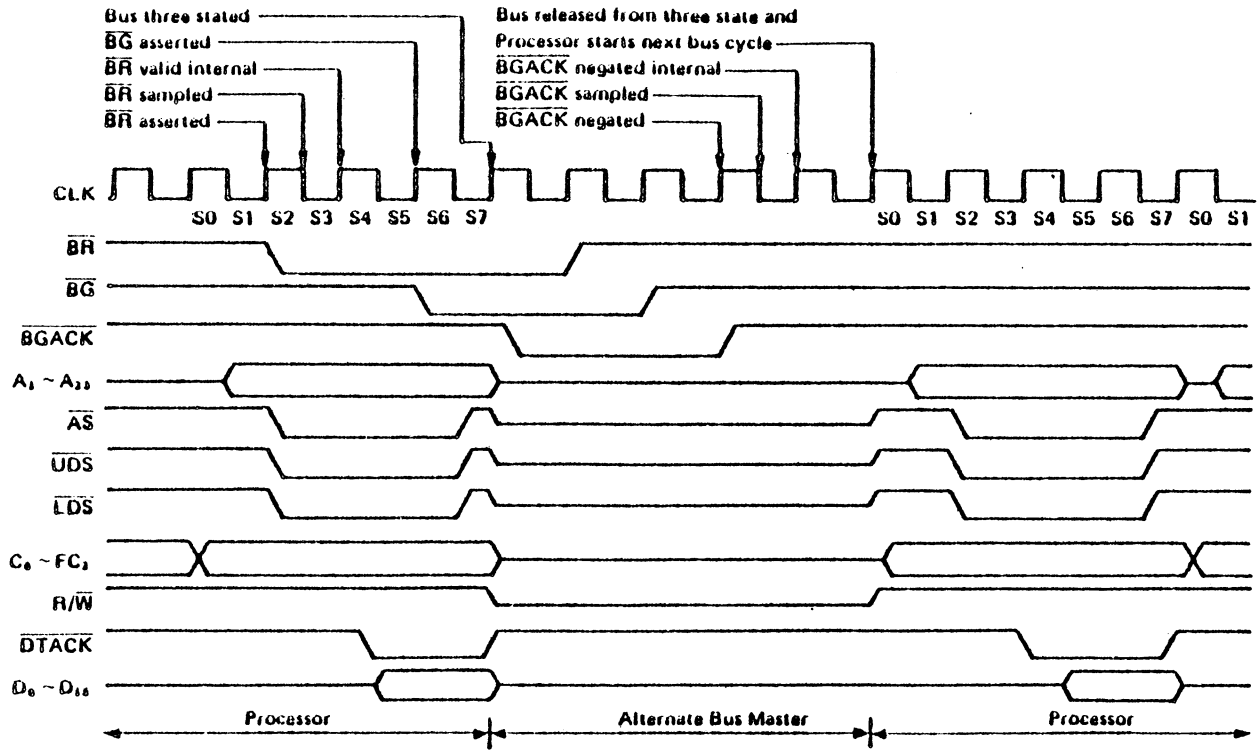


FIGURE 23

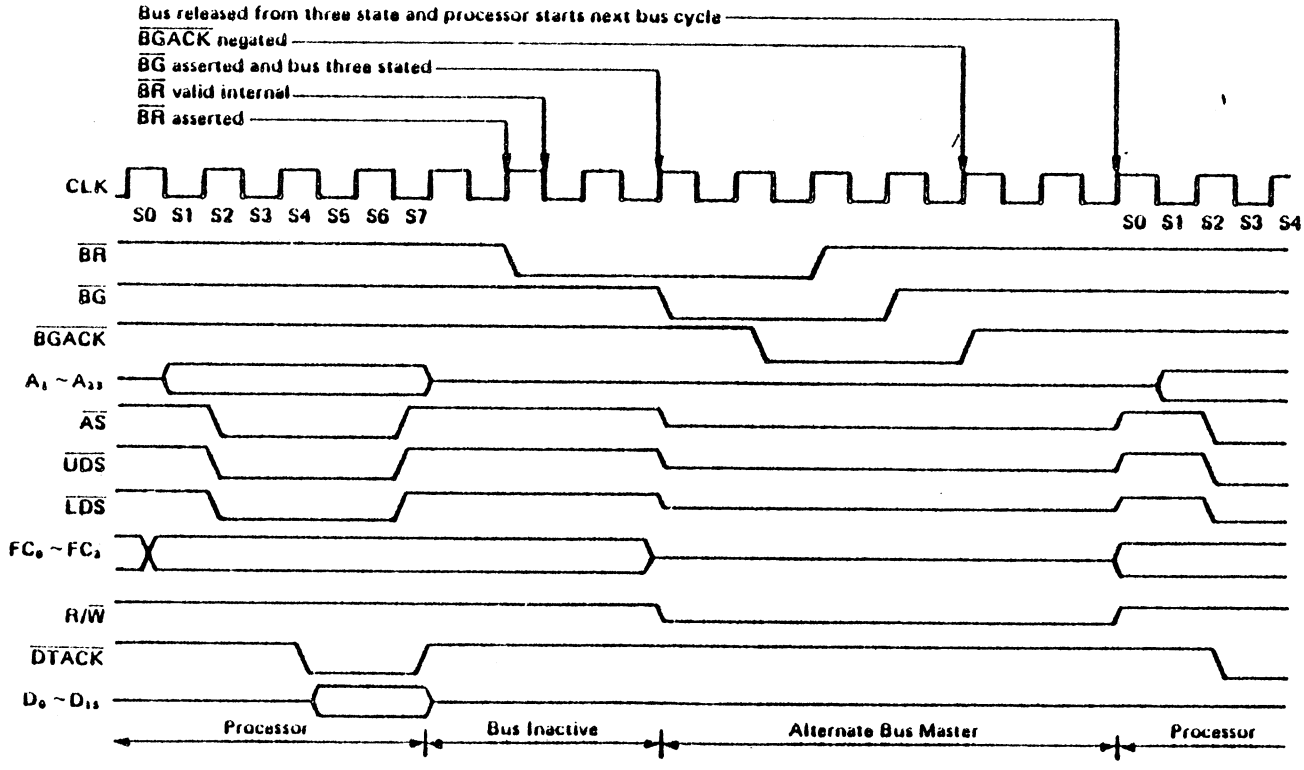


FIGURE 24

FIGURE 25

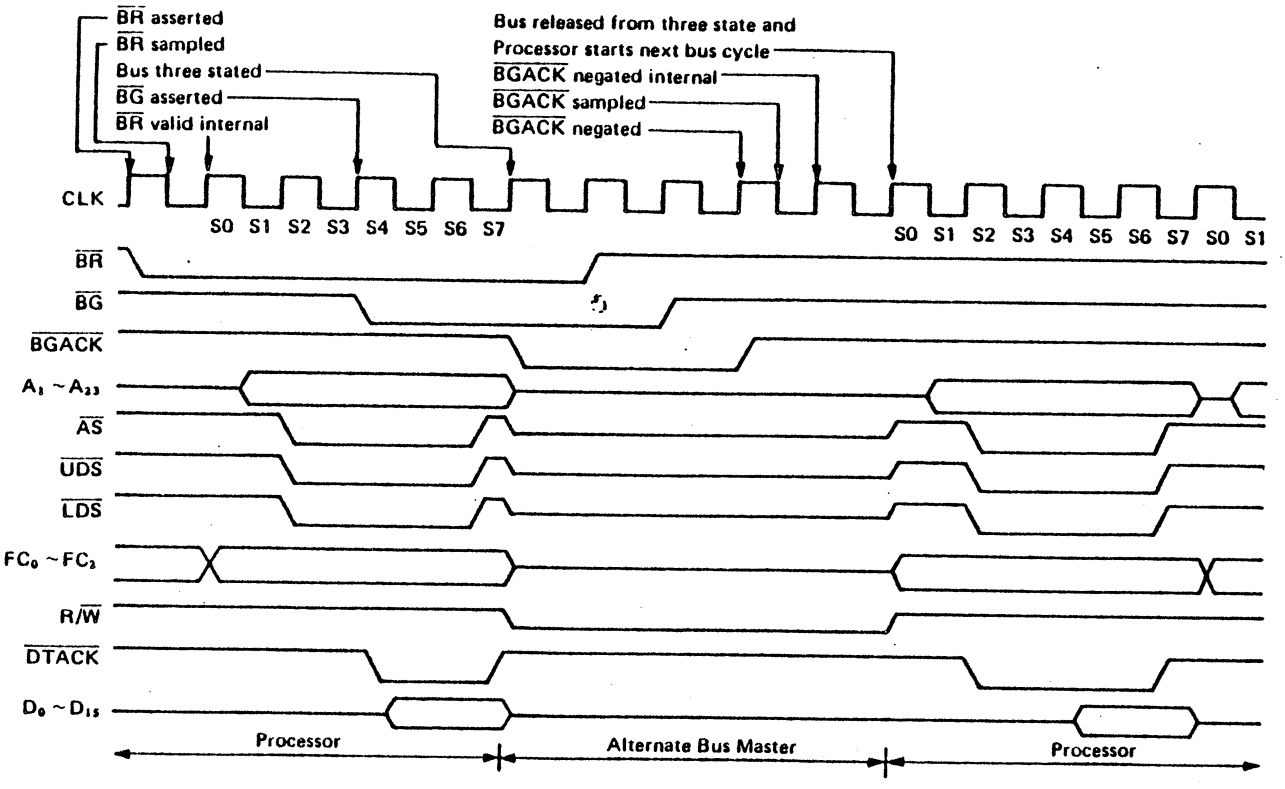
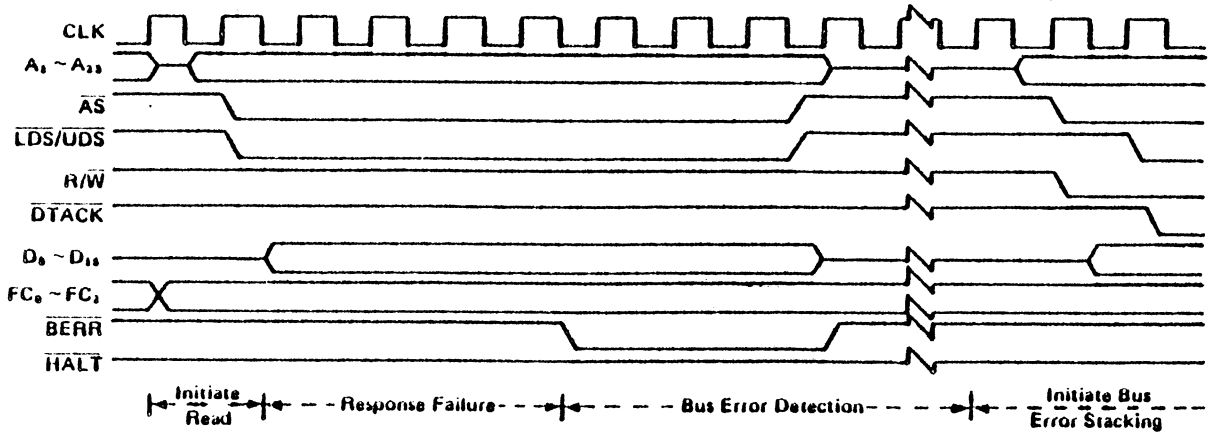

BR asserted
BR sampled
Bus three stated
BG asserted
BR valid internal
Bus released from three state and
Processor starts next bus cycle
BGACK negated internal
BGACK sampled
BGACK negated
CLK
S0 S1 S2 S3 S4 S5 S6 S7
S0 S1 S2 S3 S4 S5 S6 S7 S0 S1
BR
BG
BGACK
A1 ~A23
AS
UDS
LDS
FC0 ~FC2
R/W
DTACK
D0 ~D15
Processor
Alternate Bus Master
Processor

FIGURE 26



CHRONOGRAMME D'UNE ERREUR SUR UN BUS

CYCLE DE RE RUN DE BUS

FIGURE 27

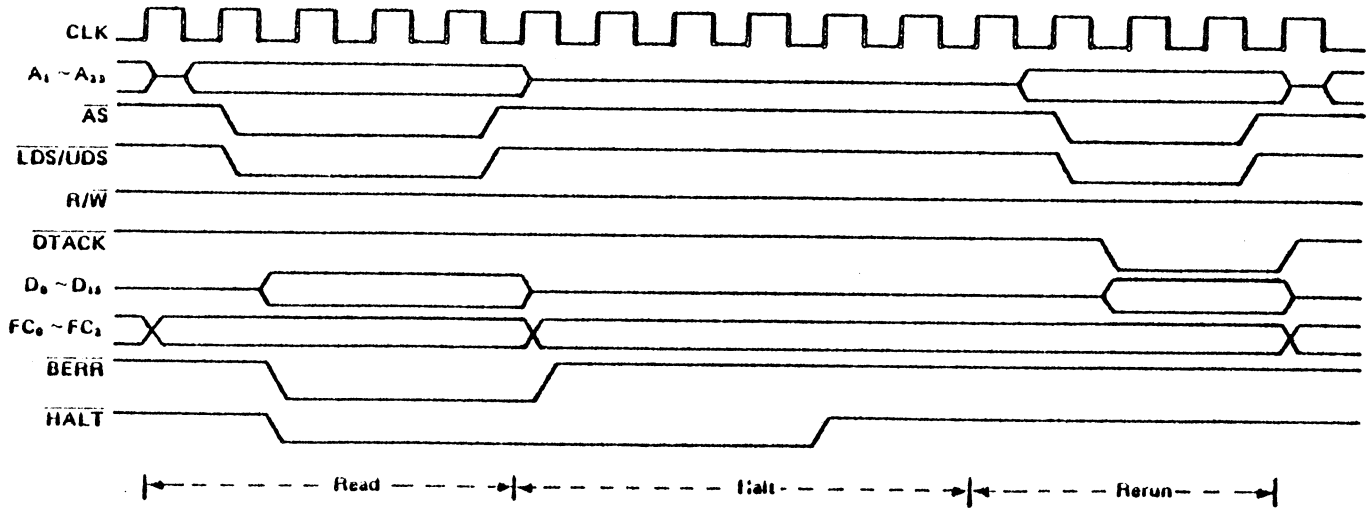
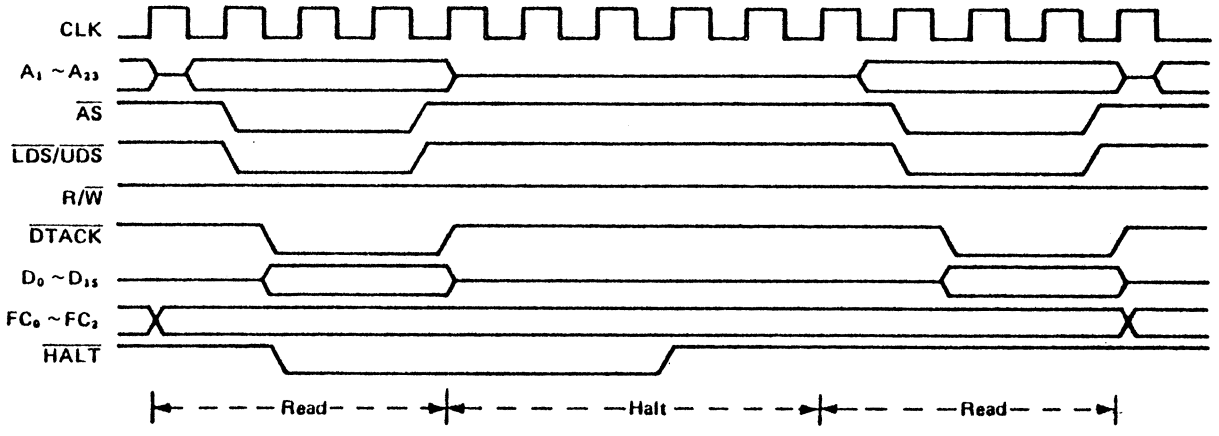
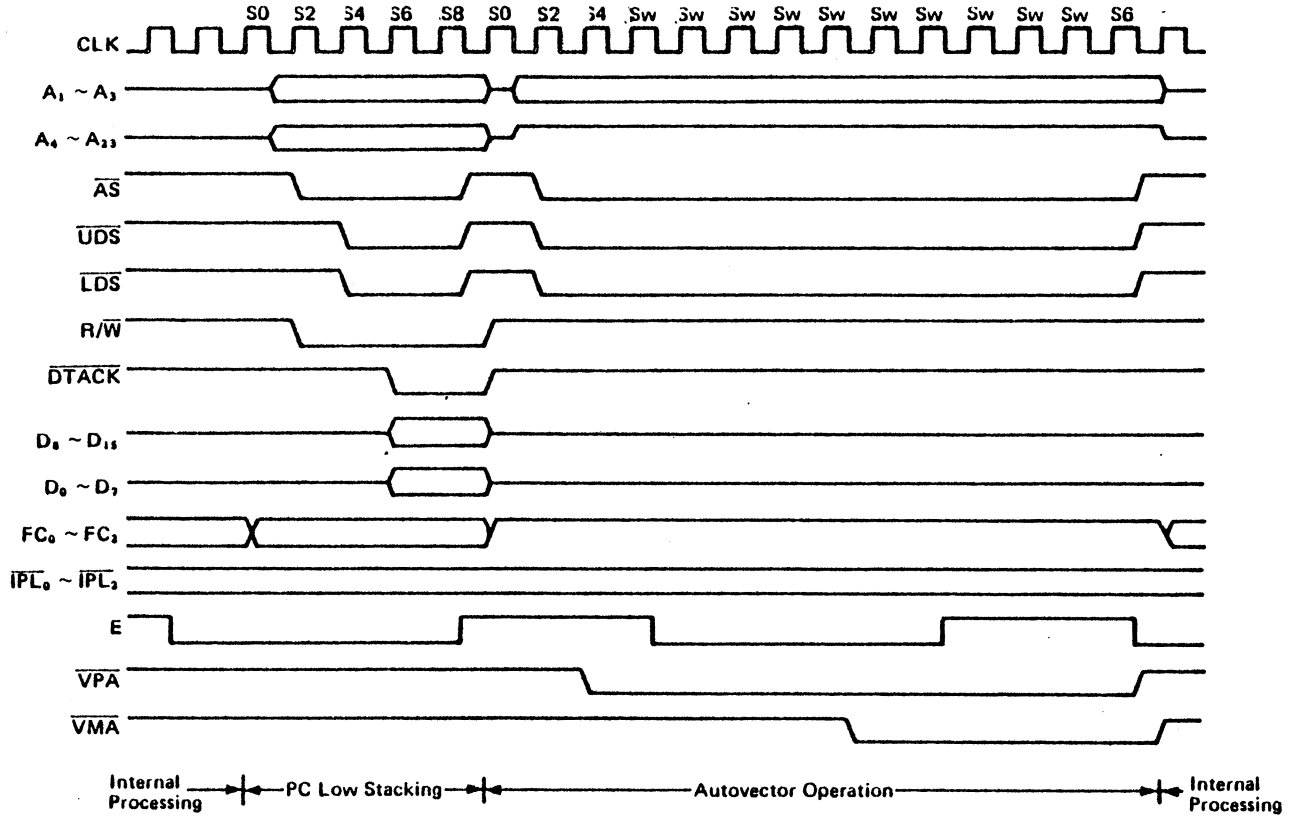


