Conception de circuits intégrés autotestables pour des hypothèses de pannes analytiques

Michel Nicolaides

THESE

présentée à

l' Institut National Polytechnique de Grenoble

par

Michel NICOLAIDES

CONCEPTION DE CIRCUITS INTEGRES AUTOTESTABLES POUR DES HYPOTHESES DE PANNES ANALYTIQUES.

Thèse soutenue le 6 janvier 1984 devant la commission d'examen.

L. BOLLIET	Président
J. BOREL	Examinateurs
A. COSTES	
J. LEROUDIER	
J.M. RATA	
B. COURTOIS	Rapporteur

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Année universitaire 1982-1983

Président de l'Université : D. BLOCH

Vice-Président : René CARRE
Hervé CHERADAME
Marcel IVANES

PROFESSEURS DES UNIVERSITES :

ANCEAU François	E.N.S.I.M.A.G.
BARRAUD Alain	E.N.S.I.E.G.
BAUDELET Bernard	E.N.S.I.E.G.
BESSON Jean	E.N.S.E.E.G.
BLIMAN Samuel	E.N.S.E.R.G.
BLOCH Daniel	E.N.S.I.E.G.
BOIS Philippe	E.N.S.H.G.
BONNETAIN Lucien	E.N.S.E.E.G.
BONNIER Etienne	E.N.S.E.E.G.
BOUVARD Maurice	E.N.S.H.G.
BRISSONNEAU Pierre	E.N.S.I.E.G.
BUYLE BODIN Maurice	E.N.S.E.R.G.
CAVAIGNAC Jean-François	E.N.S.I.E.G.
CHARTIER Germain	E.N.S.I.E.G.
CHENEVIER Pierre	E.N.S.E.R.G.
CHERADAME Hervé	U.E.R.M.C.P.P.
CHERUY Arlette	E.N.S.I.E.G.
CHIAVERINA Jean	U.E.R.M.C.P.P.
COHEN Joseph	E.N.S.E.R.G.
COUMES André	E.N.S.E.R.G.
DURAND Francis	E.N.S.E.E.G.
DURAND Jean-Louis	E.N.S.I.E.G.
FELICI Noël	E.N.S.I.E.G.
FOULARD Claude	E.N.S.I.E.G.
GENTIL Pierre	E.N.S.E.R.G.
GUERIN Bernard	E.N.S.E.R.G.
GUYOT Pierre	E.N.S.E.E.G.
IVANES Marcel	E.N.S.I.E.G.
JAUSSAUD Pierre	E.N.S.I.E.G.
JOUBERT Jean-Claude	E.N.S.I.E.G.
JOURDAIN Geneviève	E.N.S.I.E.G.
LACOUME Jean-Louis	E.N.S.I.E.G.
LATOMBE Jean-Claude	E.N.S.I.M.A.G.

.../...

LESSIEUR Marcel	E.N.S.H.G.
LESPINARD Georges	E.N.S.H.G.
LONGEQUEUE Jean-Pierre	E.N.S.I.E.G.
MAZARE Guy	E.N.S.I.M.A.G.
MOREAU René	E.N.S.H.G.
MORET Roger	E.N.S.I.E.G.
MOSSIERE Jacques	E.N.S.I.M.A.G.
PARIAUD Jean-Charles	E.N.S.E.E.G.
PAUTHENET René	E.N.S.I.E.G.
PERRET René	E.N.S.I.E.G.
PERRET Robert	E.N.S.I.E.G.
PIAU Jean-Michel	E.N.S.H.G.
POLOUJADOFF Michel	E.N.S.I.E.G.
POUPOT Christian	E.N.S.E.R.G.
RAMEAU Jean-Jacques	E.N.S.E.E.G.
RENAUD Maurice	U.E.R.M.C.P.P.
ROBERT André	U.E.R.M.C.P.P.
ROBERT François	E.N.S.I.M.A.G.
SABONNADIERE Jean-Claude	E.N.S.I.E.G.
SAUCIER Gabrielle	E.N.S.I.M.A.G.
SCHLENKER Claire	E.N.S.I.E.G.
SCHLENKER Michel	E.N.S.I.E.G.
SERMET Pierre	E.N.S.E.R.G.
SILVY Jacques	U.E.R.M.C.P.P.
SOHM Jean-Claude	E.N.S.E.E.G.
SOUQUET Jean-Louis	E.N.S.E.E.G.
VEILLON Gérard	E.N.S.I.M.A.G.
ZADWORNY François	E.N.S.E.R.G.

PROFESSEURS ASSOCIES

BASTIN Georges	E.N.S.H.G.
BERRIL John	E.N.S.H.G.
CARREAU Pierre	E.N.S.H.G.
GANDINI Alessandro	U.E.R.M.C.P.P.
HAYASHI Hirashi	E.N.S.I.E.G.

PROFESSEURS UNIVERSITE DES SCIENCES SOCIALES (Grenoble II)

BOLLIET Louis
Chatelin Françoise

PROFESSEURS E.N.S. Mines de Saint-Etienne

RIEU Jean
SOUSTELLE Michel

CHERCHEURS DU C.N.R.S.

FRUCHART Robert	Directeur de Recherche
VACHAUD Georges	Directeur de Recherche

.../...

ALLIBERT Michel	Maître de Recherche
ANSARA Ibrahim	Maître de Recherche
ARMAND Michel	Maître de Recherche
BINDER Gilbert	
CARRE René	Maître de Recherche
DAVID René	Maître de Recherche
DEPORTES Jacques	
DRIOLE Jean	Maître de Recherche
GIGNOUX Damien	
GIVORD Dominique	
GUELIN Pierre	
HOPFINGER Emil	Maître de Recherche
JOUD Jean-Charles	Maître de Recherche
KAMARINOS Georges	Maître de Recherche
KLEITZ Michel	Maître de Recherche
LANDAU Ioan-Dore	Maître de Recherche
LASJAUNIAS J.C.	
MERMET Jean	Maître de Recherche
MUNIER Jacques	Maître de Recherche
PIAU Monique	
PORTESEIL Jean-Louis	
THOLENCE Jean-Louis	
VERDILLON André	

CHERCHEURS du MINISTERE de la RECHERCHE et de la TECHNOLOGIRE (Directeurs et Maîtres de Recherches, ENS Mines de St. Etienne)

LESBATS Pierre	Directeur de Recherche
BISCONDI Michel	Maître de Recherche
KOBYLANSKI André	Maître de Recherche
LE COZE Jean	Maître de Recherche
LALAUZE René	Maître de Recherche
LANCELOT Francis	Maître de Recherche
THEVENOT François	Maître de Recherche
TRAN MINH Canh	Maître de Recherche

PERSONNALITES HABILITEES à DIRIGER des TRAVAUX de RECHERCHE (Décision du Conseil Scientifique)

ALLIBERT Colette	E.N.S.E.E.G.
BERNARD Claude	E.N.S.E.E.G.
BONNET Rolland	E.N.S.E.E.G.
CAILLET Marcel	E.N.S.E.E.G.
CHATILLON Catherine	E.N.S.E.E.G.
CHATILLON Christian	E.N.S.E.E.G.
COULON Michel	E.N.S.E.E.G.
DIARD Jean-Paul	E.N.S.E.E.G.
EUSTAPOPOULOS Nicolas	E.N.S.E.E.G.
FOSTER Panayotis	E.N.S.E.E.G.

.../...

GALERIE Alain	E.N.S.E.E.G.
HAMMOU Abdelkader	E.N.S.E.E.G.
MALMEJAC Yves	E.N.S.E.E.G. (CENG)
MARTIN GARIN Régina	E.N.S.E.E.G.
NGUYEN TRUONG Bernadette	E.N.S.E.E.G.
RAVAINE Denis	E.N.S.E.E.G.
SAINFORT	E.N.S.E.E.G. (CENG)
SARRAZIN Pierre	E.N.S.E.E.G.
SIMON Jean-Paul	E.N.S.E.E.G.
TOUZAIN Philippe	E.N.S.E.E.G.
URBAIN Georges	E.N.S.E.E.G. (Laboratoire des ultra-réfractaires ODEILLON)
GUILHOT Bernard	E.N.S. Mines Saint Etienne
THOMAS Gérard	E.N.S. Mines Saint Etienne
DRIVER Julien	E.N.S. Mines Saint Etienne
BARIBAUD Michel	E.N.S.E.R.G.
BOREL Joseph	E.N.S.E.R.G.
CHOVET Alain	E.N.S.E.R.G.
CHEHIKIAN Alain	E.N.S.E.R.G.
DOLMAZON Jean-Marc	E.N.S.E.R.G.
HERAULT Jeanny	E.N.S.E.R.G.
MONLLOR Christian	E.N.S.E.R.G.
BORNARD Guy	E.N.S.I.E.G.
DESCHIZEAU Pierre	E.N.S.I.E.G.
GLANGEAUD François	E.N.S.I.E.G.
KOFMAN Walter	E.N.S.I.E.G.
LEJEUNE Gérard	E.N.S.I.E.G.
MAZUER Jean	E.N.S.I.E.G.
PERARD Jacques	E.N.S.I.E.G.
REINISCH Raymond	E.N.S.I.E.G.
ALEMANY Antoine	E.N.S.H.G.
BOIS Daniel	E.N.S.H.G.
DARVE Félix	E.N.S.H.G.
MICHEL Jean-Marie	E.N.S.H.G.
OBLED Charles	E.N.S.H.G.
ROWE Alain	E.N.S.H.G.
VAUCLIN Michel	E.N.S.H.G.
WACK Bernard	E.N.S.H.G.
BERT Didier	E.N.S.I.M.A.G.
CALMET Jacques	E.N.S.I.M.A.G.
COURTIN Jacques	E.N.S.I.M.A.G.
COURTOIS Bernard	E.N.S.I.M.A.G.
DELLA DORA Jean	E.N.S.I.M.A.G.
FONLUPT Jean	E.N.S.I.M.A.G.
SIFAKIS Joseph	E.N.S.I.M.A.G.
CHARUEL Robert	U.E.R.M.C.P.P.
CADET Jean	C.E.N.G.
COEURE Philippe	C.E.N.G. (LETI)

.../...

DELHAYE Jean-Marc	C.E.N.G. (STT)
DUPUY Michel	C.E.N.G. (LETI)
JOUVE Hubert	C.E.N.G. (LETI)
NICOLAU Yvan	C.E.N.G. (LETI)
NIFENECKER Hervé	C.E.N.G.
PERROUD Paul	C.E.N.G.
PEUZIN Jean-Claude	C.E.N.G. (LETI)
TAIEB Maurice	C.E.N.G.
VINCENDON Marc	C.E.N.G.

LABORATOIRES EXTERIEURS

DEMOULIN Eric	C.N.E.T.
DEVINE	C.N.E.T. (R.A.B.)
GERBER Roland	C.N.E.T.
MERCKEL Gérard	C.N.E.T.
PAULEAU Yves	C.N.E.T.
GAUBERT C.	I.N.S.A. Lyon

RESUME

Des études récentes montrent que le modèle de collage logique ne convient pas pour représenter les défauts réels qui peuvent survenir dans les circuits intégrés. Ceci a amené certains auteurs à chercher des méthodes de test basées sur des hypothèses de pannes analytiques. Dans cette étude, le problème de conception des circuits autotestables vis à vis d'hypothèses de pannes analytiques, est abordé. Des méthodes et des règles générales de conception de circuits fonctinnels N-MOS "Stongly Fault Secure" sont proposées. De nouveaux codes sont introduits. La définition de la plus grande classe de contrôleurs, "Strongly Code Disjoint", qui peuvent exister est donnée. Des examples validant les méthodes sont donnés. Enfin, les méthodes sont utilisées pour l'étude d'un microprocesseur MC 68000 autotestable.

MOTS-CLES : Circuits autotestables - Circuits "Stongly Fault Secure" - Hypothèses de pannes - Codes - Contrôleurs - Conception VLSI - Microprocesseurs.

Στούς γονείς μου,

AVANT-PROPOS

Je tiens à remercier,

- *Monsieur Louis BOLLIET, Professeur à l'Université de Grenoble, pour l' honneur qu'il me fait en acceptant de présider le jury de cette thèse.*
- *Monsieur Bernard COURTOIS, Chargé de Recherche au CNRS et rapporteur de cette thèse, pour ses conseils, ses encouragements et sa confiance qui m'ont permis de mener à bien la présente étude.*
- *Monsieur François ANCEAU, Professeur à l'Institut National Polytechnique de Grenoble, pour m'avoir accueilli dans son Equipe de Recherche.*
- *Messieurs Joseph BOREL, Directeur Technique à la Société EFCIS, Alain COSTES, Professeur à l'Institut National Polytechnique de Toulouse, Jaques LEROUDIER, Responsable du laboratoire d' Architecture de Systémes Informatiques à la société THOMSON, Jean Marie RATA, Chargé des missions au service Informatique et Mathématiques Appliques à l'EDF, qui ont bien voulu participer au jury de cette thèse malgré leurs nombreuses occupations.*
- *tous les chercheurs de l'Equipe de Recherche en Architecture d'Ordinateurs, sans les travaux desquels ce document n'aurait pu voir le jour; et particulierement Pierre MARCHAL, Laurent BERGHER et Carlos POSTIGO pour leur aide et leurs précieux conseils et Ingrid JANSCH pour sa collaboration.*
- *la secretaire du laboratoire Hélène DIAZ qui a assuré la totalité de la dactylographie.*
- *la Service de Reprographie de l' IMAG : D. IGLESIAS ainsi que C.ANGUILLE, C.LABORIE, J.M. MOLLIER, P.MOUNET. La présentation de cette thèse est reflet de leur compétence.*

La deuxième partie de cette thèse a été réalisée dans le cadre d'un contrat de recherche soutenu par l' EDF.

PREMIERE PARTIE

CONCEPTION DES SYSTEMES AUTOTESTABLES POUR DES HYPOTHESES DE PANNES ANALYTIQUES

SOMMAIRE

CONCEPTION DE SYSTEME AUTOTESTABLE POUR DES HYPOTHESES DE PANNES ANALYTIQUES

I - INTRODUCTION

Des études récentes montrent que le modèle de collages logiques ne convient pas pour représenter les défauts réels qui peuvent survenir dans les circuits intégrés. Ceci est mis en évidence par GALIAY et al. dans [GAL 80] pour la technologie des transistors MOS monocanal. Certaines pannes ne sont pas modélisables par des collages logiques . D'autre part, certains collages logiques ne sont pas représentatifs de pannes réelles intervenant au niveau du transistor. Les problèmes spéciaux qui peuvent apparaître dans les circuits C.MOS sont exposés par WADSACK dans [WAD 78]. Ceci a amené quelques auteurs à chercher des vecteurs de test et/ou des simulations basées sur des hypothèses de pannes de niveau inférieur: collage ouvert [ELZ 81] ou "conductance" fault model [MAL 82] pour C.MOS, collage fermé/collage ouvert de transistors MOS [GAL 80], [ELZZ 79], [BOS 82], [CHI 82]. Pour les autres technologies comme TTL, on peut citer [HLA 76] et [BEH 82].

En conclusion:
- le modèle de collage logique n'est pas représentatif pour la génération des vecteurs de test.
- la conception des circuits basée sur ce modèle ne peut pas être efficace et particulièrement, la conception des circuits autotestables basée sur de telles hypothèses de pannes.

Dans cette étude, des règles générales de conception sont données pour la conception de circuits N.MOS "strongly fault secure" qui sont la plus large classe de circuits pouvant atteindre le "totaly self-checking goal".

Des exemples d'application de ces règles sont donnés pour des PLAs pour lesquels des schémas nouveaux sont donnés et des schémas

connus sont retrouvés de manière rapide.

Ensuite, les contrôleurs "strongly code disjoint" sont définis. Ces contrôleurs sont la classe la plus large de contrôleurs qui peuvent être définis. Des contrôleurs "strongly code disjoint" sont les compagnons naturels et nécessaires des circuits "strongly fault secure", avec lesquels ils forment une classe de systèmes atteignant le "totaly self checking goal". Des exemples de conception des contrôleurs "strongly code disjoint" pour quelques codes classiques sont donnés, basés sur des hypothèses de pannes analytiques.

Dans la deuxième partie de cette etude, la méthode développée ici est utilisée pour l'étude d'un microprocesseur MC 68000 autotestable.

I.1. Hypothèses de pannes retenues

Dans la suite, on ne considérera que la technologie de MOS monocanal. Les hypothèses de pannes retenues sont les différentes classes définies dans [COU 81]. Ces classes sont listées ci-dessous et sont regroupées à la table I. La classe 1 est retenue pour la conçeption de circuits autotestables mais l'extension de l'étude aux autres classes est aussi discutée.

On considérera que les pannes modélisées par ces classes ne provoquent que des erreurs logiques (à une tension sera toujours associé un niveau logique 0 ou 1 selon un processus déterministe).

Classe 0

Cette classe est celle des défauts simples physiques, tels par exemple qu'un contact défaillant, un MOS défaillant, un conducteur coupé. Les courts- circuits ne font pas partie de cette classe: on considère aue deux "défauts" sont nécessaires. En cas de grille flottante, on considère que le mode de panne associé est un MOS s-on ou s-open. La notion de défaut simple est naturellement relative

à un circuit de référence.

Classe I

Par rapport à la classe 0, on admet les courts-circuits entre Al, uniquement et entre un Al et l'autre Al le plus proche géographiquement. Bien que le cas similaire entre deux diffusions soit sans doute moins probable, on le considérera néanmoins, car la difficulté du problème est comparable.

Classe II

Cette classe contient naturellement ce qui n'a pas été inclus dans les classes 0 et I, tels que les courts-circuits entre un Al et un autre Al quelconque dans le circuit. Ces courts-circuits sont également étendus à plusieurs participants: 3, 4... Le cas des diffusions est étendu de façcon semblable. Les défauts sont considérés comme multiples, au sens classique du terme. Seuls les défauts classés "fabrication seulement" restent exclus (courts-circuits Al -diffusion ...).

classe 0	classe I	classe II
un défaut physique simple: 1 contact, 1 pré-contact 1 MOS s-on ou s-open, 1 Al coupé... grille flottante->MOS s-	classe 0+ : 1 court-circuit entre 1 Al et l'autre Al le plus proche géographiquement. Idem pour diffusions	classe I+ : courts-circuits entre Al quelconques idem diffusion défauts multiples

Table I - Classes d'hypothéses de pannes [COU 81]

I.2. Définitions de base

Les conventions qui suivent sont reprises de [SMI 78]. Nous

considérons un circuit G en logique combinatoire possédant des entrées multiples et des sorties multiples. Si on a un circuit G avec r entrées primaires, les 2^r vecteurs binaires de longueur r forment l'ensemble X des vecteurs d'entrée. L'ensemble Y de vecteurs de sortie est formé par les 2^q vecteurs binaires de longueur q, le nombre des sorties du circuit G étant égal à q.

Pendant le fonctionnement normal (avant l'occurrence des défauts) , le circuit G reçoit à ses entrées un sous-ensemble A de vecteurs de X, appelé code d'entrée et il produit un sous-ensemble B de vecteurs de Y, appelé code de sortie. En présence de défauts, le circuit peut produire des sorties en dehors du code de sortie B.

En présence d'un défaut f, la sortie du circuit G qui reçoit un vecteur d'entrée x, peut être dénotée G(x,f). Avant l'occurrence des défauts, elle est dénotée G(x,0).

Les définitions suivantes sont dues à ANDERSON [AND 71]. On considérera un bloc fonctionnel G, avec un code d'entrée A, un code de sortie B et un ensemble de défauts F.

Définition D1

G est "self testing" pour un ensemble F des défauts si :

$\forall f \in F, \exists a \in A$ tel que $G(a,f) \notin B$.

Définition D2

Un circuit G est "fault secure" (FS) pour un ensemble de défauts F si:

$\forall f \in F, \forall a \in A$ on a soit $G(a,f) = G(a,0)$

soit $G(a,f) \notin B$

Définition D3

Un circuit G est "totaly self checking" (TSC) pour un ensemble de défauts F : si le circuit est "self testing" et "fault secure" pour

l'ensemble F.

Propriété Pr0

Un bloc fonctionnel G accomplit le "totaly self checking goal" si la première sortie erronée due aux défauts de l'ensemble F, est en dehors du code de sortie.

Hypothèse H1

Entre l'occurrence de deux défauts quelconques appartenant à l'ensemble F, il s'écoule un laps de temps suffisant pour que tous les vecteurs de code d'entrée A soient appliqués aux entrées du circuit G.

L'hypothèse H1 étant respectée, un circuit TSC accomplit le "totaly self checking goal".

Définition D4

Un circuit G est à codes disjoints si:

$\forall a \in A,\ G(a,0) \in B$

$\forall a \notin A,\ G(a,0) \notin B$

Définition D5

Un circuit G est un controleur TSC s'il est totaly self checking et à codes disjoints.

Définition D6

Un circuit G es redondant pour un défaut f si la fonction qu'il réalise n'est pas modifiée en présence du défaut f [BRE 76].

La dernière définition peut être raffinée, étant donné que G peut être redondant pour l'ensemble X ou seulement pour le code d'entrée A.

Définition D7

Un circuit G est redondant pour un défaut f et pour l'ensemble des vecteurs d'entrée X si

$\forall x \in X$, $G(x,f) = G(x,0)$.

Définition D8

Un circuit G est redondant pour un défaut f et pour le code d'entrée A si:

$\forall a \in A$, $G(a,f) = G(a,0)$

Il est évident que:

- si un circuit est redondant pour un défaut f et pour l'ensemble X, il est redondant pour le même défaut f et pour le code d'entrée A.
- un circuit peut être redondant pour un défaut f et pour le code d'entrée A, sans être redondant pour le défaut f et pour l'ensemble X.

I.3. Strongly fault secure circuits

Les circuits "strongly fault secure" sont introduits par SMITH and METZE pour surmonter la difficulté présentée par les circuits qui ne sont pas TSC à cause de perte de la propriété "self testing". Si un circuit G est "fault secure" pour un ensemble de défauts F mais s'il n'est pas "self testing", alors il existe un défaut $f1 \in F$ qui n'est pas détectable et un deuxième défaut $f2 \in F$ peut survenir dans le circuit. Dans ce cas, le défaut f1 f2 (la combinaison des deux défauts) est présent dans le circuit, mais il est possible que le défaut f1f2 n'appartienne pas à F et que les sorties erronées provoquées par f1f2 ne soient pas en dehors du code de sortie.

Ce problème est surmonté si le circuit ne perd pas la priorité "fault secure" en présence du défaut f1. Pour une séquence $\langle f1, f2, \ldots fn \rangle$ le circuit peut être "fault secure" pour chaque

subséquence initiale et "self testing" pour la combinaison des défauts de la séquence. Ceci est formalisé par la définition suivante [SMI 78]:

Définition D9

Pour une séquence des défauts <f1,f2,...fn> soit k le plus petit entier pour lequel:

$\exists a \in A,\ G(a, \bigcup_{j=1}^{k} fj) \notin G(a,0)$

Si un tel k n'existe pas, posons k=n. Alors G is "strongly fault secure" (SFS) pour la séquence <f1,f2,..fn> si:

$\forall a \in A$: soit $G(a, \bigcup_{j=1}^{k} fj) = G(a,0)$

soit $G(a, \bigcup_{j=1}^{k} fj) \notin B$

Définition D10

Un circuit G est "strongly fault secure" pour un ensemble de défauts F s'il est "strongly fault secure" pour chaque séquence constituée d'éléments appartenant à l'ensemble F.

Il est évident que chaque circuit TSC est SFS (k=1 à la définition D9).

L'hypothèse H1 etant respectée , on peut démontrer que chaque circuit SFS accomplit le "totaly self checking goal". D'autre part, un circuit qui n'est pas SFS peut produire une sortie erronée en dehors du code de sortie.

En conclusion, les circuits SFS constituent la plus grande classe de circuits qui accomplissent le "TSC goal".

Les circuits SFS sont intéressants vis à vis de défauts pour lesquels les circuits sont redondants.

Supposons un circuit FS pour un ensemble de défauts F, supposons aussi que ce circuit est redondant pour un défaut $f \in F$, alors ce circuit n'est pas TSC, mais il peut être SFS.

Dans [SMI 78] il est indiqué comment constituer un circuit G' réduit, à partir d'un circuit SFS G redondant pour certains défauts f∈F (∀a∈A,G(A,f) = G(a,0)). Le circuit obtenu est SFS pour un ensemble de défauts F' réduit. On peut appliquer itérativvement ce processus jusqu'à obtenir un circuit TSC.

Ce processus est basé sur le modèle du collage logique. Si le modèle de défaut est assez général, il peut être impossible de construire un circuit TSC, bien qu'il soit posible de construire un circuit SFS.

Ultérieurement, les circuits SFS seront utilisés car il ne sera pas toujous possible d'éliminer les redondances, si on considère des hypothèses de pannes au niveau du MOS, comme elles sont définies au chapître II.

II - CIRCUITS SFS ET HYPOTHESES DE PANNES DE BAS NIVEAU

La structure générale des circuits SFS est donnée à la figure 1.

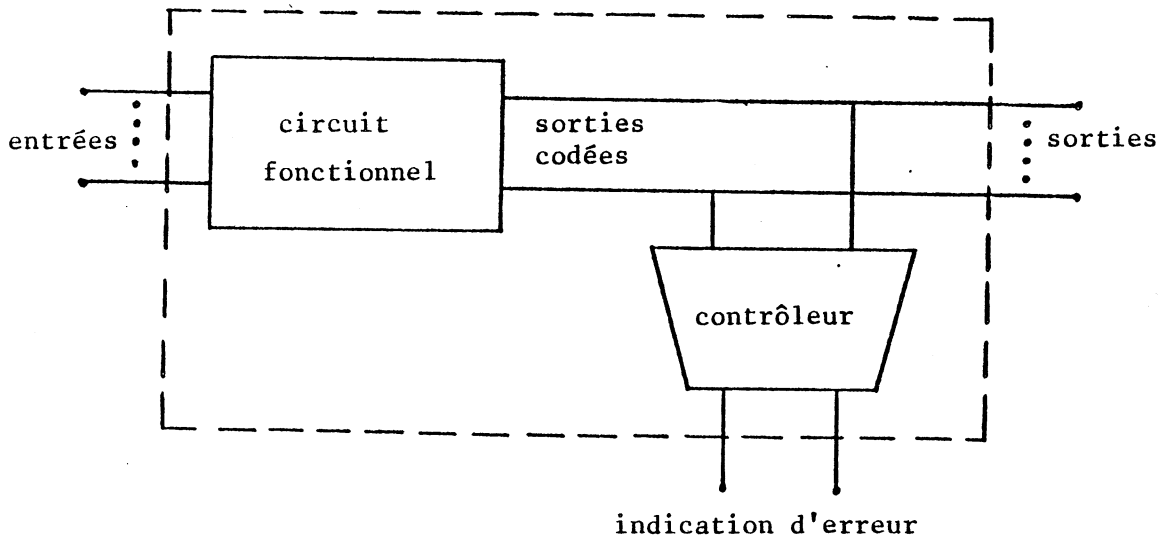


Figure 1- Structure générale d'un systeme SFS

Les indicateurs d'erreur sont utilisés pour indiquer la manifestation des défauts.

Le circuit fonctionnel transpoe des entrées encodées en sorties encodées et le contrôleur transpose les sorties encodées en signaux d'indication d'erreur.

Pour avoir un bloc global qui accomplit le "TSC goal", il est nécessaire d'utiliser un circuit fonctionnel SFS. Le circuit de contrôle pourrait être TSC. De toute façon le circuit de contrôle doit être "Strongly code disjoint". Ce type de contrôleurs est introduit à la fin de cette étude et ils forment la plus grande classe de contrôleurs qui puissent garantir le "totaly self checking goal" du bloc global. Le problème de conception des systèmes autotestables sera abordé en accord avec cette structure générale.

II.1. Circuits SFS pour une classe C des hypothèses de pannes

En ayant comme base:
- la théorie des circuits TSC/SFS,
- la technologie des circuits intégrés (NMOS, CMOS...),

on doit concevoir des circuits autotestables pour une des technologie existantes, en considérant des hypothèses de pannes au niveau du transistor. On sait que pour les hypothèses de pannes de ce niveau, on ne peut pas toujours éliminer les redondances, c'est pourquoi la conception des circuits SFS doit être le but global.

Pour les hypothèses de pannes du niveau des transistors, le nombre de défauts est beaucoup plus élevé que le nombre de défauts pour le modèle du collage logique (par exeple le nombre des courts-circuits pour un circuit de m lignes croît selon le carré de m). Pour éviter l'utilisation de listes énormes de défauts individuels, la définition des circuits SFS est modifiée pour couvrir un classe des hypothèses de pannes. Ultérieurement, une classe des hypothèses de pannes sera dénotée par C.

L'introduction de quelques définitions es nécessaire avant de donner une définition des circuits SFS pour une classe C.

Définition D11

Un circuit est "strongly redundant" pour une séquence de défauts <f1,f2,..fn> et pour le code d'entrée A (resp. pour l'ensemble de vecteurs d'entrée X), si le circuit est edondant pour les n séquences <f1>, <f1,f2>,..., <f1,f2,...fn> et pour le code d'entrée A (resp. pour X).

Dans cette définition l'ordre des défauts dans la séquence <f1,f2,...fn> est utilisé pour définir l'ordre d'occurrence des défauts c'est à dire le défaut f1 survient le premier, le défaut f2 le deuxième etc... On sait que la propriété de redondance est dynamique [FRI 67], [BRE 76], un circuit peut être strongly redundant pour la séquence <f1,f2, fi,fj...fn> sans être strongly redundant pour la séquence <f1,f2,..fj,fi,..,fn >. Un circuit peut être redondant pour un défaut individuel fi selon l'existence d'autre défauts dans le circuit.

Définition D12

Un circuit est "falt secure" pour une classe C d'hypothèses de pannes s'il est "fault secure" pour tous les défauts dûs à la classe C.

Définition D13

Un circuit G est "strongly fault secure" pour une classe C d'hypothèses de pannes si:

- avant l'occurrence de défauts, G est "fault secure" pour la classe C.
- en présence de chaque séquence <f1,f2...fn> de défauts dûs à la classe C telle que le circuit G est "strongly redundant" pour la séquence <f1,f2...fn> et pour le code d'entrée A, le circuit G reste "fault secure" pour la classe C.

En assumant l'hypothèse H1 on peut démontrer que les circuits SFS pour une classe C accomplissent le "TSC goal".

On a précisé que notre but est la conception des circuits SFS pour une classe C d'hypothèses de pannes (au niveau du transistor). Le but des chapîtres qui suivent est de donner des règles générales de conception de circuits SFS selon la définition D13. Ces règles seront retenues pour:
- la classe I d'hypothèses de pannes (table I) pour la technologie NMOS.
- trois types de codes de sortie, correspondant aux trois types d'erreurs:
* erreurs simples,
* erreurs unidirectionnelles,
* erreurs multiples,
(ces erreurs peuvent être détectées respectivement par les codes de parité, par les codes k/n ou par les codes de BERGER, par les codes de duplication ou par les double-rail codes).

Par la suite sont présentées:
- les conséquences des défauts de la classe I pour la technologie NMOS,
- les règles de propagation des erreurs au niveau du transistor,
- les règles de conception des circuits SFS pour la classe I en utilisant des codes de sortie détectant des erreurs simples unidirectionnelles ou multiples.

II.2. Erreurs dûes à la classe I

Les erreurs provoquées par les défauts dûs à la classe I sont détaillés dans cette section. Elles seront utilisées plus tard pour obtenir les règles de conception des circuits SFS. On considère que les lignes d'alimentation sont faites soit en Alu, soit en diffusion.

1 - un contact, un précontact, un MOS s-on ou un MOS s-open -> une erreur simple sur la ligne concernée ou à la sortie de la porte concernée par le défaut.

2 - une ligne de diffusion coupée:
2a - une ligne de signal -> une erreur simple sur la ligne concernée ou sur la porte concernée par la coupure.
2b - une ligne d'alimentation -> une erreur multiple unidirectionnelle aux sorties des portes alimentées par la ligne.

3 - une ligne de poly coupée -> une erreur simple sur cette ligne.

4 - une ligne d'ALU coupée:
4a - une ligne de signal -> une erreur simple sur cette ligne
4b - une ligne d'alimentation -> une erreur multiple unidirectionnelle aux sorties des portes alimentées par ces lignes.

5 - un court-circuit entre une ligne d'Alu et une autre ligne d'Alu la plus proche géographiquement, ou un court-circuit entre une ligne de diffusion et une autre ligne de diffusion la plus proche géographiquement:
5a - un court-circuit entre deux lignes VDD ou deux lignes VSS -> pas d'erreur.
5b - un court-circuit entre une ligne VDD et une ligne VSS -> on considère que la destruction du circuit est totale.
5c - un court-circuit entre une ligne de signal et une ligne d'alimentation -> une erreur simple sur la ligne de signal (collage à la valeur logique VSS ou VDD).
5d - un court-circuit entre une ligne interne d'une porte et une ligne d'alimentation -> une erreur simple à la sortie de la porte.
5e - un court-circuit entre deux lignes de signal -> une erreur qui peut concerner les deux lignes mais seulement l'une des deux à chaque fois -> une erreur simple.
5f - un court-circuit entre deux lignes internes de deux portes différetes - > une erreur qui peut concerner les sorties de deux portes mais seulement l'une des deux à chaque fois -> une erreur

simple.
5g - un court-circuit entre une ligne de signal et une ligne interne d'une porte -> une erreur qui peut concerner la ligne de signal et la sortie de la porte mais l'une des deux à chaque fois, alors une erreur simple.
5k - un court-circuit entre deux lignes internes d'une porte -> une erreur simple à la sortie de la porte.

Tous ces cas peuvent être regroupés en quatre groupes listés ci-dessous:

A - Hypothèses de pannes qui peuvent provoquer des erreurs simples concernant une ligne de signal ou une sortie d'une porte (1, 2a, 3, 4a, 5c, 5d, 5h).

B - Hypothèses de pannes qui peuvent provoquer des erreurs concernant deux lignes signal/sorties de portes, mais l'une des deux à chaque fois -> une erreur simple (5e, 5f, 5g).

C - Hypothèses de pannes qui peuvent provoquer des erreurs multiples unidirectionnelles (2b, 4b).

D - Hypothèses de pannes ne provoquant pas d'erreurs (5a).

II.3. Propagation d'erreurs

Les défauts des types A, B et C provoquent des erreurs, à l'origine de leur occurrence, avec des propriétés définies à la section précédente. Les types d'erreurs qui seront produits aux sorties primaires d'un circuit fonctionnel dépendent de la propagation des erreurs depuis l'origine de leur apparition jusqu'aux sorties primaires du bloc fonctionnel.

Les propriétés de propagation d'erreurs sont utilisées dans [SMI 77] et [CRO 78], mais ces propriétés sont dérivées au niveau des portes logiques, en considérant par exemple des portes OR, NAND

etc... Par conséquent on doit transposer ces propriétés au niveau du transistor car on considère des hypothèses de pannes à ce niveau. Par la suite, nous définissons le chemin entre une ligne interne et les sorties primaires des circuits NMOS, le degré de divergence d'une ligne, le degré de divergence maximal d'un bloc et la parité d'inversion d'un chemin; toutes ces définitions sont faites au niveau du transistor.

II.3.1. Modélisation des circuits NMOS.

Un circuit N-MOS peut être modélisé par un réseau ouvert des articulations [KUN 72]. Les entrées du réseau ouvert sont les entrées primaires du circuit et certaines lignes d'alimentation et les sorties du réseau sont les sorties primaires du circuit et certaines lignes d'alimentation.

Les articulations d'un tel circuit sont divisées en deux groupes : les étoiles et les noeuds. Les étoiles sont :

MOS de signal

Le degré d'une telle articulation est 3. La grille de poly et la source du MOS sont des entrées et le drain du MOS est la sortie. Ceci est représenté à la figure 2a.

MOS de charge

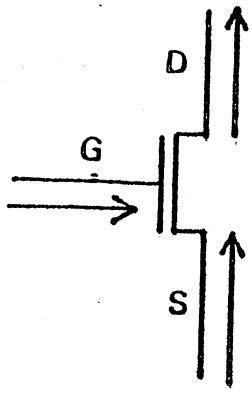
Le degré d'une telle articulation est 2. Le drain du MOS est une entrée et la source du MOS est une sortie. Ceci est présenté à la figure 2b.

MOS de précharge

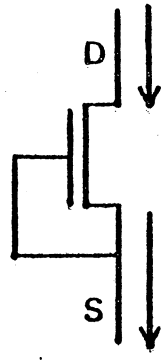
Le degré de cette articulation est 3. La grille de poly et le drain sont des entrées, la source est la sortie. Ceci est représenté à la figure 2c.

MOS interrupteur

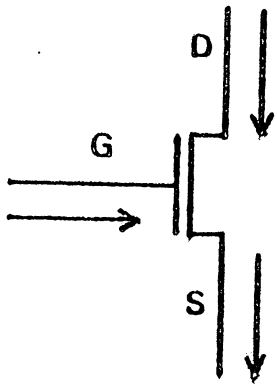
Le degré de cette articulation est 3. La source et le drain ne sont pas généralement définis vis à vis de la propriété d'orientation (entrée/sortie). La grille de poly est une entrée. Ceci est représenté à la figure 2d.



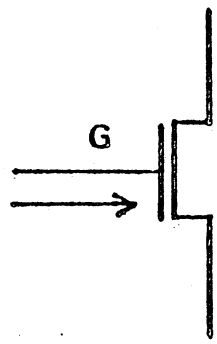
a/ Un MOS de signal



b/ Un MOS de charge



c/ Un MOS de précharge



d/ Un MOS interrupteur

Figure 2- Les articulations de type etoile

Les noeuds sont les parties équipotentielles du circuit. Le degré de ces articulations est supérieur ou égal à 2. Les noeuds sont constitués par des connecteurs et par des lignes inernes connectant les connecteurs entre eux.

Les lignes d'alimentation sont des noeuds spéciaux avec quelques connecteurs qui doivent être orientés par définition. Toutes les extrémités des lignes d'alimentation d'un bloc connectant le bloc à la source d'alimentation sont des entrées, toutes les extrémités des lignes d'alimentation d'un bloc qui sont utilisées pour l'alimentation d'autres blocs sont des sorties. L'extrémité d'une ligne VDD (VSS) connectée au drain d'un MOS de charge ou de précharge (à la source d'un MOS signal). est une sortie.

Tous les connecteurs qui ne sont pas orientés par définition seront orientés selon les règles suivantes:

- un connecteur qui n'est pas orienté par définition, sera une entrée s'il est connecté avec une sortie et il sera une sortie s'il est connecté avec une entrée.
- si un seul connecteur d'un noeud n'est pas orienté et que tous les autres sont des entrées (sorties) alors le connecteur non orienté sera une sortie (entrée).

Tous les autres connecteurs qui ne sont pas orientés ni par définition ni par application des deux règles précédentes, restent non orientés.

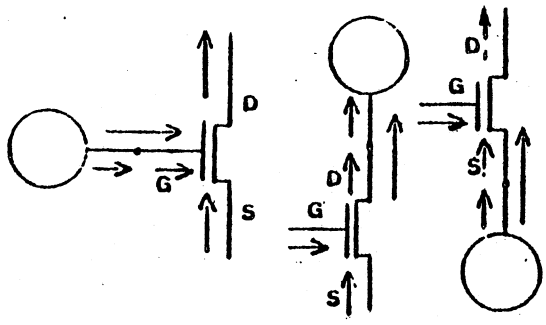
Par la suite on définit les lignes d'un circuit. Une ligne d'un circuit peut être:

Type 1: une connection entre un noeud et une etoile.