FIGURE 28



CHRONOGRAMME D'UN HALT

FIGURE 29



CHRONOGRAMME D'UNE AUTOVECTORISATION

CHRONOGRAMMES

- ARBITRE DE BUS

  = (C1-BR)+(C1-BGACK)

  = (C2-BR)+(C2-BGACK)

  = (C3-BR)+(C3-BGACK)

- ERREUR DE BUS

  = (C1-BERR)

- RERUN

  = (C2-BERR)+(C1-HALT)

- HALT

  = (C2-HALT)

- INTERRUPTION

  = (C1-IPL)

- PERIPHERIQUE

  = (C1-VPA)

- AUTOVECTPORISATION

  = (C2-VPA)+(C2-IPL)

CHAPITRE 4

INTRODUCTION

Actuellement le développement considérable des circuits périphériques de microprocesseur amène à se préoccuper de leur test. Ces dernières années des méthodes de test destinées aux microprocesseurs ont été largement développées en laissant le plus souvent dans l'ombre le problème des circuits périphériques.

La famille des circuits périphériques englobe une telle quantité de types de circuits qu'il est nécessaire pour conserver une bonne cohérence au niveau des méthodes de test à appliquer, d'effectuer une classification des circuits périphériques.

Nous restreignons donc dans un premier temps notre étude à des circuits directement commandables par le microprocesseur associé.

Dans ce chapitre, nous proposons un modèle de référence illustré par une description de type automate à deux niveaux ; la structure du test est basée sur un test de conformité et sur un test de balayage.

La mise en oeuvre du test se fait au moyen de l'équipement de test développé en collaboration avec Electronique Serge Dassault (MEY84). Le circuit sous test est lié à un microprocesseur de la même famille de circuit ; le microprocesseur dirige l'ensemble du test.

I - MODELE DE REFERENCE

Le modèle de référence des circuits périphériques sera constitué d'un graphe de contrôle a deux niveaux.

I - 1. RAPPELS (BEL79)(BEL81)

Le graphe de contrôle d'un automate est un graphe biparti. Les noeuds de ce graphe sont :

- des phases correspondant à la notion classique d'état ou de phase. Conventionnellement on fait figurer les opérations élémentaires

- des transitions étiquetées par des prédicats de transitions entre les états. Ces prédicats peuvent porter sur des ordres de haut niveau, sur des compte-rendus ou sur des fonctions temporelles.

Généralement les automates comportent une phase initiale d'attente et une phase finale.

I - 2. HIERARCHISATION DU MODELE

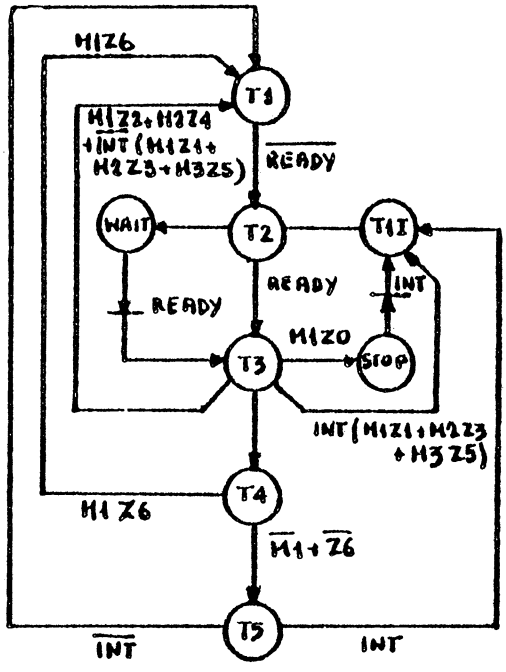
Le graphe général d'un circuit périphérique est complexe. On cherchera donc à partitionner ce graphe en un ensemble de sous-graphes: les macrophases.

Une macrophase est un graphe connexe, possédant une phase initiale et une phase finale uniques ; une macrophase se déroulera donc elle même comme un graphe de contrôle.

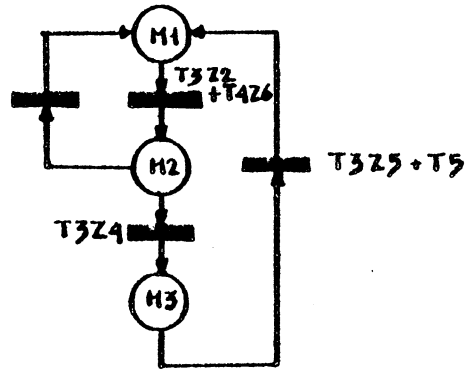
On peut alors introduire les deux niveaux de graphe du modèle de référence:

- Premier niveau : On considère le graphe dont les noeuds sont les macrophases.
- Deuxième niveau : On considère le graphe interne à une macrophase.

C.Piguet et al. ont utilisé une approche annalogue pour le séquenceur du microprocesseur 8008 INTEL. (PIG77)



GRAPHE DU GENERATEUR D'ETATS



GRAFCET DU GENERATEUR DE CYCLES

DECOMPOSITION DU SEQUENCEUR DU 8008 INTEL

FIGURE 1

Dans cette approche, instructions et fonctionnements asynchrones du circuit sont décomposés en états (macrophases). Un graphe de séquencement entre ces états correspond au graphe de séquencement entre macrophases. Un second graphe donne le séquencement en cycle des instructions et des fonctionnements asynchrones.

I - 3. GRAPHE DE SEQUENCEMENT DES MACROPHASES

I.3.1 Ensemble des macrophases

Dans le cas des circuits périphériques, une macrophase est déterminée par :

- les modes de fonctionnement du circuit.
- des indicateurs autorisant l'activation d'opérateurs spécifiques du circuit comme l'indicateur d'autorisation des interruptions.

I.3.2 Les transitions

Les transitions sont étiquetées par des prédicats de changements de macrophases portant sur les signaux et mots de commandes dont les valeurs sont stockées dans un registre de commande.

Tout changement de macrophase se fait par modification de ce registre de commande.

Exemple : Soit un circuit possédant deux modes de fonctionnement : entrée d'octets et sortie d'octets et une logique d'interruption indépendante.

M0 M1 définit le mode de fonctionnement.
I valide la logique d'interruption.

Soit le mot d'état

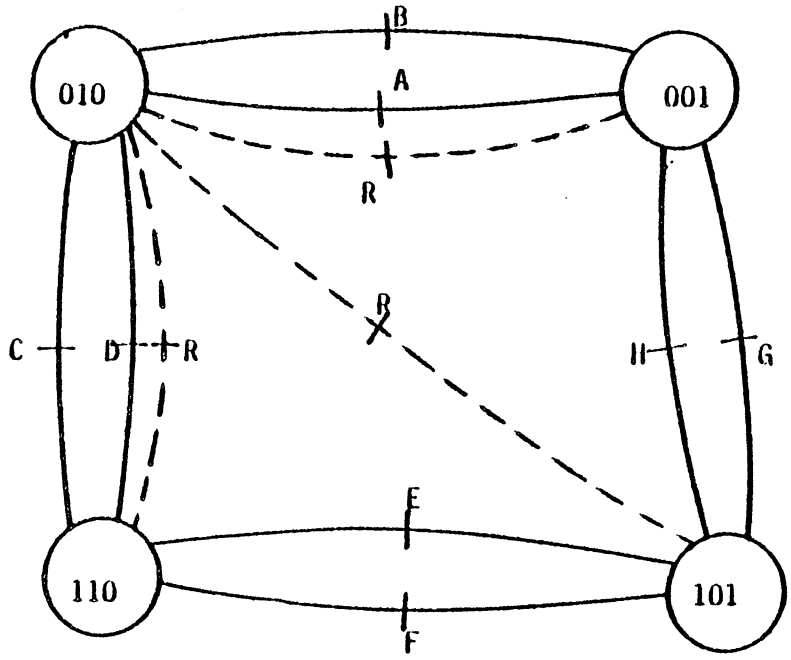
I	M0	M1

110	mode entrée d'octet	interruption autorisée
101	mode sortie d'octet	interruption autorisée
010	mode entrée d'octet	interruption inhibée
001	mode sortie d'octet	interruption inhibée

L'état premier du circuit est 010

Les commandes liées aux interruptions sont distinctes de celles liées aux modes de fonctionnement.

La transition d'un état à un autre est assurée par un ordre de haut niveau: un mot de commande entraînant un changement du mot d'état.



GRAPHE DES ETATS DU CIRCUIT

FIGURE 2

Après une réinitialisation le circuit est en son état premier 010 .

B-F	passage en mode entrée
A-E	passage en mode sortie
C-G	autorisation des interruptions
D-H	inhibition des interruptions
R	réinitialisation

I - 4. GRAPHE DE SEQUENCEMENT DES PHASES

Pour chaque macrophase un graphe de séquencement des phases précise l'évolution du circuit périphérique. Une phase est caractérisée par les actions effectuées au cours de cette phase (interprétation). Il s'agit de transfert de données ou d'affectation de valeurs aux commandes. Les prédicats de transition portent sur les signaux et mots de contrôle émis par le microprocesseur. Un état initial unique est activé dans ce graphe quand la macrophase correspondante est activée dans le graphe de séquencement des macrophases.

Exemple :

FIGURE 2 BIS

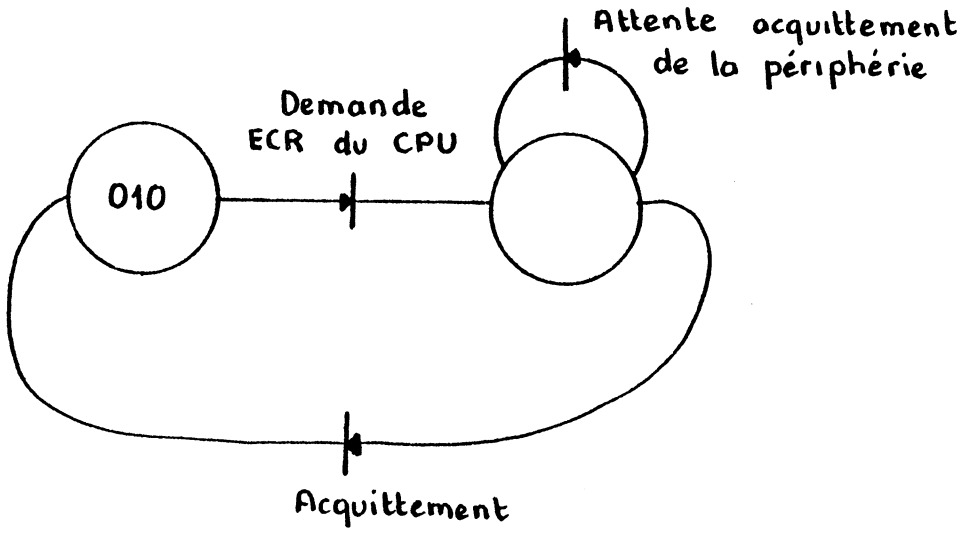
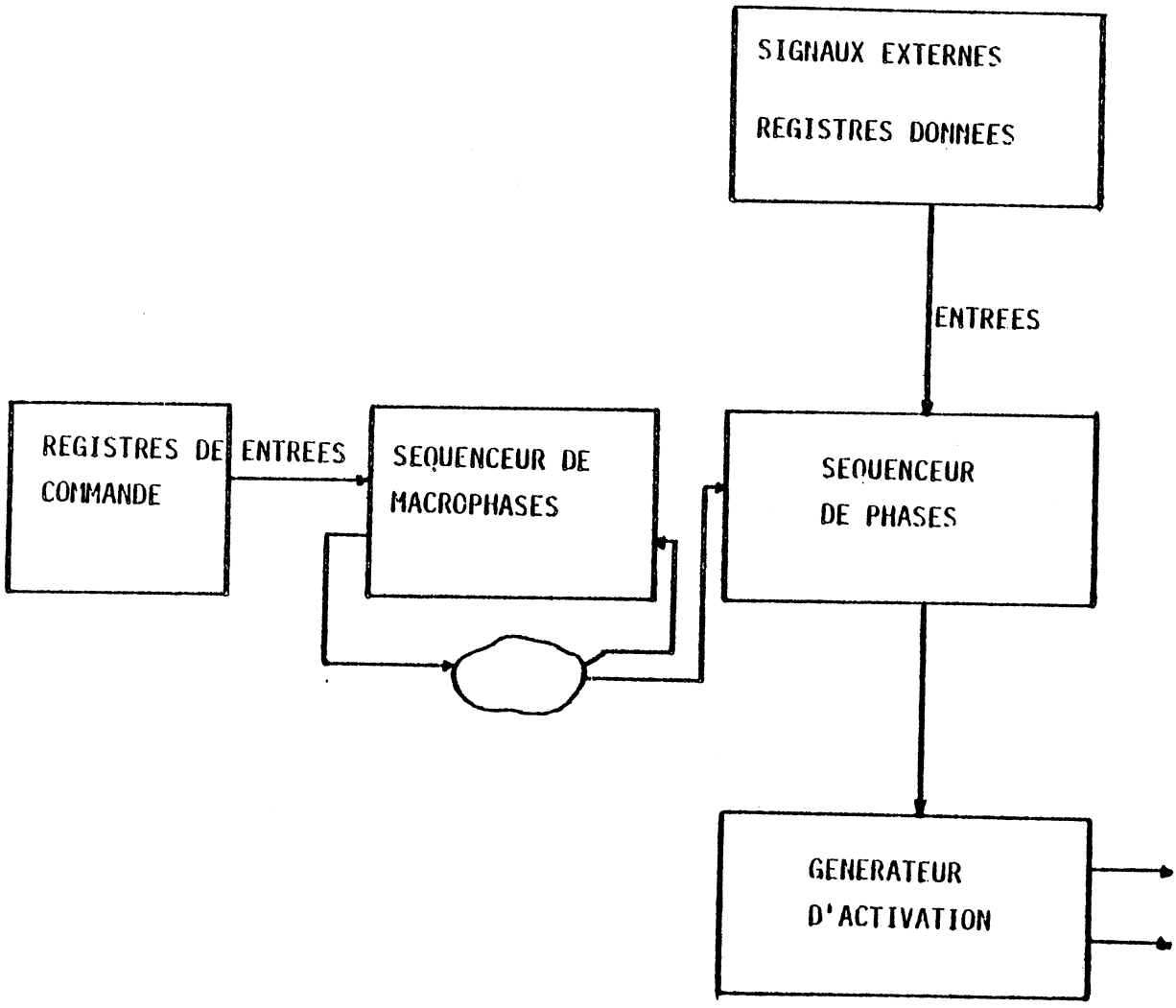


FIGURE 3


SIGNAUX EXTERNES
REGISTRES DONNEES
ENTREES
REGISTRES DE COMMANDE
ENTREES
SEQUENCEUR DE MACROPHASES
SEQUENCEUR DE PHASES
GENERATEUR D'ACTIVATION

I - 5. TEMPORISATION DU MODELE

La notion de temporisation de modèle concerne les transitions étiquetées par des prédicats. Dans la description temporelle de ces prédicats on distingue deux cas :

- Ces prédicats sont fournis par la liaison microprocesseur ou sont image fonctionnelle du déroulement d'un ensemble d'instructions. Il n'est pas nécessaire d'assurer la description temporelle de ces prédicats.

- Ces prédicats sont fournis par des organes externes. On effectue alors une description des chronogrammes à appliquer au circuit suivant le modèle développé au chapitre3. Ce modèle permet d'obtenir les données nécessaires à la simulation de ces fonctions temporelles, compte-rendus ou ordre de haut niveau.

I - 6. EXEMPLE D'APPLICATION

REFERENCE DU CIRCUIT : ZILOG Z8420-PIO (ZIL83)

C'est un circuit d'entrée-sortie parallèle de données avec une gestion indépendante des interruptions. On considère dans tout ce qui suit uniquement le port A d'échanges.

Dans le schéma récapitulatif donné FIGURE 4 on précise la signification des entrées.

ENTREES DU SEQUENCEUR DE MACROPHASES

$\overline{BA}$ Signal de sélection du circuit d'échange (A ou B).

$\overline{CD}$ Signal indicateur du type de transfert (commandes ou données).

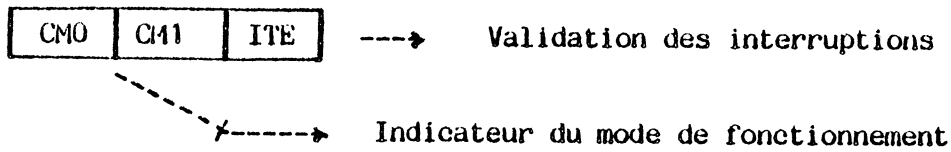
$\overline{CE}$ Validation du circuit.

$\overline{M1}$ Signal de synchronisation du circuit.

$\overline{IORQ}$ Signal de demande d'entrée-sortie.

$\overline{RD}$ Indicateur d'une lecture effectuée par le microprocesseur associé.

Rc Registre de commande.



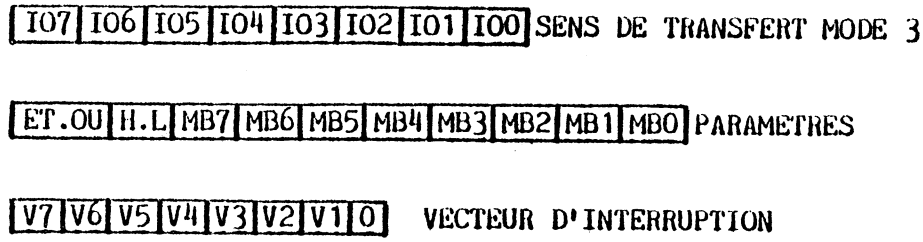
ENTREES DU SEQUENCEUR DE PHASE

$\overline{CD}$ Signal indicateur du type de transfert.

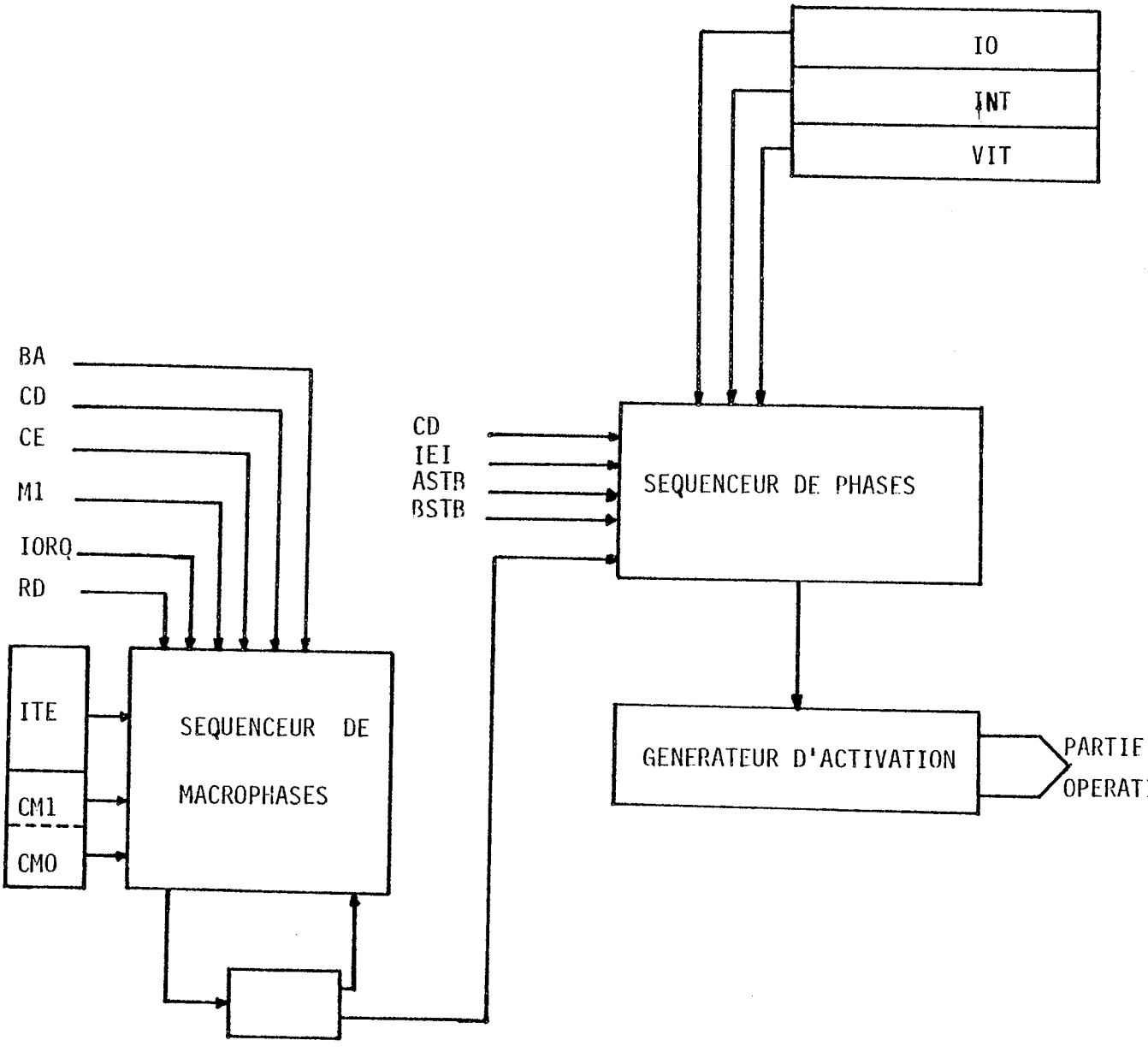
IEI Signal de hiérarchisation des interruptions.

$\overline{ASTB}$ } Signal d'acquittement de l'environnement dédié
$\overline{BSTB}$ } au port A ou au port B.

Registres de données



SCHEMA RECAPITULATIF



Pour des commodités d'écriture dans le graphe des macrophases on remplace certains ensembles de transitions par une transition unique.

$$R=(M1=0)\hat{}((RD=1)v(IORQ=1))$$

La transition CMi remplace l'ensemble de transitions suivant :

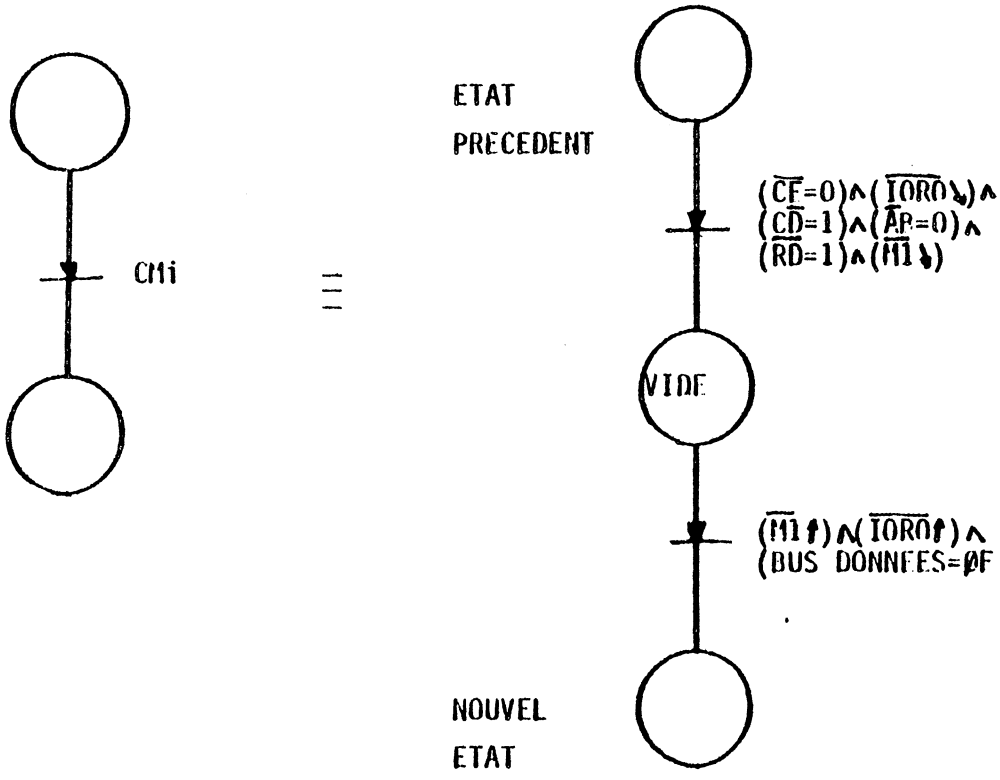


FIGURE 5

La transition ITE remplace l'ensemble de transitions suivant :

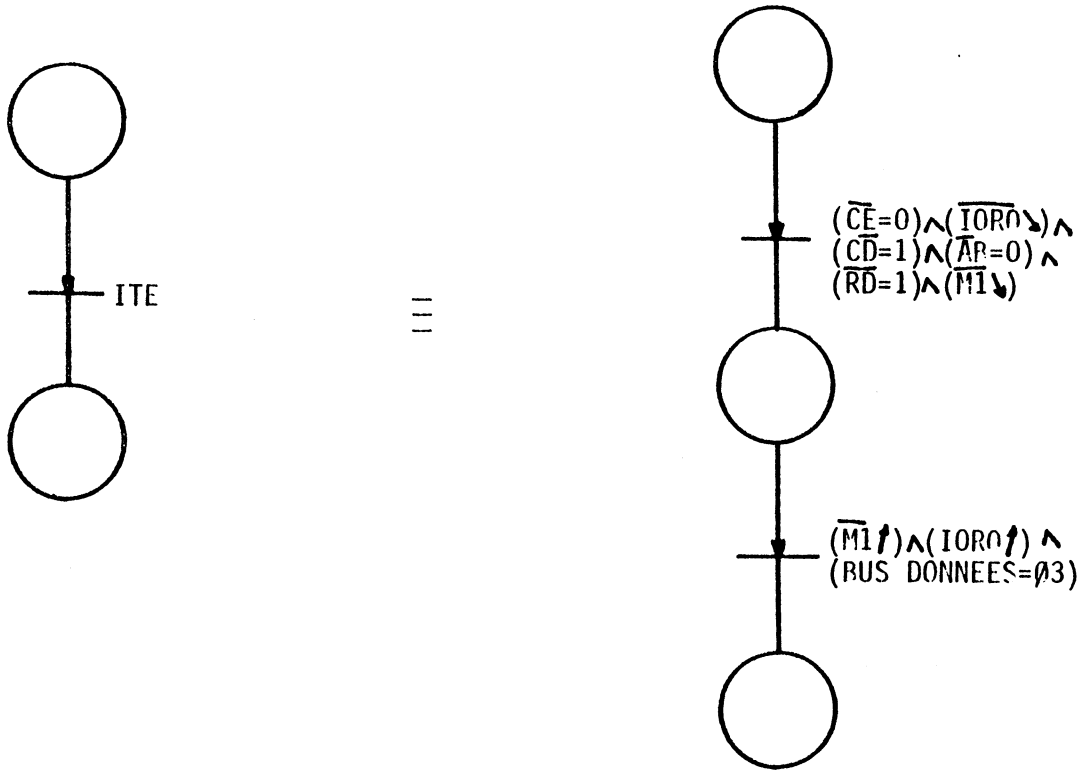
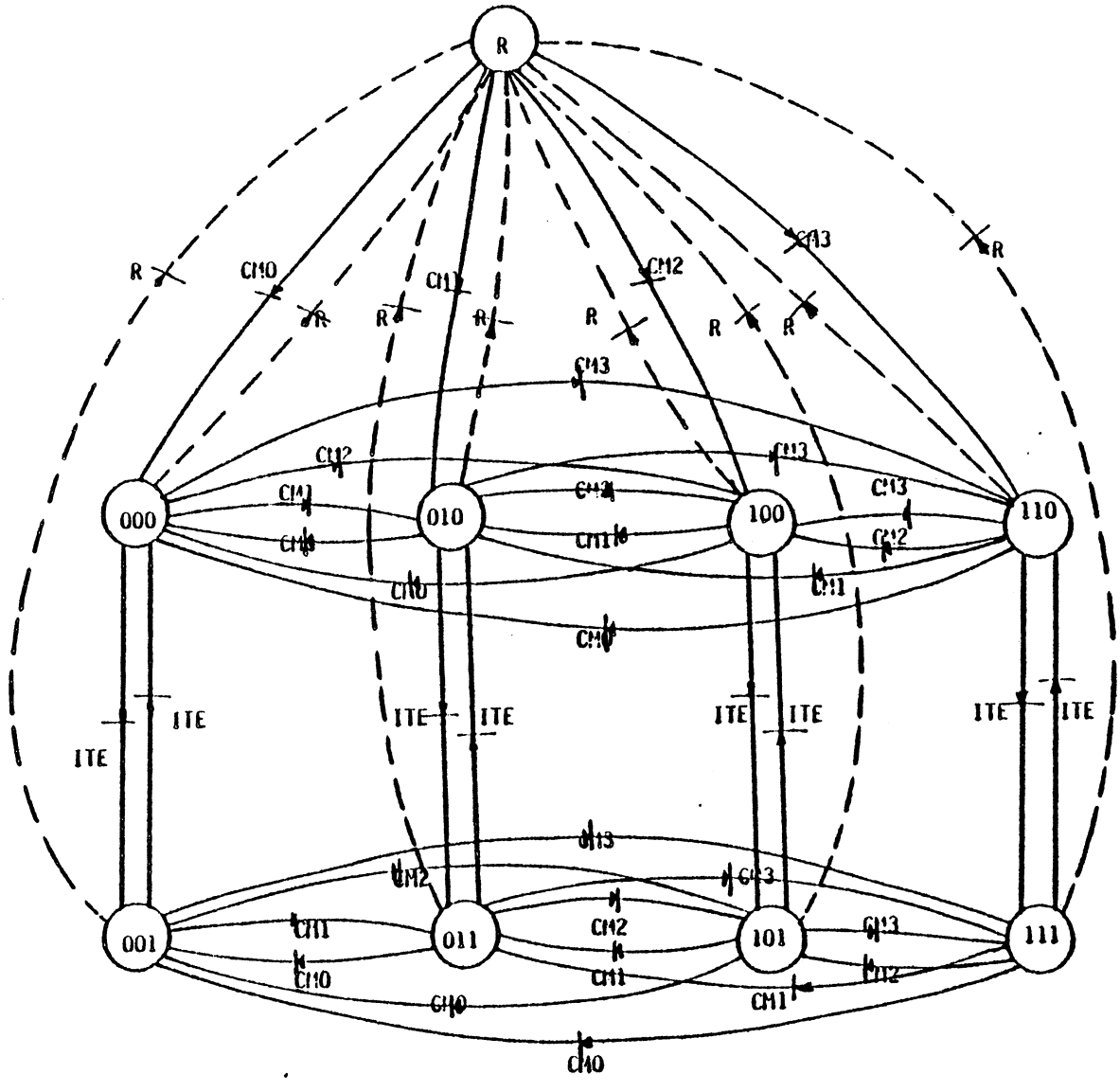