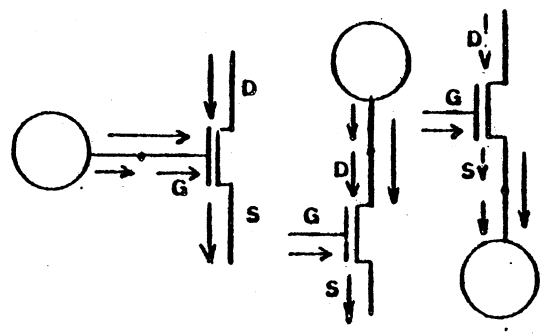
Type 2: la compression d'un noeud de degré égal à 2 et des deux connexions de ce noeud.

Type 3: une ligne interne d'un noeud.

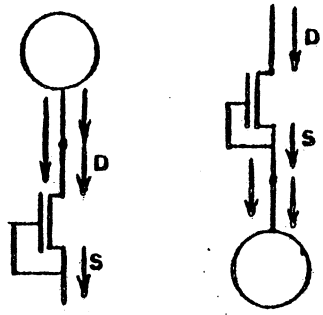
La figure 3a....f présente quelqeus cas de ces trois types de lignes.



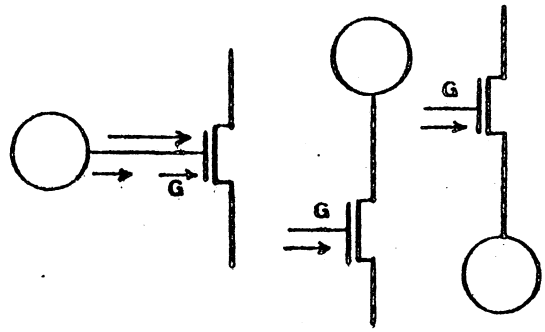
a/ Lignes de type 1 associées à un MOS de signal



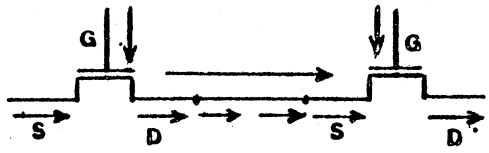
b/ Lignes de type 1 associées à un MOS de précharge



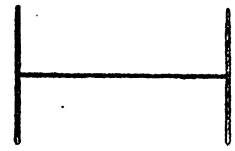
c/ Lignes de type 1 assocées à un MOS de charge



d/ Lignes de type 1 associées à un MOS interrupteur



e/ Ligne de type 2



f/ Ligne de type3

Figure 3- Les lignes d'un circuit N-MOS

II.3.2. La relation de succession

On considère deux lignes Ii, Ij ; la ligne Ij sera un successeur de la ligne Ii si:

- il existe une articulation A telle que la ligne Ii est une ligne de type 1 ou de type 2, associée avec une entrée de A et la ligne Ij est une ligne associée avec une sortie de A. Si A est un noeud, les lignes Ii et Ij doivent être connectées électriquement directement (sans utiliser une ligne interne de A).

- ou il existe un noeud A tel que la ligne Ii est une ligne de type 1 ou de type 2 associée avec une entrée de A et la ligne Ij est une ligne associée avec une ligne interne de A, ayant un point commun avec la ligne Ii.

- ou il existe un noeud A tel que la ligne Ij est une ligne de type 1 ou de type 2 associée avec une sortie de A et la ligne Ii est une ligne interne de A ayant un point commun avec la ligne Ij.

- ou il existe un noeud A tel que les lignes Ii et Ij sont des lignes internes de A ayant un point commun.

- ou il existe une étoile du type MOS interrupteur avec deux connecteurs non orientés tels que:
* soit la ligne Ii est une ligne de type 1 ou de type 2 associée à la grille de A et la ligne Ij est une ligne de type 1 ou de type 2 associée avec un des autres connecteurs de A ;
* soit les lignes Ii et Ij sont deux lignes de type 1 ou de type 2 associées avec les deux connecteurs non orientés de A.

Définition D14

Un chemin P est un ensemble ordonné des lignes <Ii1, Ii2,...,Iir> tel que Iij+1 est un successeur de Iij pour j=1 jusqu'à r-1.

II.3.3 La relation de succession inversant.

On considère deux lignes Ii et Ij, la ligne Ij est un successeur de la ligne Ii.

Le couple (Ii,Ij) sera appelé inversant si une erreur de type D "niveau haut à la place du niveau bas" (resp $\overline{D}$ "niveau bas à la place du niveau haut") sur la ligne Ii ne peut produire qu'une erreur de type $\overline{D}$ (resp. D) sur la ligne Ij.

Définition D16

Le couple (Ii,Ij) sera appelé non inversant si une erreur de type D (resp.$\overline{D}$) sur la ligne Ii ne peut produire qu'une erreur de type D (resp. $\overline{D}$) sur la ligne Ij.

Le couple (Ii,Ij) sera indéfini vis à vis de la propriété d'insertion s'il n'est ni inversant ni non inversant.

Les couples inversant/non inversant sont définis ci-après:

* le couple (Ii,Ij) de la figure 4a où l'étoile est un MOS de signal, est inversant.
* le couple (Ii,Ij) de la figure 4b où l'étoile est un MOS interrupteur, est inversant.
* le couple (Ii,Ij) de la figure 4c où l'étoile est un MOS interrupteur, est non inversnat.
* le couple (Ii,Ij) de la figure 4d où l'étoile est un MOS interrupteur et la ligne Ik est suceptible d'être portée aux deux valeurs logiques (0 ou 1) est indéfini vis à vis de la propriété d'insertion.
* tous les autres couples possibles sont non inversant.

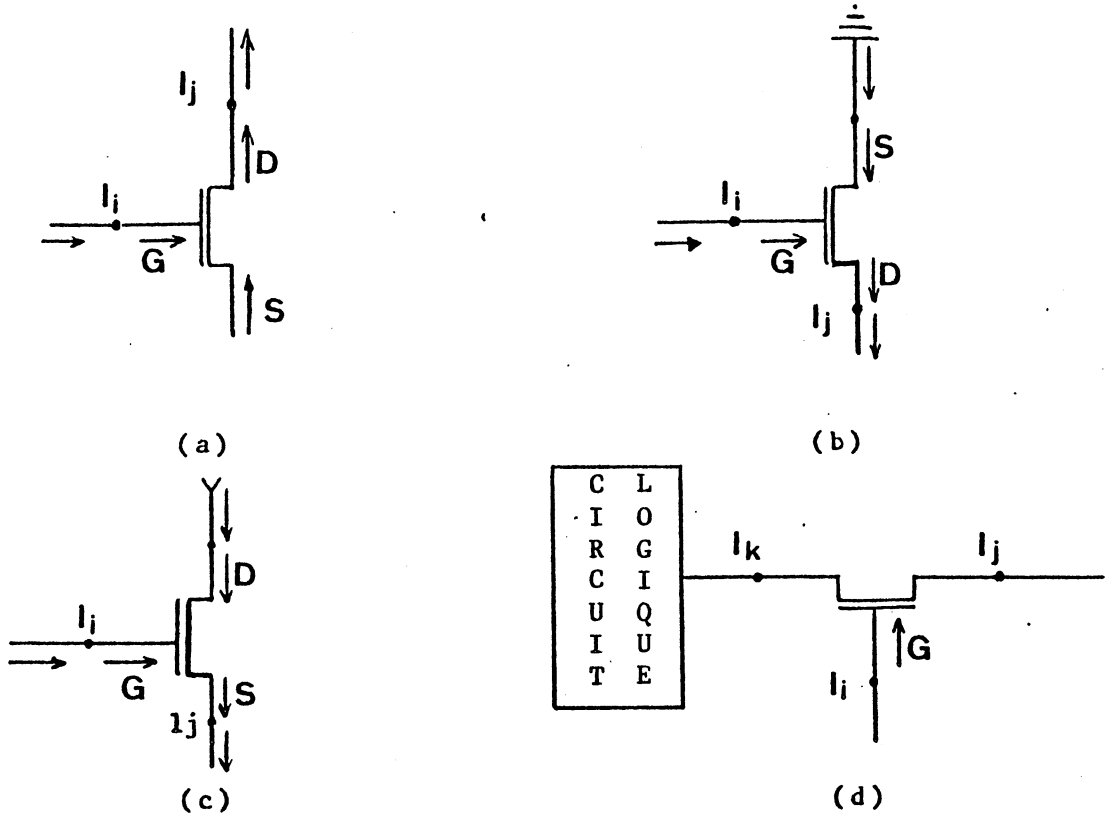


Figure 4- Couples inversant/non inversant

II.3.4. La parité d'inversion et le degré de divergence

Définition D17

La parité d'inversion IP(p) d'un chemin p, dans lequel la propriété d'inversion est définie pour tous les couples de deux lignes successives, est le nombre binaire $IP(p) \in \{0,1\}$, tel que IP(p) = n modulo 2, n étant le nombre des couples inversant de lignes successives du chemin p.

Si la propriété d'inversion est indéfinie pour quelques couples du chemin, la parité d'inversion du chemin est indéfinie.

Ces définitions permettent de considérer la parité d'inversion de chemins entre une ligne interne d'une porte et une sortie primaire, entre une ligne d'alimentation et une ligne de signal etc... Ceci n'était pas possible avec la définition de parité d'inversion au niveau logique.

Dans le cas des portes constituées par un MOS de charge et un réseau séries parallèles de MOS de signal, la parité d'inversion des chemins entre une ligne interne d'une porte et une sortie primaire du circuit est égale à la parité d'inversion (donnée par la definition au niveau de portes [SMI 77]) du chemin entre la sortie de la porte et de la sortie primaire.

Ce n'est pas le cas d'autres portes. La figure 5 est un exemple de porte pour laquelle les parités d'inversion des chemins entre une entrée et la sortie ne sont pas égales. Cette porte est un exemple pour lequel les définitions de [SMI 77] ne peuvent pas être utilisées.

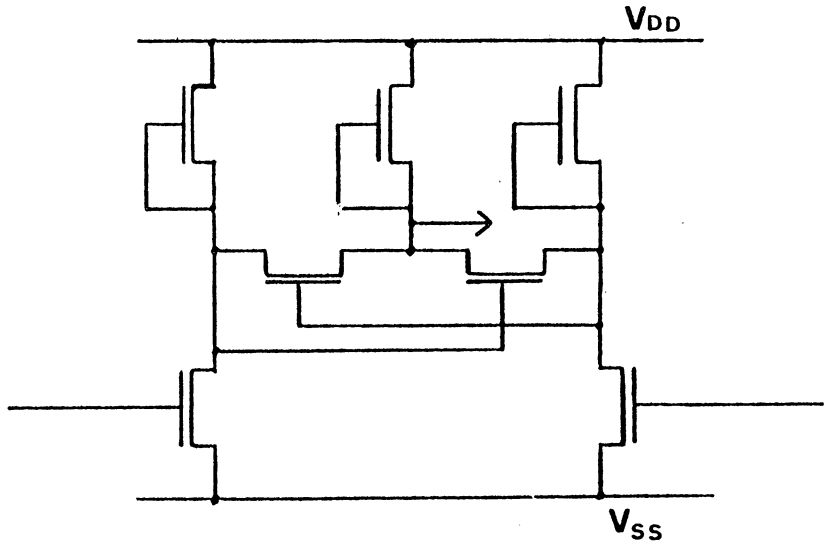


Figure 5- La définition classique de parité d'inversion ne doit pas être appliquée pour cette porte.

Définition D18

Le degré de divergence d'une ligne Ii,1 interne d'un bloc fonctionnel est le nombre des sorties primaires Ii,r du bloc pour lesquelles il existe des chemins <Ii,1,...,Ii,r>.

Définition D19

Le degré de divergence maximal d'un bloc fonctionnel, pour un ensemble de lignes internes I1,I2,...,Ik est égal au maximum des degrés de divergence des lignes I1,I2,...,Ik.

Remarque

La définition D14 du chemin concerne principalement la propagation des erreurs depuis les lignes internes de portes jusqu'aux sorties des portes et ensuite jusqu'aux sorties primaires du bloc. En effet les sorties primaires des blocs fonctionnels sont les points d'observabilité des circuits autotestables. Ainsi il existe des propagations d'erreurs pour lesquelles la définition D14 n'est pas convenable. La propagation présentée figure 6 en est un exemple: il existe une propagation possible depuis la ligne Ii jusqu'à la ligne

Ij qui n'est pas couverte par la définition D14. Une autre définition du chemin sera nécessaire pour d'autres points d'observabilité, par exemple pour les observations par le microscope électronique à balayage où les Ii et les Ij sont observables [BAI 83].

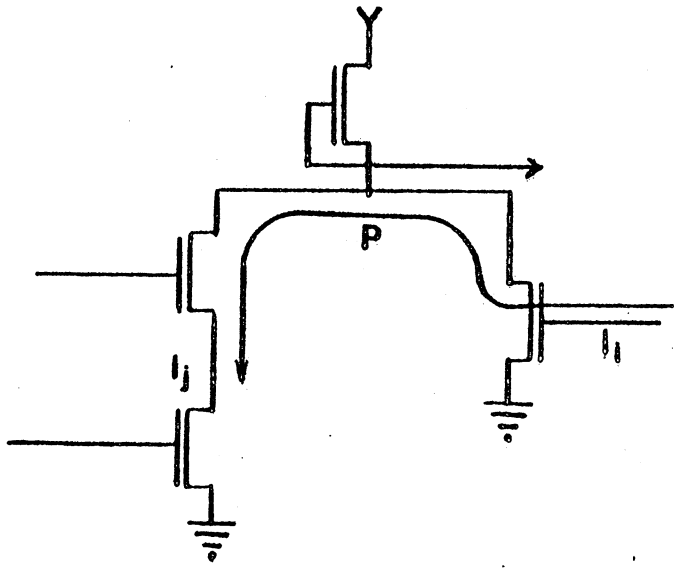


Figure 6- Une propagagation d'erreurs non considerée par la définition D14.

III - REGLES DE CONCEPTION DES CIRCUITS SFS POUR LA CLASSE I.

On propose successivement les règles de conception des blocs fonctionnels SFS dont le code de sortie détecte des erreurs simples, des erreurs unidirectionnelles ou des erreurs multiples.

Il s'agit de règles du dessin similaires aux règles données dans [GAL 80]. Dans [GAL 80] les règles sont données afin de concevoir des circuits testables, tandis que dans l'étude présente les règles sont données afin de concevoir des circuits SFS. On peut noter de façon approximative, que les règles données ici sont moins fortes que les règles données dans [GAL 80]. Par exemple, il est peut-être souhaitable dans [GAL 80] d'éviter la possibilité d'occurrence d'un défaut, car ce défaut n'est pas facilement détectable dans le cas où il n'est pas représentable par le modèle du collage logique. Un tel défaut n'empêche pas nécessairement un circuit d'être SFS.

En accord avec la Définition D13, deux groupes de règles sont utilisées. Le premier groupe introduit la propriété "fault secure" avant l'occurrence de défauts. Le deuxième groupe garantit que le circuit reste "fault secure" en présence de séquences de défauts pour lesquels le circuit est "strongly redundant".

III.1. Erreurs simples

Dans cette section on définit les règles de conception des circuits SFS dont le code de sortie détecte les erreurs simples.

III.1.1. Circuits "fault secure"

Règle R1

Le degré de divergence maximal du bloc fonctionnel, pour l'ensemble de toutes les lignes du bloc [illegible]

Le règle R1 est donnée pour assurer que toutes les erreurs simples internes au circuit seront propagées en erreurs simples aux sorties primaires du circuit. Cette règle est a niveau logique ; si elle est combinée avec la "réduction" des circuits [SMI 76] (élimination des redondances), l'ensemble est suffisant pour concevoir des circuits TSC pour le modèle du collage logique. Pour les modèles de pannes au niveau du transistor, quelques autres règles de conception sont nécessaires pour concevoir des circuits FS et SFS.

Règle R2

Une seule sortie primaire du bloc fonctionnel peut être erronée à cause d'une erreur multiple unidirectionnelle provoquée dans le circuit par un défaut du type C (section II.2).

La règle R2 étant très générale, il est très difficile de la mettre en oeuvre. Nous donnons ci-après des règles afin de faciliter cette tâche.

Règle R2'

Chaque ligne d'alimentation (diffusion ou aluminium) du bloc est utilisée pour alimenter des groupes de portes qui sont liées (par l'intermédiaire de chemins) à une seule sortie primaire du bloc fonctionnel.

L'application de la règle R2' est possible étant donné que chaque porte du circuit est reliée à une seule sortie primaire (règle R1).

Règle R2"

Une seule ligne VSS (diffusion ou aluminium) et une seule ligne VDD (diffusion ou aluminium) sont utilisées. L'extrèmitè de ces lignes est utilisée pour alimenter des portes d'un contrôleur.

Proposition P1

Si la règle R2' est vérifiée pour un circuit, alors la règle R2 est aussi vérifiée.

La proposition P1 est vraie, car les seuls défauts simples de la

classe I qui peuvvent provoquer des erreurs multiples unidirectionnelles sont les coupures des lignes d'alimentation (type C).

Proposition P2

Si la règle R2" est vérifiée pour un circuit, alors les coupures des lignes d'alimentation peuvent être retirées des hypothèses de pannes affectant le bloc fonctionnel.

La proposition P2 est vraie car les coupures considérées seront détectées immédiatement par le contrôleur.

Proposition P3

Si un circuit G est conçu de façon à ce que les règles R1 et R2 ou R2' ou R2" soient vérifiées et que son code de sortie détecte les erreurs simples, alors le circuit G est "fault secure" pour la classe I.

Preuve

La règle R1 assure que toutes les erreurs aux sorties primaires du circuit, provoquées par des défauts du type A et du type B, sont des erreurs simples. Les règles R2 ou R2' assurent que toutes les erreurs aux sorties primaires du circuit provoquées par les défauts du type C, sont des erreurs simples. La règle R2" assure que les défauts du type C sont retirés des hypothèses de pannes pour le bloc fonctionnel.
Les défauts du groupe D ne provoquent pas d'erreurs.
En conclusion, le circuit G est "fault secure" pour la classe I (la classe I est constituée par les défauts du type A, B, C et D, section II.2).

Finalement, on peut aussi utiliser une combinaison des règles R2' et R2" de façon à ce que toutes les lignes d'alimentation qui ne vérifient pas la règle R2' soient utilisées pour alimenter avec leurs extrémités de portes de contrôleurs.

On peut noter que les défauts du type A et du type B sont

équivalents vis à vis de la propriété "fault secure" (la même règle R1 est destinée à couvrir les défauts du type A et du type B). Les défauts du type D ne sont pas pris en compte (aucune règle n'est nécessaire pour couvrir les défauts du type D).

La règle R1 donne une condition suffisante pour assurer que les erreurs simples dans un circuit seront propagées en erreurs simples aux sorties primaires du circuit, mais cette condition n'est pas nécessaire et certains circuits qui ne respectent pas la règle R1 peuvent accomplir le même but. Un tel circuit est le décodeur présenté à la section V.1. (Applications).
Pour introduire la condition nécessaire et suffisante qui assure le but référé, on doit utiliser les chemins sensibles électriques [COU 81].
La structure du circuit ainsi que les vecteurs appliqués au circuit pendant son fonctionnement normal (code d'entrée A) doivent être pris en compte dans ce cas.

Définition D18'
Le degré de divergence effectif d'une ligne Ii est le nombre maximal des sorties primaires du circuit qui peuvent être connectées avec la ligne Ii par l'intermédiaire des chemins sensibilisés par un vecteur d'entrée $a \in A$.

Définition D19'
Le degré de divergence effectif maximal d'un bloc fonctionnel, pour un ensemble de lignes internes I1,I2,...,Ik est égal au maximum des degrés de divergence effectifs des lignes I1,I2,...,Ik.

La règle R1.1 donne la condition suffisante et nécessaire qui garantit la propagation des erreurs simples dans un circuit en erreurs simples aux sorties primaires du circuit.

Règle R1.1
Le degré de divergence effectif maximal du bloc pour l'ensemble de toutes les lignes du bloc, est égal à 1.

III.1.2. Circuits "strongly fault secure"

Soit un circuit G conçu de façon à ce que son code de sortie détecte les erreurs simples et que les règles R1 et R2 soient vérifiées (R2 est assurée par l'intermédiaire de la règle R2' ou R2"). Par la suite nous présentons les conditions qui peuvent empêcher G d'être SFS et les règles qui peuvent garantir que G est SFS. Selon la définition D13, ces règles doivent garantir que la propriété "fault secure" validée par les règles R1 et R2 (R2' et R2") ne sera pas invalidée en présence des séquences pour lesquelles le circuit G est "strongly redundant".

Les quatre groupes d'hypothèses de pannes données en section II.2 sont affinés ci-après.

A - hypothèses de pannes qui peuvent provoquer des erreurs simples concernant une ligne de signal ou une sortie d'une porte (1, 2a, 3, 4a, 5c, 5d, 5h).

B - hypothèses de pannes qui peuvent provoquer des erreurs concernant deux lignes de signal/sorties de portes, mais une seule des deux à la fois (5e, 5f, 5g).

B1 - hypothèses de pannes B, telles que les deux lignes concernées sont liées , par l'intermédiaire de chemins, avec la même sortie primaire du circuit.

B2 - hypothèses de pannes B, telles que les deux lignes concernées sont liées , par l'intermédiaire de chemins, avec deux sorties primaires différentes.

C - hypothèses de pannes qui peuvent provoquer des erreurs multiples unidirectionnelles (2b, 4b).

D - hypothéses de pannes ne provoquant pas d'erreurs (5a).

D1 - hypothèses de pannes D telles que les deux lignes

d'alimentation concernées sont utilisées pour alimenter des portes dont les sorties sont connectées, par l'intermédiaire des chemins, avec la même sortie primaire du circuit.

D2 - hypothèses de pannes D telles que les deux lignes d'alimentation concernées sont connectées, par l'intermédiaire de chemins, avec deux sorties primaires différentes.

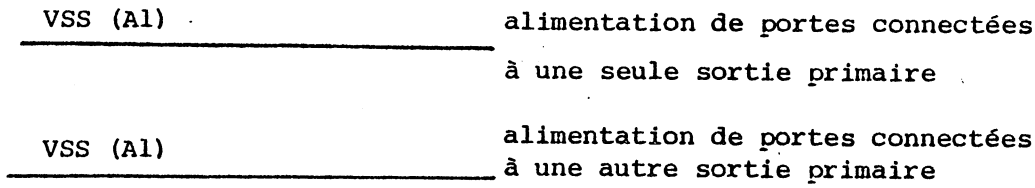
Soit une séquence de défauts <f1,f2,...,f k-1, fk> telle que le circuit est:
- "strongly redundant" pour la séquence <f1,f2,...,f k-1>,
- non redondant pour la séquence <f1,f2,...,f k-1, fk>.

Les défauts f1,f2,...,f k-1,fk sont de types A, B1, B2, C, D1, D2 et la séquence peut être représentée en utilisant des expressions régulières, par (A+B1+B2+C+D1+D2)*.

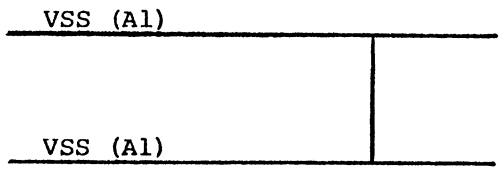
Proposition P4

Si des défauts du type D2 peuvent survenir, le circuit peut ne pas être SFS.

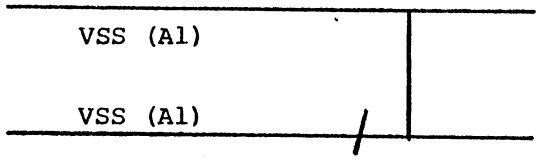
La proposition P4 peut être illustrée très facilement par un exemple. Le défaut de type D2 peut conduire à la situation présentée figure 7.



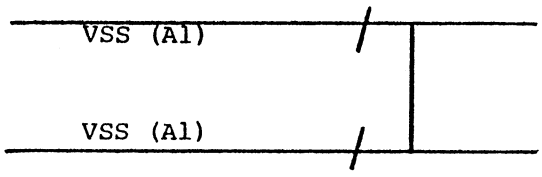
a/ Situation initialle (pas de défauts)



b/ Un défaut du type D2 survient (ou il peut provenir de la conception), le circuit est redondant pour ce défaut.



c/ Une coupure survient, le circuit reste redondant.



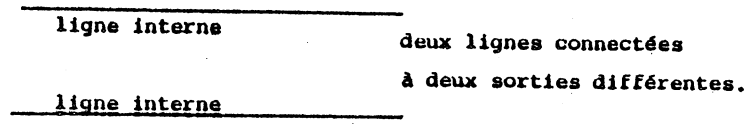
d/ Une deuxième coupure survient, deux sorties primaires peuvent être erronées.

Figure 7- Illustration de la proposition P4.

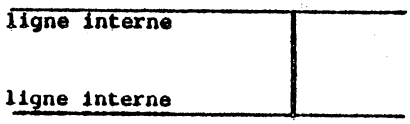
Proposition P5

Si une séquence <f1,f2,...,f k-1> du type (A+B1+C+D)*B2, pour laquelle le circuit est "strongly redundant" peut survenir alors le circuit peut ne pas être SFS.

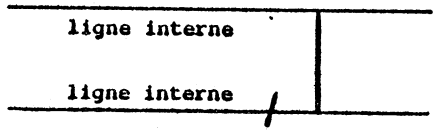
La proposition P5 peut être illustrée par une situation donnée figure 8, de façon similaire à la proposition P4.



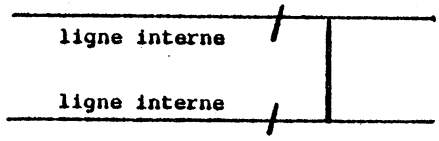
a/ Situation initiale (pas de défauts).



b/ Un défaut du type B2, pour lequel le circuit est redondant, survient.



c/ Une coupure survient, le circuit reste redondant.



d/ Une deuxième coupure survient, deux sorties primaires peuvent être erronées.

Figure 8- Illustration de la proposition P5.

Notons que deux cas peuvent survenir dans la figure 8:

1/ les deux lignes possèdent systématiquement les mêmes valeurs logiques avant l'apparition du court-circuit. Dans ce cas, le circuit est automatiquement redondant pour le court-circuit et pour la première coupure et la situation donnée à la figure 8 peut survenir.

2/ les deux lignes ne possèdent pas systématiquement les mêmes valeurs logiques mais le circuit est redondant pour le court circuit. Dans ce cas, le circuit peut être ou non redondant pour la première coupure, s'il est redondant la situation donnée à la figure 8 peut survenir.

Proposition P6

Si des séquences <f1,f2,...,f k-1> contenant des défauts de type B2 et pour lesquelles le circuit est "strongly redundant", ne peuvent pas survenir, et si des défauts du type D2 ne peuvent pas non plus survenir, alors un circuit conçu de façon à ce que son code de sortie détecte les erreurs simples et de façon à ce que les règles R1 et R2' soient vérifiées, est SFS pour la classe I.

Preuve

La séquence <f1,f2,...,f k-1> est du type (A+B1+C+D1)*. Avant l'occurrence de défauts le circuit est "fault secure" car les règles R1 et R2' ou R2" sont vérifiées (proposition P3). Par les règles R1 et R2' on peut vérifier que la structure du circuit est une structure "bit slice". Elle est donnée par un nombre de blocs dont chacun contient une seule sortie primaire. Chaque ligne et chaque porte du circuit appartient à un seul bloc et il n'existe pas d'interconnexion entre deux blocs.

Si un défaut f1 du type A ou B1 ou C ou D1 survient, alors ce défaut ne peut qu'affecter une porte, ou uneligne d'un bloc, ou une paire de portes/lignes du même bloc.

Donc, en présence du défaut f1, la structure bit-slice n'est pas modifiée et le circuit vérifie les règles R1, R2'.

Si la règle R2" est vérifiée à la place de la règle R2' alors il n'existe pas de défaut du type D et on peut montrer qu'en présence du défaut f1 la règle R2" n'est pas invalidée.

Récursivement, on peut montrer qu'en présence d'une séquence <f1,f2,...,f k-1> du type (BA+B1+C1+D1)*, les règles R1 et R2' ou R2" restent valides. D'après la proposition P6, les séquences dues à la classe I pour lesquelles le circuit est "strongly redundant" ne sont que du type (A+B1+C+D1)*. Selon la définition D13 la proposition P6 es donc vraie.

Les propriétés de redondance d'un défaut dépendent de la présence d'autres défauts, il n'est donc pas suffisant d'éliminer tous les défauts du type B2 pour lesquels le circuit est primitivement redondant puisqu'un défaut du type B2 pour lequel le circuit n'est pas redondant, peut avoir des propriétés de redondance en présence d'autres défauts. Ce n'est pas le cas des défauts du type D2 car un circuit est toujours redondant pour ce type de défaut. Les défauts du type D2 peuvent exister dans un circuit dès la conception car ils ne modifient pas le fonctionnement.

La détermination des défauts du type B2 qui peuvent appartenir aux séquences pour lesquelles un circuit est "strongly redundant" est un problème n-p complexe ; il nécessite la recherche de toutes les séquences pour lesquelles le circuit est "strongly redundant". Ceci étant en général très difficile à réaliser (sinon impossible pour la classe I) on utilise les méthodes suivantes pour éviter une telle travail.

- on regarde si le circuit peut être conçu de façon à être "self testing". Cette méthode est utilisée dans le cas du décodeur présenté aux applications (section V.1).

- on démontre en utilisant des conditions générales (sans chercher

les séquences) que les séquences pour lesquelles le circuit est "strongly redundant" ne contiennent pas de défauts du type R2. Cette méthode est utilisée dans la section V.2.1 pour la conception des PLAs SFS en utilisant le code de parité.

- on utilise des règles de dessin comme la règle R3 pour éliminer tous les défauts du type B2, cette méthode est générale et facile à appliquer mais elle peut être coûteuse en surface. Cette méthode est utilisée dans la section V.2.2 pour la conception des PLAs SFS en utilisant le code de BERGER. Les défauts éliminés dans cette application sont du type B2' qui sont similaires aux défauts du type B2. Dans ce cas, cette méthode est utilisée sans augmentation de la surface du circuit.

Règle R3

Pour éliminer tous les défauts du type B2 d'un circuit, il est nécessaire de répertorier tous les courts-circuits entre deux lignes (de diffusion ou d'aluminium) qui sont liées (par l'intermédiaire des chemins) avec deux sorties primaires différentes. Pour de tels courts-circuits, il faut soit déplacer les lignes respectives, soit les implanter avec des matériaux non court-circuitables. Par exemple, on peut implanter une ligne en diffusion et l'autre en aluminium ou en polysilicon.

Règle R4

Pour éliminer tous les défauts du type D2, il faut répertorier tous les courts-circuits entre deux lignes d'alimentation du même type (VSS ou VDD), servant à alimenter des portes dont les sorties sont liées par l'intermédiaire de chemins avec deux sorties primaires différentes. Ensuite on doit éliminer tous ces courts-circuits soit en déplaçcant quelques lignes, soit en utilisant des matériaux non court-circuitables.

Proposition P7

Si un court-circuit est conçu de façon à ce que son code de sortie détecte les erreurs simples et les règles R1, R2', R3, R4 soient vérifiées, alors le circuit est SFS pour la classe I.

Cette proposition vient facilement par la proposition P6 et par la définition des règles R3 et R4.

Proposition P8

Si un circuit est conçu de façon à ce que son code de sortie détecte les erreurs simples et que les règles R1, R2", R3 soient vérifiées, alors le circuit est SFS pour la classe I.

Cette proposition est déduite facilement de la proposition P6 et de la définition de la règle R3. La règle R4 n'est pas nécessaire, étant donné que selon la règle R2", une seule ligne VSS et une seule ligne VDD sont utilisées.

Quelques remarques peuvent être faites, cocernant la manière d'implémentatuion des circuits afin d'assurer les règles du dessin. Supposons que le concepteur a conçu un bloc SFS constitué par des portes alimentées par une distribution régulière de lignes d'alimentation (les portes peuvent être implantées avec une logique régulière ou non). Le circuit est conçu selon la structure générale représentée figure 9, où chaque zone de portes contient une seule sortie primaire et aucune interaction existe entre deux zones. Par une telle structure les règles R1 et R2' sont vérifiées, mais la propriété du circuit d'être SFS dépend de la façon d'impléementer les lignes d'alimentation. Ceci est illustré à la figure 10.

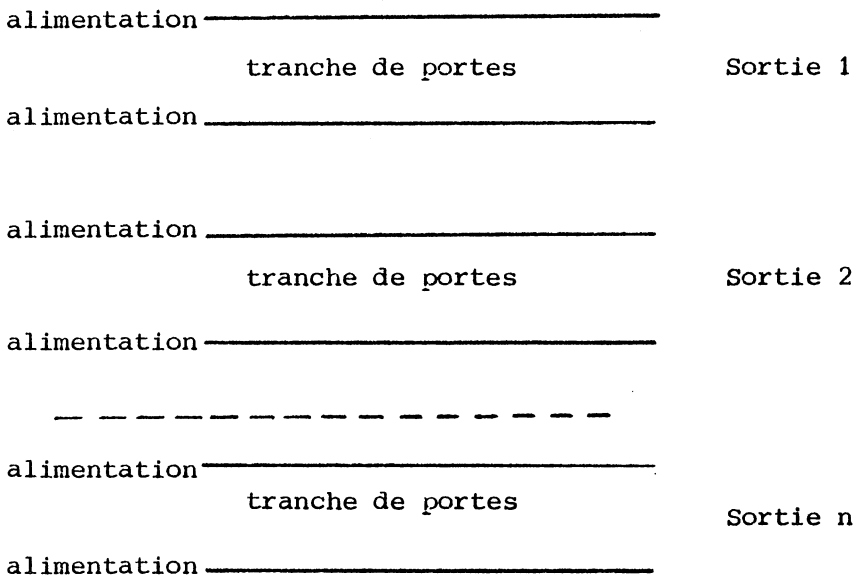


Figure 9-

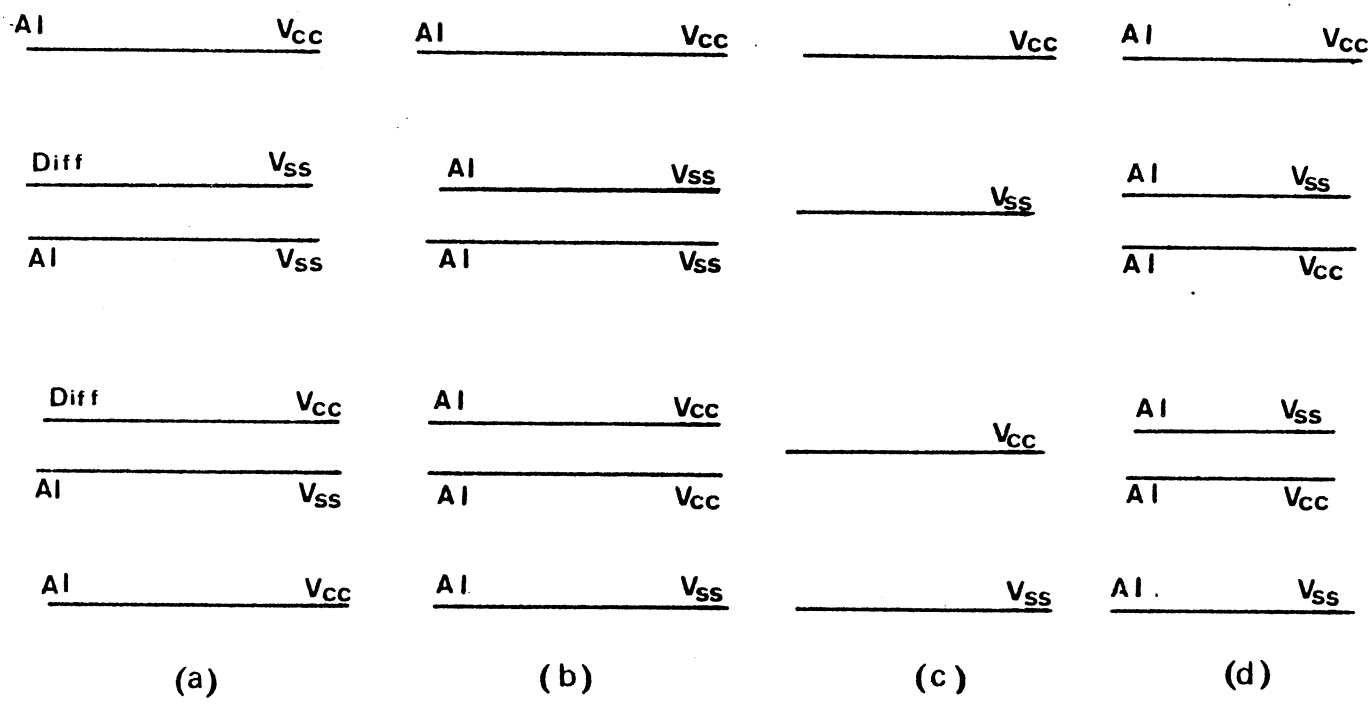


Figure 10- Schémas possibles
a/ schéma convenable
b/ règle R4 violée
c/ règle R2' violée
d/ schéma convenable

III.2. Erreurs unidirectionnelles

Dans cette section les règles de conception des circuits SFS dont le code de sortie détecte les erreurs unidirectionnelles sont définies.

III.2.1. Circuits "fault secure"

La règle suivante a été proposée dans [CRO 78] pour assurer la propagation des erreurs simples dans un circuit en erreurs unidirectionnelles aux sorties primaires du circuit.

Règle R5

Tous les chemins entre une ligne divergente et les sorties primaires du circuit ont la même parité d'inversion.

La règle R5 étant vérifiée, les lignes du circuit seront partitionnées en deux groupes; l'un regroupe les lignes qui sont liées avec les sorties primaires, par l'intermédiaire des chemins dont la parité d'inversion est égale à 0, l'autre regroupe les lignes pour lesquelles les parités d'inversion décrites ci-dessus sont égales à 1.

La règle R5 est au niveau logique ; si elle est combinée avec la "réduction" des circuits [SMI 76] (élimination des redondances), l'ensemble est suffisant pour concevoir des circuits TSC pour le modèle du collage logique. Pour les modèles au niveau du transistor, quelques autres règles de dessin sont nécessaires pour concevoir des circuits FS et SFS.

Règle R6

Pour assurer la propagation des erreurs multiples unidirectionnelles, provoquées par les défauts du type C, en erreurs unidirectionnelles aux sorties primaires du circuit, tous les chemins entre les lignes d'occurrences de ces erreurs et les sorties primaires, doivent avoir la même parité d'inversion.

La règle R6 étant très générale, elle sera assurée dans la pratique par les règles suivantes:

Règle R6'

Chaque ligne d'alimentation du bloc est utilisée pour alimenter des groupes de portes dont les sorties sont liées aux sorties primaires du bloc par l'intermédiaire de chemins qui ont la même parité d'inversion.

Cette règle peut être appliquée étant donné que tous les chemins entre une sortie d'une porte et les sorties primaires ont la même parité d'inversion (la règle R5 étant vérifiée). La règle R6' étant vérifiée, les lignes d'alimentation sont partitionnées en deux groupes. L'un des groupes contient les lignes alimentant les portes pour lesquelles les parités d'inversion sont égales à 0 et l'autre groupe contient les lignes alimentant les portes pour lesquelles les parités d'inversion sont égales à 1.