FIGURE 6

GRAPHE DES MACROPHASES: NIVEAU 1

FIGURE 7



Les places correspondent aux états définis par le registre de commandes

ITE	CM0	CM1

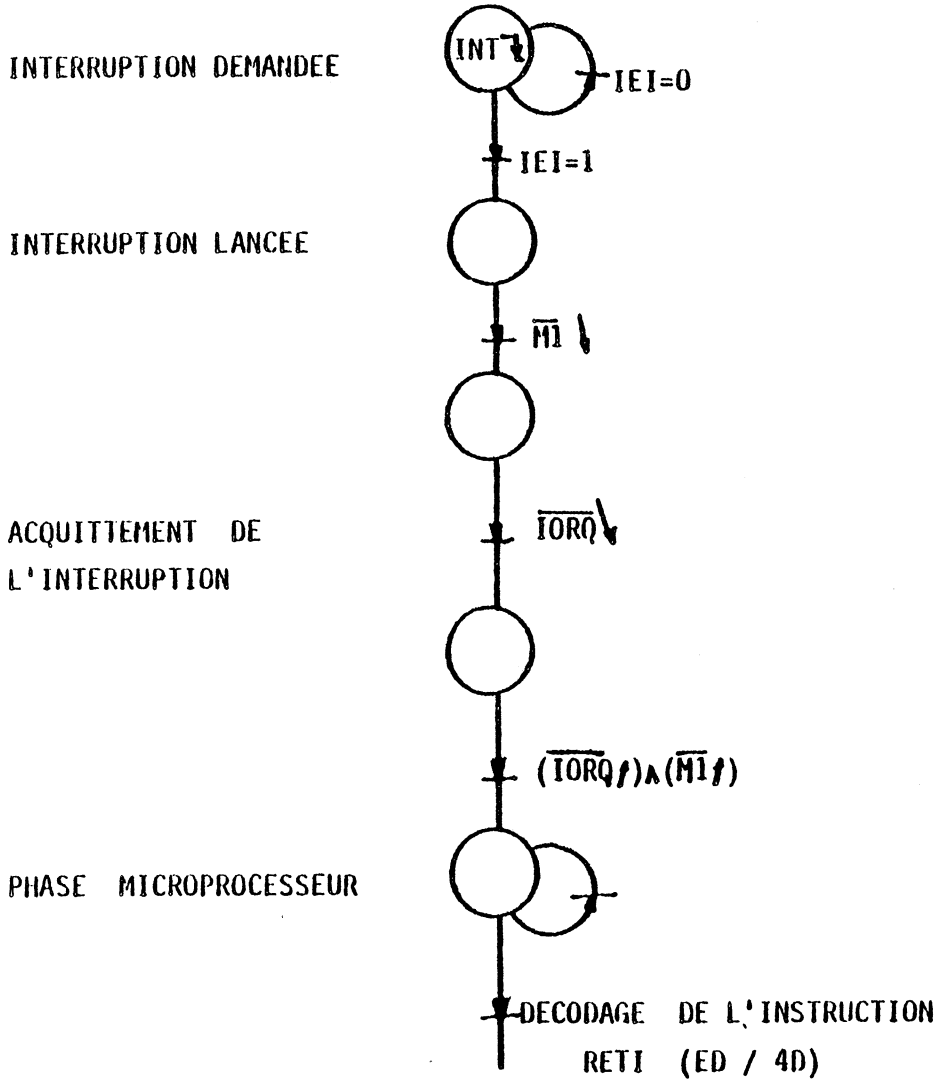
GRAPHE DES PHASES : (SAD81)

A chaque mode de fonctionnement correspond deux macrophases, l'une sans génération d'interruption l'autre avec génération d'interruption. La génération d'interruption est automatique en mode 0,1,2 ; elle est calculée en mode 3.

On propose dans un premier temps le graphe des phases correspondant à une interruption générée automatiquement.

Graphe d'interruption.

FIGURE 8


INTERRUPTION DEMANDEE
INT
IEI=0
IEI=1
INTERRUPTION LANCEE
M1
ACQUITTEMENT DE
L'INTERRUPTION
IORQ
(IORQ↓)∧(M1↓)
PHASE MICROPROCESSEUR
DECODAGE DE L'INSTRUCTION
RETI (ED / 4D)

A des fins de simplicité nous remplacerons ce graphe par :

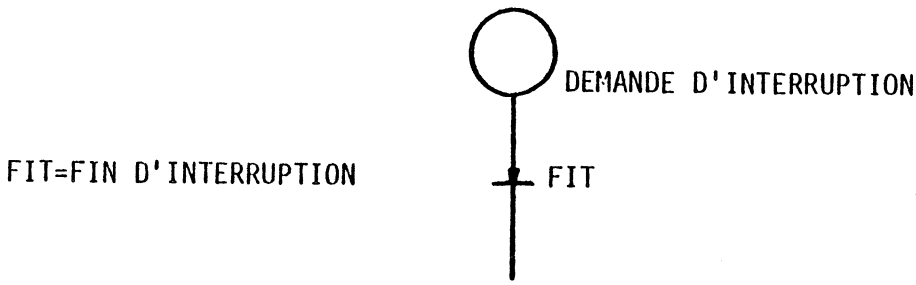


FIGURE 9

Les graphes des phases ainsi définis correspondant aux modes de fonctionnement sont donnés aux figures 10 à 16.

On déduit le graphe avec interruption du graphe sans interruption pour un mode de fonctionnement donné en insérant le graphe des interruptions.

Graphes des phases du mode sortie d'octet. MODE 0

000 code du mode 0 sans interruption
001 code du mode 0 avec interruption

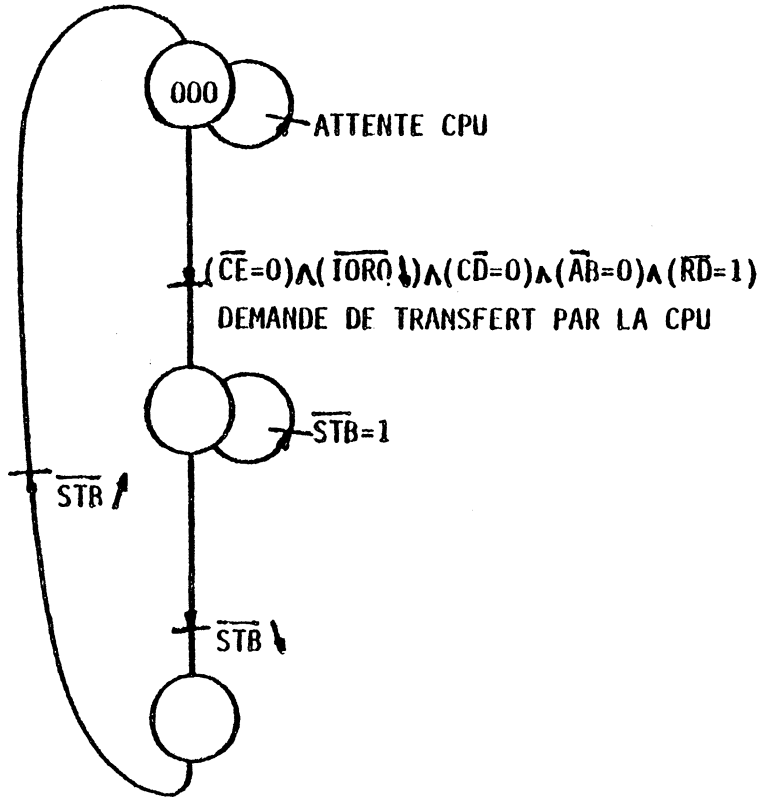
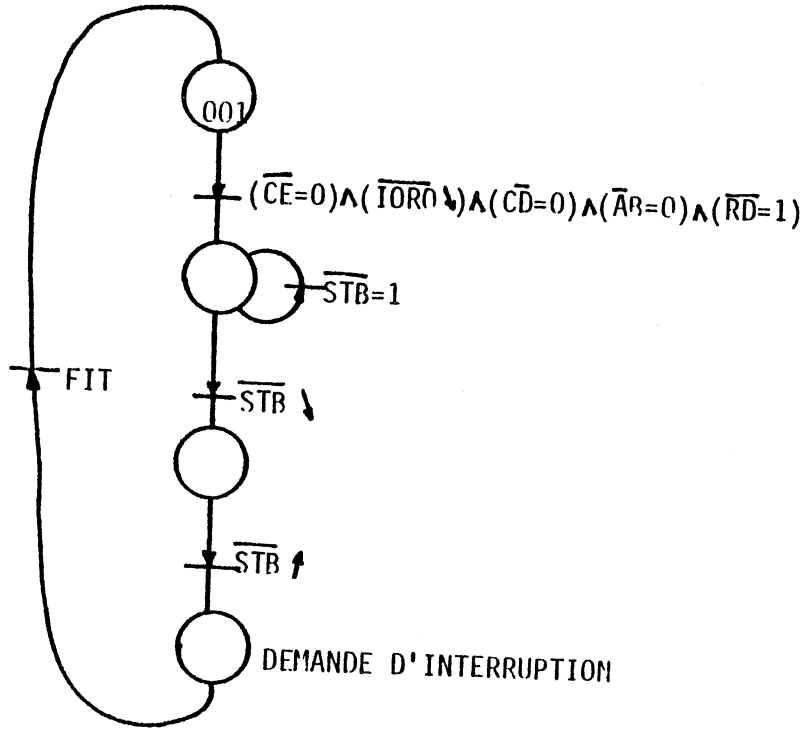


FIGURE 10

FIGURE 11


001
$(\overline{CE}=0)\wedge(\overline{IORQ}\searrow)\wedge(\overline{CD}=0)\wedge(\overline{AB}=0)\wedge(\overline{RD}=1)$
$\overline{STB}=1$
FIT
$\overline{STB}\searrow$
$\overline{STB}\nearrow$
DEMANDE D'INTERRUPTION

Graphes des phases du mode entrée d'octet. MODE 1

010 code du mode 1 sans interruption
011 code du mode 1 avec interruption

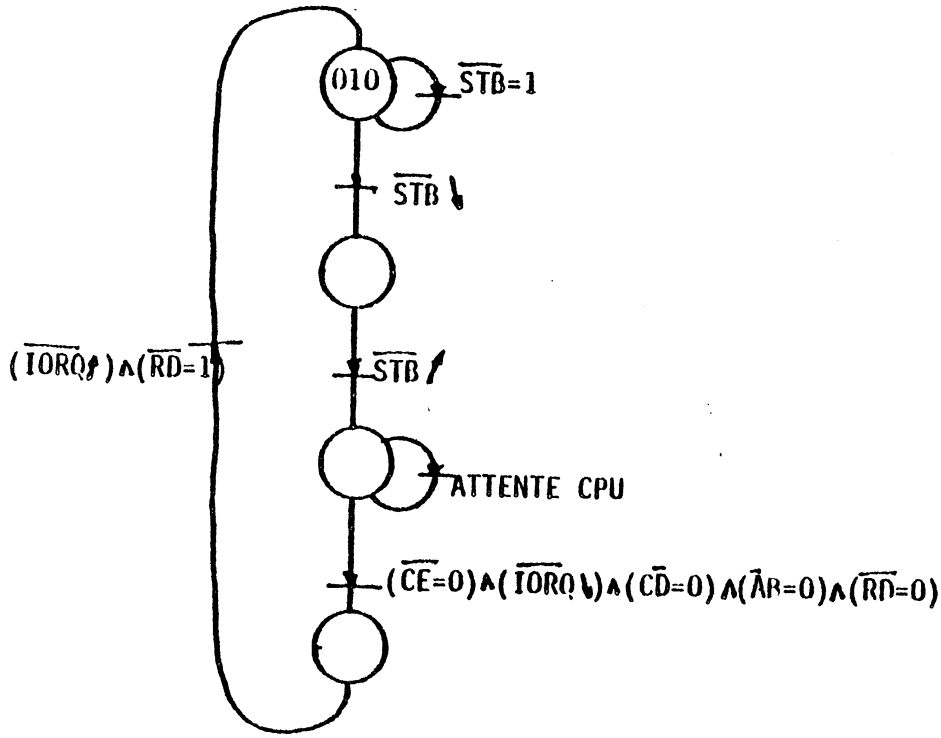
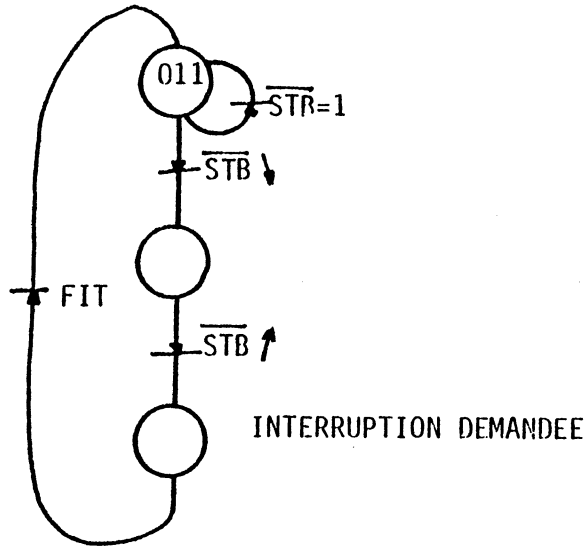


FIGURE 12

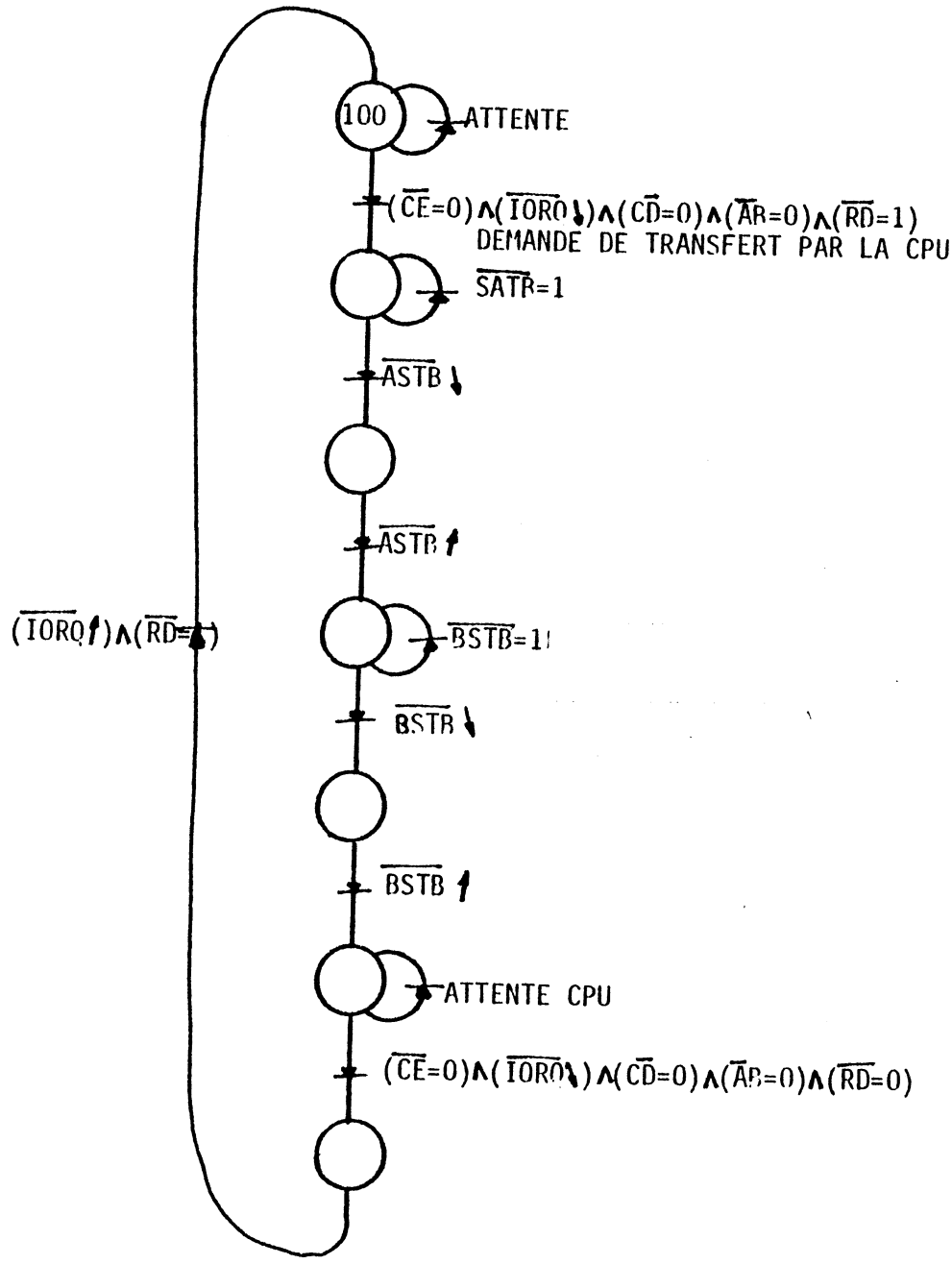
FIGURE 13


011
$\overline{STB}$=1
$\overline{STB}$
FIT
$\overline{STB}$
INTERRUPTION DEMANDEE

Graphes des phases du mode sortie d'octet. MODE 2

100 code du mode 2 sans interruption
101 code du mode 2 avec interruption

FIGURE 14


100
ATTENTE
$(\overline{CE}=0)\wedge(\overline{IORQ}\downarrow)\wedge(\overline{CD}=0)\wedge(\overline{AR}=0)\wedge(\overline{RD}=1)$
DEMANDE DE TRANSFERT PAR LA CPU
$\overline{SATR}=1$
$\overline{ASTB}\downarrow$
$\overline{ASTB}\uparrow$
$(\overline{IORQ}\uparrow)\wedge(\overline{RD}=1)$
$\overline{BSTB}=1$
$\overline{BSTB}\downarrow$
$\overline{BSTB}\uparrow$
ATTENTE CPU
$(\overline{CE}=0)\wedge(\overline{IORQ}\downarrow)\wedge(\overline{CD}=0)\wedge(\overline{AR}=0)\wedge(\overline{RD}=0)$

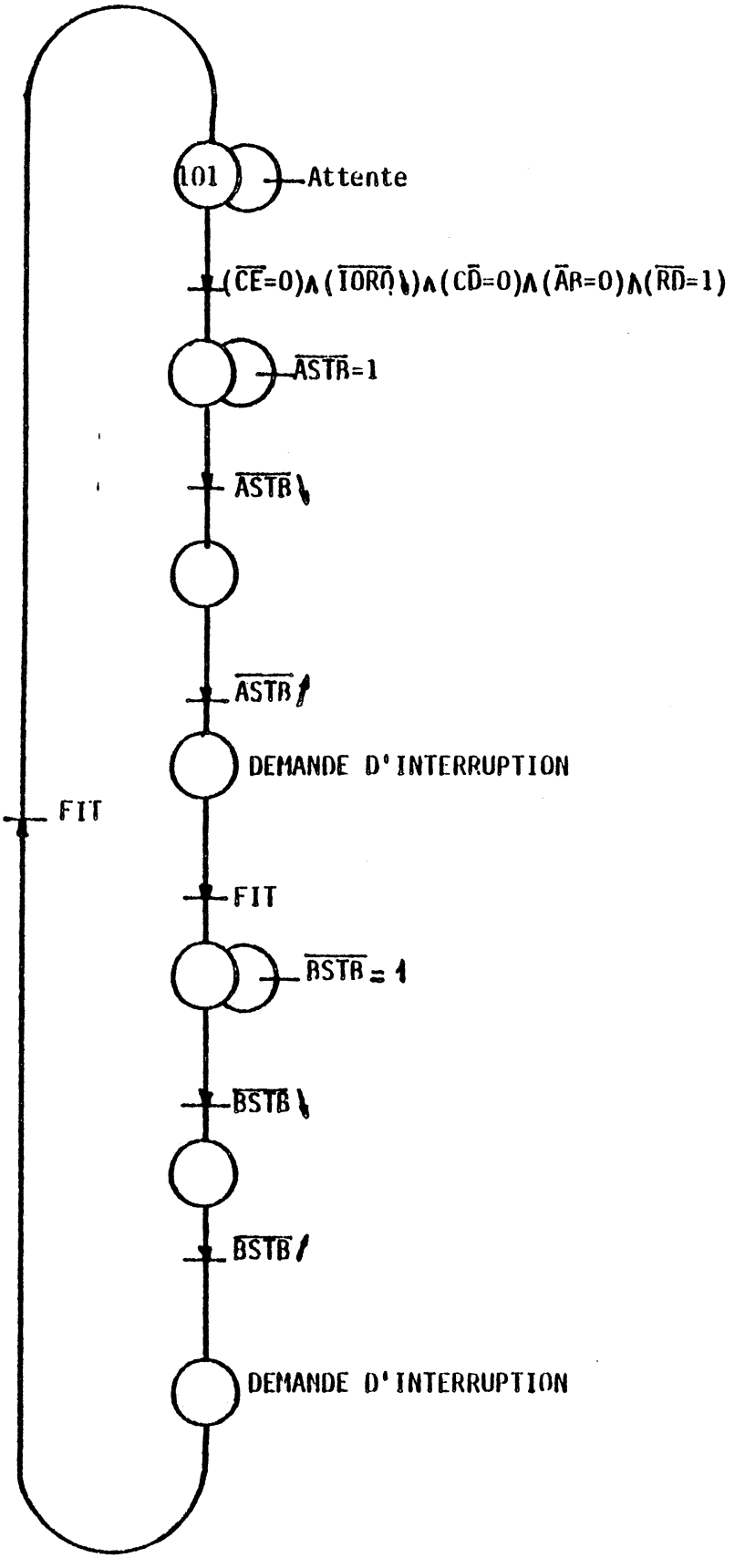
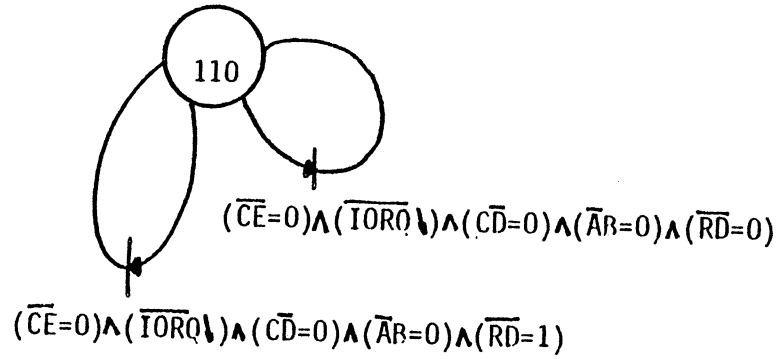

101
Attente
(CE=0)∧(IORQ↓)∧(CD=0)∧(AR=0)∧(RD=1)
ASTB=1
ASTB↓
ASTB↑
DEMANDE D'INTERRUPTION
FIT
FIT
BSTB = 1
BSTB↓
BSTB↑
DEMANDE D'INTERRUPTION

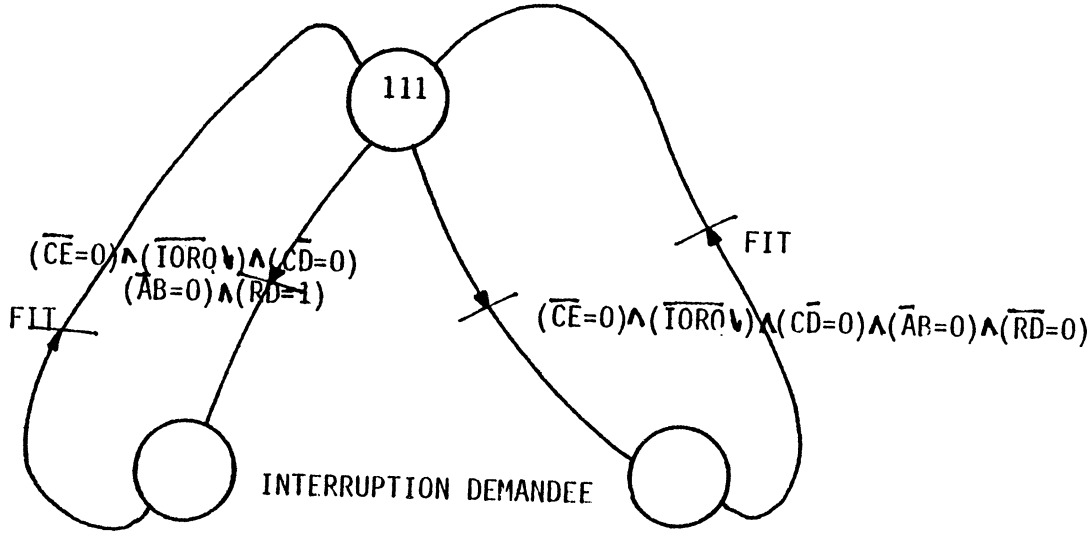
FIGURE 15



Graphes des phases du mode 3.

Le mode 3 du circuit se caractérise par le fait qu'il n'utilise aucun signal de synchronisation; de plus la génération des interruptions est de type calculé ; soit F la fonction d'interruption.

110 code du mode 3 sans interruption
111 code du mode 3 avec interruption



Cet ensemble de graphe donne la structure du test de conformité qui sera appliqué au circuit.

II - LE TEST

II - 1. LE TEST DE CONFORMITE

Etant donnés les deux niveaux de graphe du modèle de référence, le test de conformité se déroulera en deux étapes.

II.1.1 Test des macrophases

Les transitions d'une macrophase à une autre sont modèlisées par le graphe orienté des macrophases. On réalise le test des macrophases en effectuant un test de couverture de tous les arcs du graphe.

Le problème se ramène donc à un parcours des arcs d'un graphe orienté. (SAK82)(PAY75)(BER70)

Etant données les hypothèses faites (voir chapitre II) on n'effectue pas de test de séquences d'arcs dans le graphe des macrophases.

II.1.2 Graphes des phases

La nature des graphes des phases est très différente de celle des graphes des macrophases. Ces graphes ont un état initial ; le parcours de ces graphes se termine toujours à cet état initial. De plus tous les états de ces graphes excepté l'état initial sont des états inconnus déduits des chronogrammes illustrant le fonctionnement du circuit. On effectue un test formel des graphes des phases par activation des chemins de l'arbre d'exécution symbolique. (BAS82)

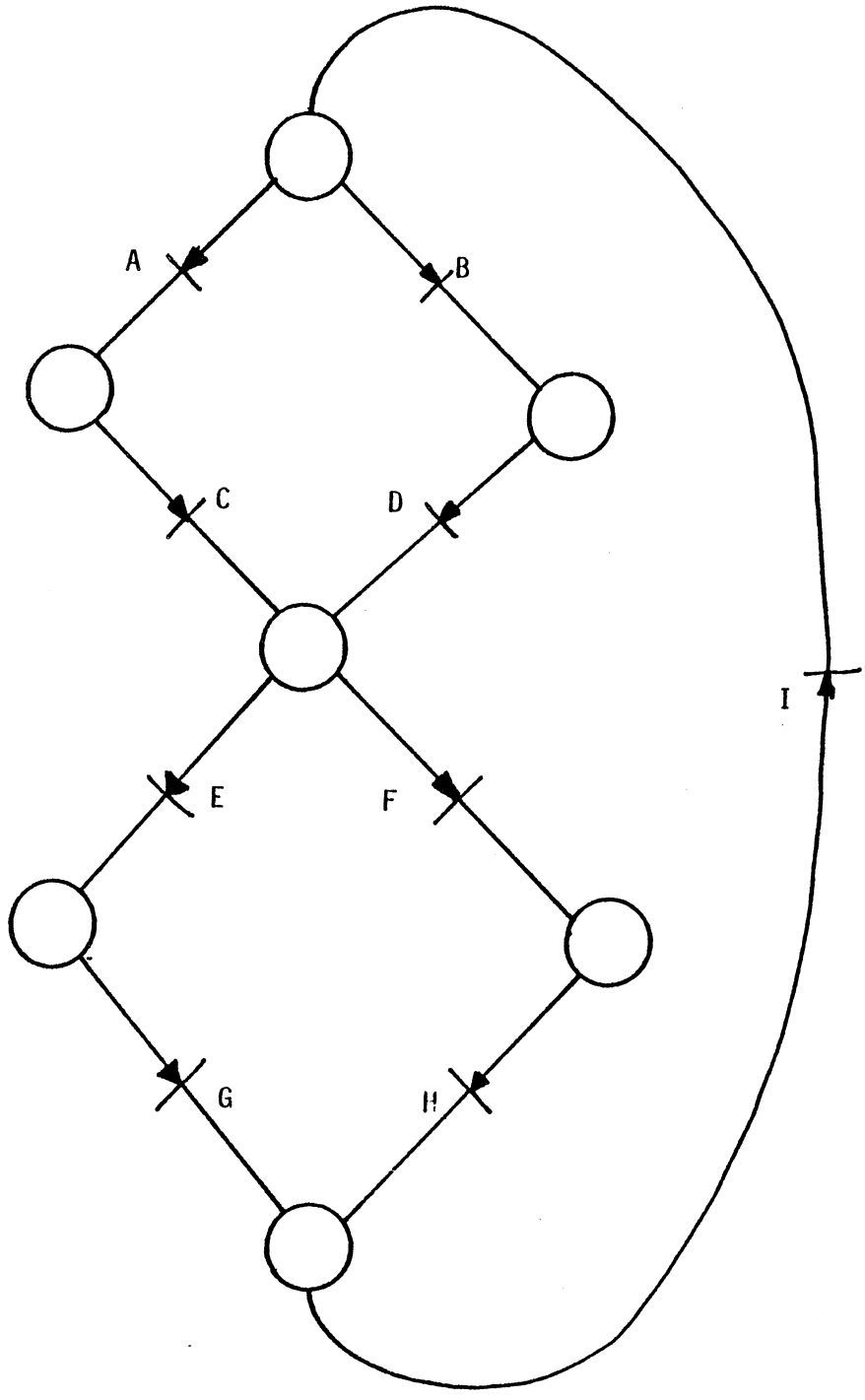
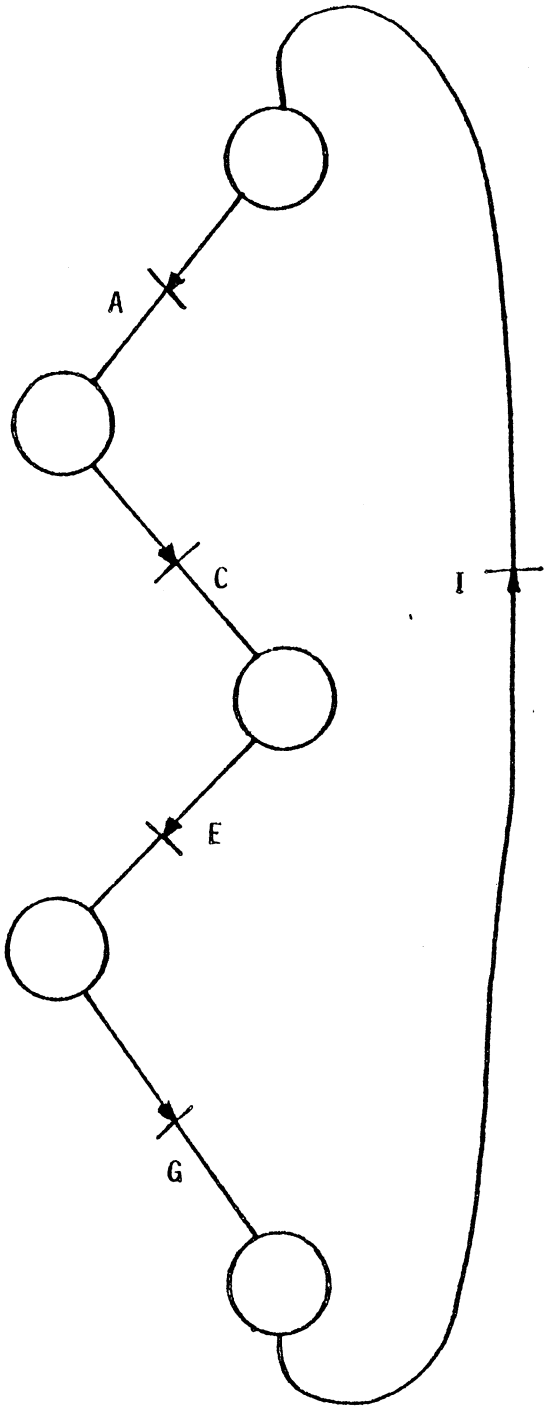


FIGURE 17

Un tel graphe (voir figure 17) serait testé en parcourant chacun des quatre chemins possibles.