Règle R6"

Une seule ligne VSS et une seule ligne VDD sont utilisées pour alimenter le bloc, les extrémités de ces lignes sont utilisées pour alimenter de portes d'un contrôleur.

Proposition P9

Si la règle R6' est vérifiée pour un circuit alors la règle R6 est aussi vérifiée.

La proposition P9 est vraie car les seuls défauts simples de la classe I qui peuvent provoquer des erreurs multiples unidirectionnelles sont les coupures des lignes d'alimentation (type C).

Proposition P10

Si la règle R6" est vérifiée pour un bloc fonctionnel, les coupures des lignes d'alimentation (défauts du type C) peuvent être retirées des hypothèses de pannes affectant le bloc fonctionnel.

La proposition P10 est vraie, car les coupures considérées seront détectées immédiatement par le contrôleur.

Proposition P11

Si un circuit est conçu de façon à ce que les règles R5 et R6, ou R6' ou R6", soient vérifiées, et de façon à ce que le code de sortie détecte les erreurs unidirectionnelles, alors le circuit est "fault secure" pour la classe I.

Preuve

La règle R5 assure que toutes les erreurs aux sorties primaires du circuit , provoquées par les défauts du type A et du type B, sont des erreurs unidirectionnelles. Les règles R6 ou R6' assurent que toutes les erreurs aux sorties primaires du circuit provoquées par les défauts du type C, sont des erreurs unidirectionnelles. La règle R6" assure que les erreurs du type C seront détectées immédiatement par un contrôleur et retirées des hypothèses de pannes. Les défauts du type D ne provoquent pas d'erreurs.
En conclusion, le circuit est "fault secure" pour la classe I (la classe I est constituée par les défauts du type A, B, C et D).

Finalement, la règle R6 peut être aussi assurée par une combinaison des règles R6' et R6" de façon à ce que toutes les extrémités des lignes d'alimentation qui ne vérifient pas la règle R6' soient utilisées pour alimenter des portes de contrôleurs.

On peut remarquer que les défauts du type A et du type B sont équivalents vis à vis de la propriété "fault secure" (la même règle R5 est destinée à couvrir les défauts du type A et du type B). Les défauts du type D ne sont pas pris en compte (aucune règle n'est nécessaire pour couvrir les défauts du type D).

III.2.2. Circuits "strongly fault secure"

Soit un circuit G conçu de façon à ce que son code détecte des erreurs unidirectionnelles et de façon à ce que les règles R5 et R6

soient vérifiées (R6 est vérifiée par l'intermédiaire des règles R6' ou R6"). Par la suite nous présentons les conditions qui peuvent empêcher G d'être SFS et les règles qui peuvent garantir que G est SFS. Selon la définition D13, ces règles doivent garantir que la propriété "fault secure" validée par les règles R5 et R6 (R6', R6") ne sera pas invalidée en présence de séquences pour lesquelles le circuit G est "strongly redundant".

Les quatre groupes d'hypothèses de pannes données en section II.2 sont affinées ci-après:

A - hypothèses de pannes qui peuvent provoquer des erreurs simples concernant une ligne du circuit (1, 2a, 3, 4a, 5c, 5d, 5h).

B - hypothèses de pannes qui peuvent provoquer des erreurs concernant deux lignes du circuit, mais une seule des deux à chaque fois (5e, 5f, 5g).

B1' - hypothèses des pannes B, telles que les deux lignes concernées soient regroupées dans le même groupe de lignes (étant donné que la règle R5 est vérifiée, deux groupes de lignes sont définis selon des parités d'inversion qui peuvent être égales à 0 ou à 1).

B2' - hypothèses de pannes B, telles que l'une des lignes concernées appartient au groupe des lignes defini par des parités d'inversion égales à 0 et l'autre ligne appartient au groupe défini par des parités d'inversion égales à 1.

C - hypothèses de pannes qui peuvent provoquer des erreurs multiples unidirectionnelles (2b, 4b).

D - hypothèses de pannes ne provoquant pas d'erreurs (5a).

D1' - hypothèses de pannes de type D telles que les deux lignes concernées appartiennent au même groupe (étant donné que la règle R6 est vérifiée ; deux groupes de lignes d'alimentation sont

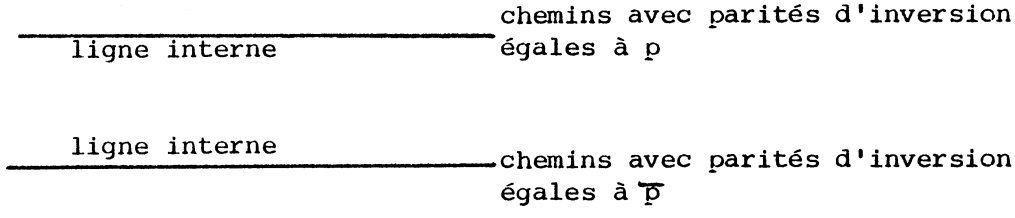
définis selon des parités d'inversion qui peuvent être égales à 0 ou à 1).

D2' - hypothèses de pannes de type D telles qu'une ligne concernée appartient au groupe des lignes d'alimentation définie par des parités d'inversion qui peuvent être égales à 0 et l'autre ligne appartient au groupe des lignes défini par des parités d'inversion égales à 1.

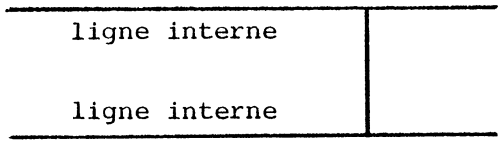
Soit une séquence de défauts <f1,f2,...,f k-1> telle que le circuit est:
- "strongly redundant" pour la séquence <f1,f2,...,f k-1>
- non redondant pour la séquence <f1,f2,...,f k-1, fk>,
les défauts f1,f2,...,f k-1,fk sont de types A,B1', B2', C, D1', D2' et la séquence peut être représentée , en utilisant les expressions régulières par (A+B1'+B2'+C+D1'+D2')*.

Proposition P12
Si des défauts du type B2' peuvent survenir, dans la séquence <f1,f2,...,f k-1>, alors le circuit peut ne pas être SFS.

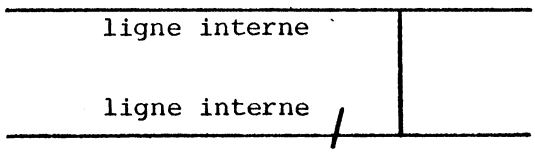
La proposition P12 peut être illustrée de façon similaire à la proposition P5, par une situation donnée figure 11.



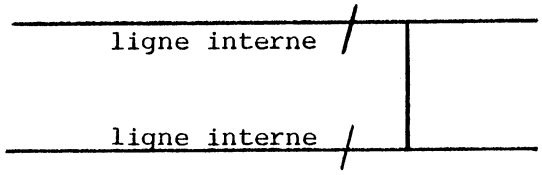
a/ Situation initiale (pas de défauts).



b/ Un défaut de type B2', pour lequel le circuit est redondant, survient.



c/ Une coupure survient, le circuit reste redondant.



d/ Une deuxième coupure survient, une erreur multiple mais non unidirectionnelle, peut être produite.

Figure 11- Illustration de la proposition P12.

Notons que deux cas peuvent survenir dans la figure 11:

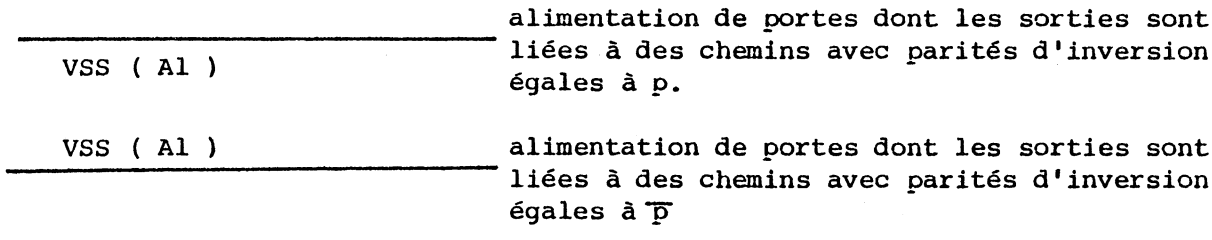
1/ les deux lignes possèdent systématiquement les mêmes valeurs logiques avant l'apparition du court-circuit. Dans ce cas le circuit est automatiquement redondant pour le court-circuit et pour la première coupure et la situation donnée à la figure 11 peut survenir.

2/ les deux lignes ne possèdent pas systématiquement les mêmes valeurs logiques mais le circuit est redondant pour le court-circuit. Dans ce cas, le circuit peut être ou non redondant pour la première coupure, s'il est redondant, la situation donnée à la figure 11 peut survenir.

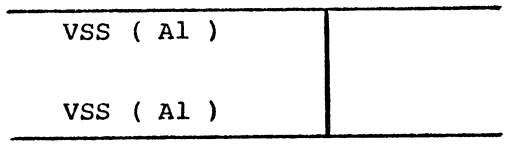
Proposition P13

Si des défauts du type D2' peuvent survenir, le circuit peut ne pas être SFS.

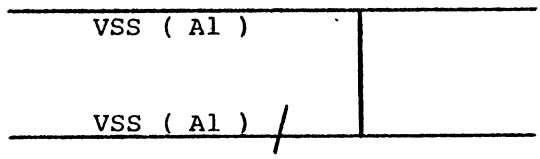
La proposition P13 peut être illustrée de façon similaire à la proposition P4, par une situation donnée à la figure 12.



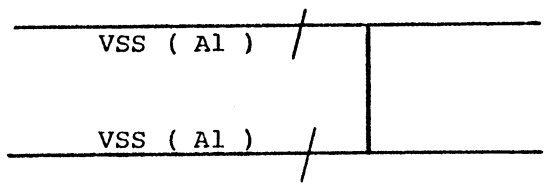
a/ Situation initiale (pas de défauts)



b/ Un défaut de type D2' survient (ou il peut provenir de la conception), le circuit est redondant pour ce défaut.



c/ Une coupure survient, le circuit reste redondant.



d/ Une deuxième coupure survient, une erreur multiple, mais non unidirectionnelle, peut être produite.

Figure 12- Illustration de la proposition P13.

Proposition P14

Si des séquences <f1,f2,...,f k-1> contenant des défauts du type B2' et pour lesquelles le circuit est "strongly redundant" ne peuvent pas survenir et si de défauts du type D2' ne peuvent pas non plus survenir, alors un circuit conçu de façon à ce que son code de sortie détecte les erreurs unidirectionnelles et les règles R5 et R6' ou R6" sont vérifiées, est SFS pour la classe I.

Preuve

La séquence <f1,f2,...,f k-1> est du type (A+B1'+C+D1')*. Avant l'occurrence de défauts le circuit est "fault secure" car les règles R5 et R6' ou R6" sont vérifiées (proposition P11).

La règle R6' peut être invalidée seulement par les défauts du type D2', car les défauts de types A et B n'affectent pas les lignes d'alimentation et les défauts de types C et D1' ne peuvent pas invalider la règle R6' (dans le cas de la règle R6" il n'existe pas de défauts du type D). Alors les règles R6' et R6" ne sont pas mises en défaut par les séquences du type (A+B1'+C+D1')*.

Pour la classe I, de nouveaux chemins ne peuvent pas être créés dans un circuit. Par conséquent la règle R5 peut être invalidée seulement par insertion entre deux lignes d'un chemin déjà existant dans le circuit. Cette insertion peut être produite seulement par des courts-circuits. La règle R5 peut être invalidée seulement par un court-circuit du type B2'.

Après ces constatations on peut prouver récursivement que: En présence d'une séquence <f1,f2,...,f k-1> du type (A+B1'+C+D1')* les règles R5 et R6 (R6', R6") ne sont pas mises en défaut.

Selon la proposition P14 les séquences dues à la classe I pour lesquelles le circuit est "strongly redundant" sont du type (A+B1+C+D1)*. En présence de telles séquences, le circuit reste "fault secure" et d'après la définition D12 le circuit est SFS.

Les propriétés de redondance d'un défaut dépendent de la présence d'autres défauts, il n'est donc pas suffisant d'éliminer tous les défauts du type B2' pour lesquels le circuit est primitivement redondant puisqu'un défaut du type B2' pour lequel le circuit n'est pas redondant, peut avoir des propriétés de redondance en présence d'autres défauts. Ce n'est pas le cas des défauts du type D2' car un circuit est toujours redondant pour ce type de défauts. Les défauts du type D2' peuvent exister dans un circuit depuis la conception car ils ne modifient pas le fonctionnement.

La détermination des défauts du type B2' qui peuvent appartenir aux séquences pour lesquelles un circuit est "strongly redundant" est un problème n-p complexe, il nécessite la recherche de toutes les séquences pour lesquelles le circuit est "strongly redundant". Ceci étant en général très difficile à réaliser (sinon impossible pour la classe I), on utilise comme dans le cas de détection des erreurs simples (section III.2) les méthodes suivantes pour éviter ce type de travail:
- on regarde si le circuit peut être conçu de façon à être "self testing",
- on démontre , en utilisant des conditions générales (sans chercher des séquences), que les séquences pour lesquelles le circuit est "strongly redundant" ne contiennent pas de défauts du type B2',
- on utilise des règles de dessin comme la règle R7 pour éliminer tous les défauts du type B2'.

Règle R7

Pour éliminer tous les défauts du type B2' il faut répertorier tous les courts-circuits entre deux lignes (de diffusion ou d'aluminium) tels que: les parités d'inversion des chemins entre l'une des deux lignes et les sorties primaires sont égales à 0 et les parités d'inversion des chemins entre l'autre ligne et les sorties primaires sont égales à 1 . Pour de tels courts-circuits, il faut soit déplacer les lignes respectives, soit les implanter avec des matériaux non court-circuitables. Par exemple une ligne peut être implantée en diffusion et l'autre en aluminium ou en polysilicon.

Les circuits étant toujours redondants pour les défauts du type D2', il faut éliminer ce type de défauts en utilisant les règles du dessin.

Règle R8

Pour éliminer tous les défauts du type D2' on doit lister tous les courts-circuits entre deux lignes d'alimentation du même type (VDD ou VSS) telles que: l'une des lignes alimente des portes dont les sorties sont liées avec les sorties primaires par l'intermédiaire de chemins ayant des parités d'inversion égales à 0 et l'autre ligne alimente des portes dont les sorties sont liées avec les sorties primaires par l'intermédiaire de chemins ayant des parités d'inversion égales à 1. Ensuite tous ces courts-circuits doivent être éliminés, soit en déplaçant quelques lignes, soit en utilisant des matériaux non court-circuitables.

Proposition P15

Si un circuit est conçu de façon à ce que son code de sortie détecte les erreurs unidirectionnelles et de façon à ce que les règles R5, R6', R7 et R8 soient vérifiées, alors le circuit est SFS pour la classe I.

Cette proposition découle de la proposition P14 et de la définition des règles R7 et R8.

Proposition P16

Si un circuit est conçu de façon à ce que son code de sortie détecte les erreurs unidirectionnelles et de façon à ce que les règles R5, R6", R7 et R8 soient vérifiées, alors le circuit est SFS pour la classe I.

La proposition P16 découle de la proposition P14 et de la définition des règles R7 et R8.

III.3. Erreurs multiples

Dans cette section on propose des règles de conception des circuits SFS dont le code de sortie détecte les erreurs multiples. Deux types de code de sortie sont considérés: l'un est le code de duplication, l'autre est le code "double- rail".

Pour les deux codes, les sorties primaires des circuits sont partitionnées en deux groupes. Chaque sortie primaire d'un groupe est associée avec une sortie primaire dupliquée (resp. "double-rail") de l'autre groupe pour le code de duplication (resp. pour le code "double-rail"). On peut considérer que chaque sortie de porte est associée ou non avec une sortie de porte dupliquée (resp. complémentaire) pour le code de duplication (resp. pour le code "double-rail"). De plus, on peut considérer que chaque ligne interne est associée ou non à une ligne interne dupliquée pour le code de duplication.

III.3.1. Circuits "fault secure"

Règle R9
Chaque ligne est liée, par l'intermédiaire de chemins, avec des sorties primaires qui appartiennent au même groupe.

Règle R10
Le circuit doit être conçu de façon à ce que les erreurs multiples unidirectionnelles provoquées dans le circuit par les défauts du type C soient propagées aux sorties primaires en erreurs détectables par le code de duplication (ou par le code double-rail).

La règle R10 étant très générale, il est difficile de la mettre en oeuvre.

Règle R10'

Chaque ligne d'alimentation est utilisée pour alimenter des portes dont les sorties sont liées (par l'intermédiaire de chemins) avec des sorties primaires du même groupe.

Règle R10"

Une ligne VDD et une ligne VSS sont utilisées. L'extrémité de chacune de ces lignes est utilisée pour alimenter deux portes dont les sorties sont des sorties primaires complémentaires. On considère que le code de sortie est le code double rail.

Règle R10'''

Une seule ligne VSS et une seule ligne VDD sont utilisées pour alimenter le circuit. L'extrémité de ces lignes est utilisée pour alimenter les portes d'un controleur.

Proposition P17

Si la règle R10' est vérifiée pour un circuit, alors la règle R10 est aussi vérifiée.

La proposition P17 est vraie car les seuls défauts simples de la classe I qui peuvent provoquer des erreurs multiples unidirectionnelles sont les coupures des lignes d'alimentation (type C).

Proposition P18

Si la règle R10'' est vérifiée pour un circuit dont le code de sortie est un double-rail code, alors la règle R10 est aussi vérifiée.

La proposition P18 est vraie car les coupures des lignes d'alimentation (défauts du type C) provoqueront entre autres le collage à la même valeur des deux sorties complémentaires. Ceci est détecté par le code double-rail.

Proposition P19

Si la règle R10''' est vérifiée pour un circuit, alors les coupures des lignes d'alimentation peuvent être retirées des hypothèses de

pannes.

La proposition P19 es vraie car les coupures des lignes d'alimentation seront détectées immédiatement par un contrôleur.

Finalement, on peut utiliser une combinaison des règles R10', R10" et R10''' pour assurer la règle R10. Chaque ligne d'alimentation doit respecter une des trois règles.

On peut démontrer qu'un circuit qui vérifie les règles R9 et R10 (R10', R10", R10''') est un circuit "fault secure".

III.3.2. Circuits "strongly fault secure"

Les quatre groupes d'hypothèses de pannes donnés à la section II.2. sont affinés ci-après:

A - hypothèses de pannes qui peuvent provoquer des erreurs simples concernant une ligne de signal ou une sortie d'une porte (1, 2a, 3, 4a, 5c, 5d, 5h).

B - hypothèses de pannes qui peuvent provoquer des erreurs concernant deux lignes mais une seule des deux à chaque fois (5e, 5f, 5g).

B1" - hypothèses de pannes B telles que les deux lignes concernées sont liées avec des sorties primaires du même groupe.

B2" - hypothèses de pannes B telles que les deux lignes concernées sont liées avec des sorties primaires des deux groupes.

C - hypothèses de pannes qui peuvent provoquer des erreurs unidirectionnelles multiples dans les circuits (2b, 4b).

D - hypothèses de pannes ne provoquant pas d'erreurs (5a).

D1" - hypothèses de pannes D telles que les deux lignes

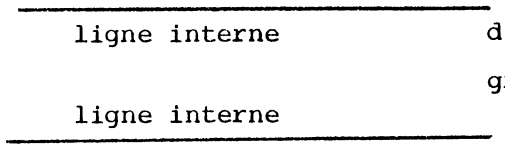
d'alimentation concernées sont utilisées pour alimenter des portes dont les sorties sont liées avec le même groupe des sorties primaires.

D2" - hypothèses de pannes D telles que les deux lignes d'alimentation concernées sont utilisées pour alimenter des portes dont les sorties sont liées avec des sorties primaires des deux groupes.

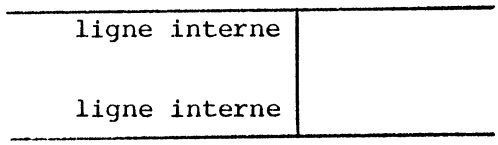
Soit une séquence de défauts <f1,f2,...,f k-1, fk> telle que le circuit est:
- "strongly redundant" pour la séquence <f1,f2,...,f k-1>,
- non redondant pour la séquence <f1,f2,...,f k-1,fk>
les défauts f1,f2,...,f k-1,fk sont de types A,B1",B2",C,D1",D2" et la séquence peut être représentée, en utilisant les expressions régulières, par (A+B1"+B2"+C+D1"+D2")*.

Proposition P20
Si une séquence <f1,f2,...,f k-1> du type (A+B1"+C+D)*B2", pour laquelle le circuit est "strongly redundant" peut survenir, alors le circuit peut ne pas être SFS.

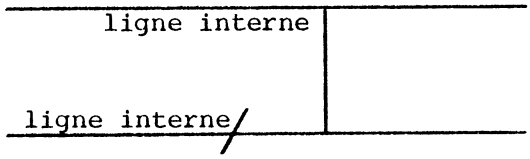
La proposition P20 peut être illustrée de façon similaire à la proposition P8, par une situation donnée figure 13.



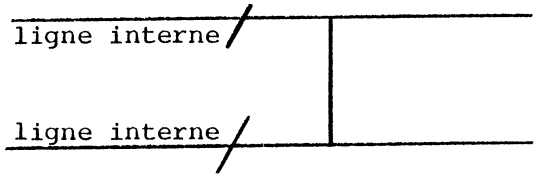
a/ Situation initiale (pas de défauts).



b/ Un défaut de type B2" survient, pour lequel le circuit est redondant.



c/ Une coupure survient le circuit reste redondant.



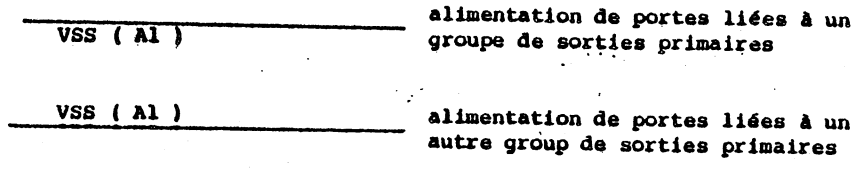
d/ Une deuxième coupure survient,des erreurs multiples, affectant les deux groupes de sorties, peuvent être produits.

Figure 13- Illustration de la proposition P20.

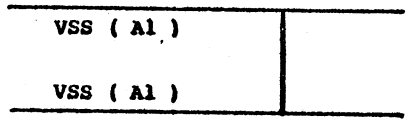
Proposition P21

Si des défauts du type D2" peuvent survenir, le circuit peut ne pas être SFS.

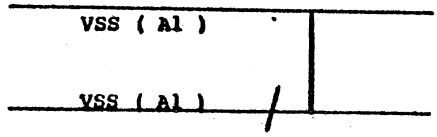
La proposition P21 peut être illustrée de façon similaire à la proposition P4, par une situation présentée figure 14.



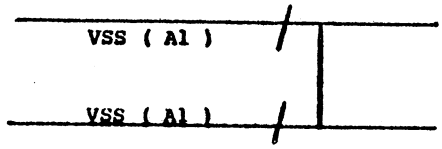
a/ Situation initiale (pas de défauts)



b/ un défaut de type $D_2^{"}$ survient (ou il peut provenir de la conception), le circuit est redondant pour ce défaut.



c/ une coupure survient, le circuit reste redondant.



d/ une deuxième courpire survient, des erreurs multiples affectant les deux groupes de sorties peuvent être produites.

Figure 14- Illustration de la proposition P21

Proposition P22

Si les séquences ⟨f1,f2,...,f k-1⟩ contenant des défauts du type B2", pour lesquels le circuit est "strongly redundant", ne peuvent pas survenir et si des défauts du type D2" ne peuvent pas non plus survenir, alors un circuit conçu de façon à ce que son code de sortie détecte des erreurs multiples et de façon à ce que les règles R9 et R10' soient vérifiées, est SFS pour la classe I.

Cette proposition peut être prouvée en remarquant que la règle R9 peut être mise en défaut seulement par des séquences contenant des défauts du type B2" et la règle R10' peut être invalidée uniquement par des défauts du type D2".

Proposition P23

Si les séquences ⟨f1,f2,...,f k-1⟩ contenant des défauts du type B2", pour lesquelles le circuit est "strongly redundant" ne peuvent pas survenir, alors un circuit conçu de façon à ce que les règles R9 et R10" ou R10''' soient vérifiées et de façon à ce que son code de sortie détecte des erreurs multiples (code double-rail si R10" est vérifiée), est SFS pour la classe I.

Cette proposition peut être prouvée en remarquant que la règle R9 peut être mise en défaut seulement par des séquences contenant des défauts B2" et les règles R10", R10''' ne peuvent pas être mises en défaut (la règle R10" ou R10''' étant vérifiées il n'existe pas de défauts du type D2".

Règle R11

Pour éliminer tous les défauts du type B"2 d'un circuit, il est nécessaire de répertorier tous les courts-circuits entre deux lignes qui sont liées avec des sorties primaires appartenant à deux groupes différents. Pour de tels courts-circuits, il faut, soit éloigner les lignes respectives, soit les implanter avèc des matériaux non court-circuitables. Par exeple, une ligne peut être implantée en diffusion et l'autre en aluminium ou en polysilicon.

Règles R12

Pour éliminer tous les défauts du type D2", il faut répertorier tous les courts-circuits entre deux lignes d'alimentation du même type (VSS ou VDD), servant à alimenter des portes dont les sorties sont liées avec des sorties primaires appartenant à deux groupes différents. Ensuite il faut éliminer tous ces courts-circuits soit en déplaçant quelques lignes, soit en utilisant des matériaux non court-circuitables.

III.3.3. La méthode de duplication

La méthode classique de duplication des blocs fonctionnels est discutée ci après.

On sait que cette structure offre aux circuits une très grande protection vis à vis des pannes possibles. En fait, le bloc dupliqué est protégé vis à vvis de chaque type de défaut, affectant à chaque fois, un seul des blocs dupliqués. Le circuit est protégé même si des défauts multiples surviennent sur l'un des blocs.

Un autre avantage de cette structure est la facilité de concevoir les blocs dupliqués.

L'inconvvénient de cette structure est la grande consommation en surface. La duplication est en général utilisée lorsque la protection offerte par les autres méthodes n'est pas suffisante.

On a déjà remarqué que la duplication offre une protection suffisante vis à vis des hypothèses de pannes beaucoup plus larges que la classe I, mais les règles données pour les codes détectant les erreurs multiples peuvent être mises en défaut par quelques défauts simples de la classe I. En effet, un court circuit entre deux lignes de deux blocs dupliqués ou une coupure d'une ligne d'alimentation alimentant de portes appartenant aux deux blocs dupliqués, peuvent mettre en défaut la condition que chaque défaut affecte un seul des blocs dupliqués . Par conséquent, il est

nécessaire d'examiner si les règles données aux sections III.3.1 et III.3.2 sont vérifiées par la structure de la duplication:

- la règle R9 est toujours vérifiée.
- la règle R10' peut être mise en défaut si certaines lignes d'alimentation alimentent de portes des deux blocs dupliqués. On peut permettre l'existence des lignes d'alimentation communes aux deux blocs si la règle R10''' est vérifiée par ces lignes (les extrémités de ces lignes sont utilisées pour alimenter des portes d'un contrôleur), dans le cas de la structure double-rail la règle R10" peut être aussi aplliquée.

- les courts-circuits entre les lignes des deux blocs (défauts du type B2") pour lesquels les blocs dupliqués sont redondants, peuvent mettre en défaut la propriété SFS. De tels courts-circuits existent par définition dans le cas de la duplication pure (tous les courts-circuits entre deux lignes respectives des deux blocs dupliqués).

- les courts-circuits D2" peuvent mettre en défaut la règle R12.

Il est donc nécessaire d'éliminer certains courts-circuits en déplaçant quelques lignes ou en utiliant des matériaux non court-circuitables. Une méthode plus efficace consiste à séparer les deux blocs en plaçant un autre bloc entre eux. Un tel bloc pourrait être par exemple le controleur des deux blocs doublés.

Un dernier point de la discussion relative à la méthode de duplication concerne la conception des circuits duaux.

III.3.3.1. La conception des circuits MOS duaux

Successivement nous abordons les définitions des circuits duaux et des réseaux de transistors série-parallèles, la conception de parties classiques duales, N.MOS et C.MOS des circuits intégrés. Enfin, nous donnons des conclusions concernant le test et la CAO de réseaux duaux. Des exemples sont extraits de parties du MC

68000.

III.3.3.1.1. Définitions

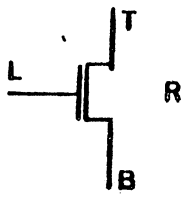
Deux définitions de base sont données dans cette section à propos des circuits duaux et des réseaux des transistors série-parallèles.

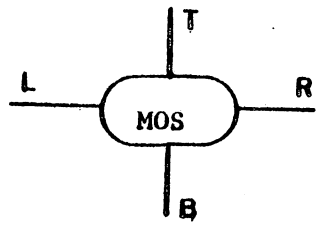
Circuits duaux:

Définition D20
Deux circuits G1 et G2 sont dits duaux si des sorties complémentaires sont produites lorsque des entrées complémentaires sont appliquées à ces circuits.

Réseaux de transistors série-parallèles:

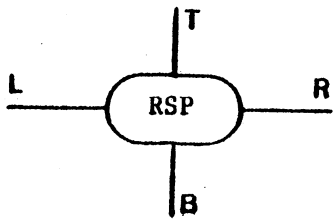
Les réseaux des transistors série-parallèles sont définis depuis [KUN 70]. Un réseau est défini successivement depuis un réseau élémentaire constitué d'un seul transistor,

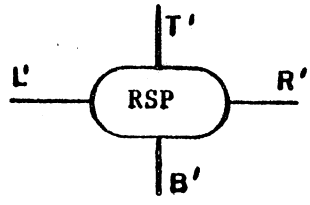




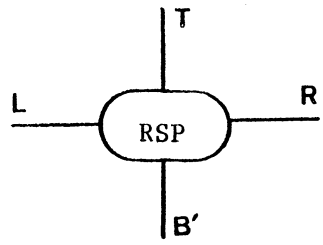
En utilisant les deux opérations suivantes:

- Le câblage en série:

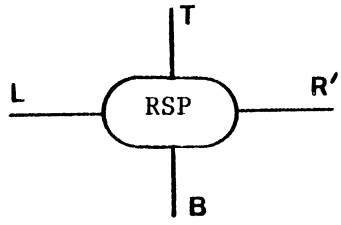




L et L', R et R', B et T' sont connectés ensemble et on obtient



- Le câblage en parallèle:
T et T', B et B', R et L' sont connectés et on obtient:



Chacune de ces opérations est commutative et associative (lorsqu'on considère que les réseaux élémentaires sont les transistors). On commence ainsi par un réseau élémentaire, c'est à dire un transistor, les règles suivantes peuvent alors être appliquées:
- remplacement d'un transistor du réseau par deux transistors séries,
- remplacement d'un transistor du réseau par deux transistors parallèles.

L'ensemble des réseaux parallèles-séries de transistors est ainsi généré. Chacun des réseaux séries-parallèles peut être exprimé en utilisant des expressions à l'intérieur desquelles une lettre correspond à un transistor. Une opération parallèle est faite par la somme de chaque expression représentant le réseau constituant, et une opération série est faite par des produits d'expressions représentant le réseau constituant (les parenthèses étant utilisées comme facteur irréductible pour une lettre). Réciproquement, à chaque expression formée avec des sommes et des produits dans laquelle une lettre est utilisée une et une seule fois, un réseau de transistors séries- parallèle peut être attaché.

Maintenant nous allons considérer la conception des circuits duaux en technologie N.MOS et C.MOS.

Le réseau dual d'un réseau de transistors parallèles séries est défini comme un réseau obtenu en échangeant les opérations séries et les opérations parallèles amenant au réseau original. Ceci constitue la définition D21:

Définition D21
Un réseau dual est obtenu à partir de l'expression représentant le réseau original, à l'intérieur de laquelle chaque somme a été remplacée par un produit et chaque produit a été remplacé par une somme.

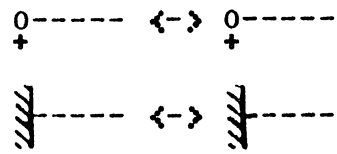
III.3.3.1.2. Conception des circuits duaux en technologie N.MOS
Un seul type de transistors (MOS de type N) est utilisé dans cette technologie.

Par la suite, quelques éléments de la technologie N.MOS sont présentés.

III.3.3.1.2.1. Portes à charge passive
Ces portes sont constituées de transistors MOS signal et de transistors MOS de charge. Deux types de circuits sont considérés ici: les portes constituées d'un MOS de charge et d'un réseau parallèle séries et les autres portes.

A - Portes constituées par un MOS de charge
et un réseau parallèle séries de MOS signal
La porte dual d'une telle porte est trivialement obtenue en remplaçant le réseau séries parallèle (SP) original par le réseau SP dual du premier.

L'ensemble de règles complet est alors donné par:



MOS de charge <-> MOS de charge

réseau SP de MOS signaux <-> réseau SP dual de MOS signaux

B - Autres portes

Aucune règle générale ne peut être donnée pour les portes à charge passive qui ne correspond pas au schéma donné par le point A. Plusieurs exemples de telles portes peuvent être donnés. Le premier est montré à la figure 15. Il correspond au cas pour lequel la porte possède un MOS de charge, mais le réseau de MOS signaux n'est pas SP.

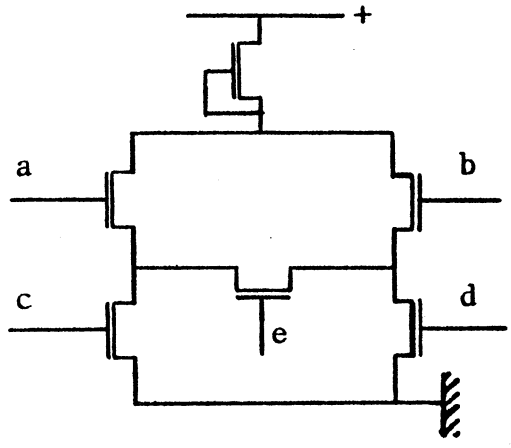


Figure 15 - Réseau de MOS signal non SP.

Le circuit dual d'une telle porte peut être facilement trouvé et il est donné par la figure 16.

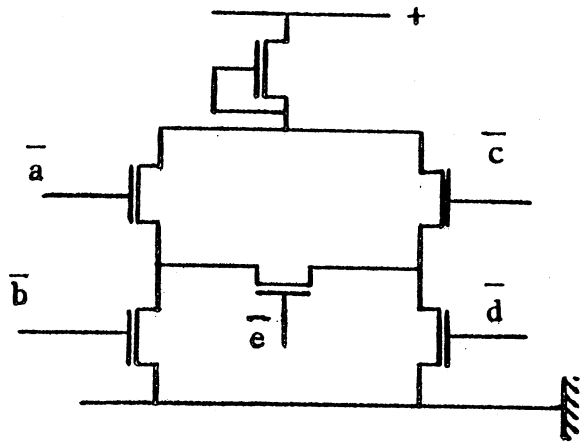


Figure 16- Circuit dual pour l'exemple de la figure 15.

Un autre type de portes à charge passive qui ne correspond pas au schéma donné au point A est le type de porte pour lequel il n'existe pas un MOS de charge associé à un réseau de MOS (SP ou non). De telles portes associent des MOS de charge à des MOS de signal. L'exemple d'une telle porte est donné à la figure 17. Plusieurs exemples de circuits duaux peuvent être trouvés pour cette porte, mais quelques uns de ceux-ci posent des problèmes électriques (figure 18).

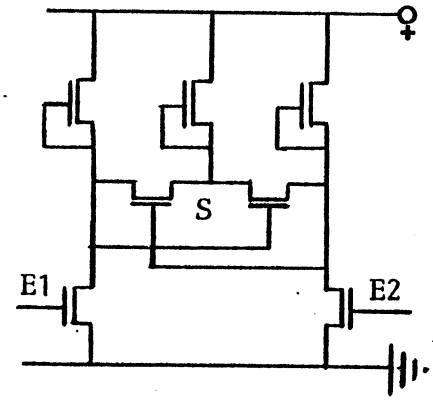


Figure 17 - Porte constituée de MOS de charge et de MOS de signal.

En effet, tous les circuits de la figure 18 possèdent au total 8 transistors alors que le circuit original n'en possède que 7 au total. On peut remarquer que pour la même fonction, il existe deux circuits (l'original et son dual) ayant au total 9 transistors. Ces circuits sont montrés en figure 19. Malgré tout il faut noter que les entrées complémentaires sont supposées disponibles.

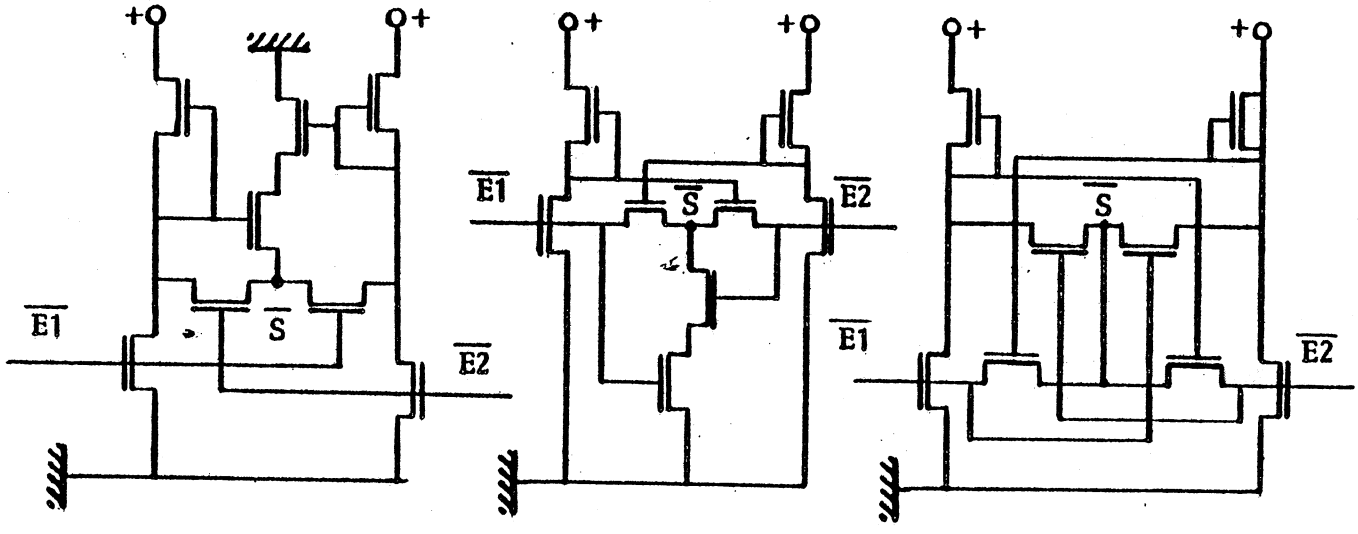


Figure 18 - Circuits duaux pour l'exemple de la figure 17.

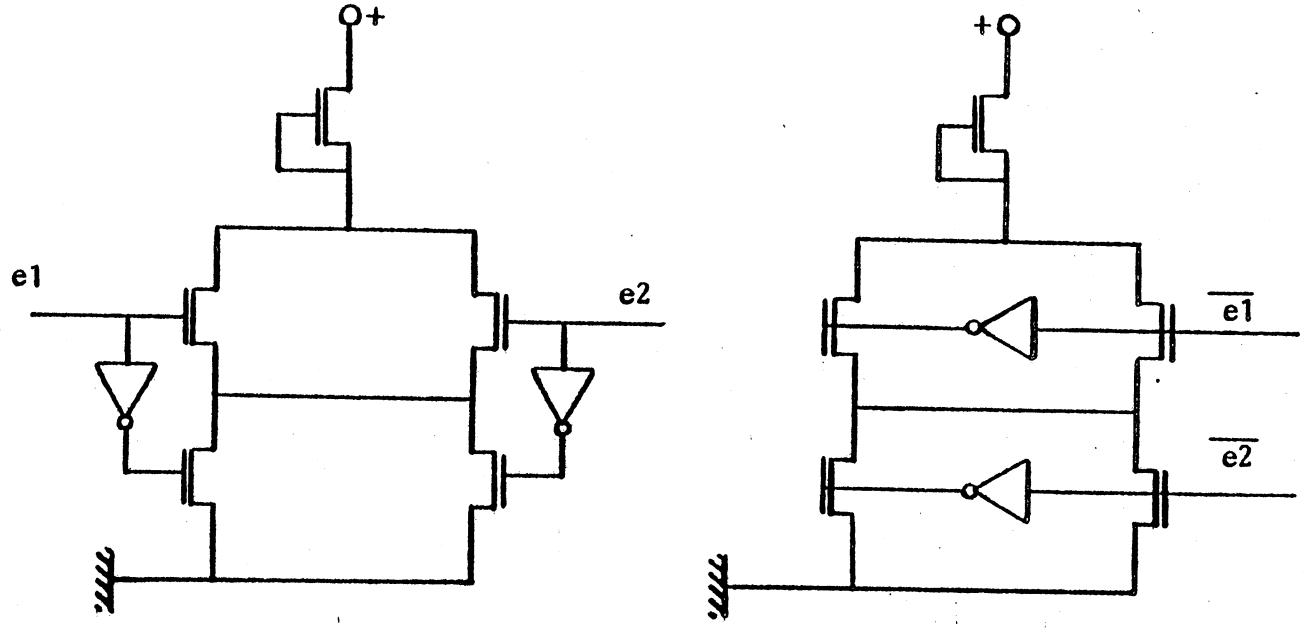


Figure 19 - Circuit original et circuit dual pour la fonction du circuit de la figure 17.

III.3.3.1.2.2. Portes à charge commandées

Ces portes sont caractérisées par le fait que, comparées aux portes précédentes (section III.3.3.1.2.1.), elles ont un MOS de précharge à la place d'un MOS de charge. Evidemment, toutes les règles, données par les portes à charge passive, sont applicables à ces portes. Ces règles s'appliquent aussi pour la conception de portes duales, de portes NMOS dynamiques, comme représentées à la figure 20.

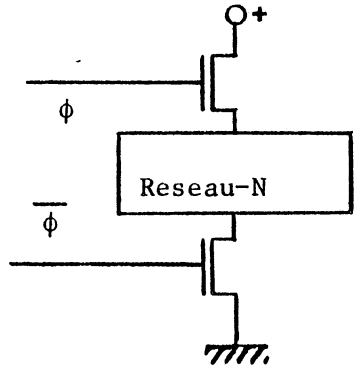


Figure 20 - Circuit N.MOS dynamique.

III.3.3.1.2.3. Transistors séries

Cette section concerne des transistors séries utilisés quelques fois pour assembler les portes. Il n'y a pas de problème spécifique, pour assembler les portes duales, lorsque chaque porte originale a une sortie utilisée comme entrée classique d'une autre porte. Mais il est nécessaire de faire attention lorsque par exemple la sortie de la porte originale est connectée directement à la diffusion appartenant à un transistor d'une autre porte. Ces deux possibilités sont illustrées par la figure 21. La figure 21(a) et 21(b) sont deux implémentations amenant à des fonctions différentes. Pour la figure 21(b), on suppose aue le mot d'entrée: d=1, e=1, Si-1=1 n'arrivera jamais (sinon, l'étage i du circuit aura une influence sur l'étage i-1 : Si-1 sera forcé à 0). Dans ce cas, on peut considérer que la fonction NAND est réalisée entre d et Si-1 pour la figure 21(b), au lieu de NAND entre d et Si-1 pour

la figure 21(a).

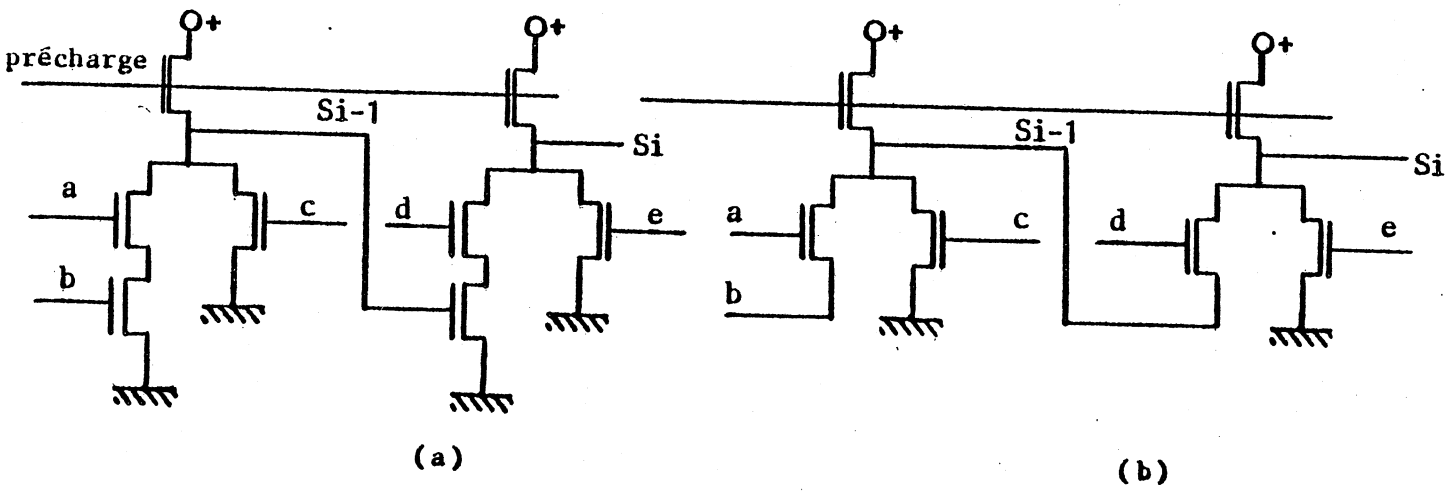


Figure 21 - Un circuit original:
(a) sans MOS séries
(b) avec des MOS séries.

En supposant qu'un circuit dual peut être conçu en accord avec les règles indiquées précédemment, soit que les cablages séries deviennent des cablages paralléles, etc... et donc un circuit dual du cicuit de la figure 21(b) peut être représenté comme à la figure 22.

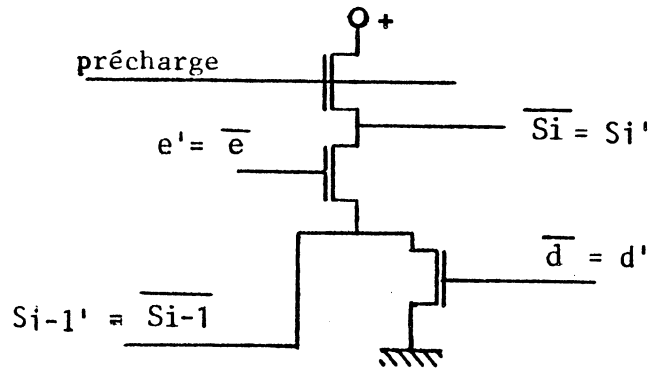


Figure 22 - Circuit dual attendu pour la figure 21(b).

Mais pour ce circuit dual, le vecteur d'entrée interdit ne correspond pas aux valeurs complémentées des entrées interdites pour le circuit original. Les valeurs d'=1, S'i-1=1 doivent être interdites pour le circuit dual alors que les valeurs interdites e=1 d=1 Si-1=1 du circuit original mènent à une valeur non interdite e'=0 d'=0 S'i-1=0 pour le circuit dual. Ceci sous-entend que pour la valeur e=0 d=0 Si-1=0 du circuit original, le circuit dual donnerait une réponse erronée en sortie S'i-1=0 au lieu de S'i-1=1. Ce cas correspondrait à un décalage à gauche dans une Alu. Cet exemple montre que la conception de circuits duaux, dans le cas des transistors séries, doit être faite avec beaucoup d'attention. L'application des règles générales doit être faite avec précaution. Dans l'exemple utilisé précédemment, il est possible d'ajouter un transistor (commande par le signal d) au schéma de la figure 22 et d'obtenir ainsi un circuit dual. Ce circuit dual peut être représenté par la figure 23.

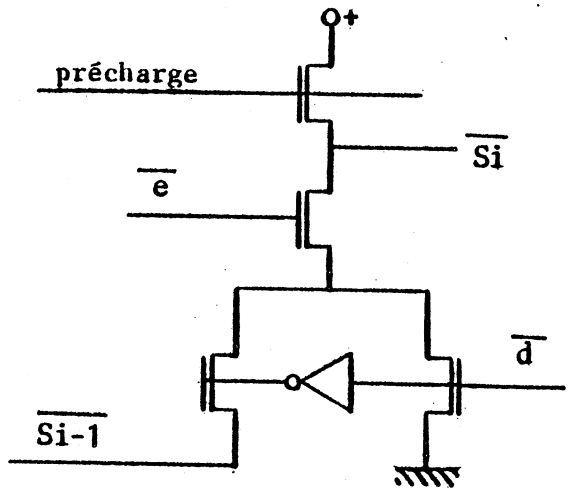


Figure 23 - Circuit dual pour la figure 21(b).

Il faut noter que pour un circuit d'ALU réelle, en prenant en compte que les valeurs e=1 d=1 Si-1=1 n'arrivent jamais, d'autres modifications peuvent être faites pour éviter de trop longs temps de propagation. Le circuit original et le circuit dual sont représentés à la figure 24.

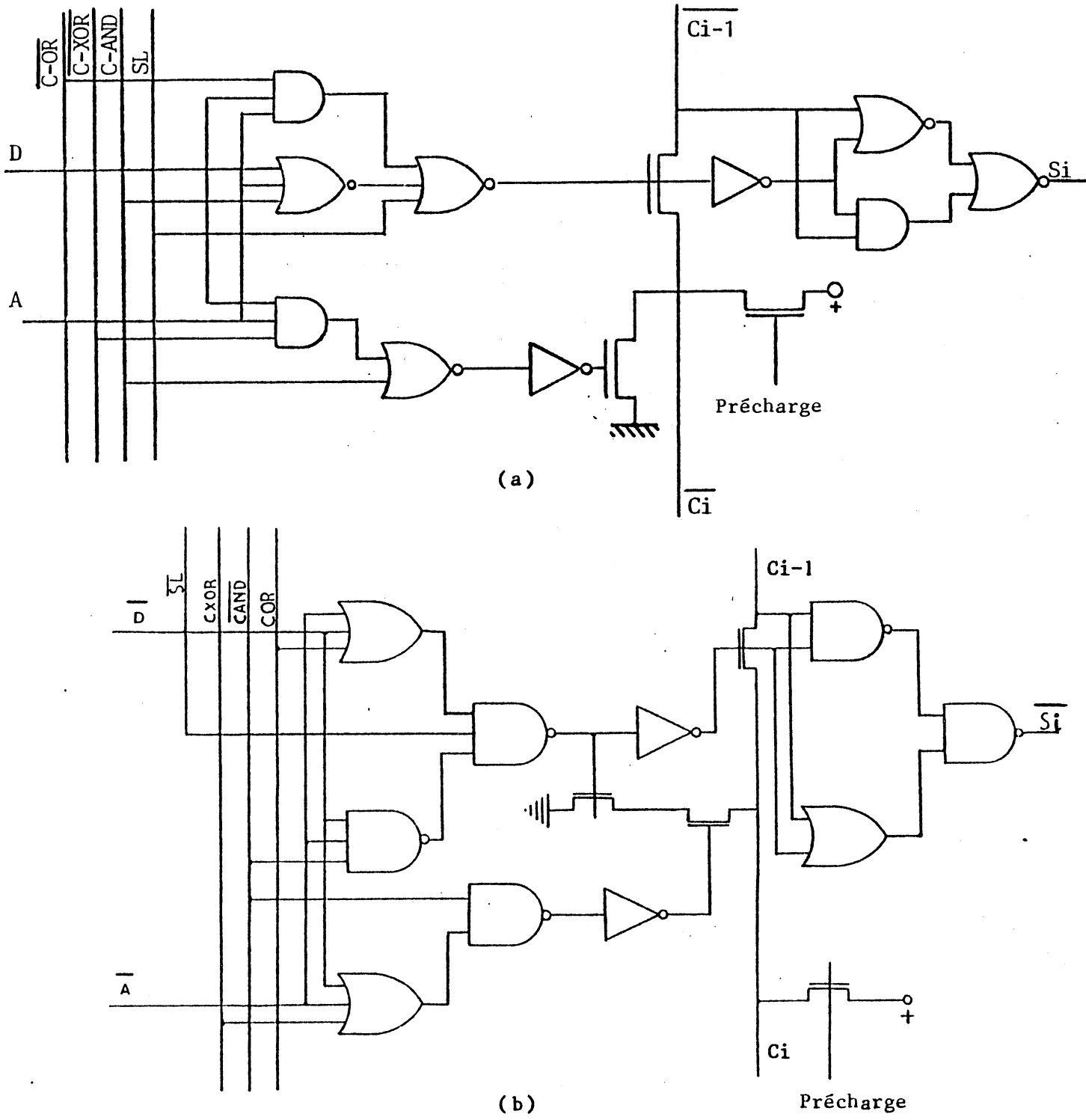


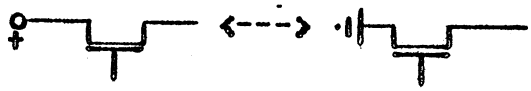
Figure 24 - ALU original (a) et dual (b).

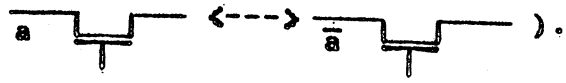
III.3.3.1.2.4. Portes de transfert

Le fonctionnement de la porte de transfert duale doit rester identique au fonctionnement des portes de transfert sur le circuit

original: elles doivent être toutes les deux ouvertes ou fermées simultanément. Ceci implique que les lignes de contrôle des portes de transfert ne sont pas complémentées pour les circuits duaux et ceci est dû au fait qu'aucun transistor dual n'est disponible en technologie N.MOS. Bien entendu, si les lignes de contrôle sont issues complémentées, les inverseurs peuvent être utilisés.

Un exemple de circuit utilisant de telles portes de transfert est donné à la figure 25(a) (cette figure est adaptée à partir du schéma de génération de la retenue entrant dans l'ALU du MC 68000). La figure 25(b) représente alors le circuit dual obtenu. On peut noter que les lignes de transfert et les portes de transfert n'ont pas d'éléments duaux, mais les valeurs transportées par ces lignes sont complémentées (on doit alors utiliser la règle



ou de façon plus générale la règle ).

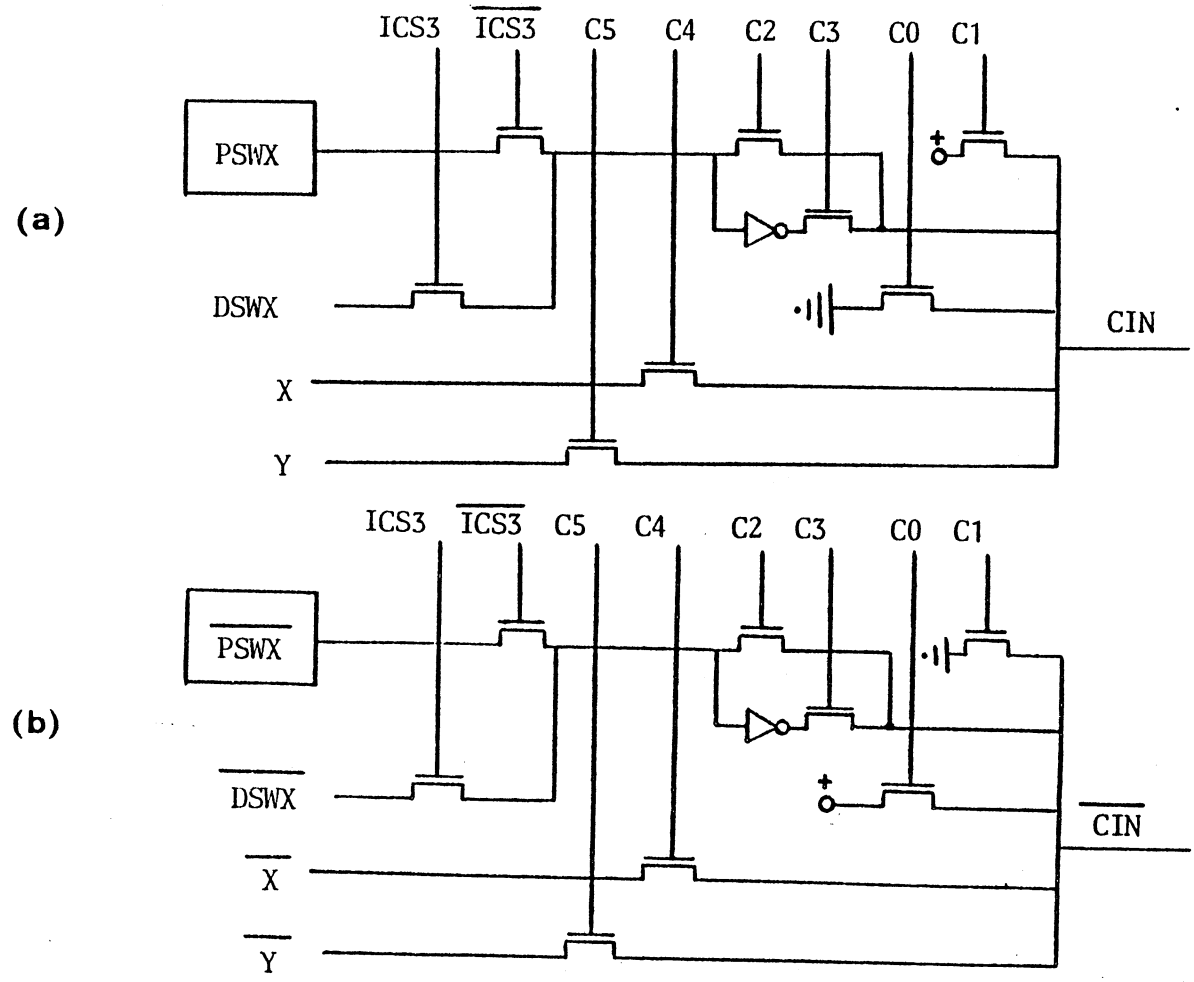


Figure 25 - Portes de transfert:
(a) circuit original
(b) circuit dual.

III.3.3.1.2.5. Registres

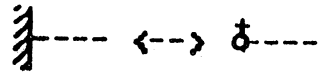
Les circuits pour le stockage de données sont identiques pour les circuits originaux et les circuits duaux. Donc, les registres ne sont pas modifiés dans les circuits duaux. Les lignes de contrôle pour les opérations de lecture et d'écriture sont identiques.

III.3.3.1.3. Conception des circuits C.MOS duaux

La technologie C.MOS est bien entendu d'un intérêt primordial pour la conception des circuits duaux, en raison de l'utilisation de deux types de transistors (N.MOS et P.MOS) qui sont duaux. Les

transistors N.MOS et P.MOS sont intéressants du fait qu'ils ont un fonctionnement identique lorsqu'ils sont controlés par des entrées complémentaires. En association avec la dualité des lignes d'alimentation, les circuits duaux peuvent être facilement conçus. Les règles de base sont:

N-MOS <--> P-MOS



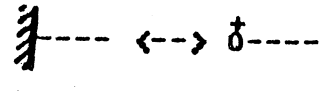
Nous examinons maintenant différents types de circuits C.MOS.

III.3.3.1.3.1. Portes à réseaux entrièrement complémentées

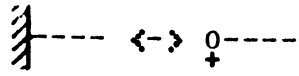
Les portes entièrement complémentées sont constituées de:
- un réseau de transistors P.MOS, entre l'alimentation (VCC) et la sortie.
- un réseau de transistors N.MOS, entre la masse (VSS) et la sortie.

Ces réseaux sont tels que pour chaque transistor N.MOS correspond un transistor P.MOS recevant la même entrée. Ces réseaux sont caractérisés par le fait que, pour chaque valeur d'entrées, lorsqu'il existe un chemin fermé entre la masse VSS et la sortie, il n'existe pas de chemin fermé entre l'alimentation (VCC) et la sortie, et lorsqu'il n'existe pas de chemin fermé entre la masse (VSS) et la sortie, il en existe un entre l'alimentation (VCC) et la sortie.

Si dans ces réseaux on applique la règle N.MOS <-> P.MOS et si on applique à la porte ainsi obtenue des entrées complémentaires par rapport aux entrées de la porte originale, alors les mêmes chemins seront fermés, les mêmes chemins seront ouverts, et les sorties des deux portes seront identiques.

Si ensuite on applique la règle 

pour des valeurs complémentaires aux entrées des deux portes, on aura les valeurs complémentaires aux sorties de ces deux portes. Par conséquent, l'application des règles: N.MOS <-> P.MOS



donne des portes duales selon la définition D21.

La figure 26 représente la structure générale d'une porte C.MOS entièrement complémentée. On peut noter que des réseaux duaux sont utilisés (réseaux N, réseau P). Ils ne sont pas nécessairement séries parallèles comme défini à la section III.3.3.1.1.

Pour des réseaux séries parallèles, réseau N ou réseau P, les règles données à la section III.3.3.1.2.1. sont utilisées pour la conception desportes originales C.MOS entièrement complémentées.

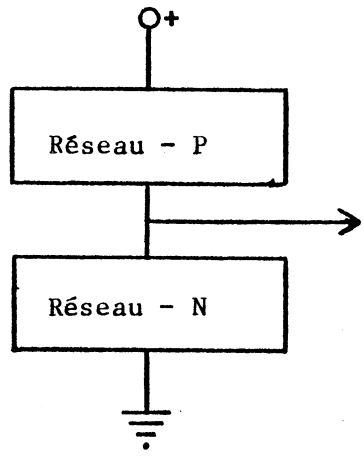


Figure 26 - Schéma général d'une porte entièrement complémentée.

La figure 27 illustre (a) des portes NAND-NOR duales et (b) la porte NOT auto- duale. La figure 28 illustre une porte duale entièrement complémentée plus complexe.

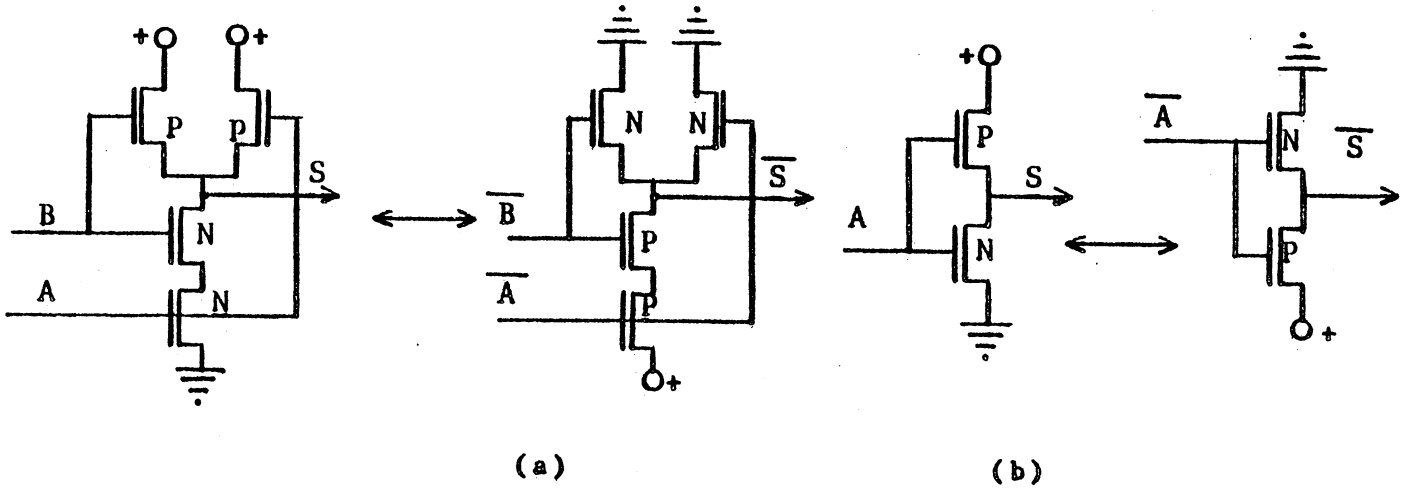


Figure 27 - Portes classiques entièrement complémentées.

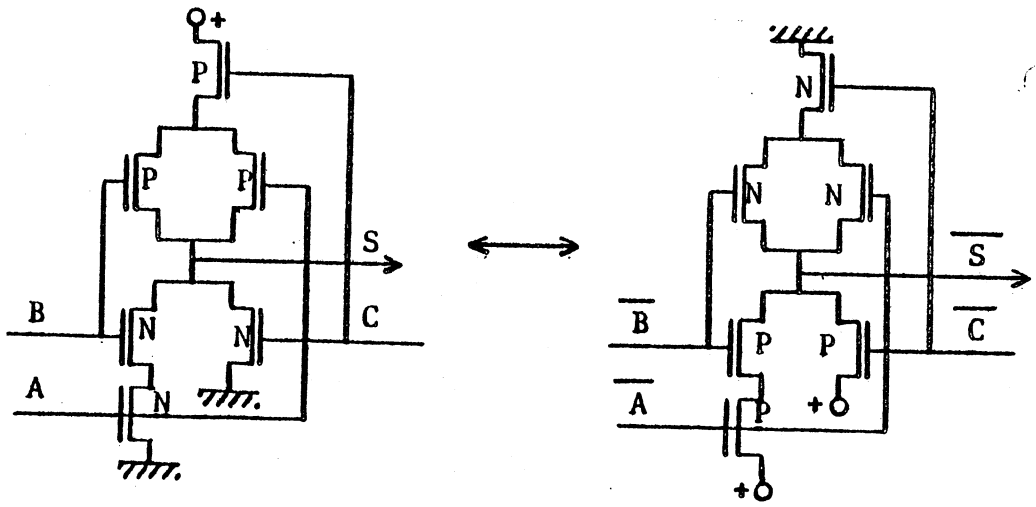


Figure 28 - Portes entièrement complémentées plus complexes.

III.3.3.1.3.2. Portes non entièrement complémentées

Les portes non entièrement complémentées ont un schéma général semblable à celui montré figure 26 mais les réseaux N et P ne sont pas des réseaux duaux comme défini dans la section précédente (portes entièrement complémentées), à un transistor N.MOS correspond, ou non, un transistor P.MOS et vice-versa. Dans ce cas,

les circuits duaux P.MOS et N.MOS doivent être utilisés.

III.3.3.1.3.3. Portes C.MOS à charge commandé

La figure 29 donne le schéma général d'une porte C.MOS à charge commandé.

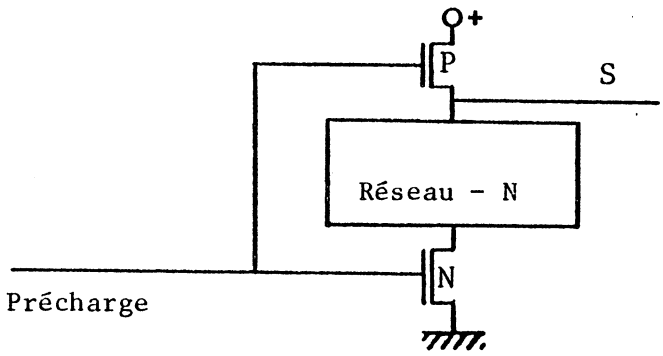


Figure 29 - Schéma général d'une porte C.MOS à charge commandé.

La fonction logique est réalisée en utilisant seulement un réseau de transistors N.MOS. Le circuit dual de cette porte C.MOS sera donc obtenu en utilisant le dual du réseau N. Les règles et remarques de la section III.3.3.1.2.1. sont appliquées pour obtenir un tel réseau N dual. La figure 30 représente le schéma général (a) du circuit original et (b) de la porte duale et la figure 31 donne un exemple.

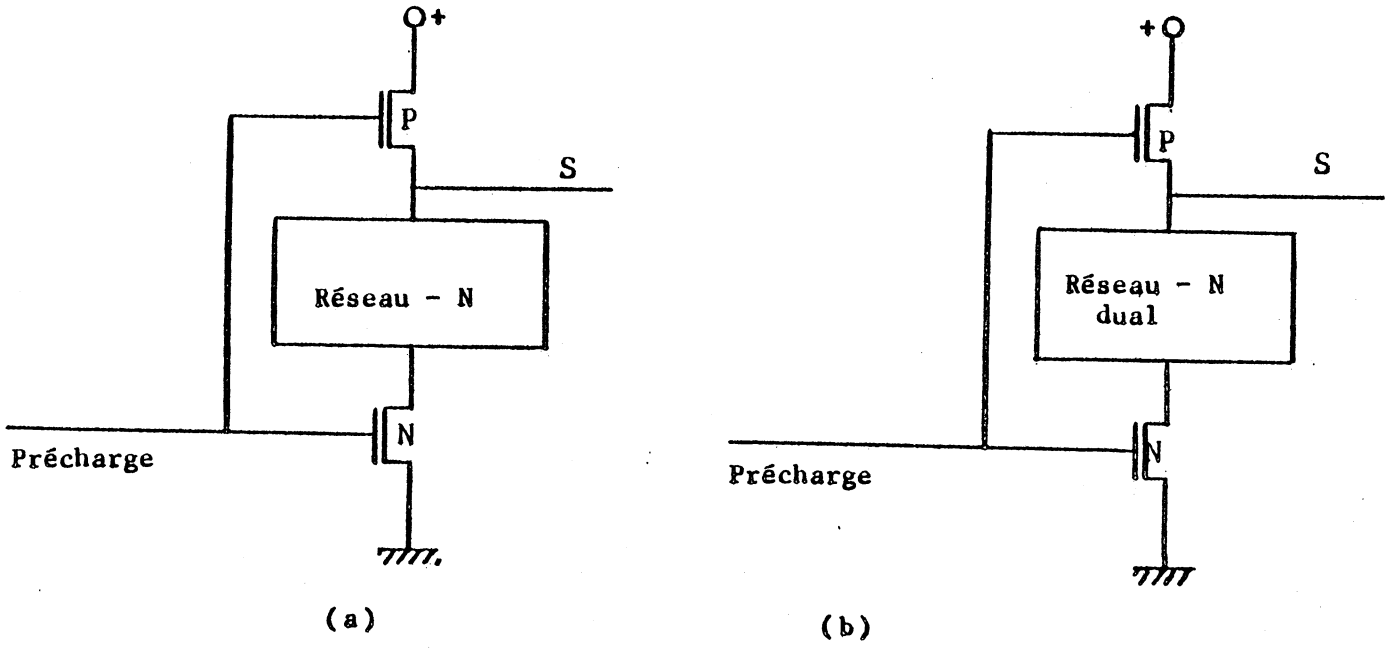


Figure 30 - Porte originale et porte duale.

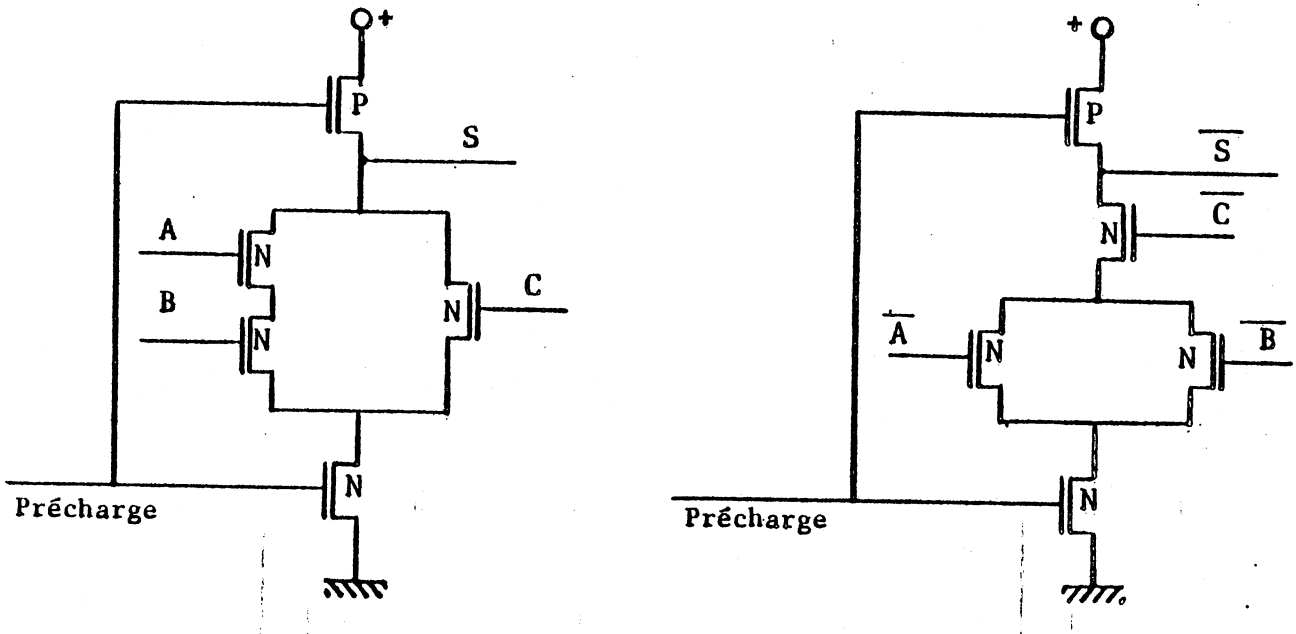


Figure 31 - Exemple d'une porte originale et d'une porte duale.

Les règles et remarques précédentes peuvent être ainsi appliquées aussi à la logique "DOMINO" dynamique ou statique, de [KRA 82] ou

au "NP-CMOS" de [GON 82]. La figure 32 donne le schéma général de ces deux types de conception C.MOS.

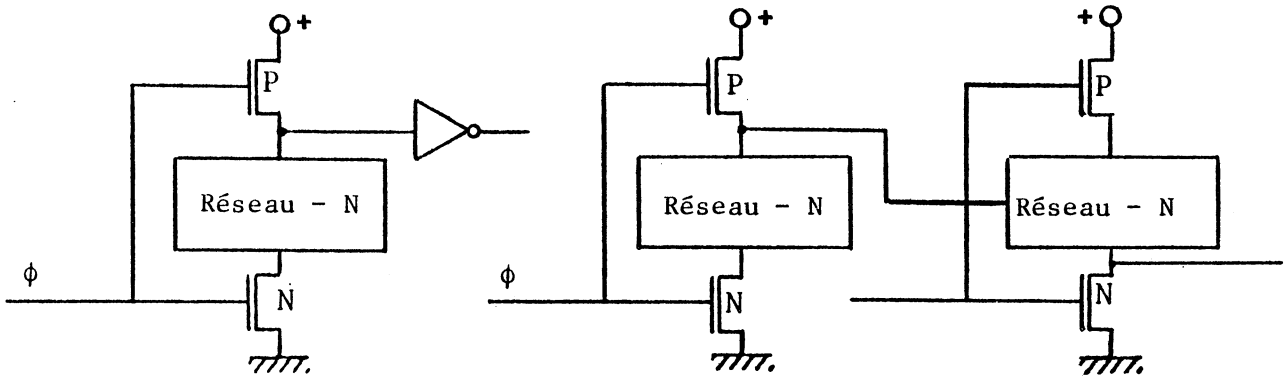


Figure 32 - (a) domino CMOS
(b) NP CMOS

Enfin, les portes C2 MOS [SUW 73] suivraient des règles de la section complète.

III.3.3.1.3.4. Portes de transfert

La figure 33 représente (a) la porte de transfert originale et (b) la porte de transfert duale obtenue, en utilisant les règles générales de la section III.3.3.1.3.

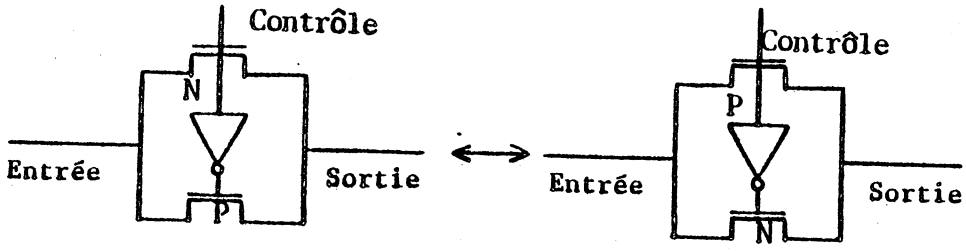


Figure 33 - (a) porte de transfert originale
(b) porte de transfert duale.

Donc, pour des circuits duaux incluant des portes de transfert, pour des entrées complémentées et des lignes de contrôle complémentées, les sorties sont complémentées et des circuits duaux réels peuvent être ainsi conçus. Ceci n'est pas le cas pour la technologie N.MOS dans laquelle il n'était pas possible d'utiliser des lignes de contrôle complémentaires. Ceci est dû à l'existence en technologie C-MOS de deux transistors (N.MOS et P.MOS) qui ont des fonctions duales.

III.3.3.1.3.5. Registres

Comme il a été noté en section III.3.3.1.2.5., la fonction des registres (stockage des données) doit être identique dans les deux circuits (circuit original et circuit dual). Pour la technologie N.MOS on a remarqué que les registres doivent être identiques associés à des lignes de contrôle identiques (c'est à dire non complémentées). Ceci n'est pas le cas pour la technologie C.MOS pour laquelle les circuits duaux et les lignes de contrôle duales peuvvent être utilisées. La figure 34 représente (a) une conception classique d'un registre et (b) le circuit dual, associé aux lignes de contrôle duales (c'est à dire complémentées).

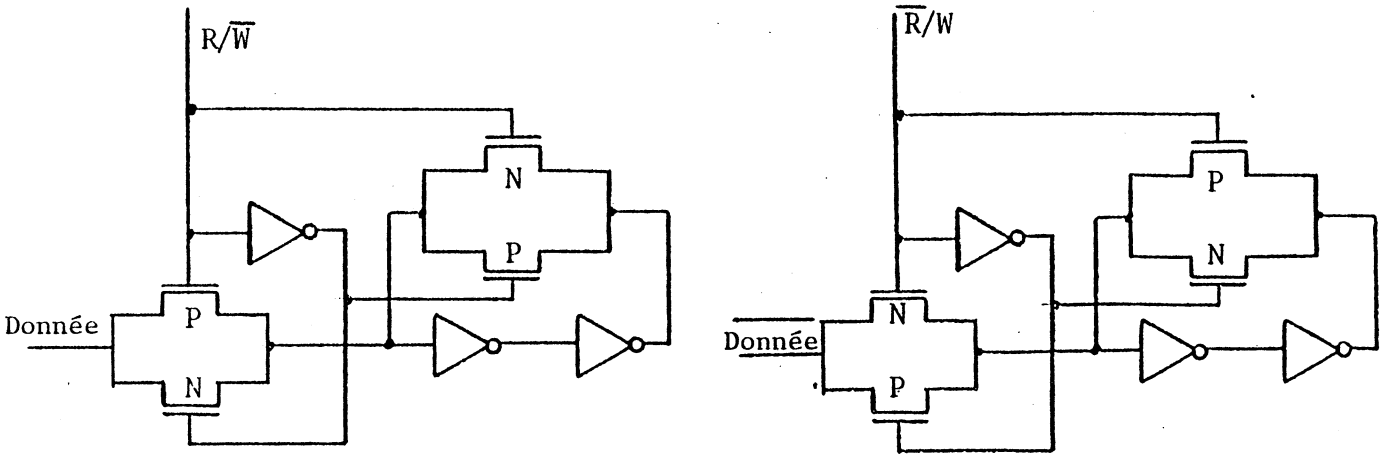


Figure 34 - (a) registre classique original
(b) registre dual.