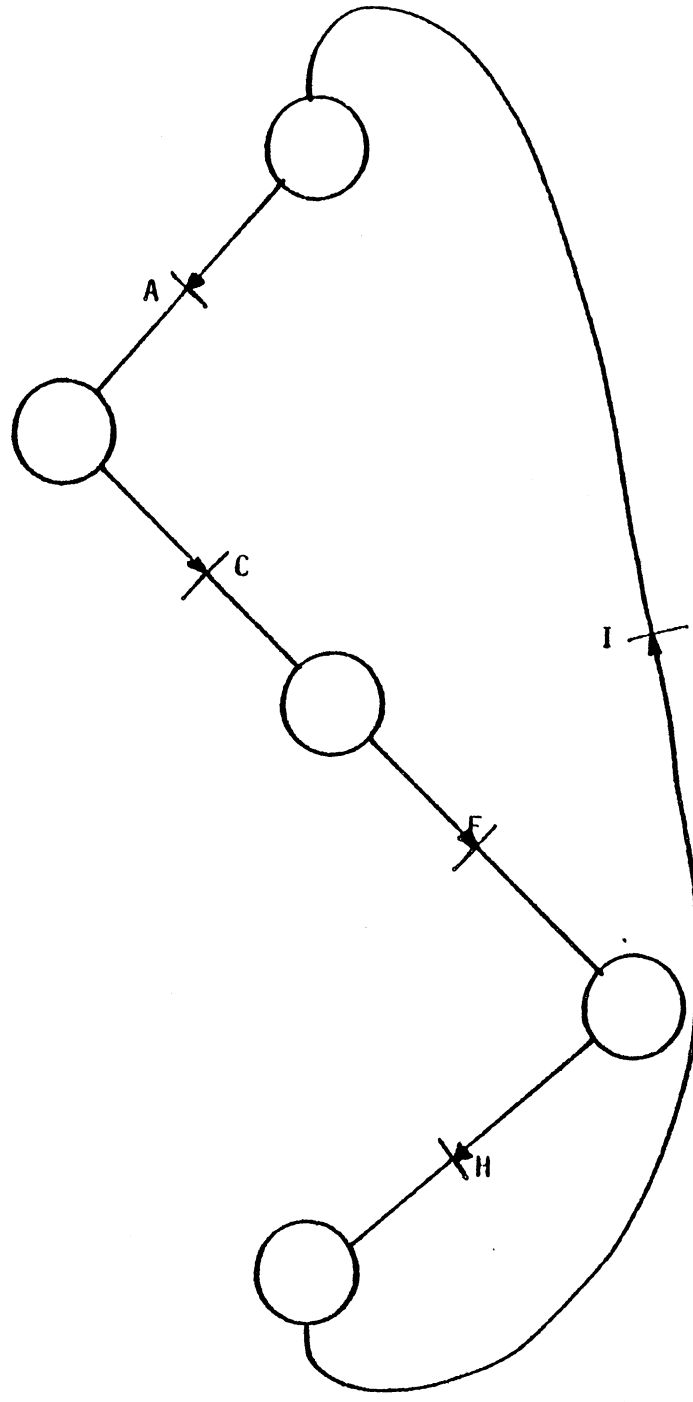
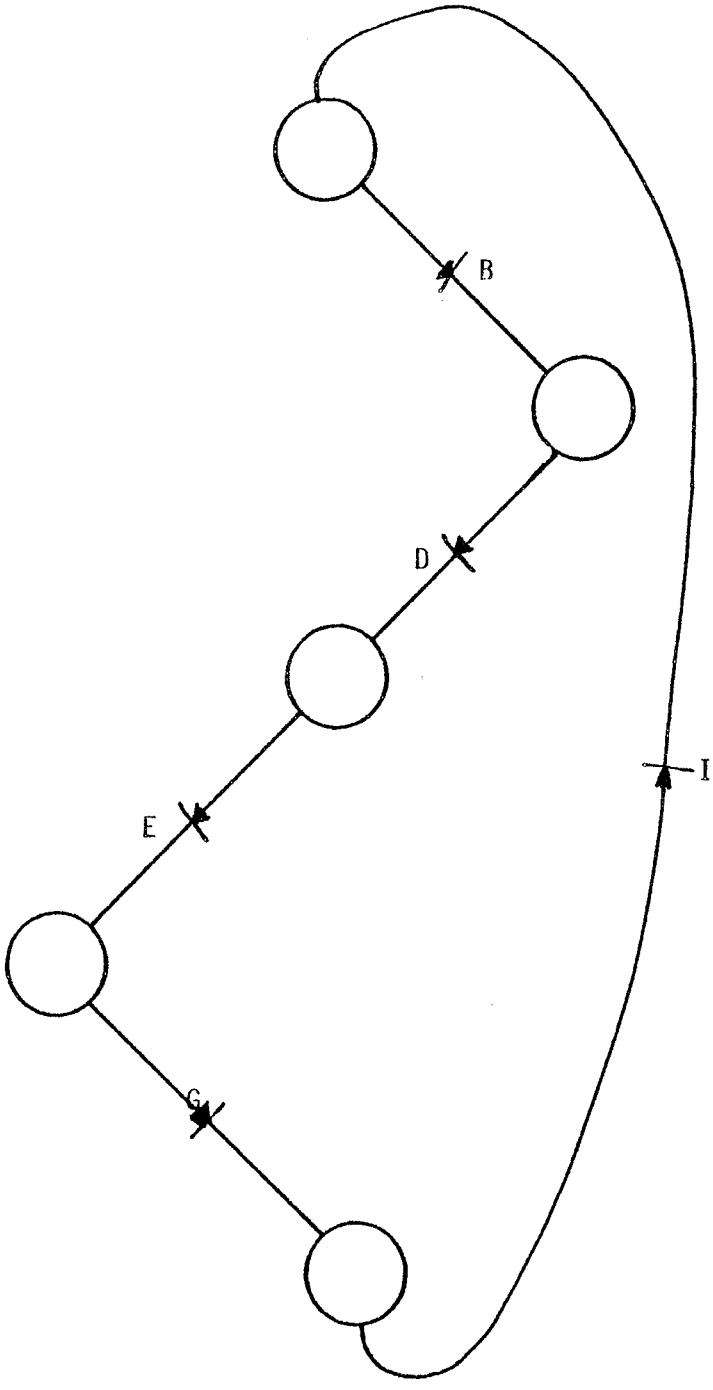


FIGURE 18-1



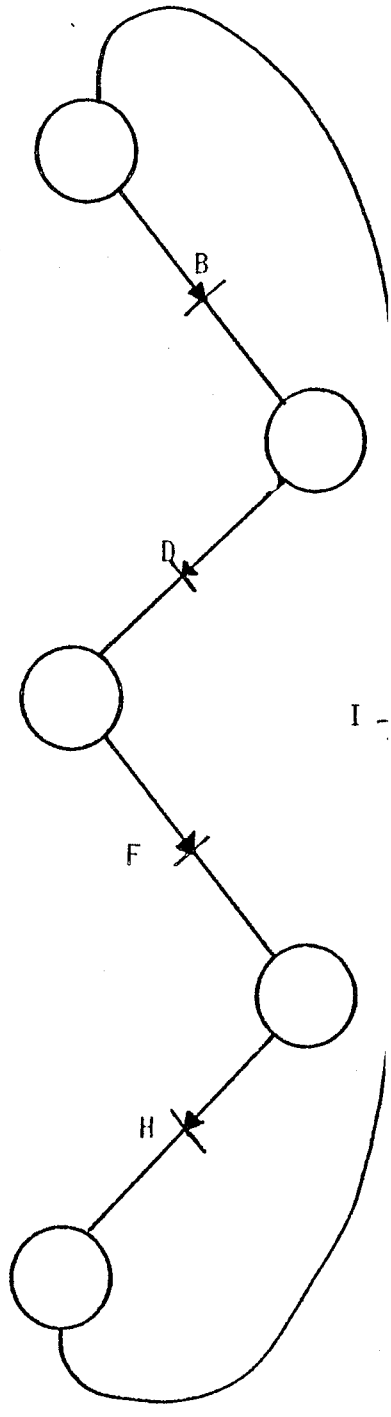


FIGURE 18-2

II - 2. LE TEST DE BALAYAGE

Comme il a été précisé dans le chapitre II, le test de balayage concerne différents modules matériels définis par l'étude de l'architecture du circuit.

Dans le cas de circuits périphériques, on peut généralement isoler :

- des bancs de registres.
- des opérateurs de calcul de fonctions spécifiques.
- des mémoires.
- des timers.
- etc...

Les opérandes de test sont déterminés par le type de test envisagé sur le module matériel considéré. Ce test peut être exhaustif, classique ou aléatoire.

Exemple d'application :

CIRCUIT Z8420 PIO.

Les registres suivant sont testés à l'aide d'une méthode classique dérivée du WALKING 1/0 ; ces registres contiennent les mots d'entrée du séquenceur de phase.

(I07-I00)	sens de transfert des données en mode3.
(MB7-MB0)	masque des bits de la fonction d'interruption.
(V7-V1-0)	vecteur d'interruption.

* Registres de transfert: ces registres sont testés à l'aide de la même méthode.

D17-D10	registres d'entrée.
D07-D00	registres de sortie.

* Le registre contenant les entrées du séquenceur de macrophases

CM0	CM1	ITE

est testé exhaustivement.

REMARQUE: Ce test est réalisé lors du test de conformité.

Le registre contenant les paramètres de la fonction de validation des interruptions est exhaustivement lors du test de celle-ci.

* Opérateur de la fonction de validation des interruptions:

- en mode 0,1,2 la fonction d'interruption est réduite à l'expression:

$$F(012) = ITE$$

- en mode 3 des paramètres supplémentaires interviennent: MBi, ET/OU, H/L.

NOTATION: H= ϕ L= $\overline{\phi}$ ET= ψ OU= $\overline{\psi}$

bi les bits de la donnée transférée.

$$F(3)=ITE\left[\left(\sum_{0}^{7} MBi(bi\,\phi+\overline{bi}\,\overline{\phi})\right)\overline{\psi} + \left(\prod_{0}^{7} MBi(bi\,\phi+\overline{bi}\,\overline{\phi})\right)\psi\right]$$

La fonction globale s'écrit:

$$F=F(012)(MODE0+MODE1+MODE2)+F(3)MODE3$$

La partie F(U12) est testée exhaustivement par le test de conformité qui intéresse les graphes de phase. Dans l'expression de F(3) les MBi sont testés de manière classique en balayage. Les paramètres ET/OU et H/L sont testés exhaustivement. Lors de ce test les valeurs des MBi et des bi sont déterminées aléatoirement dans un ensemble de vecteurs permettant la validation et la non-validation des interruptions.

* Opérateur de hiérarchisation des interruptions: La hiérarchisation des interruptions générées par le circuit se situe à deux niveaux.
- au niveau externe: plusieurs circuits périphériques peuvent être chaînés hiérarchiquement.
- au niveau interne: le circuit d'accès A est prioritaire sur le circuit d'accès B.

NIVEAU EXTERNE: L'entrée de l'opérateur est le signal IEI.
Si IEI=1 pendant tout le déroulement de l'interruption le circuit demandeur reste le plus prioritaire. Le déroulement de l'interruption est normal.
Si pendant le déroulement de l'interruption IEI passe à 0 cela signifie qu'un circuit plus prioritaire est demandeur; tout le temps pendant lequel IEI reste à 0, le déroulement du programme d'interruption du premier demandeur est stoppé au profit du second demandeur.

Si IEI=0 l'interruption est mise enattente jusqu'à ce que IEI passe à 1.

NIVEAU INTERNE: Les entrées de l'opérateur sont IEI et ASTB.BSTB. Le signal IEI reste le plus prioritaire, mais lorsque les signaux d'acquittement ASTB relatif au port A et BSTB relatif au port B sont simultanés, l'interruption provoquée par le signal relatif au port A,ASTB, sera prise en compte en premier.