Un autre dessin original, souvent utilisé pour une tension d'alimentation VCC de 12 volts au lieu de 5 volts, est représenté à la figure 35 (a). pour ce registre encore, le circuit dual est obtenu en appliquant des règles générales données à la section III.3.3.1.3. La figure 21 (b) représente le circuit dual qui doit être associé aux lignes de contrôle complémentées.

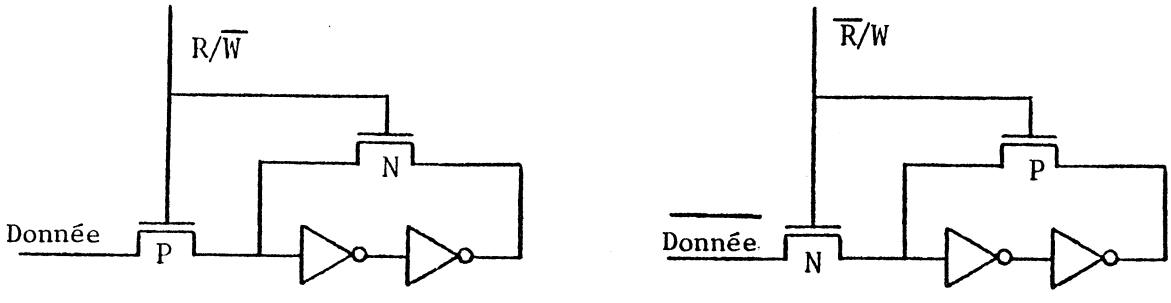


Figure 35 - (a) registre original
(b) registre dual.

On doit noter que ces registres consomment une certaine quantité de courant.

III.3.3.1.4. Conclusions

Il est nécessaire de rappeler que les règles générales ne sont pas toujours applicables pour la conception de circuits duaux. Nous développons maintenant deux sections, une concerne le test des circuits duaux et la seconde concerne les problèmes de CAO.

III.3.3.1.4.1. Test des circuits duaux

Les circuits duaux ont des caractéristiques spécifiques en ce qui concerne le test. Successivement les circuits N.MOS et C.MOS sont examinés.

A - Circuits N.MOS

Dans ce paragraphe on ne considère que les portes N.MOS à charge passive ou à charge commandée qui sont constitutées d'un MOS de charge ou de précharge associé à un réseau séries parallèles dans lequel une entrée est utilisée seulement une fois (seulement un transistor est controlé pour une entrée). Cette entrée peut être représentée par une variable directe ou complémentée (la fonction réalisée est ainsi "unate". De cette façon, à chaque transistor appartenant à la porte originale correspond un transistor appartenant à la porte duale. Ainsi, un modèle de défaut peut être considéré pour les deux portes: transistors s-on et s-open dans le réseau séries parallèles. D'autres types de défauts tels que coupures de connexions ne sont pas considérés lorsqu'il n'existe pas d'élément correspondant dans les portes duales (les coupures équivalentes aux collages ouverts de MOS sont implicitement considérées).

Pour le modèle mentionné ci-dessus, si un vecteur d'entrée est un vecteur de test d'une panne de la porte originale, le vecteur complémentaire est un vecteur de test pour la porte duale. Ceci est facile à vérifier. Supposons qu'un vecteur d'entrée est un vecteur de test pour un transistor s-on ou s-open. Cela veut dire qu'il existe des valeurs d'entrée a1, a2,..., an telles que:
$S(a1,a2,\ldots,0,\ldots.an) = 1$

$S(a1,a2,\ldots,1,\ldots,an) = 0$
L'entrée qui varie est supposée être directe. Ces deux vecteurs d'entrée sont des vecteurs de test pour un MOS s-on ou s-open. Pour la porte duale, ces entrées sont telles que:
$S'(\overline{a1},\overline{a2},\ldots,1,\ldots\overline{an}) = 0$
$S'(\overline{a1},\overline{a2},\ldots,0,\ldots\overline{an}) = 1$

Ainsi ces vecteurs d'entrée sont des vecteurs de test pour un MOS s-open ou s- on.

Si on se réfère aux vecteurs de test donnés dans [ELZ 79] pour le test des réseaux séries parallèles des portes N.MOS, il est facile de vérifier que les vecteurs de test pour la porte duale (vecteurs complémentaires) sont donnés par les matrices transposées.
$M^{t}01$, $M^{t}02,\ldots,M^{t}11$, $M^{t}12\ldots$ (notations de [ELZ 79].

D'autres portes ne sont pas considérées dans ce rapport.

B - Portes C.MOS

Dans ce paragraphe, on ne considère que les portes C.MOS entièrement complémentaires. La même démonstration que celle utilisée pour des réseaux séries parallèles des portes N.MOS peut être faite, il est alors facile de vérifier que: si un vecteur d'entrée est un vecteur de test pour un MOS s-on du réseau N (et un vecteur de test pour un MOS s-open du reseau P), le vecteur complémentaire est un vecteur de test pour un MOS s-open du réseau N de la porte duale (et un vecteur de test pour un MOS s-on du circuit P de la porte duale).

Il faut noter qu'en fait des séquences de test doivent être considérées, sans aucun problème, pour le test réel de MOS s-open. Les autres types de portes C.MOS ne sont pas examinées dans ce rapport.

Les résultats mentionnés ci-dessus sont une extension des résultats dans [CRO 78].

III.3.3.1.4.2. CAO et circuits duaux

Le deuxième point de la conclusion concerne un aspect essentiel des circuits VLSI d'aujourd'hui: la CAO. Deux facettes différentes sont examinées. La première concerne la génération automatique des circuits duaux d'un circuit original. Pour quelques portes, il est possible d'automatiser la structure de la porte duale (c'est le cas des réseaux séries parallèles pour les portes N.MOS ou des portes C.MOS entièrement complémentées etc...) . Pour d'autres portes, des systèmes experts pourront être un moyen d'obtenir cette structure. Après avvoir obtenu la structure, des outils de CAO peuvent être utilisés pour automatiser les calculs électriques qui sont nécessaires à la détermination des dimensions exactes des transistors (c'est le cas des portes N.MOS pour lesquelles les dimensions dépendent de la structure séries parallèles et des portes C.MOS pour lesquelles les dimensions dépendent de la structure et du type N.MOS/P.MOS de transistors.

L'autre facette concerne plus particulièrement le développement de la conception des circuits duaux simultanément avec l'évolution de la conception des VLSI. Il est clair que l'évolution est dirigée vers une conception structurée qui implique l'utilisation des cellules de haut niveau appartenant à des bibliothèques.

Dans ce contexte, il serait intéressant d'avoir des cellules duales disponibles pour être capable de concevoir des circuits originaux à partir de cellules originales en bibliothèque et des circuits duaux à partir des cellules duales en bibliothéque.

IV - CONCEPTION DE SYSTEMES SFS: CONDITIONS, NOUVEAUX CODES, INTERCONNEXIONS.

Dans les chapîtres précédents nous avons donné des règles pour la conception des systèmes autotestables controlés vis à vis d'erreurs simples, unidirectionnelles et multiples. Le but du présent chapître est de donner des conditions nécessaires et suffisantes d'applicabilité de certaines de ces règles ainsi que des moyens de réalisation lorsque ces conditions ne sont pas vérifiées. Ainsi sont utilisés le contrôle des entrée primaires, de modifications des circuits ou des fonctions pour des circuits controlés vis à vis d'erreurs simples et unidirectionnelles.

Enfin nous introduisons une nouvelle famille de codes qui s'avèrent trés utiles pour la conception de certains circuits autotestables.

IV.1. Circuits autotestables controles vis à vis des erreurs simples

IV.1.1. Préliminaires

La règle R1, proposée dans le chapître III.1. pour assurer la propagation des erreurs simples dans un bloc fonctionnel en erreurs simples aux sorties primaires du bloc, est redonnée ci-dessous.

Règle R1

Le degré de divergence maximal du bloc fonctionnel, pour l'ensemble de toutes les lignes du bloc, est égal à 1.

Cette règle peut être remplacée par une règle équivalente R1'.

Règle R1'

Le degré de divergence maximal du bloc fonctionnel, pour l'ensemble de toutes les entrées primaires du bloc est égal à 1.

Lemme 1

Les règles R1 et R1' sont équivalentes.

Preuve

Le Lemme 1 est démontré facilement. Si la règle R1 est vérifiée il est évident que la règle R1' est aussi vérifiée. Si la règle R1 n'est pas vérifiée, il existe au moins une ligne dont le degré de divergence est plus grand que 1. Cette ligne est soit une entrée primaire, soit une ligne interne. S'il s'agit d'une entrée primaire, alors la régle R1' n'est pas vérifiée. S'il s'agit d'une ligne interne alors chaque entrée primaire connectée (par l'intermédiaire de chemins) avec cette ligne interne possède un degré de divergence plus grand que 1 ce qu'il faut démontrer.

Une autre règle non équivalente aux règles R1 et R1' sera utilisée.

Règle R1"

Le degré de divergence maximal du bloc fonctionnel pour l'ensemble de toutes les lignes du bloc (les entrées primaires étant exceptées) est égal à 1.

Un circuit pour lequel la règle R1 (R1') n'est pas vérifiée mais pour lequel la règle R1" est vérifiée, est un circuit tel qu'il existe au moins une entrée primaire dont la parité de divergence est plus grande que 1 et tel que le degré de divergence de toutes les lignes internes est égal à 1.

IV.1.2. Remarques concernant la conception des systèmes autotestables controlés vis à vis des erreurs simples

La conception des systèmes pour lesquels la règle R1 (R1') est vérifiée, doit être telle que si une entrée Ii apparaît dans la fonction boolèenne représentant une sortie, cette entrée ne doit pas apparaître dans la fonction boolèenne d'autres portes (dans le cas contraire, le degré de divergence de l'entrée Ii sera plus grand que 1).

De plus, les sorties doivent être codées pour donner dans le cas présent un code de sortie détectant des erreurs simples (code de parité) ; la façon la plus simple d'avoir un tel code de sortie est

de générer une sortie supplémentaire (le bit de parité des sorties). Par conséquent certaines entrées auront un degré de divergence égal à 2, même si le degré de divergence maximal du circuit initial étant égal à 1.

Des méthodes permettant de surmonter ce problème sont données ci-après.

IV.1.3. Partition

Si la règle R1 (R1') n'est pas vérifiée, la première méthode pour résoudre le problème consiste à diviser le bloc fonctionnel en blocs tels que chacun de ces blocs vérifie la règle R1 (R1'), la figure 36 donne un tel exemple.

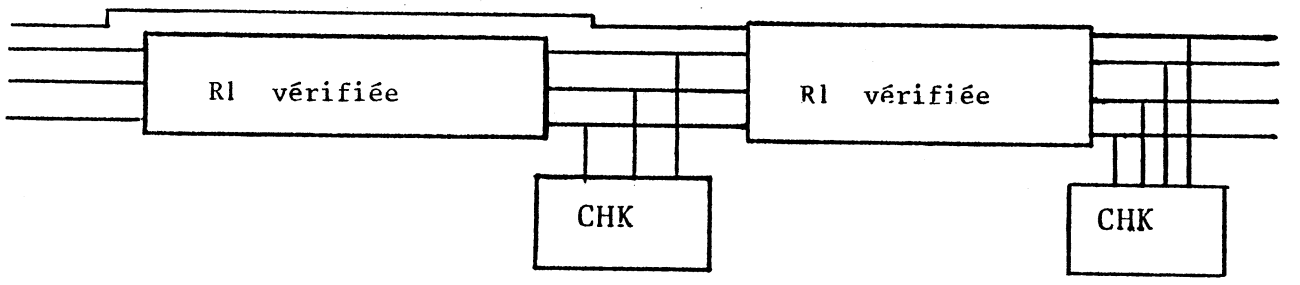


Figure 36 - Partition

Cette méthode est utilisée dans les applications (chapître V) pour la conception des PLAs contrôlés pour des erreurs simples.

IV.1.4. Contrôle des entrées primaires

Dans cette section, on considère des circuits pour lesquels la règle R1 (R1') n'est pas vérifiée, mais la règle R1'' est vérifiée. Le degré de divergence de tels circuits est supérieur à 1, mais ceci est dû seulement aux entrées primaires, étant donné que le degré de divergence de toute autre ligne est égal à 1. Un tel circuit est représenté figure 37.

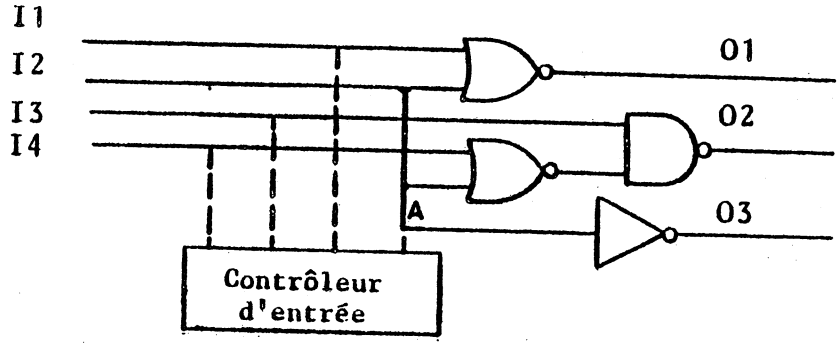


Figure 37 - Circuit pour lequel la règle R1' n'est pas vérifiées, mais la règle R1" est vérifiée.

Dans ce circuit, le degré de divergence de l'entrée I2 est égal à 3. Dans le cas où cette entrée est conrolée jusqu'au point A, le controleur de sortie doit controler uniquement les erreurs simples sur une des 3 branches de la ligne I2. Ainsi le bloc controlé par le controleur de sortie (bloc initial en exception de l'entrée primaire I2) vérifie la règle R1 (R1').

IV.1.5. Contrôle des entrées primaires et redondance

Dans cette section on considère des circuits tels que ni la règle R1 (R1'), ni la règle R1", sont vérifiées. Le degré de divergence de tels circuits est supérieur à 1 ; ceci est dû aux lignes internes du circuit. Un tel circuit est présenté figure 38.

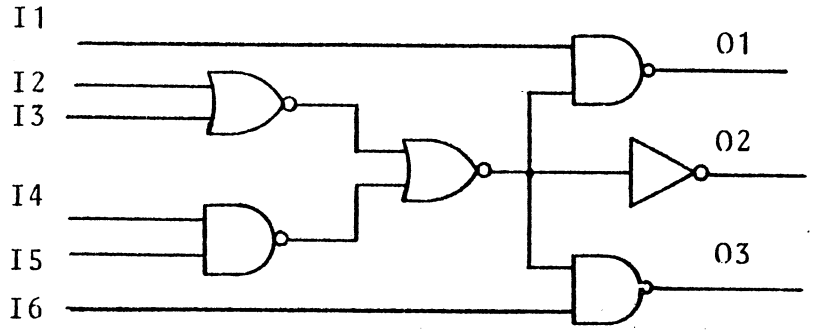


Figure 38 - Un circuit original

L'idée développée à la section IV.1.4. est reprise ici mais dans ce cas la replication de certaines parties du circuit est nécessaire pour obtenir un circuit qui respecte la règle R1".

On repère toutes les lignes dont le degré de divergence est supérieur à 1. Pour de telles lignes, certaines parties du circuit doivent être repliquées; le nombre des répétitions d'une partie sera égale au nombre de sorties accédées par cette partie. Quelques entrées primaires sont aussi repliquées. Enfin les repliques de chaque entrée primaire sont connectées et les entrées primaires sont controlées de la façon proposée à la section IV.1.4.

La figure 39 représente la modification du circuit de la figure 38 obtenue par cette méthode.

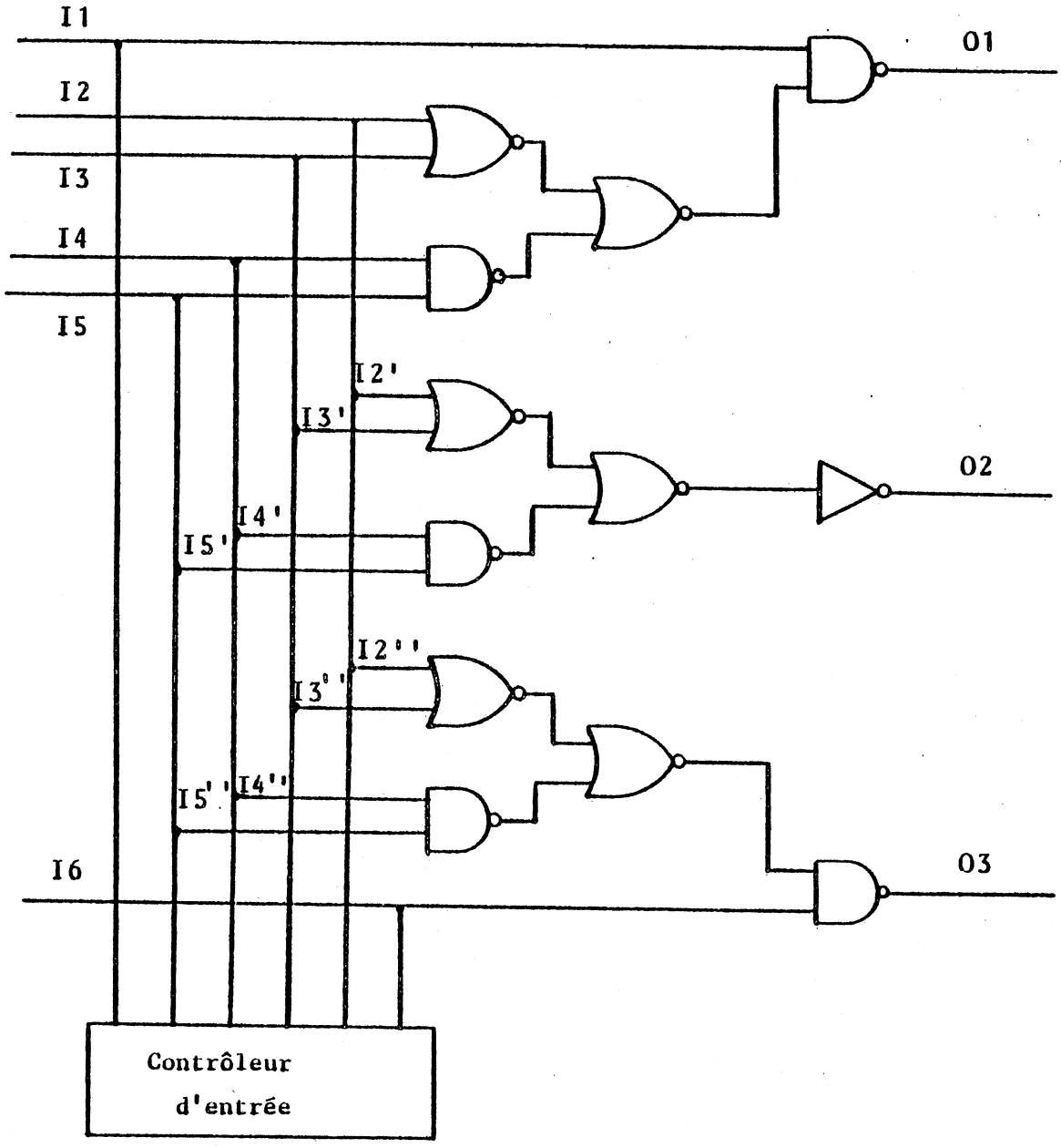


Figure 39 - Circuit modifié pour être contrôlé vis à vis des erreurs simples.

IV.1.6. Contrôle des entrées primaires et redondance logique

Certains inconvénients de la méthode donnée à la section IV.1.5. sont les suivants:

- il n'est pas possible de dessiner directement le circuit autotestable ; on est obligé de dessiner d'abord un circuit ordinaire (figure 38) et de dessiner ensuite le circuit modifié

(figure 39).
- la méthode n'est pas facilement automatisable pour les systèmes de CAO.
- quelques fois cette méthode produit des redondances non nécessaires, il faut alors redessiner le circuit afin de l'optimiser.

Dans cette section nous proposons une procédure pour éliminer ces inconvénients.

Procédure Pc1

Première étape
On considère les q fonctions booléennes des q sorties primaires du circuit.
Chaque occurrence d'une entrée divergente Ii (Ii apparaît dans deux ou plusieurs fonctions booléennes), dans la fonction booléenne d'une sortie primaire Ok est remplacée par Ii^k.

Après la première étape, on possède les fonctions redondantes des sorties primaires. Dans ces fonctions, chaque ligne Ii^k n'est pas divergente et la règle R1 (R1') est vérifiée.

Deuxième étape
On réalise le dessin optimal pour les fonctions redondantes.

Pour le dessin obtenu par la deuxième étape, la règle R1 (R1') est vérifiée mais quelques entrées primaires sont repliquées.

Troisième étape
Pour k=1 jusqu'à k=q on interconnecte toutes les lignes Ii^k.

Le circuit donné par la troisième étape réalise les fonctions booléennes initiales et la règle R1" est vérifiée. La méthode proposée dans la section IV.1.4. peut maintenant être appliquée.

La procédure proposée dans cette section est très facilement

automatisable pour les systèmes de CAO et le circuit augmenté est dessiné et optimisé directement. L'application de cette procédure au circuit de la figure 3 donne les fonctions redondantes suivantes (première étape):

$$O_1 = \overline{I_1(I_{21} + I_{31})I_{41}\ I_{51}}$$

$$O_2 = \overline{(I_{22} + I_{32})I_{42}\ I_{52}}$$

$$O_3 = \overline{I_6(I_{23} + I_{33})I_{43}\ I_{53}}$$

Par les étapes 2 et 3 on obtient le circuit de la figure 40 qui nécessite 17 MOS au lieu de 35 MOS pour le circuit de la figure 39 (le circuit initial (figure 38) possède 17 MOS).

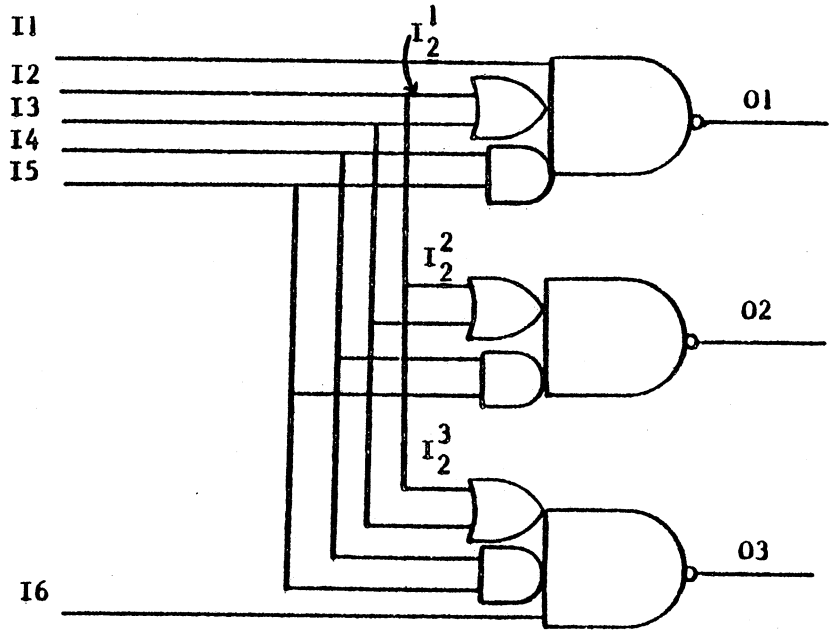


Figure 40 -

IV.1.7. Remarques et conclusions

Notons que pour certains circuits on peut utiliser soit la méthode de la section IV.1.3. soit la méthode donnée aux sections IV.1.4, IV.1.5, IV.1.6. Dans ce cas, un choix reste à faire. Le critère de ce choix est l'augmentation globale de la surface. Dans d'autres

cas, une combinaison de la méthode de la section IV.1.3. et des méthodes des sections IV.1.4. , IV.1.5 et IV.1.6. peut être utilisée.

IV.2. Circuits autotestables controlés vis à vis des erreurs unidirectionnelles

IV.2.1. Préliminaires

La règle R5 est proposée dans le chapître III.2. pour assurer la propagation des erreurs simples dans un bloc fonctionnel en erreurs unidirectionnelles aux sorties primaires du bloc. Cette règle est rappelée ci-après.

Règle R5

Tous les chemins entre une ligne divergente et les sorties primaires du bloc ont la même parité d'inversion.
Cette règle peut être remplacée par une règle équivalente R5'.

Règle R5'

Tous les chemins entre une entrée primaire divergente et les sorties primaires du bloc ont la même parité d'inversion.

Lemme 2

Les règles R5 et R5' sont équivalentes.

Preuve

Le Lemme 2 est démontré facilement. Si la règle R5 est vérifiée, il est évident que la règle R5' est aussi vérifiée. Si la règle R5 n'est pas vérifiée, il existe au moins une ligne divergente telle qu'il existe au moins deux chemins entre cette ligne et les sorties primaires du bloc, avec des parités d'inversion différentes. Cette ligne divergente est soit une entrée primaire, soit une ligne interne. S'il s'agit d'une entrée primaire alors la règle R5' n'est pas vérifiée.S'il s'agit d'une ligne interne, alors on peut voir que chaque entrée primaire connectée (par l'intermédiaire des chemins) avec cette ligne interne, est une entrée divergente et

qu'il existe au moins deux chemins entre cette entrée divergente et les sorties primaires avec des parités d'inversion différentes. C.q.f.d.

La conception d'un circuit G tel que la règle R5' soit vérifiée et tel que le code de sortie B détecte les erreurs unidirectionnelles, impose des restrictions au code d'entrée et à la fonction implémentée. Par la suite ces restrictions seront examinées.

Notons que généralement les entrées primaires ont un degré de divergence supérieur à 1, étant donné que les entrées primaires sont utilisées pour générer la partie information et la partie codage des sorties primaires.

IV.2.2. Une conditions nécessaire et suffisante pour concevoir des circuits dont le code de sortie détecte les erreurs unidirectionnelles et tels que la règle R5' (R5) soit vérifiée

Un Lemme simple est d'abord donné.

Lemme 3

Pour chaque couple de vecteurs d'entrée $(a,a') \in A^2$, a et a' étant différents pour une seule entrée Ii, les vecteurs de sortie doivent être identiques pour que la règle R5' (R5) soit vérifiée et pour que le code de sortie détecte les erreurs unidirectionnelles.

Preuve

Soit deux vecteurs d'entrée a et $a' \in A$ qui sont différents seulement pour l'entrée Ii. Soit un défaut f qui ne modifie que l'entrée Ii du vecteur d'entrée a. Le vecteur a' sera alors appliqué au circuit au lieu du vecteur a. On a alors

$G(a,f) = G(a',0)$

mais une erreur simple sur une entrée Ii ne peut provoquer que des erreurs unidirectionnelles aux sorties primaires du circuit (la règle R5 etant vérifiée), cette erreur sera donc détectée par le code B de sortie, c'est à dire

$G(a,f) = G(a,0)$ ou $G(a,f) \notin B$
Etant donné que par hypothèse $G(a',0) \in B$ on en déduit que
$G(a,f) = G(a,0) = G(a',0)$

Par la suite on donne un Lemme plus général. On considére un circuit G qui assure la règle R5' (R5). Toutes ses entrées primaires peuvent être alors divisées en deux groupes : g1 et g2. Le groupe g1 contient toutes les entrées primaires telles que tous les chemins entre une telle entrée et les sorties primaires ont une parité d'inversion égale à 1. Le groupe 2 contient les entrées primaires telles que tous les chemins entre une telle entrée et les sorties primaires ont une parité d'inversion égale à 0.

Lemme 4
Soit G un circuit tel que :
- la règle R5' est vérifiée.
- le code de sortie B détecte les erreurs unidirectionnelles.

Alors pour chaque vecteur d'entrée $a \in A$, soit $G(a',0) \notin B$ soit $G(a',0)=G(a,0)$. a' étant un vecteur d'entrée ($a' \in X$) tel que a couvre a' pour toutes les entrées du groupe g1 et a' couvre a pour toutes les entrées du groupe g2 ou le contraire (a couvre a' pour une entrée Ii si a prend la valeur 1 pour l'entrée Ii, lorsque a' prend la valeur 1 pour la même entrée).

Preuve
Soit un défaut f qui produit aux entrées primaires du groupe g1 une erreur unidirectionnelle du type 0 -> 1 (resp. 1 -> 0) et aux entrées primaires du groupe g2 une erreur unidirectionnelle du type 1 -> 0 (resp. 0 -> 1).

Selon les propriétés des vecteurs a et a': en présence d'un tel défaut f le vecteur a' peut être appliqué au circuit à la place du vecteur a. On a alors $G(a,f)=G(a',0)$. Un tel défaut f ne peut provoquer aux sorties primaires du circuit que des erreurs unidirectionnelles à cause des parités d'inversion des chemins entre les entrées du groupe g1 et des sorties primaires et à cause

des parités d'inversion des chemins entre les entrées du groupe g2 et les sorties primaires du circuit. Ces erreurs sont détectées par B. On a alors:

$G(a,f) = G(a,0)$ ou $G(a,f) \notin B$

et en replaçant $G(a,f)$ par $G(a',0)$ on obtient

$G(a',0) = G(a,0)$ ou $G(a',0) \notin B$

CQFD.

Lemme 5

S'il existe une partition des entrées primaires d'un circuit en deux groupes g1 et g2 de façon à ce qu'il n'existe pas deux vecteurs d'entrée $(a, a') \in A^2$ telles que a couvre a' pour toutes les entrées du groupe g1 et a' couvre a pour toutes les entrées du groupe g2 (ou le contraire). Alors pour chaque fonction logique on peut concevoir un circuit qui assure la règle R5' (R5).

Pour avoir un circuit tel que la règle R5' soit vérifiée, il est nécessaire que les parités d'inversion des chemins entre une entrée primaire et les sorties primaires, sont égales à 0 ou ils sont égales à 1. Pour un tel circuit, si une entrée primaire est modifiée, la modification des sorties primaires est exclusivement du type 0 -> 1 ou exclusivement du type 1 -> 0.

Par la suite, nous présentons la définition et une propriété des fonctions monotones, puis la procédure qui permettra de construire un circuit qui vérifie la règle R5', selon le Lemme 5.

Définition D20

Une fonction booléenne est :

- croissante monotone si $x \lessdot y$ implique $f(x) \leq f(y)$
- décroissante monotone si $x \lessdot y$ implique $f(x) \geq f(y)$

($x \lessdot y$ signifie que y couvre x)

Théorème T1 [HAR 65]

Une fonction booléenne est croissante (décroissante) monotone si elle peut être représentée par une somme de monômes composés par des variables non compléementées (complémentées) uniquement.

Procédure Pc2

Soit les fonctions des sorties du circuit, exprimées comme la somme des monômes complets. Par ces monômes on élimine toutes les variables du groupe gl qui sont complémentées (resp. incomplémentées) et toutes les variables du groupe g2 qui sont incomplémentées (resp. complémentées). Ainsi les fonctions de sorties ne sont modifiées que pour des vecteurs d'entrée (monômes) qui n'appartiennent pas au code d'entrée A, car il n'existe pas a et a'$\in$A tels que a couvre a' (resp. a' couvre a) pour les entrées du groupe gl et a' couvre a (resp. a couvre a') pour les entrées du groupe g2. Les fonctions de sortie sont donc monotomes. La règle R5 est vérifiée pour ces fonctions et le Lemme 5 est vérifié.

Le Lemme 5 est une extension des résultats donnés dans [MAG 73] ; d'après ces résultats toutes les fonctions peuvent être réalisées de façon monotone si un code d'entrée non ordonné (aucun a$\in$A ne couvre un a'$\in$A) est utilisé.

Lemme 6

Si les entrées primaires d'un circuit G peuvent être divisées en deux groupes gl et g2 de façon à ce que pour chaque couple de vecteurs d'entrée (a,a')$\in A^2$, tels que a couvre a' (resp. a' couvre a), pour toutes les entrées du groupe gl et a' couvre a (resp. a couvre a') pour toutes les entrées du groupe g2, nous avons G(a,0) = G(a',0), alors pour chaque fonction G on peut concevoir un circuit qui assure la règle R5' (R5).

Une procédure similaire à la procédure Pc2 peut être utilisée pour démontrer le Lemme 6.

Procédure Pc3

Soit les fonctions des sorties du circuit, exprimées comme la somme des monômes complets (vecteurs d'entrée), appartenant au code d'entrée A. Soit a un tel monôme qui apparaît dans la fonction de la sortie Ok, alors nous avons Ok(a,0) = '1'.

Pour chaque entrée Ii appartenant au groupe g1 on élimine les occurrences de $\overline{Ii}$ (resp. Ii) dans les monômes des fonctions des sorties. Pour chaque entrée Ij appartenant au groupe g2 on élimine les occurrences de Ij (resp. $\overline{Ij}$) dans les monômes des fonctions des sorties.

Si le monôme a est modifié par cette procédure, la fonction Ok est modifiée uniquement pour des monômes a' tels que a' couvre a (resp. a couvre a') pour toutes les entrées du groupe g1 et a couvre a' (resp. a' couvre a) pour toutes les entrées du groupe g2. Par cette modification Ok est forcée à la valeur '1' pour les monômes a' ; si $a' \notin A$ alors il n'y a pas de problème ; si $a' \in A$, par les hypothèses du Lemme 6 on a Ok(a',0) = Ok(a,0) = 1 et donc Ok est forcée à la valeur correcte.

De façon similaire au Lemme 5, la règle R5' est assurée.

Proposition P23

Une condition nécessaire et suffisante qui permet d'implémenter un circuit G tel que la règle R5' soit vérifiée et le code de sortie B détecte les erreurs unidirectionnelles est que: les entrées primaires du circuit peuvent être divisées en deux groupes g1, g2 tels que pour chaque couple de vecteurs $(a,a') \in A^2$ tels que a couvre a' (resp. a' couvre a) pour les entrées du groupe g1 et a' couvre a (resp. a couvre a') pour les entrées du groupe g2 on a G(a,0) = G(a',0).

La proposition P23 est vraie car le Lemme 4 donne une condition nécessaire tandis que le Lemme 6 donne une condition suffisante.

Noton que cette condition doit être respectée par la fonction modifiée G' (G' est la fonction G, modifiée de façon à ce que les sorties du circuit soient codées en un code non ordonné, en réalité on doit implémenter la fonction G'), mais si la condition est respectée par la fonction initiale G elle sera aussi respectée par la fonction modifiée G', car G(a,0) = G(a',0) implique G'(a,0) = G'(a',0).

La proposition P23 donne une condition qui permet de vérifier si une fonction peut être ou non réalisée par un circuit tel que la règle R5 (R5') soit vérifiée et tel que son code de sortie détecte les erreurs unidirectionnelles. Cette condition est très stricte et par conséquent elle sera très rarement vérifiée.

Une façon directe pour atteindre ce but est de générer un code d'entrée A qui force la condition nécessaire (redondance du code d'entrée). Ces codes sont les codes qui ne contiennent pas deux vecteurs a et a' tels que a couvre a' pour un groupe d'entrées g1 et a' couvre a pour le groupe g2 des entrées restantes. Une étude plus systématique de ces codes sera présentée plus tard.

Par la suite nous donnons d'autres méthodes pour surmonter ce problème.

IV.2.3. Partition

De façon similaire à ce qui a été proposé dans le cas des circuits testés vis à vis des erreurs simples, une première méthode est de diviser le circuit en plusieurs blocs pour lesquels la règle R5' (R5) est vérifiée. Chacun de ces blocs doit être controlé séparément.

IV.2.4. Contrôle des entrées primaires

Dans cette section on utilise le contrôle des entrées primaires de façon similaire au contrôle des entrées primaires utilisé à la section IV.1.4. La règle R5" correspondant à la règle R1" sera utilisée.

Règle R5"

Tous les chemins entre une ligne quelconque du circuit (les entrées primaires exceptées), et les sorties du circuit, ont la même parité d'inversion.

La règle R5" n'est pas équivalente à la règle R5' (R5). Dans cette section on considère des circuits qui ne vérifient pas la règle R5' mais qui vérifient la règle R5". Un tel circuit est donné figure 41.

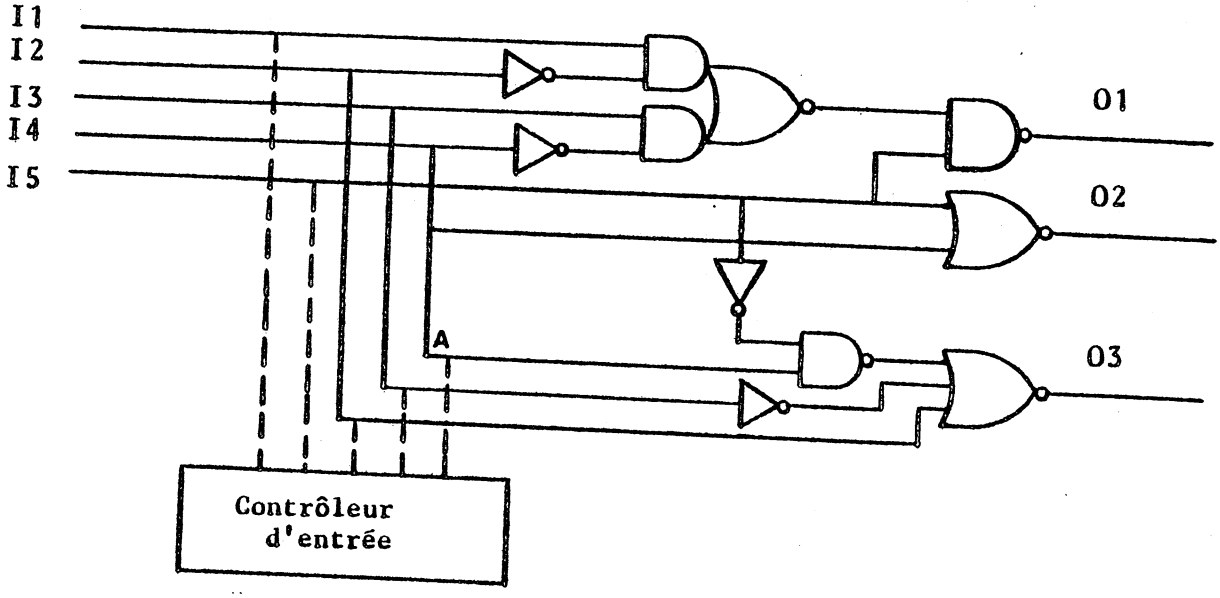


Figure 41 - Circuit pour lequel la règle R5' n'est pas vérifiée, mais la règle R5" est vérifiée.