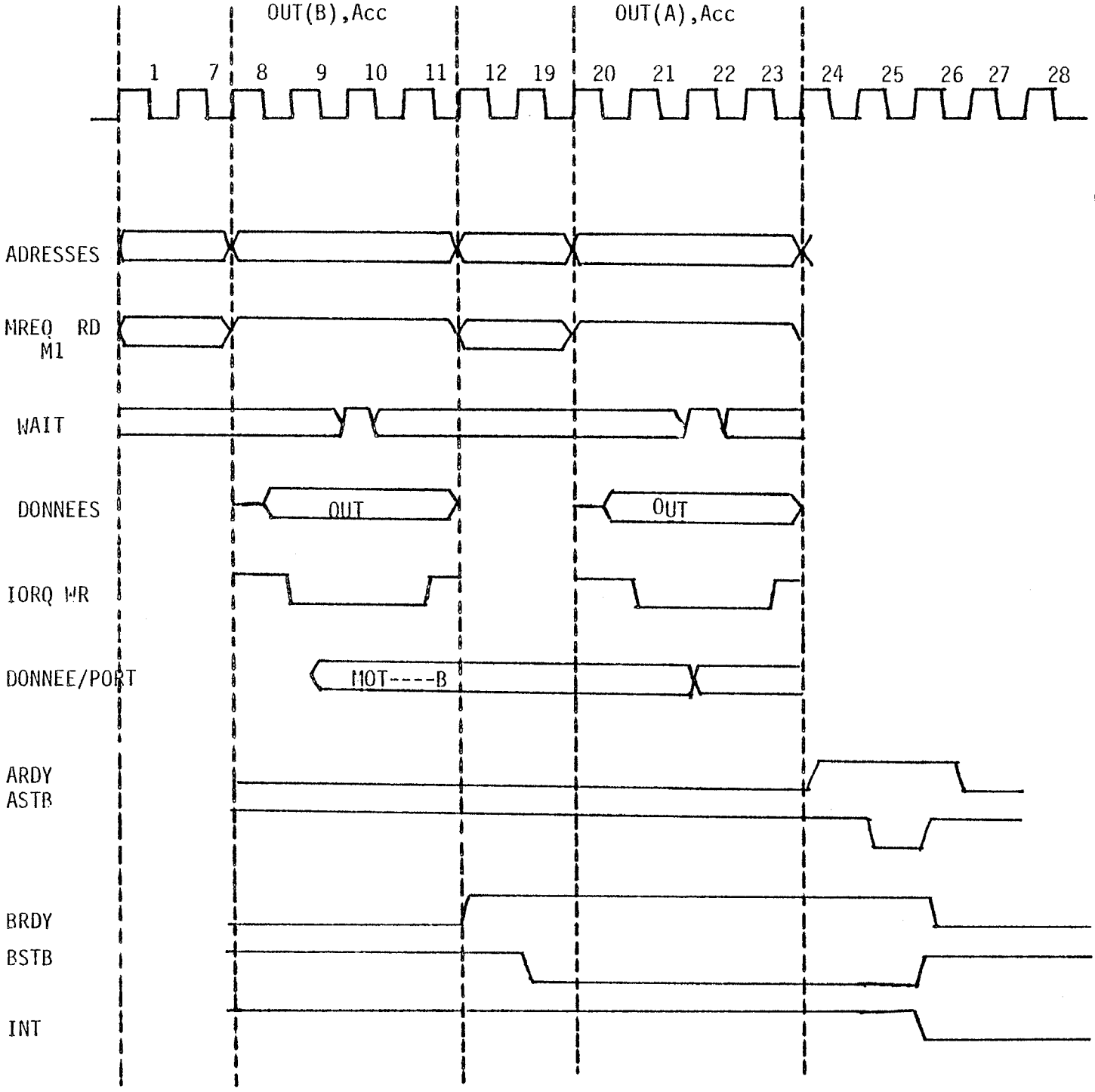

OUT(B),Acc
OUT(A),Acc
1
7
8
9
10
11
12
19
20
21
22
23
24
25
26
27
28
ADRESSES
MREQ RD
M1
WAIT
DONNEES
OUT
OUT
IORQ WR
DONNEE/PORT
MOT----B
ARDY
ASTB
BRDY
BSTB
INT

FIGURE 19

II.3 EXPRESSION TEMPORISEE DES JEUX DE TEST

Nous utilisons la modélisation definie au chapitre III; cette modélisation est employée pour le système GAPT.

Exemple: Hiérarchisation des interruptions au niveau interne.

La priorité du port A sur le port B est mise en évidence en simulant une arrivée simultanée des acquittements destinés au port A et port B.
Les signaux d'entrée de l'opérateur de hiérarchisation sont: IEI,ASTB,BSTB.
Soient les descriptions des chronogrammes correspondant.

C.IEI=1

C.ASTB=ASTB(R25+R26)

C.BSTB=BSTB(R19+R26)

Les autres signaux d'entrée sont formés par la liaison microprocesseur fournissant le protocole nécessaire.

III MISE EN OEUVRE DU TEST

III.1 MISE EN OEUVRE MATERIELLE

La mise en oeuvre du test est réalisée au moyen du testeur développé en collaboration avec Electronique Serge Dassault.(ESD82)(ESD83)(EMD81). Le circuit sous test est relié à un microprocesseur de la même famille.

Exemple: Z80 + Z8420 PIO
MC 6800 + MC6821 PIA

Le microprocesseur associé a deux missions:

- assurer la direction du test en envoyant des commandes et des données au circuit sous test.
- assurer l'interface entre le système de test et le circuit sous test.

Soit le schéma type de mise en oeuvre matérielle du test.

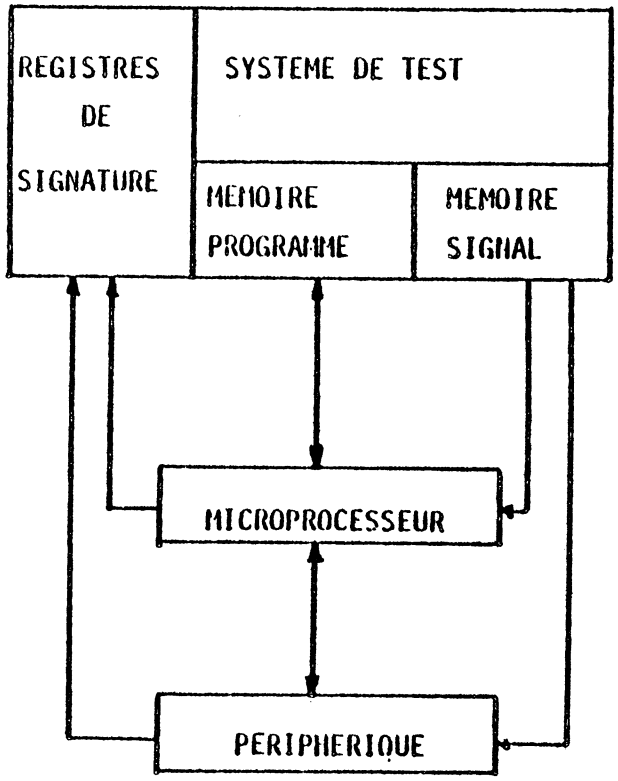


FIGURE 20

Remarque: La liaison entre un microprocesseur et un circuit périphérique de la même famille supprime les problèmes de protocole entre les deux circuits.

III - 2. MISE EN OEUVRE LOGICIELLE

La génération des programmes de test peut être en partie automatisée grâce à l'utilisation d'un outil d'aide à la génération du type ROBIN. (ROB81)

L'usage de ce type d'outil est intéressant étant donné le caractère très répétitif des programmes de test. GAPT est un instrument très performant qui pourrait également être utilisé.

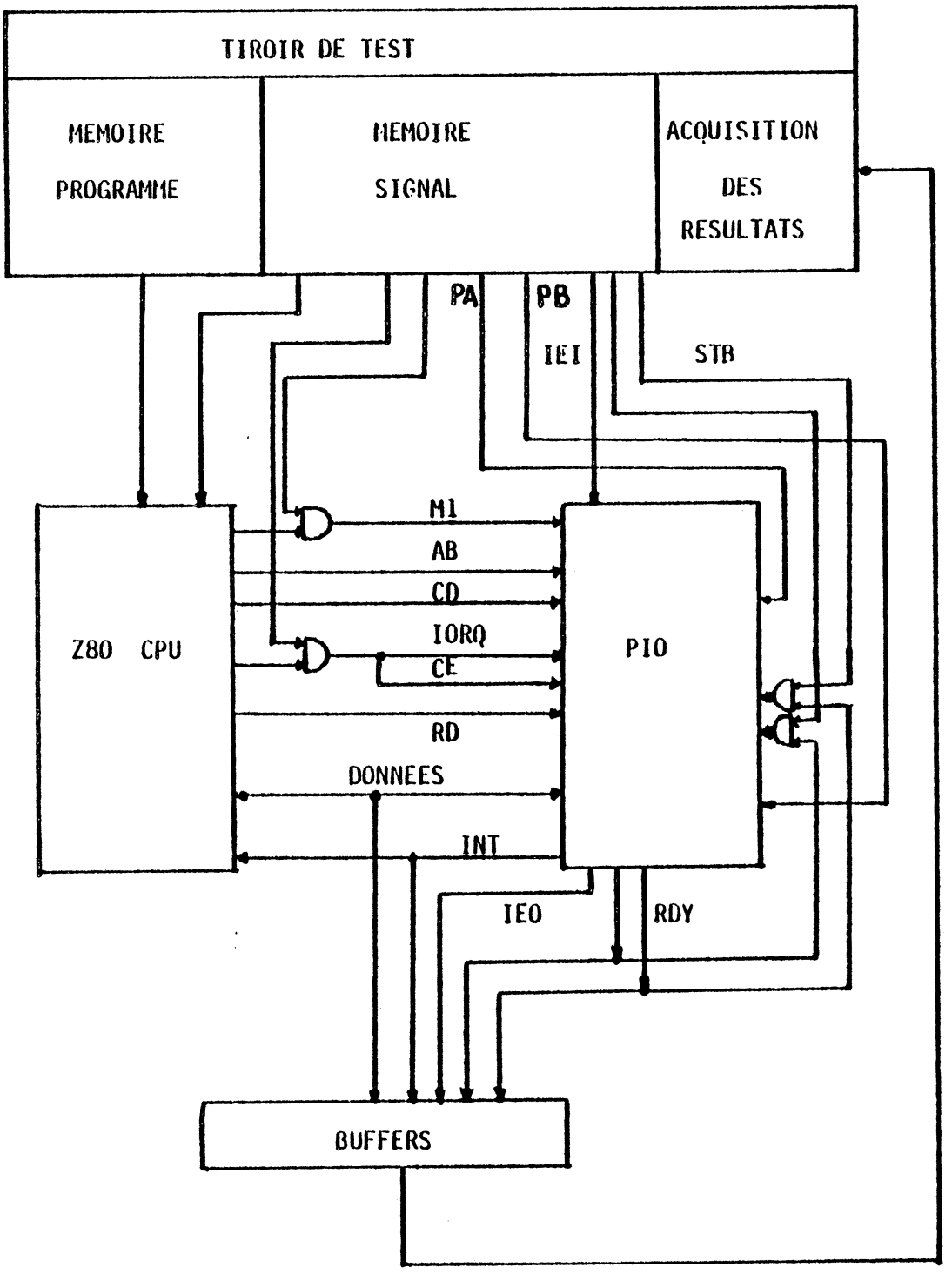

TIROIR DE TEST
MEMOIRE PROGRAMME
MEMOIRE SIGNAL
ACQUISITION DES RESULTATS
PA
PB
IEI
STB
M1
AB
CD
IORQ
CE
RD
DONNEES
INT
IEO
RDY
Z80 CPU
PIO
BUFFERS

FIGURE 21

ANNEXE AU CHAPITRE 4

PROGRAMME DE TEST DU PIO Z8420 ZILOG / CIRCUIT D'ECHANGE A

Préalablement on définit les adresses du circuit :

envoi de commandes :	PA	port A
	PB	port B
envoi de données :	pa	port A
	pb	port B

On précise le contenu des zones données :

ZD1/ 0	00000000
1	00000001
2	00000011
3	00000111
4	00001111
5	00011111
6	00111111
7	01111111
8	11111111
9	11111110
A	11111100
B	11111000
C	11110000
D	11100000
E	11000000
F	10000000

ZD2/ 0	00000000		
1	10000000		
2	01000000		
3	00100000		
4	00010000		
5	00001000		
6	00000100		
7	00000010		
8	00000001		
ZD3/ 0	11111111		
1	01111111		
2	10111111		
3	11011111		
4	11101111		
5	11110111		
6	11111011		
7	11111101		
8	11111110		
ZD4/ 0	00__1111	MOT DE COMMANDE	MODE0
1	01__1111		MODE1
2	10__1111		MODE2
3	11__1111		MODE3
ZD5/ 0	0___0011	INTERRUPTION INHIBEE	
1	1___0011	INTERRUPTION PERMISE	

ZD6/ 0	00000111	CONTROLE INTERRUPTION
1	10000111	
2	10010111	
3	10100111	
4	10110111	
5	11000111	
6	11010111	
7	11100111	
8	11110111	
ZD7/ 0	00000000	MOT DE COMMANDE ENTREE SORTIE
1	11111111	
2	01010101	
3	10101010	
ZD8/ 0	_____0	VECTEUR D'INTERRUPTION
1	_____0	
2	_____0	
3	_____0	
4	_____	REGISTRE I DU Z80
5	_____	
6	_____	
7	_____	

Structure de la mémoire signal :

E p.23
22
21
20 portA / portB
19
18
17
E p.16

CIRCUIT PERIPHERIQUE

E p.15 ASTB
E p.14 BSTB

E p.13 IEI

E p.12 M1

E p.11 IORQ

E p.10
09
08
07
06

CIRCUIT MICROPROCESSEUR

05
04
03
02
01
E p.00

Initialisations :

Le microprocesseur maître du test doit être initialisé pour se conformer aux conditions prescrites de fonctionnement associé à un circuit périphérique.

Le mode d'interruption est le mode 2: INSTRUCTION IM2

On doit charger un vecteur servant au calcul de l'adresse du programme d'interruption. VECTEUR I

On calcule les valeurs de I et de VIT pour que l'adresse générée se situe dans la partie mémoire désignée à cet effet.

I=XXXXXXXX
VECTEUR D'INTERRUPTION VIT=ZD2/3/7(4,5)

```
LD A      ZD8(i)
OUT  (PA) A              chargement de VIT

LD A      ZD8(j)
LD I,  A                 initialisation du Z80
IM2
```

Remarque : Tous les programmes sont accompagnés d'une séquence START et d'une séquence STOP pour la mémoire signal. (voir chapitre III)

Ces séquences affectent le microprocesseur et le circuit périphérique sous test.

TEST DE CONFORMITE :

MISE EN MODE / GRAPHES PRINCIPAUX

notation : MS=mémoire signal

```
LD      A    ZD4.0    MISE EN MODE 0 SANS INTERRUPTION
OUT     PA   A

LD      A    ZD4.0    MISE EN MODE 0 AVEC INTERRUPTION
OUT     PA   A
LD      A    ZD5.1
OUT     PA   A

LD      A    ZD4.1    MISE EN MODE 1 SANS INTERRUPTION
OUT     PA   A
LD      A    ZD4.1    MISE EN MODE 1 AVEC INTERRUPTION
OUT     PA   A
LD      A    ZD5.0
OUT     PA   A

LD      A    ZD4.2    MISE EN MODE 2 SANS INTERRUPTION
OUT     PA   A

LD      A    ZD4.2    MISE EN MODE 2 AVEC INTERRUPTION
OUT     PA   A
LD      A    ZD5.1
OUT     PA   A
```

```
        LD              A   ZD4.3   MISE EN MODE 3 SANS INTERRUPTION
        OUT             PA  A
        LD              A   ZD7.2   SELECTION ENTREE SORTIE
        OUT             PA  A
        LD              A   ZD6.0   SANS INTERRUPTION
        OUT             PA  A


        LD              A   ZD4.3   MISE EN MODE 3 AVEC INTERRUPTION
        OUT             PA  A
        LD              A   ZD7.2   E/S ALTERNEES
        OUT             PA  A
        LD              A   ZD6.8   ET.H.MASQUE SUIT
        OUT             PA  A
        LD              A   ZD0.4   MASQUE
        OUT             PA  A

        DECLENCHEMENT MS

        NOP
        NOP
        NOP                         PASSAGE A L'ETAT REINITIALISATION
TRANSFERT DE DONNEES    /   GRAPHES LOCAUX


        LD              A   DONNEES TRANSFERT MODE 0
        OUT             pa  A


        LD              A   DONNEES TRANSFERT MODE 1
        DECLENCHEMENT MS
        NOP
        NOP
        IN              A   pa      DONNEE FOURNIE PAR MS
```

```
LD    A     DONNEES     TRANSFERT MODE 2
OUT   pa    A
DECLENCHEMENT HS
NOP
NOP
IN    pa    A


LD    A     DONNEES
OUT   pa    A           TRANSFERT MODE 3
```

TEST DE BALAYAGE:

Eléments de memorisation

```
LD    A     ZD4.0       MODE 0      TEST DE DO EN MODE 0
OUT   PA    A                       SANS INTERRUPTION

LD    A     ZD1.i       SORTIE      i=0....F
OUT   pa    A
LD    A     ZD2.j       SORTIE      j=0....a
OUT   pa    A
LD    A     ZD3.k       SORTIE      k=0....a
OUT   pa    A
LD    A     ZD7.1       SORTIE      1=2,3
OUT   pa    A

LD    A     ZD4.1       MODE 1      TEST DE D1 EN MODE 1
OUT   PA    A                       SANS INTERRUPTION
DECLENCHEMENT HS
NOP
NOP
IN    pa    A                       DONNEES FOURNIES PAR M
                                    ZD11/ZD2j/ZD3k/ZD41
```

```
LD    A     ZD4.3      MODE 3
OUT   PA    A
LD    A     ZD7.1      CHARGEMENT DE ID/ I=2,3
OUT   PA    A
LD    A     DONNEES
OUT   pa    A



LD    A     ZD4.0      MODE 0
OUT   PA    A
LD    A     ZD5.1      AUTORISATION DES INTERRUPTIONS
OUT   PA    A
LD    A     ZD8.i      CHARGEMENT DU VIT i=0/1/2/3
OUT   PA    A
LD    A     ZD6.4      CHARGEMENT DE 1 DANS LE Z80
LD    I     A
IM2
LD    A     DONNEES    TRANSFERT
OUT   pa    A
La fonction d'interruption:

LD    A     ZD5.1      AUTORISATION DES INTERRUPTIONS
OUT   PA    A
LD    A     ZD8.i      CHARGEMENT VIT
OUT   PA    A
LD    A     ZD6.4      INITIALISATION DU Z80
LD    I     A
IM2


LD    A     ZD4.0      MODE 0
OUT   PA    A
LD    A     DONNEES    TRANSFERT
OUT   pa    A
```

```
LD    A     ZD4.1       MODE 1
OUT   PA    A
DECLENCHEMENT MS
NOP
NOP
IN    A     pa          TRANSFERT


LD    A     ZD4.2       MODE 2
OUT   PA    A
LD    A     DONNEES
OUT   PA    A
DECLENCHEMENT MS
NOP
NOP
IN    A     pa          TRANSFERT
```

La fonction d'interruption mode 3

INITIALISATION DU Z80

```
LD   A    ZD4.3      MODE 3
OUT  PA   A
LD   A    ZD7.1      TOUS LES BITS SONT EN SORTIE
OUT  PA   A
LD   A    ZD8        CHARGEMENT DU VECTEUR
OUT  PA   A
LD   A    ZD6.k      k=2/4/6/8 MOT DE CONTROLE DES
OUT  PA   A          INTERRUPTIONS
LD   A    ZD7.2      MASQUE 1
OUT  PA   A
LD   A    DONNEES    TRANSFERT DONNANT UNE INTERRUPTI
OUT  pa   A
LD   A    ZD7.3      MASQUE 2
OUT  PA   A
LD   A    DONNEES    TRANSFERT DONNANT UNE INTERRUPTI
OUT  pa   A
LD   A    DONNEES
OUT  pa   A          TRANSFERT SANS INTERRUPTION
```

Hierarchie des interruptions:

Entre portA et portB

```
LD    A     Z04.0
OUT   Pb    A
OUT   PA    A          MODE 0 PORT A / PORT B
LD    A     Z05.1
OUT   PA    A          VALIDATION DES INTERRUPTIONS
OUT   Pb    A
LD    A     Z06.i      CHARGEMENT VIT     i><j
OUT   PA    A
LD    A     Z06.j
OUT   Pb    A
INITIALISATION DU Z80

LD    A     DONNEES
DECLENCHEMENT MS
OUT   pb    A
OUT   pa    A          TRANSFERT
```

Niveau externe

```
LD          A   ZD4.0    MODE 0
OUT         PA  A
LD          A   ZD5.1    AUTORISATION DES INTERRUPTIONS
OUT         PA  A
```

INITIALISATION DU Z80

```
LD          A   ZD8.i    CHARGEMENT VIT
OUT         PA  A

LD          A   DONNEES
DECLENCHEMENT MS
OUT         pa  A
NOP
NOP
```

MS 2 CAS : IEI=0

IEI=1 PUIS PASSE A 0 ET REVIENT A 1

Remplissage de la mémoire signal :

La description des signaux s'effectue au moyen du symbolisme déjà utilisé pour les microprocesseurs. A partir des chronogrammes on établit le contenu de la mémoire signal.

CONCLUSION GENERALE

Ce travail tente de montrer que le test comportemental des circuits périphériques peut s'orienter autour de notions déjà développées pour le test des microprocesseurs comme les états comportementaux, le découpage du test en test de balayage et en test de conformité, etc...

Par contre, la description des circuits est spécifique ce qui laisse envisager une rapide application des langages de description du type CADOC.

Les équipements de test moyennant une mise en oeuvre légèrement différente peuvent également être communs.

BIBLIOGRAPHIE

ABR81 J.A ABRAHAM , K.P PARKER
Practical microprocessor testing operand closed loop approaches.
COMPCON 81 / MARS 81.

AKE83 J.M ACKEN
Testing for buying faults short in CMOS circuits.
20th DAC 1983

AKE77 S.B AKERS
Partitionning for testability.
Journal of DAFTC Vol1 / FEV 77

AKE80 S.B AKERS
Test generation technique
IEEE 1980

AND80 H.ANDO
Testing VLSI with random access scan.
COMPCON 80 / FEV 80

ARY82 C.C BEH , K.H ARYA , C.E RADKE , K.E TORKU
Do stuck at fault model reflect manufacturing defect
IEEE TEST CONFERENCE 1982

BAS82 P.BASSET
Etude de l'utilisation des écoulements fonctionnels pour le test de circuits intégrés.
DEA INPG 1982

BEL77 C.BELLON , P.CASPI , C.ROBACH , G.SAUCIER
Constructeurs et utilisateurs face au test des microprocesseurs.
RR94 IMAG 1977

BEL79 C.BELLON , G.SAUCIER
Conception de séquenceur.

BEL81 C.BELLON , JM.GOBBI , G.SAUCIER
Hardware description levels and test for complex circuits.
18th DAC 1981

BEL82-1 C.BELLON , A.LIOTHIN , S.SADIER , R.VELAZCO
Génération automatique de programmes de test pour microprocesseurs et leurs circuits annexes.
Rapport de contrat IMAG-EMD FEV82

BEL82-2 C.BELLON , A.LIOTHIN , S.SADIER , R.VELAZCO
Autmatic test generation for microprocessors.
19th DAC JUIN 1982

BEL82-3 C.BELLON , S.SADIER , R.VELAZCO
Behavioral test method of microprocessors and automated generation: GAPT system.

BEL82-4 C.BELLON , S.SADIER , R.VELAZCO
A global behavioral test method for microprocessors and VLSI circuits.

BEL83-1 C.BELLON , E.KOLOKITHAS , R.VELAZCO
Génération automatique de programme de test pour microprocesseurs.
Journées délectronique. LAUSANNE OCTOBRE 1983

BEL83-2 C.BELLON
Test en ligne et hors ligne de systèmes à circuits.
THESE D'ETAT IMAG 1983

BRE79 M.A BREUER , A.D FRIEDMAN
Test80 a proposal for an advanced automatic test generation system.
IEEE AUTESTCON 1979

BRE80 M.A BREUER , A.D FRIEDMAN
Functionnal level primitives in test generation.
IEEE TRANSACTION ON COMPUTER VOL24 MARS 1980

BOU82 M.BOURDON
Une approche unifiée des problèmes d'ordonnancement statique discret.
DEA INPG 1982

CAR82 W.C CARTER
Signature testing with guaranteed bounds for fault coverage.
IEEE TEST CONFERENCE 1982

CAS82 P.CASPI , N.HALBWACHS
A model for parallel and real time system.
RR285 IMAG JANVIER 1982

CHI81 A.C CHIANG , R MACCASKILL
Two new approaches simplify testing of microprocessors.
ELECTRONICS JANV 1976

CHI83 K.W CHIANG , Z.G VRANESIC
On fault detection in CMOS logic networks.
20th DAC 1983

COU81 B.COURTOIS
Test et LSI.
THESE D'ETAT USMG/INPG 1981

DAV79 R.DAVID , P.THEVENOD-FOSSE
Panorama des methodes de test non déterministes des circuits logiques.
RAIRO AUTOMATIQUE 1979

DAV80 R.DAVID
Feed back register testing.
IEEE TRANSACTION ON COMPUTERS VOL29 1980

DAV81 R.DAVID , P.THEVENOD-FOSSE
Random testing of the data processing section of a microprocessor.
FTCS11 PORTLAND 1981

EDW80 D.G EDWARDS
Testing for MOS integrated circuit failure modes.
IEEE TEST CONFERENCE 1980

ELE80 ELCTRONIC DESIGN INTERNATIONAL
Annual microprocessors special issue.
NOV 1980

ELZ80 Y.M ELZIQ
Testing of a special VLSI design.
COMPUTER SCIENCE PRESS 1980

EMD81 Rapport Electronique Marcel Dassault
Dossier de définition globale: Réalisation d'une maquette d'outil général de production et d'exploitation automatique de test pour microprocesseurs et circuits annexes.
NE38935 OCTOBRE 1981

ESD82 Rapport Electronique Serge Dassault
Premier rapport d'avancement de la deuxième tranche.
NE40893 SEPTEMBRE 1982

ESD83 Rapport ESD
Rapport de la deuxième tranche.
NE41549 1983

ESI79-1 Les microprocesseurs 16 bits
CONFERENCE ESIEE 1979

FRO77 R.A FROHWERK
Signature analysis: a new digital field service method.
HP JOURNAL MAI 1977

GER79 R.GERBER , C.BRIE
Les microprocesseurs en tranches.
1979 TECHNIQUE ET DOCUMENTATION PARIS

GOR77 G.GORDON , H.NADIG
Hexadecimal signatures identify trouble spots in microprocessors.
ELECTRONICS MARS 1977

HAL82 N HALBWACHS , P CASPI
An approach to real time systems modelling.
INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING SYSTEMS 82

INT82 INTEL DATA BOOK 1982

JAI83 S.K JAIN , V.D AGRAWAL
Test generation for MOS circuits using D-algorithm.
20th DAC 1983

JOH79 W.A JOHNSON
Behavioral level test developpment.
TEXAS INSTRUMENT DALLAS 1979

KIZ80 R.E KIZIS , R.T HERBERY
PLT Language and languages processing system.
IEEE TEST CONFERENCE 1980

KOH70 Z.KOHAVI
Switching and finite automata theory. Chapitre 13
MACGRAW HILL NY1778 COMPUTER SCIENCE SERIE.

LIT80 E.L LITHY , R HUSSON
Bit sliced microprocessors testing. A case study.
IEEE TEST CONFERENCE 1980

LUC79 B LUCAZEAU
Les microprocesseurs.
ESIEE CFCIC 1979

MAL82 Y.K MALAIYA , S.Y.H SU
A new fault model and testing techniques for CMOS devices.
IEEE TEST CONFERENCE 1982

MEY84 Y.MEYER

THESE CNAM A PARAITRE

MIN79 MINIS ET MICROS N°79.81.82.83.85.86
Les microprocesseurs en tranches.

MIT77 K MITCHELL
Signature analysis.
ELECTRONICS NOVEMBRE 1977

MOA81 M MOALLA
Spécification et conception sûre d'automatismes discrets complexes basées sur l'utilisation du GRAFCET et des réseaux de Petri.
THESE D'ETAT INPG 1981

MOT81 MOTOROLA MC68000
Advanced informations.

MUE81 E.I MUEHLDORF , A.D SAVKAR
LSI logic testing: an overview.
IEEE TRANSACTION ON COMPUTER VOL30 1981

NIC80 V NICKEL
VLSI The inadequacy of the stuck at fault model.
IEEE TEST CONFERENCE 1980

NSC81 NSC800 Microprocessor family handbook
NATIONAL SEMICONDUCTORS 1981

PAS80 M PASTUREL
What makes pascal a modern test language.
IEEE TEST CONFERENCE 1980

PAY75 C PAYAN , J SIFAKIS
Propriétés des séquences parcourant les arcs d'un graphe orienté étiqueté.
JOURNEE COMBINATOIRE INFORMATIQUE / BORDEAUX 1975

PIG79 PIGUET PEROTTO STAUFFER MONBARON
Séquenceur de microprocesseur.
BULLETIN ASE N°3 1979

ROB72 C ROBACH , G SAUCIER
Le test logique des circuits intégrés.
ONDE ELECTRIQUE VOL54 N°12 1972

ROB78 C ROBACH
Le test en production et en exploitation.
RR112 RR132 IMAG 1978

ROB79 C ROBACH
Test et testabilité de systèmes informatiques.
THESE D'ETAT INPG 1979

ROB80-1 C ROBACH , G SAUCIER
Microprocessors functional testing.
IEEE TEST CONFERENCE 1980

ROB80-2 C ROBACH , G SAUCIER , R VELAZCO
Flexible test method for microprocessors.
EUROMICRO 1980

ROB81 C ROBACH
Conception d'un générateur de programme de test.
PROJET SURF MARS 1981

ROT68 J.P ROTH
Diagnosis of automata failure: a calculus and a method.
IBM JOURNAL VOL10 1968

ROS82 K ROSE , M.G LIN
Applying test theory to VLSI testing.
IEEE TEST CONFERENCE 1982

SAD81 S SADIER
Test et testabilité des circuits intégrés.
DEA INPG JUIN 1981

SAK82 M SAKAROVITCH
Optimisation dans les réseaux.
ENSIMAG 1982

SAU78 G SAUCIER , C ROBACH
Le test logique des circuits intégrés.
ONDE ELECTRIQUE VOL54 N°2 1978

SAU81-1 G SAUCIER , R VELAZCO
Microprocessor functional testing using deterministic test pattern.
EUROMICRO 1981

SAU81-2 G SAUCIER
Les perspectives dans le domaine du test et de la testabilité des circuits à très haute intégration.
RR268 IMAG OCTOBRE 1981

SAU82-1 G SAUCIER
Le test en ligne.

SCR78 S.E SCRUPSKI
Why and how users test microprocessors.
ELECTRONICS MARS 1978

SIF74 J SIFAKIS , C PAYAN
Conception de systèmes logiques: Automates.
ENSIMAG 1974

SMI77 D.H SMITH
Microprocessor testing: method or madness.
COMPCON 1977

SRI81 J SRIDHAR , J.P HAYES
A functional approach to test bit sliced microprocessors.
IEEE TRANSACTION ON COMPUTERS VOL30 1981

SON82 K SOH , J.Y.O FONG
Automatic behavioral test generation.
IEEE TEST CONFERENCE 1982

THA78 S.M THATTE , J.A ABRAHAM
A methodology for functional level testing of microprocessors.
FTCS8 1978

THA79 S.M THATTE , J.A ABRAHAM
Test generation for general microprocessors architecture.
FTCS9 1979

THA80 S.M THATTE , J.A ABRAHAM
Test generation for microprocessors.
IEEE TEST CONFERENCE 1980

THE78-1 P THEVENOD-FOSSE , R DAVID
A method to analysis random testing of sequential circuits.
DIGITAL PROCESSES VOL4 1978

THE78-2 P THEVENOD-FOSSE
Contribution à l'aide du test aléatoire des circuits séquentiels et des mémoires.
THESE DE DOCTEUR INGENIEUR USMG 1978

THE80 P THEVENOD-FOSSE , R DAVID
Un outil pour l'étude du test aléatoire de la partie opérative des microprocesseurs.
COLLOQUE INTERNATIONAL SUR LA FIABILITE ET LA MAINTENABILITE
SEPTEMBRE 1980

VEL82 R VELAZCO
Test comportemental de microprocesseurs.
THESE DE DOCTEUR INGENIEUR INPG 1982

WAD78 R.L WADSACK
Fault modelling and logic simulation of CMOS and MOS integrated circuits.
BELL SYSTEM TECHNICAL JOURNAL 1978

ZIL83 ZILOG DATA BOOK 1983

AUTORISATION DE SOUTENANCE

VU les dispositions de l'article 3 de l'arrêté du 16 avril 1974,

VU les rapports de présentation de

. Madame G. SAUCIER, Professeur

. Monsieur F. GRILLOT, Ingénieur

Monsieur SADIER Sylvain

est autorisé à présenter une thèse en soutenance pour l'obtention du diplôme de DOCTEUR-INGENIEUR, spécialité "Informatique".

Fait à Grenoble, le 17 novembre 1983

Le Président de l'INPG

D. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble

P.O. le Vice-Président,

RESUME

Après avoir situé le problème du test des circuits intégrés, nous proposons une méthode de test basée sur une description fonctionnelle du circuit.
Le système de test développé en collaboration avec ESD est présenté, ainsi qu'un outil de description de signaux.
Nous développons une application à un circuit périphérique à partir d'un modè de référence comportant deux niveaux de description.

MOTS CLES : Test, Méthode de test, Outil de test, Microprocesseurs, Circuits periphériques, Automate, Niveau de description, Graphe.