Dans ce circuit les chemins entre l'entrée primaire I4 et les sorties primaires du circuit n'ont pas la même parité d'inversion. Mais si l'entrée I4 est controlée jusqu'au point A (donné figure 41), alors il n'est pas nécessaire de controler l'entrée I4 avec le controleur de sortie, ce controleur doit controler les trois branches de l'entrée I4 après le point A. Ainsi le bloc testé par le contrôleur de sortie (bloc initial l'entrée primaire I4 exceptée) vérifie la règle R5'.

Plus précisément, si une entrée est le point de départ de plusieurs chemins, le point A sera défini comme dans la figure 42.

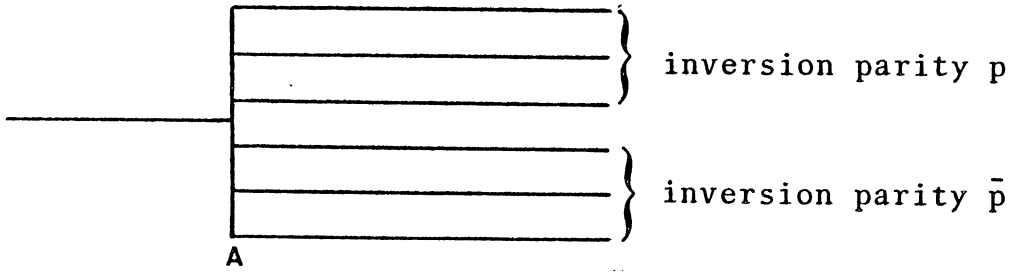


Figure 42 - Position possible du point A.

Une structure très commune qui vérifie la règle R5" est la structure des PLAs ; ceci peut être vérifié à la figure 43. Sur cette figure on remarquera le contrôleur d'entrée qui contrôle les entrées jusqu'aux points critiques.

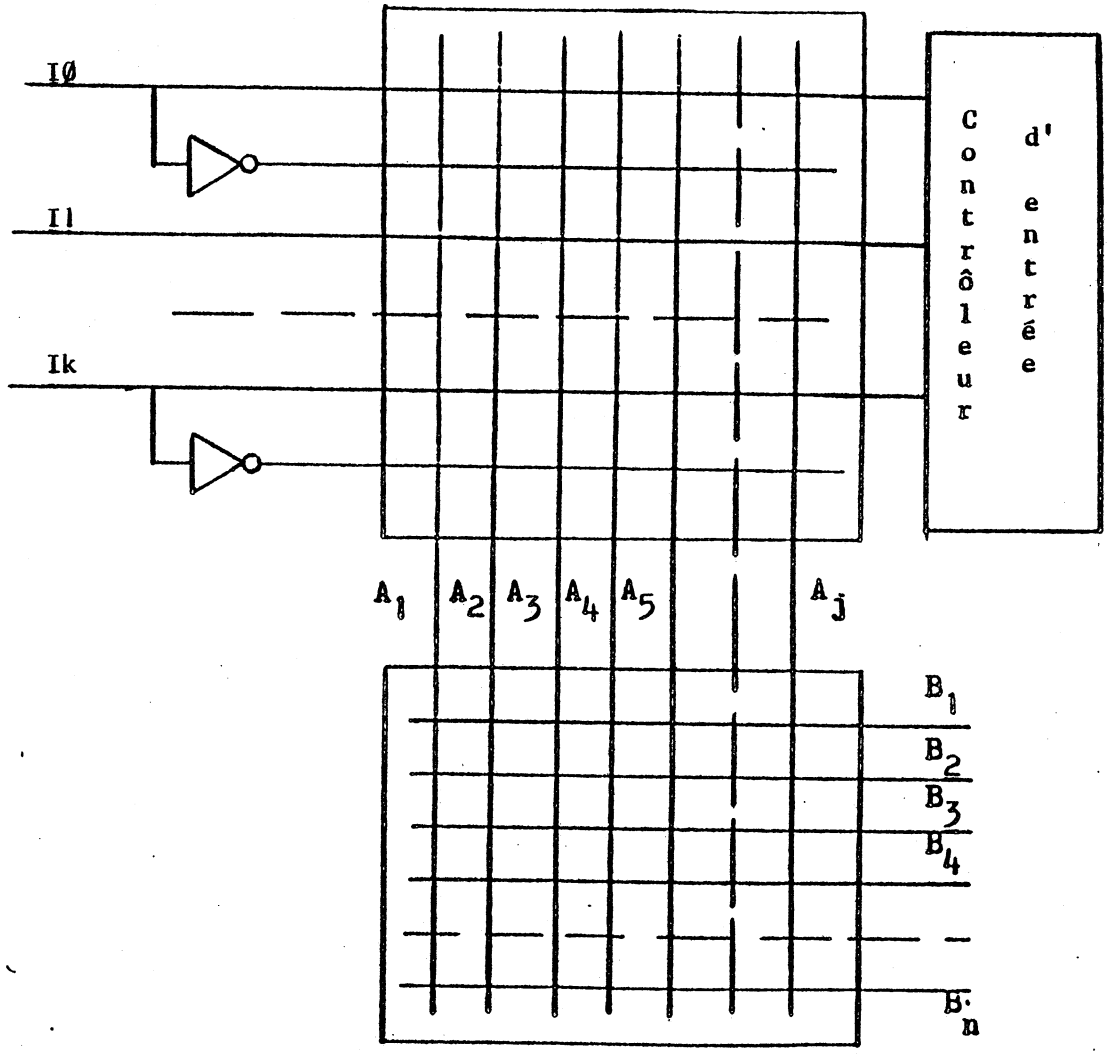


Figure 43 -

IV.2.5. Contrôle des entrées primaires et redondance

Dans cette section on considère des circuits tels que ni la règle R5' (R5), ni la règle R5" sont vérifiées. Dans de tels circuits, il existe au moins une ligne interne telle que les chemins entre cette ligne et les sorties primaires du circuit n'ont pas la même parité d'inversion. Un tel circuit est donné à la figure 44.

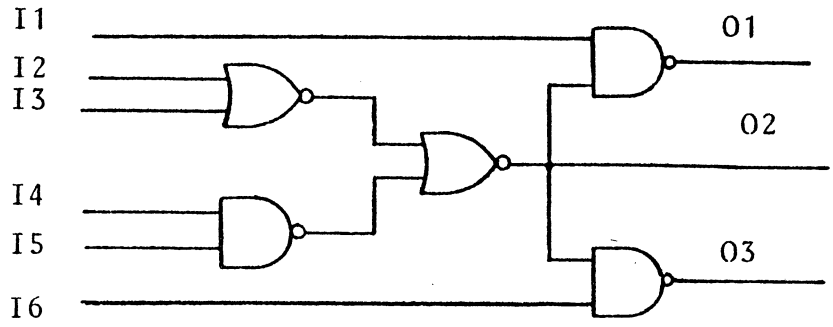


Figure 44 - Un circuit original.

Une idée similaire à l'idée utilisée dans la section IV.1.5. est reprise ici. On repère les lignes qui ne respectent pas la règle R5". Pour de telles lignes quelques parties du circuit sont dupliquées afin d'avoir deux parties dont l'une sera utilisée pour connecter une entrée quelconque Ii avec les sorties primaires du circuit, par l'intermédiaire des chemins ayant des parités d'inversion P et l'autre partie sera utilisée pour connecter l'entrée Ii avec les sorties primaires, par l'intermédiaire des chemins ayant des parités d'inversion égales à $\overline{P}$. De cette façon on aura quelques entrées primaires dupliquées ; les paires des entrées primaires dupliquées seront ensuite connectées et le circuit ainsi obtenu vérifie la règle R5". Les entrées primaires seront controlées de la façon définie dans la section IV.2.4.

La figure 45 présente le circuit obtenu en appliquant cette méthode au circuit de la figure 44.

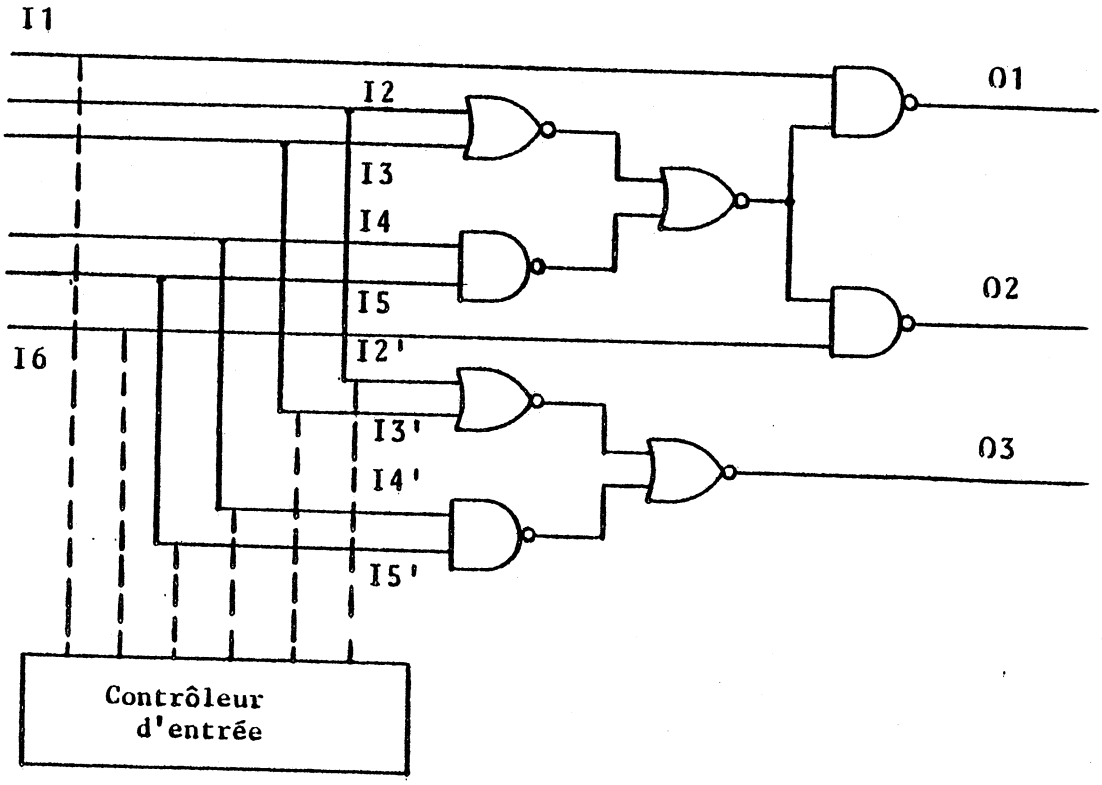


Figure 45 - Circuit modifié pour être controlé vis à vis des erreurs unidirectionnelles.

Il est évident que tous les circuits combinatoires peuvent être modifiés selon la méthode proposée dans cette section. Il est évident que cette méthode introduit une augmentation de la surface initiale.

IV.2.6. Contrôle des entrées primaire et redondance logique

Une procédure qui utilise la redondance logique de façon similaire à la méthode de la section IV.1.6. est présentée ici pour obtenir des circuits respectant la règle R5".

Procédure Pc4

Première étape

On considère les q fonctions booléennes des q sorties du circuit. Pour chaque entrée primaire Ii on vérifie si les chemins connectant l'entrée Ii avec les sorties primaires, ont la même parité d'inversion (à chaque apparition du terme Ii dans les fonctions booléennes des sorties primaires on associe une parité d'inversion; cette parité est égale à la somme modulo 2 du nombre de barres inversantes situées au-dessus du terme Ii). Si les parités d'inversion ne sont pas égales, alors on remplace par Ii' les apparitions de Ii avec un nombre impair des barres inversantes. Par exemple, par les fonctions $O1 = \overline{I1 + I2 + I2.I3}$, $O2 = \overline{I1 + I2 + I3}$, on obtient $O1 = \overline{I1 + I2 + I2'.I3}$, $O2 = \overline{I1' + I2' + I3}$.

Par l'étape 1 on obtient les fonctions redondantes des sorties primaires ; dans ces fonctions tous les chemins connectant une entrée Ii ou une entrée Ii' avec les sorties primaires ont la même parité d'inversion.

Deuxième étape

On réalise le dessin optimal qui correspond aux fonctions redondantes. La règle R5' est vérifiée pour le circuit obtenu par la deuxième étape.

Troisième étape

Dans le circuit obtenu par la deuxième étape on connecte chaque ligne Ii' avec la ligne Ii.

Le circuit obtenu par la troisième étape réalise les fonctions initiales et la règle R5" est vèrifiée.

Notons que cette procedure permet de réaliser n'apport quel circuit combinatoire de façon à ce que la régle R5" soit vérifiée. En appliquant la procédure Pc4 au circuit de la figure 44, on retrouve le circuit de la figure 45.

En général les méthodes des sections IV.2.5. et IV.2.6. ne sont pas équivalentes.

IV.2.7. Remarques

Les méthodes présentées dans ce chapître pour surmonter les difficultés de conception des circuits respectant les règles R1 et R5 consomment de la surface. Parmi les méthodes présentées (c'est à dire redondance du code d'entrée, partition, contrôle des entrées primaires, contrôle des entrées primaires et redondance, contrôle des entrées primaires et redondance logique etc...), une méthode ou une combinaison de ces méthodes doit être choisie pour obtenir le circuit optimal.

Des règles autres que les règles R1 ou R5 sont aussi nécessaires pour obtenir des circuits SFS, par exemple, les règles R2, R3, R4... etc... données aux sections III.1 et III.2. Quelques courts-circuits du type B2 ou B2' pour lesquels le circuit est redondant, peuvent être introduits par les méthodes des sections IV.1.5, IV.1.6, IV.2.5 etc... Pour des raisons données aux chapîtres III.1 et III.2, ces courts-circuits doivent être éliminés. Deux exemples de tels courts-circuits sont présentés figures 46 et 47. Ces courts- circuits ne peuvent pas survenir car les deux lignes respectives ne sont pas des lignes proches géographiquement (Hypothèses classe I).

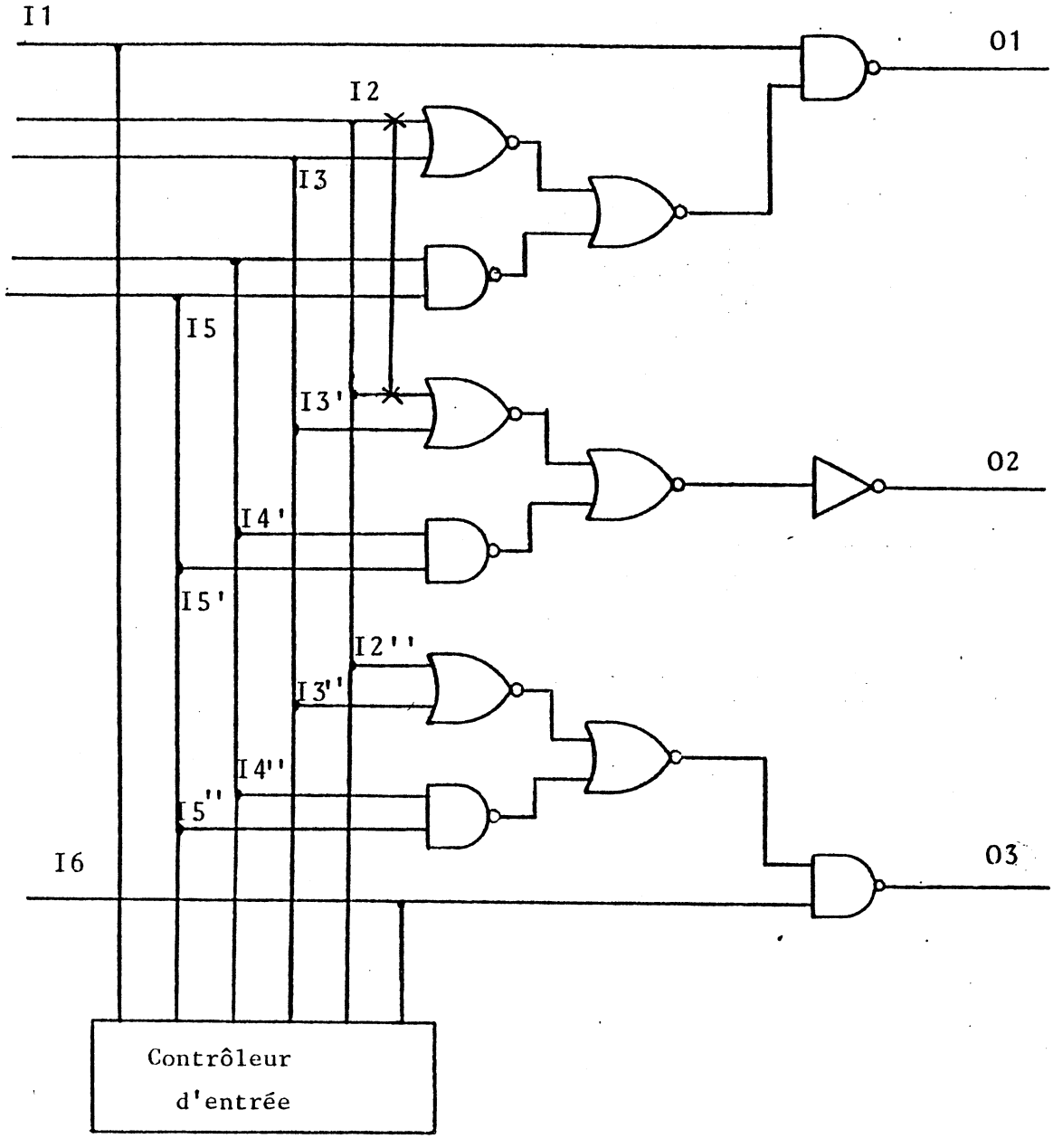


Figure 46 - Le court-circuit indiqué ne doit pas apparaître.

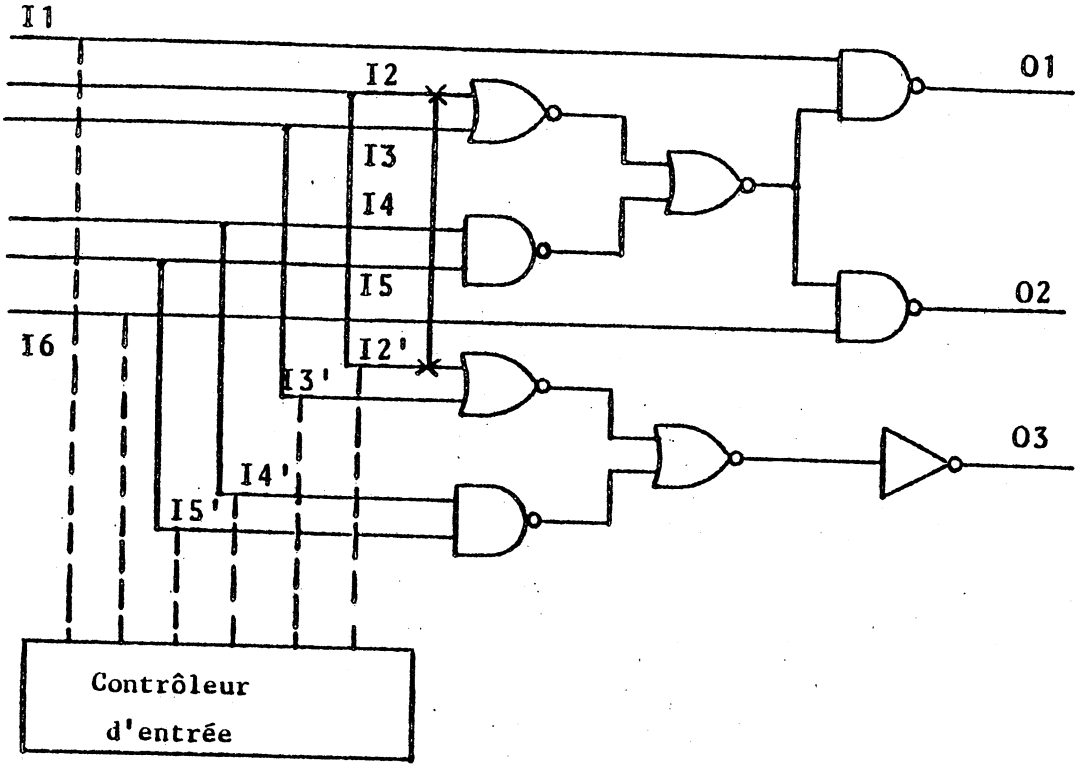


Figure 47 - Le court-circuit indiqué ne doit pas apparaître.

IV.3. Les codes non ordonnés généralisés

Dans le Lemme 5 de la section IV.2. le code d'entrée A a été codé dans un nouveau type de code. Cette classe de nouveau code est introduite ci-dessous.

Définition D22

Soit une partition des bits des vecteurs binaires de longueur n dans deux groupes p1 et p2.

Un code C de vecteurs binaires de longueur n est appelé non ordonné généralisé pour les groupes p1 et p2, s'il n'existe pas deux vecteurs v1 et v2 appartenant au code C tels que v1 couvre v2 pour tous les bits du groupe p1 et v2 couvre v1 pour tous les bits du groupe p2.

Notons que le nom "non ordonné généralisé" est utilisé car ces codes peuvent être définis en faisant une généralisation des codes non ordonnés et que deux vecteurs ordonnés peuvent appartenir à un tel code (malgré l'utilisation du terme non-ordonné). D'autre part les codes non -ordonnés peuvent être considérés comme un cas spécial de ces codes (p1 vide ou p2 vide).

Un code sera dit symétrique si:
- le code n'est pas modifié en inversant les n bits des vecteurs binaires,

Un code sera dit dissymétrique si:
- un code différent est obtenu en inversant les n bits des vecteurs binaires.

Les codes k/2k et les codes de Berger complets sont des codes symétriques. Les codes m/n avec m=2n et les codes de Berger non complets sont des codes dissymétriques.

En inversant tous les bits des vecteurs binaires d'un code C on obtient un nouveau code que l'on désignera par $\overline{C}$.
Pour un code symétrique Cs on a: $\overline{Cs} = Cs$
Pour un code dissymétrique Cd on a: $\overline{Cd} \neq Cd$
Pour les vecteurs binaires d'un code C, en inversant les bits du group p1, on obtient un code désigné par $C\overline{p_1}$.
Pour un code C quelconque on a: $\overline{C}\overline{p_1} = C\overline{p_2}$
Pour un code C quelconque on a: $\overline{C\overline{p_1}} = C\overline{p_2}$

On peut vérifier que les codes non ordonnés et les codes non ordonnés généralisés sont associés selon les propriétés suivantes.

Propriété Py2
Les codes non ordonnés sont une sous-classe des codes non ordonnés généralisés, pour lesquels le groupe p1 ou le groupe p2 est vide.

Propriété Py3
Si on inverse les bits du groupe p1 ou les bits du groupe p2 d'un code non ordonné généralisé symétrique, on obtient un code non

ordonné symétrique. Si on inverse les bits du goupe p1 d'un code non ordonné généralisé dissymétrique on obtient un code non ordonné dissymétrique et si on inverse les bits du groupe p2 on obtient un autre code non ordonné dissymétrique.

Propriété Py4

Si les groupes p1 et p2 sont déterminés:

- en inversant les bits du groupe p1 ou du groupe p2, dans un code symétrique non ordonné on obtient un code non ordonné généralisé symétrique.
- en inversant les bits du groupe p1 dans un code non ordonné dissymétrique on obtient un code non ordonné généralisé dissymétrique et en inversant les bits du groupe p2 on obtient un autre code non ordonné généralisé dissymétrique.

La démonstration de la prmière partie de la propriété Py4 est simple:

$$\left.\begin{array}{l} \overline{Cs\ \overline{p1}} = Cs\ \overline{p2} \\ \overline{Cs} = \overline{Cs} \end{array}\right\} \Rightarrow Cs\ \overline{p1} = Cs\ \overline{p2}$$

(en inversant les bits du groupe p1 ou du groupe p2 on obtient le même code)

$\overline{Cs\ \overline{p1}} = Cs\ \overline{p2} = Cs\ \overline{p1}$ (le code obtenu est donc symétrique)

La démonstration de la deuxième partie de la propriété Py4 est similaire. La démonstration de la propriété Py3 est aussi similaire.

Les propriétés Py5 et Py6 sont évidentes.

Propriété Py5

Si le nombre de bits du groupe p1 est égal à k un code symétrique non ordonné donne

$$N = C_n^k = \frac{n!}{k!(n-k)!}$$

codes non ordonnés généralisés symétriques, chacun de ces codes est

obtenu en inversant un groupe de k bits dans les vecteurs binaires de longueur n ou en inversant le groupe des n-k bits restant.

Un code dissymétrique non ordonné donne

$$N = 2C_n^k = 2\,\frac{n!}{k!(n-k)!}$$

codes non ordonnés généralisés dissymétriques. Chaque partition en groupes p1 et p2 donne un code en inversant les bits du groupe p1 et un autre code en inversant les bits du groupe p2 (propriété Py4).

Propriété Py6

Un code non ordonné symétrique des vecteurs binaires de longueur n donne au total

$$N = \frac{\sum_{m=0}^{n} C_n^m}{2} = 2^{n-1}$$

codes non ordonnés généralisés symétriques ; chacun de ces codes est obtenu en inversant un groupe de m bits ou un groupe de n-m bits. Un code non ordonné dissymétrique des vecteurs binaires de longueur n donne au total

$$N = \sum_{m=0}^{n} C_n^m = 2^n$$

codes non ordonnés généralisés dissymétriques, chacun de ces codes est obtenu en inversant un groupe de m bits.

Le code k/2k donne par exemple $N = 2^{2k-1}$. Un code de Berger complet donne $N = 2^{n-1}$; un code m/n avec $n \neq 2m$ donne $N = 2^n$; un code de Berger non complet donne $N = 2^n$ etc...

De façon similaire aux erreurs unidirectionnelles détectées par les codes non ordonnés, on peut définir les erreurs unidirectionnelles généralisées détectées par les codes non ordonnés généralisés.

Définition D23

Soit une partition des bits des vecteurs binaires de longueur n, en deux groupes p1 et p2.

Une erreur dans un vecteur binaire de longueur n est appelée unidirectionnelle généralisée pour les groupes p1 et p2, si chaque modification (due à cette erreur) sur les bits du groupe P1 est du type 0 -> 1 (resp. 1 ->0) et chaque modification sur les bits du groupe p2 est du type 1 -> 0 (resp. 0 -> 1).

D'une façon similaire à celle donnée précédemment (propriétés données pour les codes non ordonnés généralisés), les erreurs unidirectionnelles et les erreurs unidirectionnelles généralisées sont associées selon les propriétés suivantes.

Propriété Py2'

Les erreurs unidirectionnelles sont une sous-classe des erreurs unidirectionnelles généralisées pour laquelle le groupe p1 est vide ou le groupe p2 est vide.

Propriété Py3'

Les erreurs unidirectionnelles généralisées pour les groupes p1 et p2, donnent des erreurs unidirectionnelles, si on inverse les bits du groupe p1 ou les bits du groupe p2.

Propriété Py4'

Si les groupes p1 et p2 sont déterminés , les erreurs unidirectionnelles donnent des erreurs unidirectionnelles généralisées pour les groupes p1 et p2, si on inverse les bits du groupe p1 ou les bits du groupe p2.

Propriété Py5'

Si le nombre des bits du groupe p1 est égal à k , les erreurs unidirectionnelles affectant des vecteurs binaires de longueur n, donnent N types d'erreurs unidirectionnelles généralisées, avec N égal à

$$C_n^m = \frac{n!}{k!(n-k)!}$$

Chacune de ces erreurs est obtenue en inversant un groupe de k bits ou le groupe de (n-k) bits restant.

Propriété Py6'

Les erreurs unidirectionnelles affectant des vecteurs binaires de longueur n donnent au total

$$N = \frac{\sum_{m=0}^{n} C_n^m}{2} = 2^{n-1}$$

types d'erreurs unidirectionnelles généralisées, chacun de ces types est obtenu en inversant un groupe de k bits ou le groupe de (n-k) bits restant.

Il est évident que les codes des vecteurs binaires de longueur n qui sont non ordonnés généralisés pour deux groupes p1 et p2, détectent toutes les erreurs qui sont unidirectionnelles généralisées pour les mêmes groupes. Par conséquent les codes non ordonnés généralisés peuvent être utilisés pour le test des circuits qui produisent des erreurs unidirectionnelles généralisées aux sorties primaires.

IVV.3.1. Conception des circuits SFS en utilisant les codes non ordonnés généralisés

Respectivement à la règle R5 donnée pour les circuits contrôlés vis à vis des erreurs unidirectinnelles, la régle GR5 doit être utilisée pour les circuits contrôlés vis à vis des erreurs unidirectionnelles généralisés.

Règle GR5

Il existe un vecteur binaire V=(V1,V2,...,Vk) avec k égal au nobre des sorties du circuit tel que pour chaque ligne Ij du circuit, les parités d'inversion de tous les chemins reliant la ligne Ij à n'importe quelle sortie Oi sont égales à Vi ou bien les parités d'inversion de tous les chemins reliant la ligne Ij à n'importe quelle sortie Oi sont égales à $\overline{Vi}$.

On peut vérifier que les structures qui correspondent à la règle R5 (V=(0,0,...0) ou V=(1,1,...1)) représentent 2 parmi les 2^n structures représentées par la règle GR5 (2^n valeurs possibles du vecteur V). La règle GR5 représente une classe des circuits beaucoup plus large que la classe définie par la règle R5.

Si la règle GR5 est vérifiée, alors les erreurs simples internes peuvent être propagées aux sorties primaires du circuit comme erreurs unidirectionnelles généralisées pour deux groupes p1 et p2; le groupe p1 est le groupe des sorties Oi pour lesquelles Vi=0 et le groupe p2 est le groupe des sorties Oj pour lesquelles Vj=1.

Un circuit qui vérifie la règle GR5 est donné à la figure 48. Pour un tel circuit la règle R6' de la section III.2.1. doit être remplacée par la règle GR6'.

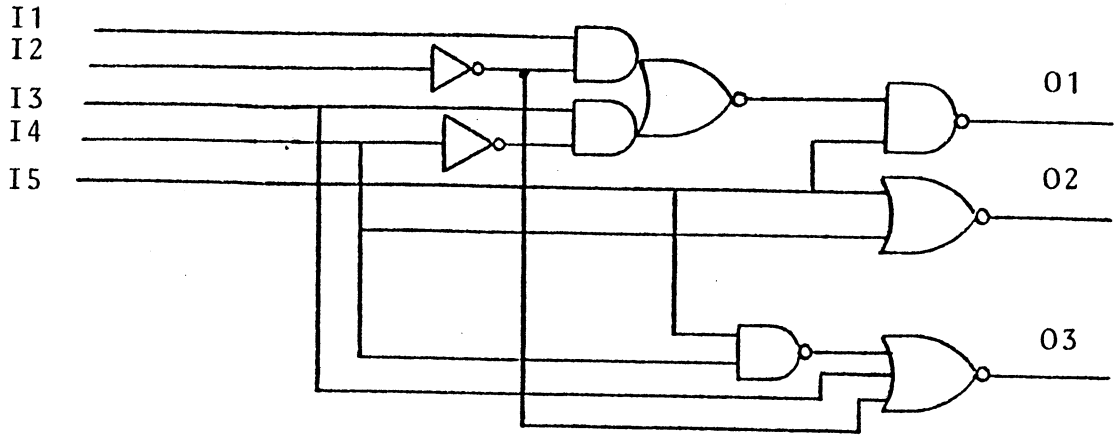


Figure 48 - Un circuit original.

Règle GR6'

Chaque ligne d'alimentation n'est utilisée que pour alimenter des portes dont les sorties sont connectées avec les sorties primaires par l'intermédiaire des chemins qui ont des parités d'inversion égales à Vi, ou pour alimenter des portes pour lesquelles les parités d'inversion respectives sont égales à $\overline{Vi}$.

De façon similaire on peut introduire des défauts de types GB1', GB2', GD1' et GD2' à la place des défauts de types B1', B2', D1' et D2' et les règles GR7 et GR8 à la place des règles R7 et R8 et on obtient ainsi l'ensemble complet des règles de conception des circuits SFS controlés vis à vis des erreurs unidirectionnelles généralisés.

La règle GR5 est équivalente à la règle GR5'.

Règle GR5'

Il existe un vecteur binaire V=(V1,V2,...,Vk), avec k égal au nombre des sorties du circuit tel que pour chaque entrée primaire

Ij du circuit, les parités d'inversion de tous les chemins reliant l'entrée Ij à n'importe quelle sortie Oi sont égales à Vi, ou bien les parités d'inversion de tous les chemins reliant la ligne Ij à n'importe quelle sortie Oi sont égales à $\overline{Vi}$.

La preuve d'équivalence des règles GR5 et GR5' est similaire à la preuve d'équivalence des règles R5 et R5'.
Enfin, une proposition similaire à la proposition P23 de la section IV.2. est donnée.

Proposition GP23

Pour pouvoir réaliser une fonction G par un circuit qui vérifie la règle GR5' et dont le code de sortie est un code non ordonné généralisé pour deux groupes p1 et p2, la condition nécessaire et suffisante est la suivante : les entrées primaires doivent pouvoir être partagées en deux groupes g1 et g2 telsque,pour chaque couple de vecteurs a et a' appartenant au code d'entrée, pour lequels a couvre a' pour toutes les entrées du groupe g1 et a' couvre a pour toutes les entrées du groupe g2 (ou l'inverse),on a G(a, $\emptyset$) = G(a', $\emptyset$).

En effet, il s'agit de la même condition donnée à la proposition P23 qui assure la vérification de la règle R5'. La demonstration de la proposition GP23 peut se faire de façon similaire à la demonstration de la proposition P23.

Si la condition suffisante et nécessaire n'est pas vérifiée, on peut utiliser des méthodes similaires aux méthodes uttilisées dans le cas de la règle R5'. On peut par exemple modifier les blocs qui génèrent les entrées du circuit afin d'obtenir un code d'entrée non ordonné généralisé car il est évident que pour un tel code d'entrée, la condition nécessaire et suf- est vérifiée.

On peut aussi utiliser le conrôle des entrées primaires, dans ce cas une règle GR5" correspondant à la règle R5" doit être utilisée.

Le dernier point à discuter est le problème des contrôleurs des codes non ordonnés généraligés. D'après les propriétés Py3 et Py3', on peut vérifier que si on utilise des inerseurs après les sorties du groupe p1 ou après les sorties du groupe p2, alors le code non ordonné généralisé est transformé en code non ordonné et les erreurs unidirectionnelles généralisées sont transforeées en erreurs unidirectionnelles. Par conséquent, le contrôle des codes non ordonnés généralisés peut se faire en utilisant quelques inverseurs et les contrôleurs des codes non ordonnés.

Dans la figure 49 on donne l'application de cette méthode pour le circuit de la figure 48. Dans la figure 49 on peut vérifier que le circuit initial (figure 48) vérifie la règle GR5, tandis que le bloc composé par le circuit initial et les inverseurs vérifie la règle R5. Cette transformation est plus générale et on peut vérifier la proposition suivante.

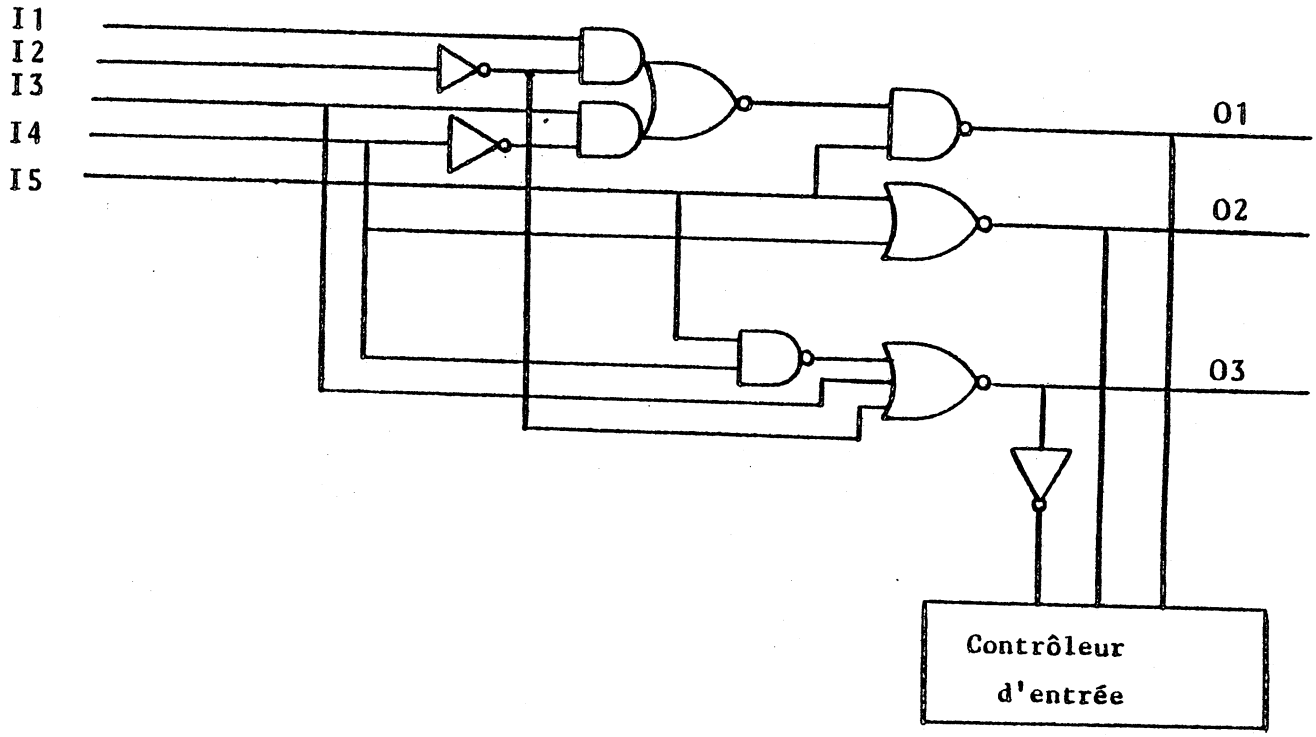


Figure 49 - Circuit modifié pour être contrôlé vis à vis des erreurs unidirectionnelles généralisées.

Proposition P24

Pour un circuit qui vérifie les règles de conception des circuits SFS contrôlés vis à vis des erreurs unidirectionnelles généralisées, l'utilisation des inverseurs après les sorties du groupe p1 ou après les sorties du groupe p2, donne les transformations suivantes: code de sortie non ordonné généralisé <-> code de sortie non ordonné. Erreurs de sortie unidirectionnelles généralisées <-> Erreurs de sortie unidirectionnelles. Règle R5 (ou R5') <-> Règle GR5 (ou GR5'). Règle R5" <-> Règle GR5". Règle R6' <-> Règle GR6'. Règle R7 <-> Règle GR7. Règle R8 <-> Règle GR8 etc...

Par la suite, nous donnons des exemples d'utilisation des codes non ordonnés généralisés.

La figure 50 présente un circuit composé d'une matrice NOR. De tels

circuits sont utilisés trés souvent comme décodeurs.

Dans la section IV.2. on a vu que les PLA peuvent être contrôlés par un code non ordonné car la règle R5" est vérifiée par un bloc de deux matrices NOR.

La génération des bits de contrôle, nécessaires pour coder les sorties du PLA, ne modifie pas la structure NOR-NOR et le PLA peut être testé en utilisant un code de sortie non ordonné. La génération des bits de contrôle pour le circuit de la figure 50 donne un circuit qui ne vérifie pas la règle R5", mais il vèrifie la règle GR5".

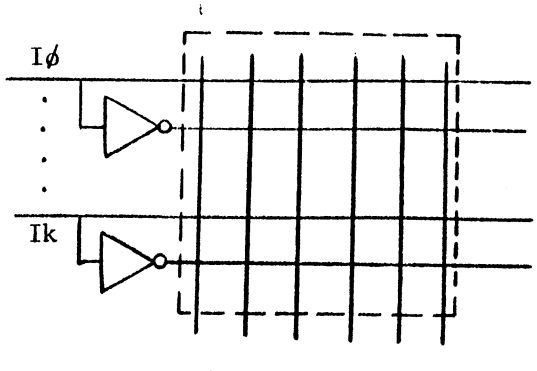


Figure 50

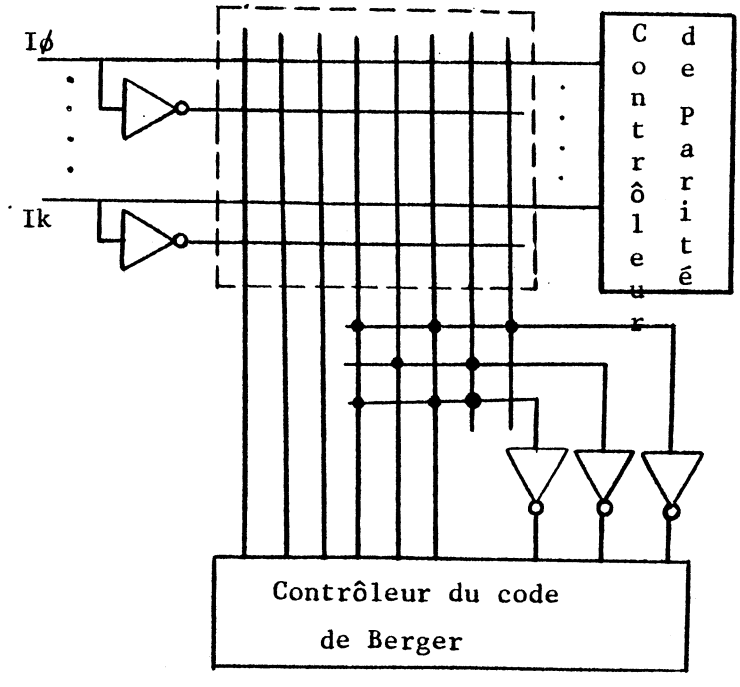


Figure 51

La figure 51 présente le circuit augmenté qui est testé en utilisant le contrôle des entrées primaires. Les sorties supplémentaires donnent les bits de contrôle pour un code de Berger généralisé (le groupe p1 est le groupe des sorties du circuit initial et le groupe p2 est le groupe des bits de contrôle).

Notons que pour un code de Berger non complet ($n-k \neq 2^m-1$) comme le code de la figure 51, il existe deux codes de Berger généralisés pour les groupes p1 et p2 (proprieté Py4). On peut alors choisir le code le plus économique (minimisation des monômes supplémentaires).

La même méthode peut être utilisée pour des circuits composés par deux matrices NOR, mais pour lesquels certaines sorties sont générées directement par la première matrice NOR. Un tel circuit est donné dans la figure 52 ; il est évident que la règle R5" n'est pas vérifiée mais la règle GR5" est vérifiée.

Dans le circuit augmenté, qui n'est pas présenté ici, le code de sortie sera un code de Berger généralisé pour le groupe p1 des sorties 01, 06, 07, et pour le groupe p2 des sorties 02, 03, 04, 05 et des bits de contrôle c1, c2 et c3.

Trois inverseurs seront utilisés après les sorties 01, 06 et 07, et finalement le circuit sera testé par un code de Berger.

Notons que pour le circuit de la figure 52 il existe un seul code de BERGER généralisé ($7 = 2^3-1$)

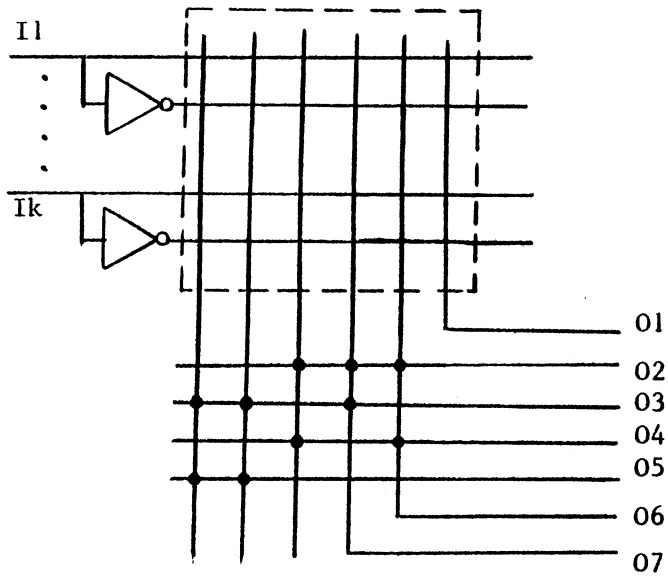


Figure 52.

IV.4. Interconnexion des blocs SFS

Supposons un circuit composé de deux blocs SFS comme celui présenté à la figure 53.

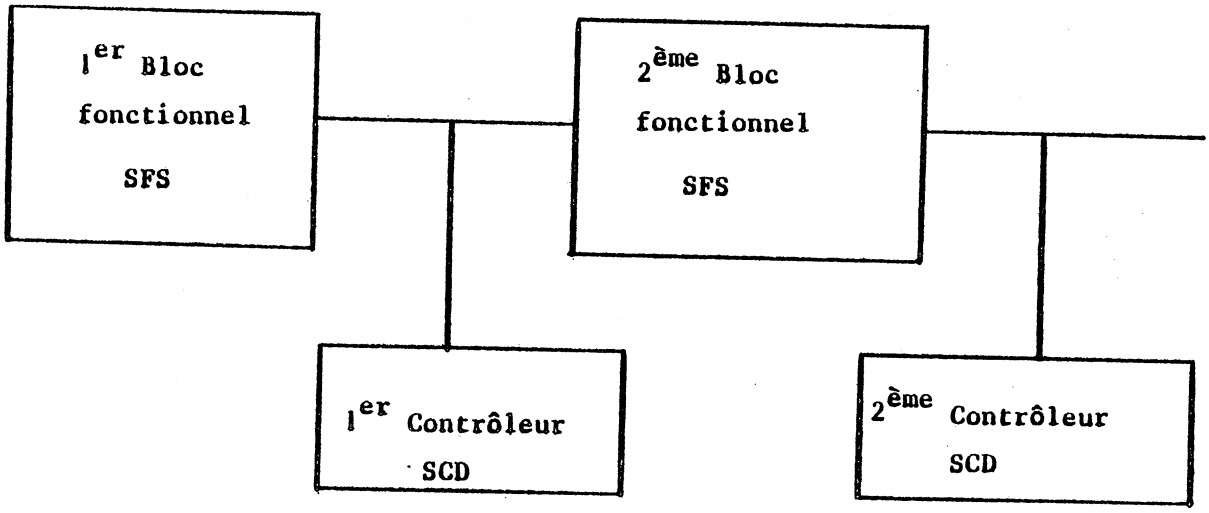


Figure 53

Pour que le bloc global de la figure 53 soit SFS, la propriété SFS doit être aussi assurée pour les défauts des interconnexions. Le premier contrôleur SCD peut être utilisé pour assurer le test des interconnexions. Selon la structure du premier et du deuxième bloc fonctionnel SFS, le premier contrôleur SCD doit être utilisé pour contrôler les lignes des interconnexions jusqu'à certains points critiques. Dans d'autres cas le contrôle des interconnexions par le premier contrôleur SCD n'est pas nécessaire et ce contrôleur peut être connecté aux sorties du premier bloc fonctionnel de la façon arbitraire. Dans certaines cas le premier contrôleur peut être retiré et le bkoc global sera alors contrôlé uniquement par le deuxième contrôleur SCD. Ces cas sont détaillés dans la suite.

IV.4.1. Contrôle des interconnexions jusqu'aux points critiques.

Le deuxième bloc fonctionnel est contrôlé en utilisant la méthode de contrôle des entrées primaires. C'est le cas de vérification des règles R1", R5" ou GR5". Dans ce cas le premier contrôleur doit

être utilisé pour contrôler les lignes des interconnectios jusqu' aux points critiques définis das les sections IV.1.4,IV.1.5,IV.2.4 et IV.2.5.

Notons que dans certaines cas (les lignes de transfert sont très longues et le code de sortie du premier bloc nécessite plusieurs bits de codage)il préferable de générer le bit de parité des lignes de transfert pour assurer le contrôle des interconnexions. Dans lecas du code de Berger (généralise ou non) ce bit est disponible car pour ce code on peut vérifier que le bit de contrôle poids faibles est aussi le bit de parité des bits d'information.

IV.4.2. Le contrôle des interconnexions n'est pas nécessaire.

Le deuxième bloc fonctionnel est testé sans utiliser le contrôle des entrées primaires.C'est le cas de vérification des règles R1 (R1'), R5 (R5'), GR5 (GR5'). Dans ce cas il n'y a pas de restrictions pour placer le premier contrôleur.

IV.4.3. Le premier contrôleur peut être retiré.

Selon la structure du premier et du deuxème bloc fonctionnel,le premier contrôleur peut être retiré dans les cas suivants.

a/ Le premier bloc fonctionnel est testé par un code détectant les erreurs simples et il ne produit que des erreurs simples à ses sorties primaires. Le deuxième bloc fonctionnel vérifie la règle R1 (R1') ou la règle R5(R5') ou la règle GR5(GR5') ; pour un tel bloc on peut vérifier la propriété suivante concernant la propagation des erreurs :

- chaque erreur simple aux entrées primaires d'un bloc qui vérifie la règle R1(R1') ou R5(R5') ou GR5(GR5') est propagée jusqu'aux sorties primaires du circuit respectivement soit comme une erreur

simple, soit comme une erreur unidirectionnelle soit comme une erreur unidirectionnelle généralisée.

Par conséquent, le premier contrôleur peut être retiré et le système global sera testé uniquement par le deuxième contrôleur.

b/ Les erreurs produits aux sorties primaires du première bloc fonctionnel sont des erreurs unidirectionnelles généralisés pour deux groupes des sorties g1 et g2 ; le premier bloc fonctionnel est testé par un code de sortie qui est non ordonné généralisé pour les groupes g1 et g2 (le groupe g1 ou g2 peut être vide et le code de sortie est non ordonné).

Le code de sortie du premier bloc fonctionnel peut être utilisé comme code d'entrée du deuxième bloc fonctionnel pour concevoir, selon la prosédure Pc2 (section IV.2.2), un circuit qui vérifie la règle R5' et dont le code de sortie détecte les erreurs unidirectionnelles. Pour un tel bloc on peut vérifier la propriété suivante concernant la propagation des erreurs :

- les erreurs unidirectionnelles généralisées pour les groupes g1 et g2, produites aux sorties primaires du premier bloc fonctionnel seront propagées comme erreurs unidirectionnelles aux sorties primaires du deuxième groupe.

Preuve: Par la procédure Pc2 on peut vérifier que pour le deuxième bloc fonctionnel, tous les chemins entre les entrées primaires du groupe g1 et les sorties primaires ont de parité d'inversion égale à 1 (resp. 0) et tous les chemins entre les entrées primaires du groupe g2 et les sorties primaires ont de parité d'inversion égale à 0 (resp. 1). Il est ensuite facile de vérifier que la propagation des erreurs s'effectue comme définie ci-dessus.

Par conséquent, le premier contrôleur SCD peut être éliminé et le code global sera testé uniquement par la deuxième contrôleur.

Au lieu de la procédure Pc2 on peut aussi utiliser une procédure similaire GPc2 de façon à ce que le deuxième bloc fonctionnel

vérifie la règle GR5' et que son code de sortie soit un code non ordonné généralisé pour deux groupes des sorties p1 et p2 (le choix des groupes p1 et p2 peut se faire selon des critères d'optimisation). Ensuite on peut vérifier que les erreurs unidirectionnelles généralisées pour les groupes g1 et g2, produites aux sorties du premier bloc fonctionnel, seront propagées aux sorties du deuxième bloc fonctionnel comme des erreurs unidirectionnelles généralisées pour les groupes p1 et p2. Par conséquent le premier contrôleur peut être retiré et le bloc global sera testé uniquement par le deuxième contrôleur.

La conclusion est la suivant:
Une fois que le code de sortie d'un bloc est un code non ordonné (généralisé ou non) le deuxième, troisième etc... bloc, peuvent être conçus pour être SFS en utilisant un code non ordonné (généralisé ou non) et ainsi le circuit global sera testé uniquement par le contrôleur du dernier bloc.

c/ Le premier et le deuxième blocs fonctionnels sont contrôlés par un code de duplication ou par un code "double-rail". Les entrées primaires du deuxième bloc fonctionnel sont partagées en deux groupes associés aux deux groupes des sorties primaires du premier bloc. Chacun des deux groupes des entrées primaires du deuxième bloc est connecté par l'intermédiaire de chemins uniquement avec l'un des deux groupes des sorties primaires du deuxième bloc. Pour un tel bloc, on peut vérifier la propriété suivante concernant la

propagation des erreurs

- Les erreurs multiples produites sur un des deux groupes des sorties du premier bloc seront propagées comme erreurs multiples affectant uniquement l'un des deux groupes des sorties du deuxième bloc. Par conséquent le premier contrôleur peut être retiré et le bloc global sera testé par le deuxième contrôleur.

Notons que dans les cas a/, b/ c/, si le premier contrôleur est retiré, de nouveaux défauts peuvent apparaître pour lesquels le premier bloc fonctonnel est redondant. En fait le premier bloc

peut ne pas être redondant pour un défaut si on considere ses propres sorties comme des points d'observation, mais il peut être redondant pour le même défaut si on considere comme points d'observation les sorties du deuxième bloc. Il pourait donc être nécessaire d' éliminer certaines redondances indesirables qui n'existaient pas avant retirer le premier contrôleur.

IV.5. Conclusion : Conception des circuits SFS pour de differents classes d'hypotheses de pannes

L'analyse présentée dans cette étude est faite en considerant la classe I d'hypotheses de pannes pour la technologie N-MOS. Il est intéressant de vérifier dans quelle mesure les r7gles de conception peuvent être ou non utilisées pour concevoir des circuits SFS si on considere d'autres classes d'hypotheses de pannes. Une telle extension est façile à faire grace à la généralité de la méthodologie utilisée pour l'extraction des règles.

Dans la section II.2 on a déterminé les conséquences des défauts de classe I. Ces défauts comprennent:
- défauts simples du type MOS colé fermet/colé ouvert,contact ou précontact défaillant,coupure de lignes d'aluminium,des lignes de diffusion, des lignes de polysilicon.
- courts-circuits entre deux lignes d'aluminium ou entre des lignes de diffusion.

Les défauts présentés ci-dessus peuvent affecter tous les types de lignes, ou bien il peuvent affecter par exemple toutes les lignes d'un circuit sauf les lignes d'alimentation.

Ilest évident que la règle Rl de la section III.1.1, la règle R5 de la section III.2.1 et la règle R9 de la section III.3.1. sont nécessaires dés que la classe introduit des erreurs simples dans les lignes internes des circuits.

La règle R2 de la section III.1.1. , la règle R6 de la section III.2.1. et la règle R10 de la section III.3.1. sont nécessaires dans la mesure où la classe considérée comprend des défauts simples affectant les lignes d'alimentation.

Les autres règles déterminées aux sections III.1.2. , III.2.2. et III.3.2. sont nécessaires dans la mesure où la classe considérée comprend des courts- circuits entre des lignes spéciales.

La table I indique les règles nécessaires pour 4 classes d'hypothèses de pannes. Ces classes sont:
- défauts simples, sauf les défauts affectant les lignes d'alimentation et les courts-circuits,
- défauts simples (les coupures des lignes d'alimentation comprises), sauf les courts-circuits: classe 0 [COU 81],
- défauts simples, y compris les coupures des lignes d'alimentation et les courts-circuits entre certaines lignes d'Alu et entre certaines lignes de diffusion ; les courts-circuits entre les lignes d'alimentation étant toutefois exclus,
- défauts simples y compris les coupures des lignes d'alimentation et les courts-circuits entre certaines lignes d'Alu et certaines lignes d'alimentation classe I [COU 81].

Dans la table I on considère les propriétés "fault secure" et "strongly fault secure".

	propriété	erreurs détectées par le code de sortie		
		simples	unidirectionnelles	multiples
défauts simples sauf lignes d'alimentation et sauf courts-circuits	FS	R1	R5	R9
	SFS	R1	R5	R9
défauts simples sauf courts-circuits (classe Ø)	FS	R1, R2	R5, R6	R9, R10
	SFS	R1, R2	R5, R6	R9, R10
défauts simples et courts-circuits sauf courts-circuits entre lignes d'alimentation	FS	R1, R2	R5, R6	R9, R10
	SFS	R1, R2, R3	R5, R6, R7	R9, R10, R11
défauts simples et courts-circuits (classe I)	FS	R1, R2	R5, R6	R9, R10
	SFS	R1, R2, R3, R4	R5, R6, R7, R8	R9, R10, R11 R12

Table I - Règles pour les différentes classes d'hypothèses de pannes au niveau physique.

Remarque:
Notons que si on considère des hypothèses de pannes au niveau logique (ce modèle n'est pas représentatif des défaillances réelles des circuits intégrés sauf si certaines règles de dessin sont utilisées), alors le modèle de collage logique peut être considéré comme une classe d'hypothèses de pannes. Un autre type de pannes dans ce niveau pourrait être le court-circuit entre deux lignes internes.

La table II indique les règles nécessaires pour deux classes d'hypothèses de pannes au niveau logique.

	propriété	erreurs détectées par le code de sortie		
		simples	unidirectionnelles	multiples
collage	FS	R1	R5	R9
	SFS	R1	R5	R9
collage et courts circuits	FS	R1	R5	R9
	SFS	R1,R3	R5,R7	R9,R11

Table II - Règles de conception pour des classes d'hypothèses de pannes du niveau logique.

On peut noter que ces règles ne sont pas suffisantes pour assurer la propriété TSC au lieu de la propr)été SFS.

Finalement, si on considère des hypothèses de pannes plus larges que la classe I, alors on doit déterminer quelques règles supplémentaires. Par exemple, si on considère des courts-circuits entre deux matériaux qui sont considérés non court-circuitables pour la classe I, e.g. courts-circuits entre deux lignes de

polisilicon, alors on sera obligé d'éliminer la possibilité de courts-circuits de types B2 ou D2, B2' ou D2', B2" ou D2" entre deux lignes de Si-poly.

Les règles nécessaires pour la conception des circuits SFS dont le code de sortie détecte des erreurs unidirectionnelles généralisées peuvent être obtenues par les tables précédentes en remplaçant les règles R5, R6, R7 et R8 par les règles GR5, GR6, GR7 et GR8.

Un dernier point à mentionner est que l'on n'a pas encore examiné si la conception de circuits SFS pour des classes de plus en plus larges, nécessite une augmentation en surface de plus en plus élevée associée à un nombre de règles de plus en plus élevé.

Pour le dévelopements futurs il est interessant de déterminer des hypothèses de pannes au niveau du MOS pour d'autres technologies, comme la technologie C-MOS (à une ou plusieurs couches d'interconnexions) et de transposer l'étude faite pour la technologie N-MOS dans d'autres technologies.

Un autre point pour les dévelopement futurs concerne les systemes séquentiels. La définition des circuits "Sequentially self-testing", "Sequentially fault secure" et "Sequentially self-checking" est donnée dans [VIA 80], la conception des circuits séquenciels autotestables est un point important qui reste à étudier.

V- APPLICATIONS

Les règles de conception des circuits SFS présentées dans cette étude seront validées dans la suite par des applications sur la conception SFS de certains circuits.

La technique de détection des erreurs simples est utilisée pour la conception SFS d'un circuit décodeur. Les trois techniques (c'est à dire la technique de détection des erreurs simples, unidirectionnelles et multiples) seront utilisées pour la conception SFS des PLAs, pour montrer que la validation des schémas déja connus peut être facilement vérifiée et pour donner de nouveaux schémas de PLAs SFS.

V.1. Un circuit décodeur

Un décodeur utilisé dans la partie opérative du microprocesseur MC 68000 est présenté figure 54. Il code la valeur binaire appliquée sur ses 4 entrées en un code 1 parmi 16.

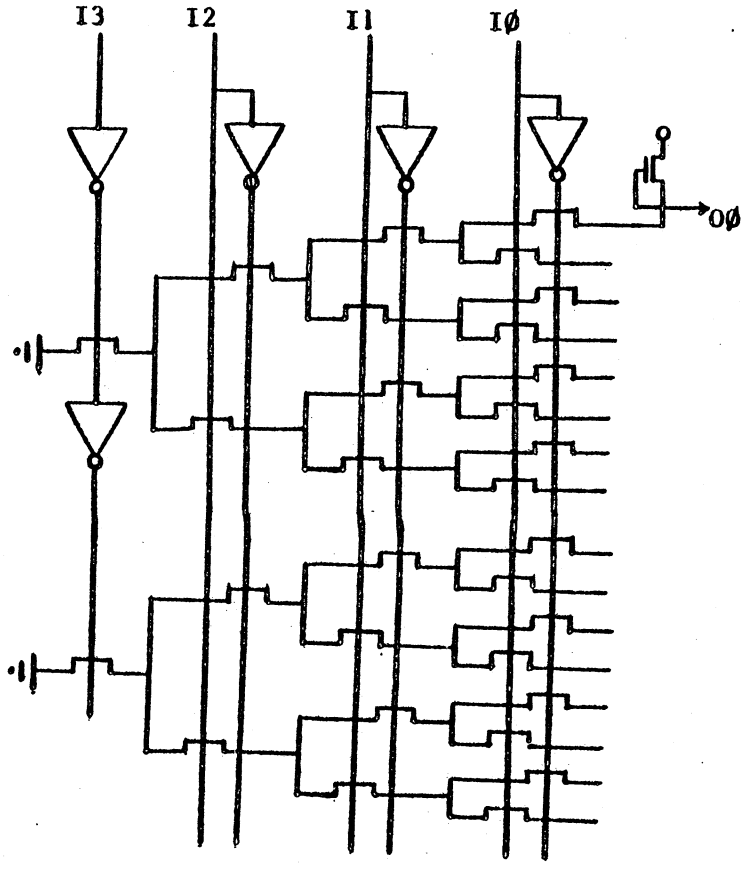


Figure 54-

V.1.1. Analyse et construction

Ce circuit est étudié ici car il présente un intérêt spécial étant donné que certaines notions classiques du chemin et du degré de divergence ne sont pas applicables dans ce circuit.

Premièrement on peut vérifier que le circuit n'est pas composé par des portes interconnectées avec des lignes mais il est composé par des réseaux MOS signal partagés entre plusieurs portes. Par conséquent la notion de chemin classique constitué par des lignes d'interconnexion des portes n'est pas applicable dans ce cas. L'utilisation du chemin au niveau du MOS, donnée dans la section

II.3, est nécessaire pour analyser le circuit.

Ensuite, on peut vérifier que ni la règle R1'(R1) ni la règle R1" ne sont vérifiées ; on doit donc utiliser une des métodes de la section IV.1 pour introduire la règle R1". Ceci nécessitera une modification totale du circuit et une augmentation importante de sa surface.

Si l'utilisation du degré de divergence pose ces problèmes, par contre l'utilisation du degré de divvergence effective (section III.1) peut amener à des solutions très simplifiées car on peut vérifier qu'un seul chemin entre une ligne (les entrées primaires exceptées) et les sorties primaires est sensibilisé à chaque fois.

Le degré de divergence effective maximal du décodeur pour l'ensemble des lignes, les entrées primaires étant exceptées, est donc égal à 1. Par conséquent, le circuit peut être conrôlé, en utilisant le contrôle des entrées primaires (section IV.1.4) et en utilisant un code de sortie détectant les erreurs simples.

V.1.2. Le circuit augmenté

Le circuit augmenté est présenté figure 55.

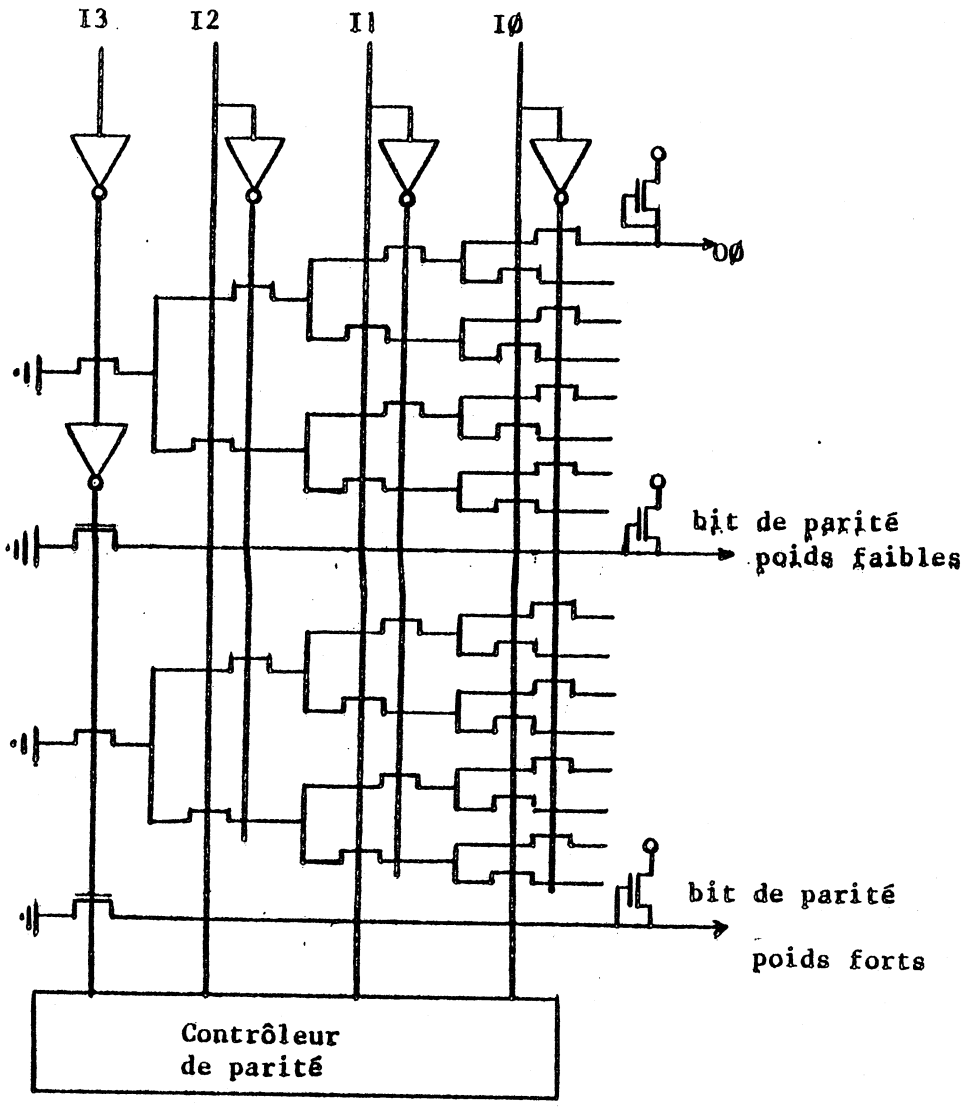


Figure 55

Les entrées primaires et les inverseurs d'entrée sont contrôlés par un contrôleur de parité ; les sorties primaires sont aussi controlées par un contrôleur de parité.

En plus des règles R1' et R1", d'autres règles doivent être vérifiées. Etant donné que deux contrôleurs sont utilisés, les règles doivent être vérifiées par chacun des deux blocs contrôlés.

La règle R2 est assurée par le bloc contrôlé par le contrôleur des entrées, en effet chaque ligne VDD ou VSS est utilisée pour aimenter un sel inverseur d'entrée (structue des lignes d'alimentation de la PO du MC 68000). La règle R2 est aussi assurée par le bloc testé par le contrôleur des sorties primaires car:
- deux lignes VSS sont utilisées, une coupure d'une ligne VSS peut

produire une erreur simple sur une ligne du réseau de MOS dont le degré de divergence effective est égal à 1 ;
- chaque ligne VDD est utilisée pour alimenter une seule porte de sortie (structure des lignes d'alimentation de la PO du MC68000).

La règle R4 est assurée pour chacun des deux blocs car les lignes VSS et les lignes VDD sont implantées alternativement (structure des lignes d'alimentation de la PO du MC 68000).

Finalement le problème des défauts du type B2 pour lesquels le circuit est redondant, est résolu car on peut vérifier que le circuit est "self testing" si tous les vecteurs d'entrée sont appliqués au circuit pendant son fonctionnement normal.

V.2. PLA

Dans l'étude suivante on considère la structure NOR-NOR des PLAs pour la technologie N-MOS.

La figure 57 présente l'implémentation de la première matrice et la figure 58 présente l'implémentation de la deuxième matrice.

Le PLA original reçoit les entrées I0, I1... Ik et il produit les sorties B1, B2... Bn (figure 56) contrairement à ce qui est indiqué dans [TSA 81] et [KHA 82] les PLAs ne seront pas nécessairement "non concurrent".

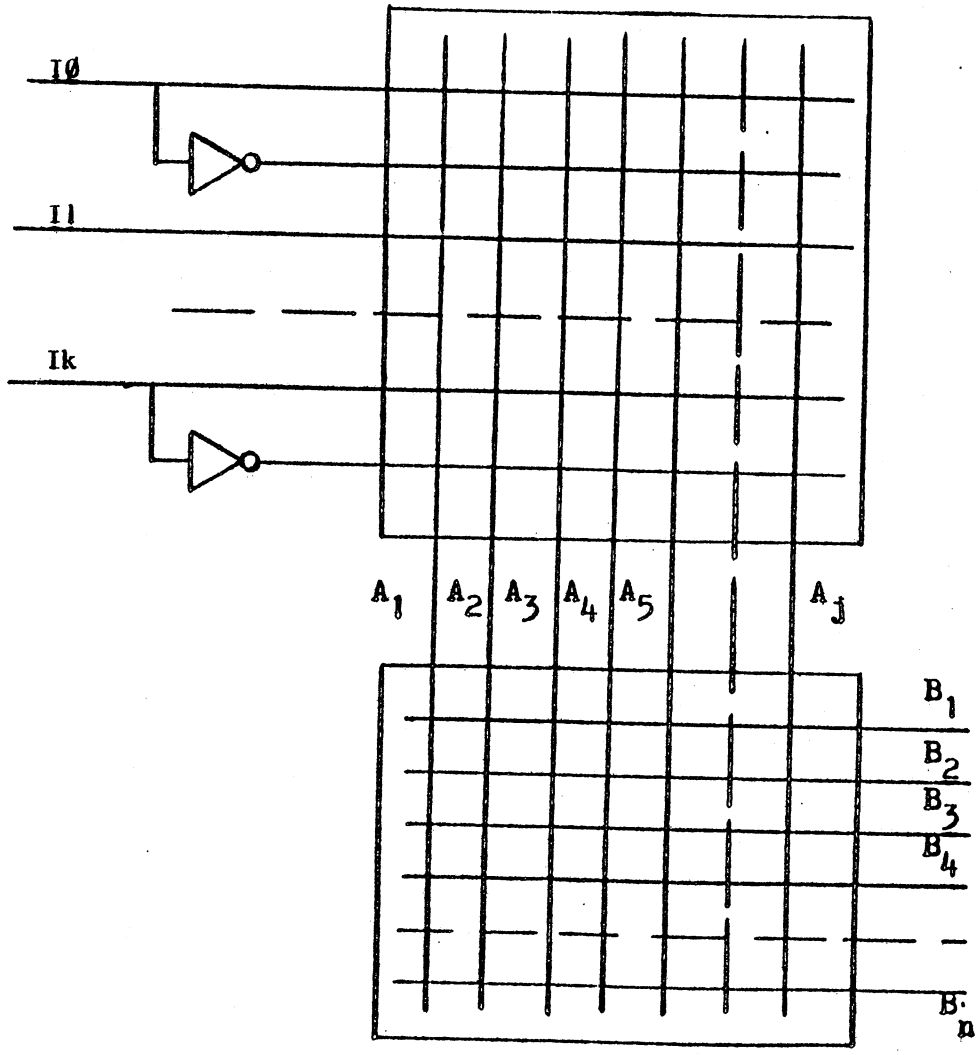


Figure 56 - PLA original

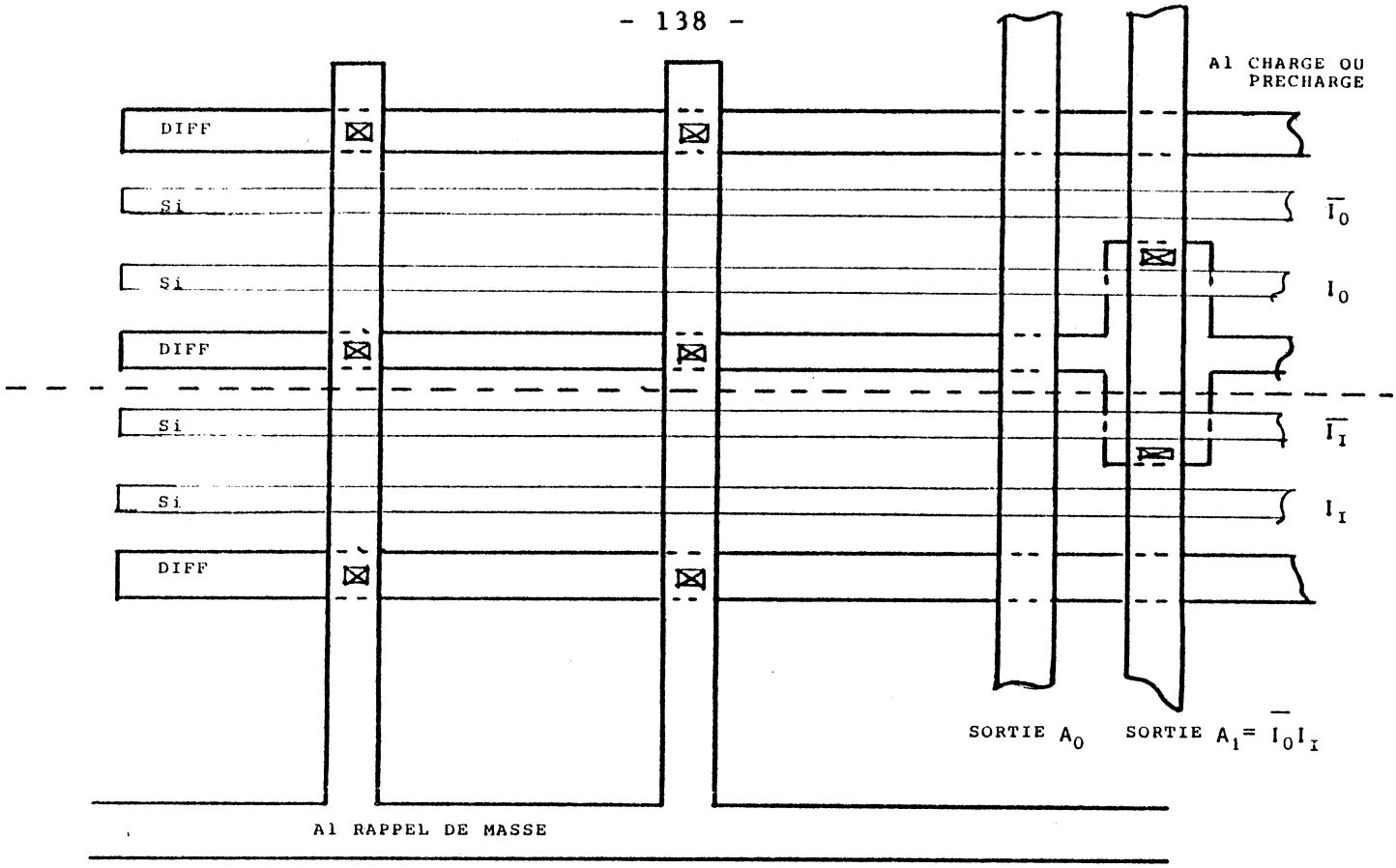


Figure 57 - Implémentation de la première matrice.

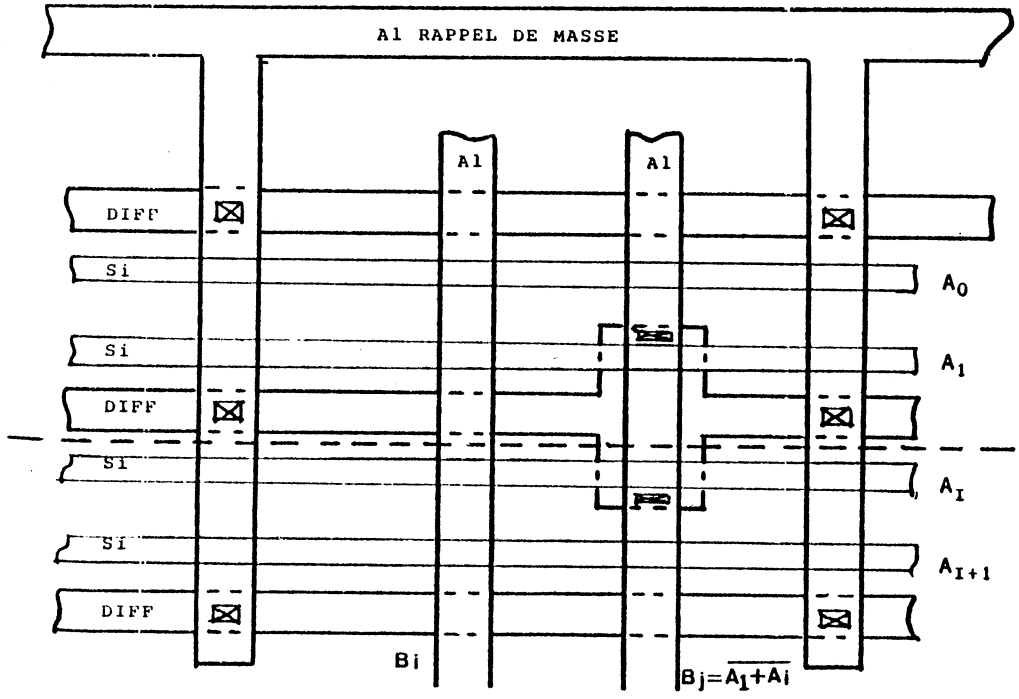


Figure 58 - Implémentation de la deuxième matrice.

V.2.1. PLA SFS pour un code détectant les erreurs simples

Par la suite on présente successivement l'analyse, la construction

et le PLA SFS augmenté.

V.2.1.1. Analyse et construction
La méthode de partition proposée à la section IV.1.3. est utilisée pour assurer la règle R1.

On divise le PLA en trois blocs: le bloc des entrées primaires, le bloc des monômes (colonnes) et le bloc des sorties primaires (lignes). On peut vérifier facilement que chacun de ces blocs vérifie la règle R1. Par conséquent chacun de ces blocs sera controlé par un controleur de parité après avoir commandé tous les MOS du bloc suivant. Ainsi, les coupures des entrées ou les coupures des colonnes ou les défaillances des MOS de la première matrice qui peuvent produire des erreurs multiples aux sorties du PLA, seront controlées.

La règle R2 n'est pas vérifiée par les lignes VDD et VSS. Par conséquent les coupures des lignes d'alimentation sont retirées des hypothèses de pannes.

Des conditions générales seront utilisées pour démontrer que les séquences pour lesquelles le circuit est "strongly redundant" ne contiennent pas de défauts du type B2 comme il es proposé à la section III.1.2.

Chacun des trois groupes de lignes, c'est à dire les entrées primaires, les monômes et les sorties primaires sont controlés directement.

Primitivement le PLA est implémenté de façon à ce qu'il n'existe pas deux entrées, ou deux monômes, ou deux sorties, qui prennent en permanence les mêmes valeurs. Par conséquent il n'existe pas de défauts du type B2 pour lesquels le circuit est redondant (les trois types de lignes étant testés directement).

Une autre conséquence du contrôle direct des trois types de lignes

est que: chaque séquence de défauts <f1,f2,...,f k-1> qui modifiera le PLA de façon à ce que deux lignes d'entrée, ou deux monômes, ou deux lignes de sortie, prennent en permanence les mêmes valeurs, sera détectée. Alors il n'existe pas de séquences de défauts qui contient des défauts du type B2 et pour lesquelles le circuit est "strongly redundant".

Finalement, la vérification de la règle R4 n'est pas nécessaire étant donné que les coupures des lignes d'alimentation sont retirées des hypothèses de pannes.

Le bit de parité des monômes et le bit de parité des lignes de sortie doivent être générés.

Soit Bn+1 une sortie supplémentaire, égale au bit de parité des sorties B1,B2...,Bn. Pour la génération de Bn+1 on peut utiliser les monômes initiaux A1,A2...,Aj et quelques monômes supplémentaires A1', A2',...,Am' (notons que le PLA pourrait être complètement réimplémenté pour des raisons d'optimisation).

Les monômes issus de la première matrice (A1,A2,...Aj,A1',A2'...,Am') doivent être controlés pour des erreurs simples. Ces monômes sont partagés en deux groupes: le premier groupe contient des monômes affectant une sele sortie (c'est par exemple le cas des monômes A1',A2',...Am'). Le deuxième groupe contient les monômes affectant au moins deux sorties.

Seul le contrôle des monômes du deuxième groupe est nécessaire, étant donné que les défauts affectant les monômes du premier groupe ne peuvent provoquer que des erreurs simples aux sorties du PLA. Soit A1",A2",...,A1" les monômes du deuxième groupe et soit Bn+2 une nouvelle sortie du PLA égale au bit de parité des monômes A1",A2"...,A1". Pour implémenter Bn+2 on peut utiliser les monômes du premier groupe et de nouveaux monômes qui devront probablement être rajoutés.

V.2.1.2. PLA augmenté SFS

Le PLA augmenté est présenté figure 59. Les trois controleurs ainsi que les bits de parité Bn+1 et Bn+2 sont présentés dans cette figure.

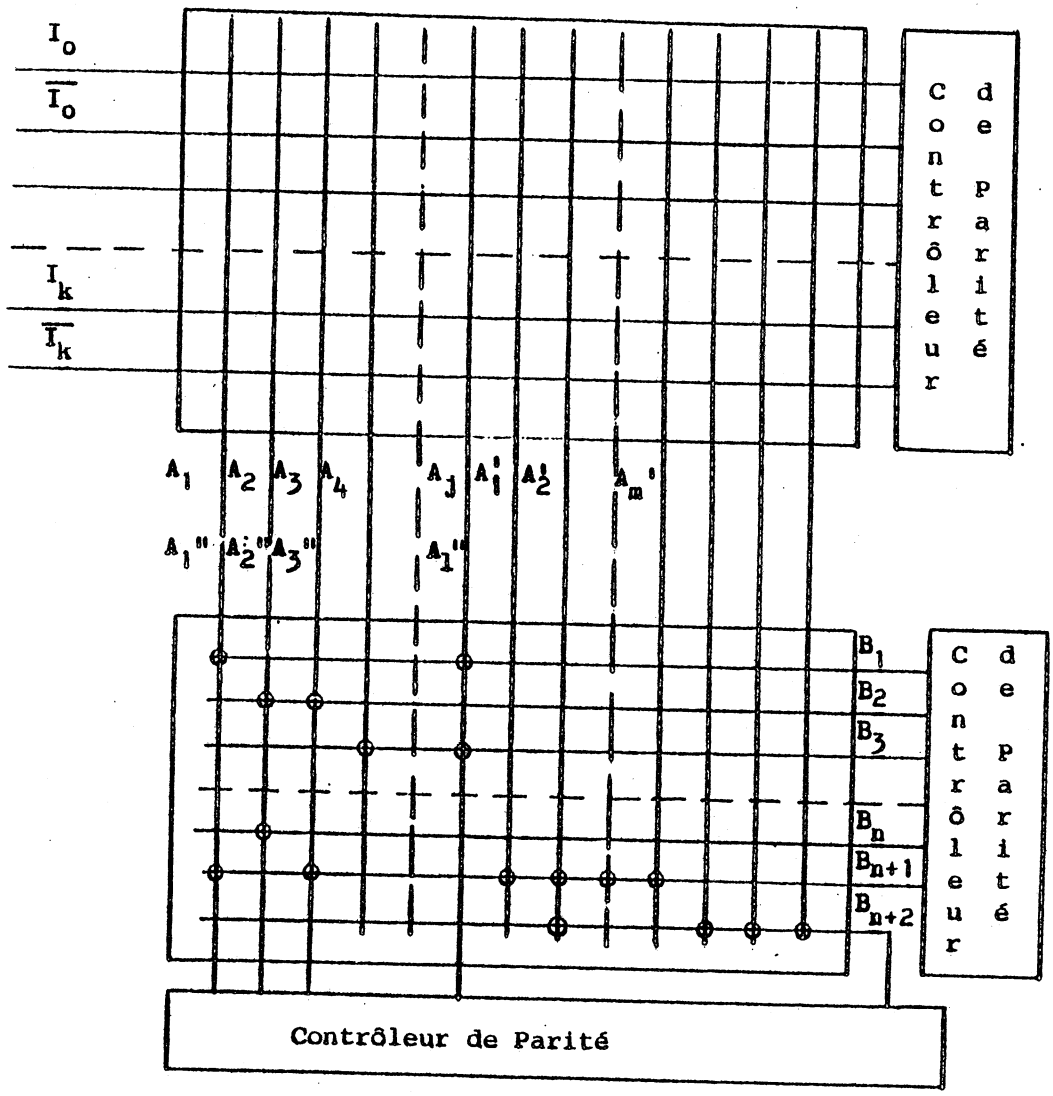


Figure 59 - PLA augmenté

Notons que:

- le controleur des entrées doit controler les entrées directes mais aussi les entrées complémentaires ;
- dans le PLA augmenté, on utilise certains monômes qui ne sont pas contrôlés directement (par le contrôleur des monômes). Par conséquent l'analyse affectant les défauts du type B2 n'est pas

valable pour ces monômes.

Les courts-circuits du type B2 qui sont critiques sont les suivants:
- courts-circuits entre deux monômes du premier groupe (voir analyse et constructions). Ces lignes doivent être séparées en implémentant d'autres lignes entre elles.
- courts-circuits entre un des nouveaux monômes utilisés pour la génération de la sortie Bn+2 (voir analyse et construction) et un des monômes du deuxième groupe. Une ligne du premier groupe peut être utilisée pour séparer les monômes du deuxième groupe et les nouveaux monômes utilisés pour la génération de la sortie Bn+2.

La séparation nécessaire de différets monômes n'apparaît pas dans la figure 59.

Par sa construction, un tel PLA est SFS pour la classe I, les coupures des lignes d'alimentation étant exceptées.

V.2.2. PLA SFS pour un code détectant les erreurs unidirectionnelles.

La détection des erreurs unidirectionnelles est considérée dans la mesure où des modèles ordinaires de défauts provoquent des erreurs unidirectionnelles aux sorties des PLAs [DON 81], [MAK 82].

V.2.2.1. Analyse et construction

Dans la section IV.2.4. on a vérifié que la règle R5" est assurée par les PLAs, par conséquent la méthode de contrôle des entrées primaires sera utilisée et les sorties primaires seront testées par un code non-ordonné.

La règle R6' est vérifiée car:
Les lignes d'alimentation de la premiére matrice alimentent les monômes (la parité d'inversion des chemins entre les monômes et les

sorties primaires est égale à 1).
Les lignes d'alimentation de la deuxième matrice alimentent les sorties du PLA (parité d'inversion égale à 0).

La règle R7' est vérifié car les courts-circuits du type B2' sont les courts- circuits entre les entrées et les monômes et les courts-circuits entre les monômes et les sorties.

Dans les figures 57 et 58 on peut vérifier que ces courts-circuits sont éliminés (pour la classe I) car sur la première matrice les entrées sont implémentées en Si-poly et les monômes en Alu. Sur la deuxième matrice, les monômes sont en Si-poly et les sorties en Alu.

Finalement, la règle R8 est vérifiée car les lignes d'alimentation de la première matrice alimentent des portes (monômes) pour lesquelles les parités d'inversion sont égales. La même constatation est valable pour les lignes alimentant la deuxième matrice.

Selon l'analyse donnée, le code de sortie du PLA doit être un code non- ordonné. On peut alors utiliser des codes séparables non-ordonnés comme le code de BERGER ou le code de BERGER modifié proposé dans [DON 82] et dans [MAK 82]. Par conséquent on doit générer des sorties supplémentaires Bn+1,Bn+2,...,Bn+p.

Notons qu'aucune modification de la structue ordinaire des PLAs n'est nécessaire ; cette structure est tout à fait convenable pour la conception des PLAs SFS.

V.2.2.2. PLA augmenté SFS

Le PLA augmenté est présenté figure 60. Ce PLA est SFS pour la classe I.

Notons que ce PLA est le PLA augmenté proposé dans [MAK 82]. L'analyse est très simple en utilisant les règles générales

proposées dans cette étude.

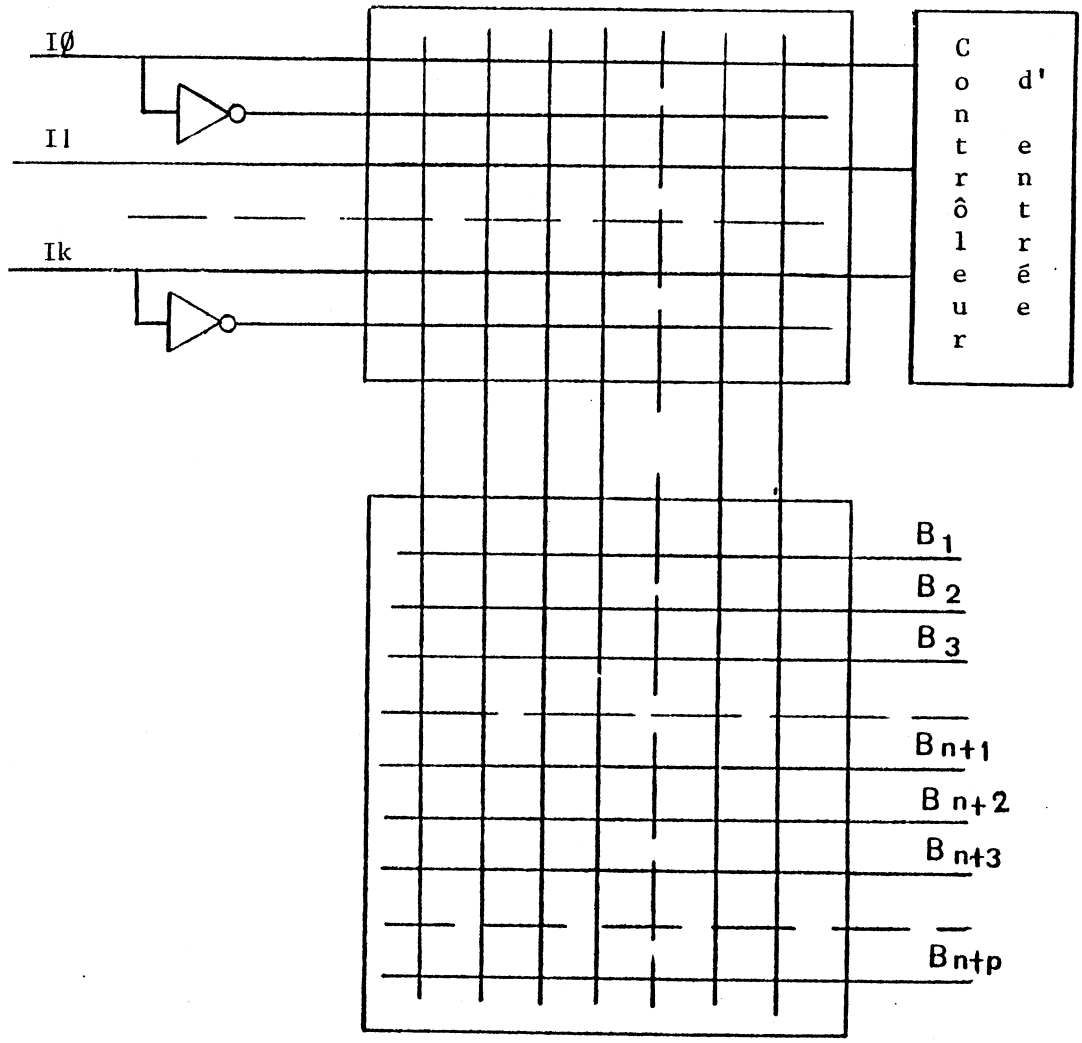


Figure 60 - PLA augmenté

Notons que la même méthode peut être utilisée pour la conception SFS des structues composées par 3, 4 (etc...) matrices NOR. Ces circuits peuvent être conçus, comme les PLAs NOR-NOR, en utilisant le contrôle des entrées primaires et un code de sortie non-ordonné. Par contre, cette analyse ne peut pas être appliquée aux circuits composés par une seule matrice NOR.

V.2.2.3. PLA composés d'une seule matrice NOR

Cette structure est utilisée très souvent pour concevoir des circuits de décodage.

Dans le chapître IV nous avons montré que la règle GR5" est vérifiée par le circuit augmenté. Par conséquent le circuit doit être testé, en utilisant le contrôle des entrées primaires et un code de sortie non-ordonné généralisé.

Une analyse similaire à l'analyse donnée pour les PLAs NOR-NOR peut être utilisée pour mettre en évidence la vérification des règles GR6', GR7 et GR8.

Le circuit augmenté est présenté figure 61 ; ce circuit était déjà déterminé à la section IV.31 (figure 50).

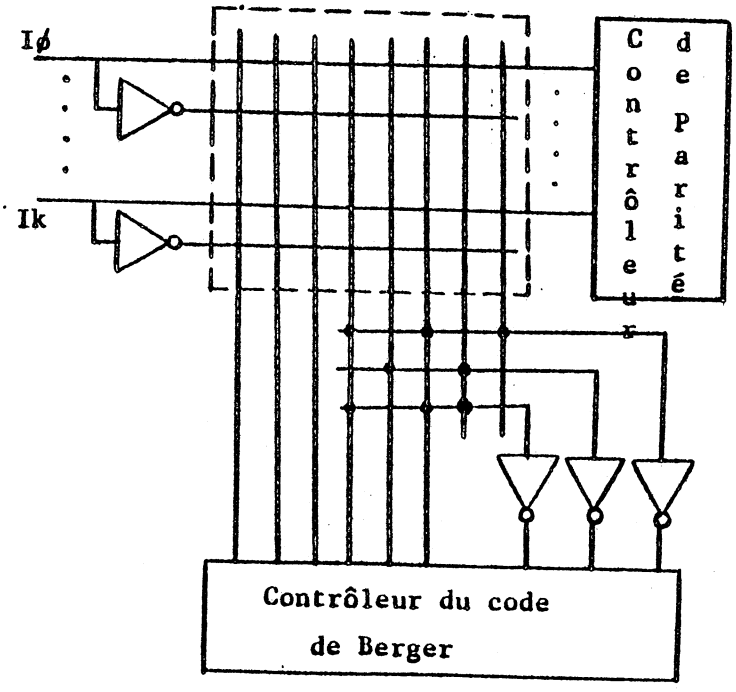


Figure 61 - PLA augmenté

V.2.3. PLA SFS pour un code détectant les erreurs multiples

On peut considérer deux types de codes de sortie:
- le code de duplication ; n sorties supplémentaires sont nécessaires (B1,B2,...,Bn) ;
- le code double-rail ; n sorties supplémentaires sont nécessaires ($\overline{B1}$,$\overline{B2}$,...,$\overline{Bn}$).

V.2.3.1. Analyse et construction

On peut montrer que les règles R9, R10, R11 et R12 ne peuvent pas être vérifiées par la structure des PLAs ; ainsi les erreurs provoquées par les défauts de la classe I seront propagées jusqu'aux deux groupes des sorties du PLA. Le code de la duplication ne convient pas pour la conception SFS des PLAs, sauf si on utilise deux PLAs séparés (méthode de la duplication classique).

D'autre part, l'analyse de la section IV.2.2. a montré qu'un code non ordonné est convenable pour la conception des PLAs SFS. Le code double-rail étant un code non ordonné, pourrait être utilisé.

La figure 61bis présente le PLA augmenté.

Il est évident que le code de BERGER, qui offre une protection similaire, est préférable car il est moins coûteux que le code double-rail.

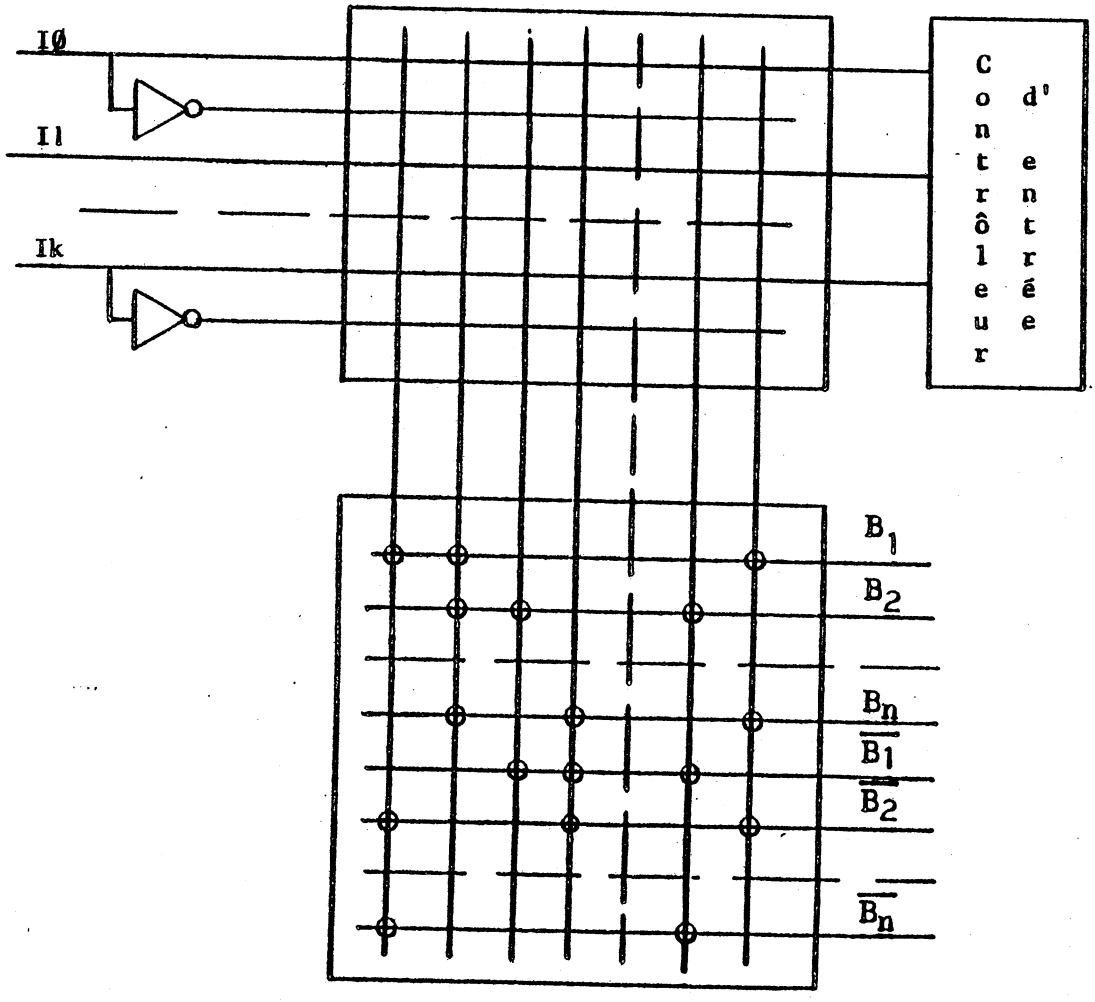


Figure 61 bis - PLA augmenté

VI - LES CONTROLEURS "STRONGLY CODE DISJOINT"

Les contrôleurs Strongly Code Disjoint (SCD) sont définis dans ce chapître. Ces contrôleurs incluent les contrôleurs Code Disjoint, Totally Self Checking. Ce type de contrôleurs est le type nécessaire pour être associé à des réseaux Strongly Fault Secure, tels que définis par SMITH-METZE.

Des définitions sont données, tout d'abord vis-à-vis d'une classe générique de défauts. Pour illustrer la conception de contrôleurs SCD, la conception d'une cellule de contrôleur double-rail SCD est donnée, vis-à-vis d'une large classe de défauts , en technologie NMOS.

Des contrôleurs SCD pour des codes systématiques, séparables ou non, sont ensuite abordés. Il est démontré que pour certaines conceptions, une condition nécessaire et suffisante donnée par ASHJAEE-REDDY n'est plus nécessaire pour des contrôleurs SCD.

Enfin, la conception de contrôleurs spécifiques SCD est examinée. Ces contrôleurs sont nécessaires pour concevoir de contrôleurs de codes arbitraires.

Les contrôleurs SCD sont la plus large classe de contrôleurs avec lesquels un système combinatoire peut atteindre le Totally Self Checking Goal.

VI.1. Définition des contrôleurs SCD

Dans la définition classique, un contrôleur est défini comme un circuit TSC à codes disjoints (def. D5).

Dans [MAK 82] un contrôleur qui est associé avec un bloc fonctionnel SFS, doit verifier les propriétés SFS et à codes disjoints pour accomplir le but de "Totally Self Checking Goal".

On peut vérifier que les propriétés proposées dans [MAK 82] ne sont pas suffisantes car la propriété SFS permet la présence de certains types de défauts, mais en présence de ces défauts il n'existe aucune précaution concernant la propriété à "codes disjoints".

Deux remarques concernant les contrôleurs sont importantes pour déterminer la définition adéquate.

Remarque 1
La propriété à "codes disjoints" n'est pas suffisante pour un contrôleur associé avec un circuit SFS.

La propriété TSC (Self testing et Fault secure) est utile car la propriété "self testing" assure la détection des défauts, mais la propriété "fault secure" n'est pas vraiment nécessaire.

Remarque 2
La propriété "fault secure" n'est pas vraiment nécessaire. Alors la recherche des propriétés "fault secure" plus fortes comme dans [SON 81] n'est pas nécessaire.

Ces remarques peuvent amener à la définition des contrôleurs "strongly code disjoint" en prenant en compte le fait d'autoriser l'existence de défauts dans le contrôleur doit être compensée par des précautions concernant la propriété "codes disjoints".

On appelle B le code d'entrée du contrôleur (code de sortie du bloc fonctionnel) et C le code de sortie du contrôleur.

Définition D24
Avant l'occurrence de défauts, le circuit G est à codes disjoints. Pour une séquence de défauts $\langle f1, f2, \ldots, fn \rangle$, soit k le plus petit entier pour lequel il existe un vecteur $b \in B$ tel que

$$G\left(b, \bigcup_{j=1}^{k} fj\right) \notin C$$

S'il n'existe pas un tel k posons k=n. Alors le circuit G est

"strongly code disjoint" pour la séquence $\langle f1, f2, \ldots, fn \rangle$ si:

$$\forall\, b \notin B,\ \forall\, m \in \{1, 2, \ldots, k-1\}$$

$$G\left(b, \bigcup_{j=1}^{m} fj\right) \notin C$$

Définition D25

Un circuit G est "strongly code disjoint" pour un ensemble de défauts F, s'il est "strongly code disjoint" pour chaque séquence de défauts dont les éléments appartiennent à l'ensemble F.

Notons que si k=1 pour toutes les séquences de défauts qui peuvent survenir dans un circuit, alors le circuit sera "self-testing" et "à code disjoint".

Une définition plus forte des controleurs est donnée ci-dessous (toutefois cette définition n'est pas nécessaire).

Définition D26

Avant l'occurrence de défauts, le circuit G est à codes disjoints. Pour une séquence de défauts $\langle f1, f2, \ldots, fn \rangle$, soit k le plus petit entier pour lequel il existe un vecteur $b \in B$ tel que:

$$G\left(b, \bigcup_{j=1}^{k} fj\right) \notin C$$

S'il n'existe pas un tel k posons k=n. Alors le circuit G est "strongly code disjoint" (SCD) pour la séquence $\langle f1, f2, \ldots, fn \rangle$ si:

$$\forall\, b \notin B,\ \forall\, m \in \{1, 2, \ldots, k\}$$

$$G\left(b, \bigcup_{j=1}^{m} fj\right) \notin C$$

Définition D27

Le circuit est SCD pour un ensemble de défauts F, s'il est SCD (selon la définition D26) pour chaque séquence de défauts dont les éléments appartiennent à l'ensemble F.

Les définitions D24 et D26 sont différentes en ce qui concerne les propriétés du contrôleur en présence de la séquence $\langle f1, f2, \ldots, fk)$. Selon la plus forte définition D26, le contrôleur en présence de la

séquence ⟨f1,f2,...,fk⟩ transpose toujours les vecteurs d'entrée en dehors du code en vecteurs de sortie en dehors du code, tandis que selon la définition D24 il est possible d'avoir certains vecteurs d'entrée transposés en vecteurs de sortie appartenant au code.

On peut noter que si k=1 pour toutes les séquences possibles, la définition D26 pourrait faire penser à un circuit "self testing" et "code disjoint" d'après [SON 81] mais qu'en fait ces deux propriétés ne sont pas compatibles (cette incompatibilité existe que le circuit soit "self testing" ou "partially self testing": la définition des circuits "completely self checking" de [SON 81] définit en fait des circuits qui ne peuvent pas exister).

Par la suite on définit successivement les propriétés de redondance pour les contrôleurs et les contrôleurs SCD pour une classe Cf d'hypothèses de pannes.

Définition D28

Un contrôleur est redondant pour un défaut f si:

$$\forall\, b \in B,\quad G(b,f) \in C,\ \forall\, y \notin B,\ G(y,f) \notin C$$

(c'est-à-dire que le contrôleur en présence du défaut f est à codes disjoints).

Définition D29

Un contrôleur est "strongly redundant" pour une séquence ⟨f1,f2,...,fn⟩ s'il est redondant pour toutes les sous-séquences ⟨f1⟩, ⟨f1,f2⟩...⟨f1,f2,...,fn⟩, (c'est à dire que le contrôleur est à codes disjoints en présence de chacune des sous séquences).

Définition D30

Un contrôleur est SCD pour une classe Cf d'hypothèses de pannes s'il est à codes disjoints et si pour chaque séquence de défauts ⟨f1,f2,...,fn⟩ due à la classe Cf:

Soit il existe k tel que :

- $\exists\, b \in B / G(b, \bigsqcup_{j=1}^{k} fj) \notin C$
- le contrôleur est "strongly redundant" pour la séquence ⟨f1,f2,...,f k-1⟩

Soit le contrôleur est "strongly redundant" pour la séquence $\langle f1, f2, \ldots fn \rangle$.

On peut noter que la propriété SCD signifie que le contrôleur est initialement à codes disjoints (avant l'occurrence de défauts) et que pour chaque séquence pour laquelle il n'existe par un vecteur $b \in B$ qui détecte une des sous- séquences, le contrôleur reste à codes disjoints.

Une définition plus forte de contrôleurs pour une classe C d'hypothèses de pannes est donnée dans [NIC 83].

Une hypothèse concernant l'occurrence de défauts dans un système composè par un bloc fonctionnel et un contrôleur est nécessaire. Une telle hypothèse est détaillée ci-dessous.

Hypothèse H2
Après l'occurrence d'un défaut dans le bloc fonctionnel, il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée A soient appliqués dans le bloc fonctionnel, avant qu'un deuxième défaut survienne au bloc fonctionnel ou au contrôleur.
Après l'occurrence d'un défaut dans le contrôleur, il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée B soient appliqués au contrôleur, avant qu'un deuxième défaut survienne dans le contrôleur ou dans le bloc fonctionnel.

Proposition P25
L'hypothèse H2 étant assurée, un système composé de:
- un bloc fonctionnel SFS,
- un contrôleur SCD,
accomplit le but de "totally self checking goal".

Preuve
Avant l'occurrence de défauts, le bloc fonctionnel est "fault secure" et le contrôleur est à codes disjoints. Le premier défaut peut survenir, soit au bloc fonctionnel, soit au contrôleur.

a/ Le défaut survient dans le bloc fonctionnel:
L'hypothèse H2 assure que tous les vecteurs $a \in A$ seront appliqués au bloc fonctionnel avant l'occurrence d'un deuxième défaut dans le système.
Si le bloc fonctionnel n'est pas redondant pour le défaut, la première sortie erronée du bloc fonctionnel sera en dehors du code de sortie B (propriété FS) et le contrôleur produira une indication d'erreur (propriété à codes disjoints).
Si le bloc fonctionnel est redondant, pour le défaut, alors il ne produira pas d'erreur et le système fonctionnera correctement. En présence d'un tel défaut, le bloc fonctionnel reste SFS.

b/ Le défaut survient dans le contrôleur:
L'hypothèse H2 assure que tous les vecteurs $b \in B$ seront appliqués au contrôleur avant l'occurrence d'un deuxième défaut dans le système.
Si le contrôleur n'est pas redondant pour le défaut, il y aura une indication d'erreur.
Si le contrôleur est redondant pour le défaut alors le contrôleur fonctionnera correctement (le bloc fonctionnel aussi).En présence d'un tel défaut le contrôleur reste SCD.

D'après cette analyse, on peut avoir deux cas possibles:
- le contrôleur produit une indication d'erreur. Dans ce cas la défaillance est détectée (le système sera mis hors de service).
- le contrôleur ne produit pas une indication d'erreur. Dans ce cas le système conserve ses propriétés (défaut non détectable dans un circuit SFS ou dans un contrôleur SCD), un deuxième défaut peut survenir et le cas a/ ou b/ sera produit.

Par récurrence on peut vérifier que le système accomplit le "TSC goal",c.q.f.d.

Notons que le "TSC goal" pour un système composé par un bloc fonctionnel et un contrôleur signifie que la première sortie erronée du bloc fonctionnel doit provoquer une indication d'erreur

aux sorties du contrôleur.

On peut vérifier qu'un contrôleur, qui n'est pas SCD, ne peut pas assurer le "TSC goal" dans un système autotestable. Par conséquent les contrôleurs SCD sont la plus large classe de contrôleurs avec lesquels un système peut accomplir le "TSC goal".

L'univers des contrôleurs est présenté figure 62. Les contrôleurs SCD sont la plus large classe ; les contrôleurs SFS/SCD sont inclus dans les contrôleurs SCD. Les contrôleurs ST/CD sont aussi inclus car la propriété FS n'est pas vraiment nécessaire (ils sont obtenus pour k=1 dans la définition D24). L'intersection de la classe SFS/SCD et de la classe ST/CD donne les contrôleurs TSC/CD ("Totally self checking checkers"). Dans la même figure apparaîtra aussi la position des contrôleurs pour la plus forte définition.

Finalement, on doit préciser que des règles générales ne sont pas nécessaires pour la conception des contrôleurs. Par opposition aux circuits fonctionnels, les contrôleurs doivent implémenter un nombre standard de fonctions. Il ne reste plus alors qu'à connaître un dessin adéquat de chacune de ces fonctions.

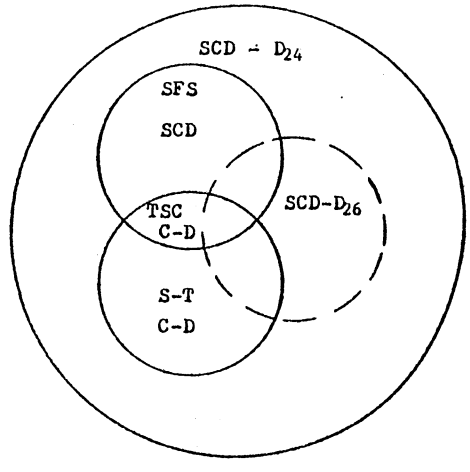


Figure 62 - L'univers des contrôleurs.

VI.2. Réalisation cellulaire des contrôleurs double-rail pour la classe I d'hypothèses de pannes.

Dans ce chapître on aborde successivement la conception SCD de la cellule de base pour la classe I ; la conception SCD (solution cellulaire) pour les contrôleurs double-rail qui reçoivent , en fonctionnement normal, tous les vecteurs du code ; la conception SCD (solution cellulaire) pour les contrôleurs double-rail qui ne reçoivent pas, en fonctionnement normal, tous les vecteurs du code.

Dans une réalisation cellulaire, la protection doit être assurée vis à vis des défauts concernant chaque cellule et vis à vis des défauts des interconnexions. Les défauts des interconnexions peuvent être [JAN 83]:

1/ coupure d'une ligne d'interconnexion ou court-circuit d'une ligne d'interconnexion et d'une ligne d'alimentation, ou court-circuit entre deux lignes d'interconnexion qui sont des entrées ou des sorties de la même cellule. Ce type de défaut est couvert par le contrôle des cellules.

2/ courts-circuits entre deux lignes VSS ou deux lignes VDD. Ces défauts ne sont pas détectables, mais ils ne posent pas de problèmes.

3/ courts-circuits entre deux lignes d'interconnexion qui ne sont pas des entrées ou sorties de la même porte, courts-circuits entre deux cellules, courts-circuits entre une ligne d'interconnexion et une cellule. En général le test des cellules n'assure pas le test de ces défauts.

VI.2.1. Conception SCD de la cellule de base

La figure 63 présente le schéma logique pour la cellule de base du contrôleur double-rail, proposé par CARTER et al. Cette cellule est à codes disjoints et TSC pour le modèle du collage logique. Cette section aborde la réalisation SCD de la cellule de base, pour la classe I d'hypothéses de pannes. Pour de telles hypothèses de pannes le schéma de l'implémentation physique doit être utilisé.

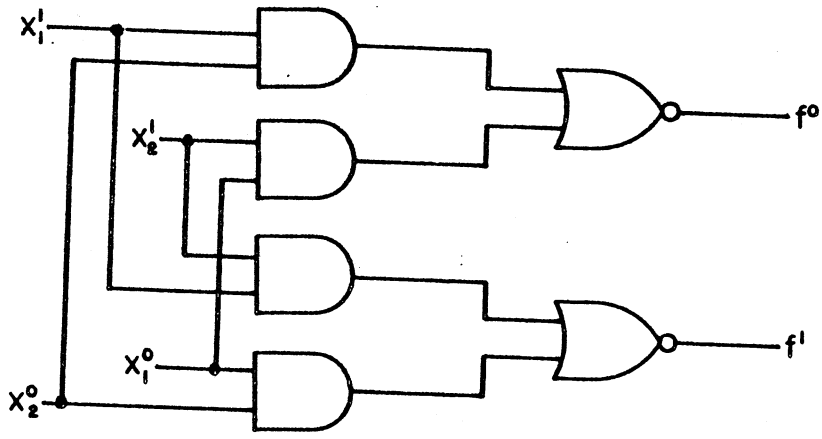


Figure 63 - Cellule de base du contrôleur double-rail.

Dans [JAN 83] des propositions de cellules double-rail SCD pour la classe I sont données. La figure 64 présente l'implémentation physique d'une telle cellule ; la propriété SCD est assurée si les quatre vecteurs possibles sont appliqués à la cellule en fonctionnement normal.

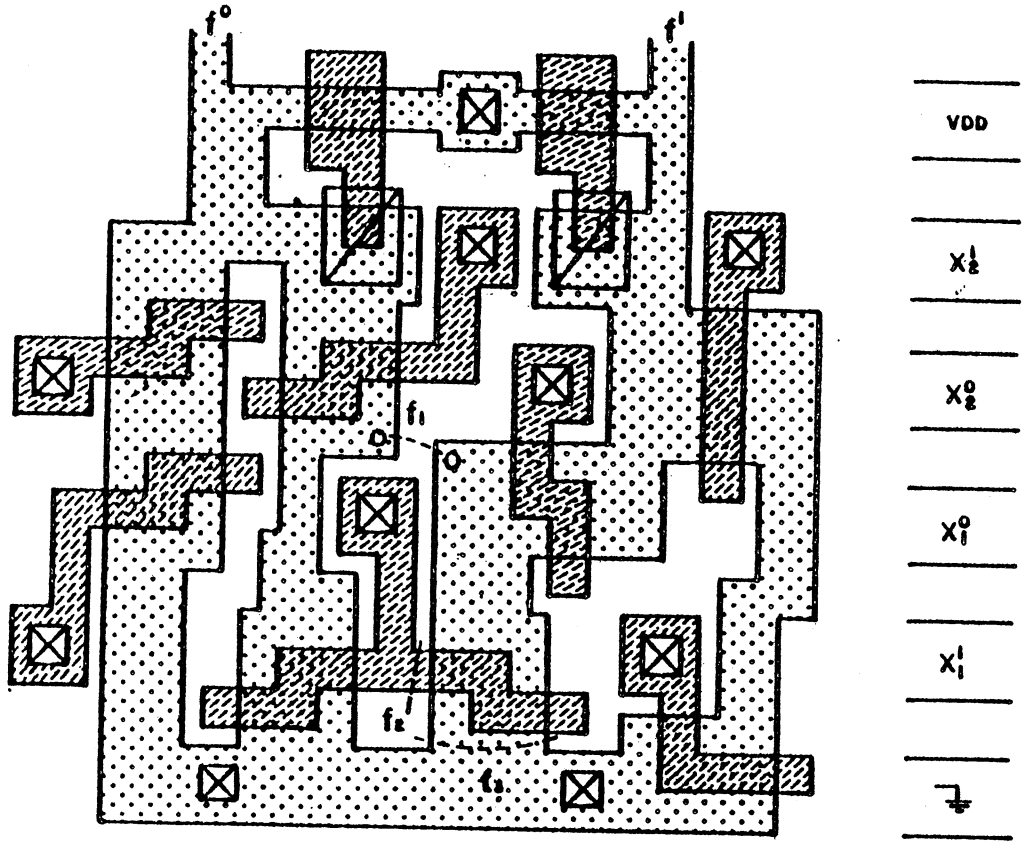


Figure 64- Cellule du contrôleur double-rail, SCD pour la classe I

Sur cette figure on peut vérifier que la cellule est redondante pour la séquence ‹f1,f2,f3›. Ceci n'est pas permis pour les contrôleurs TSC/CD, mais par contre ceci est autorisé pour les contrôleurs SCD. Cependant, cette redondance peut être éliminée en retirant un MOS commandé par l'entrée X_1^0. De façon similaire un deuxième MOS, commandé par l'entrée X_1^1, peut être aussi retiré. On obtient ainsi une cellule double-rail à 8 transistors MOS au lieu de 10 MOS de la cellule classique, le schéma électrique est présenté figure 65. Une implémentation SCD de cette cellule est présentée figure 66. Cette cellule est utile pour assurer le contrôle des interconnexions dans le cas de la réalisation cellulaire des contrôleurs double-rail qui ne reçoivent pas tous les vecteurs d'entrée en fonctionnement normal.

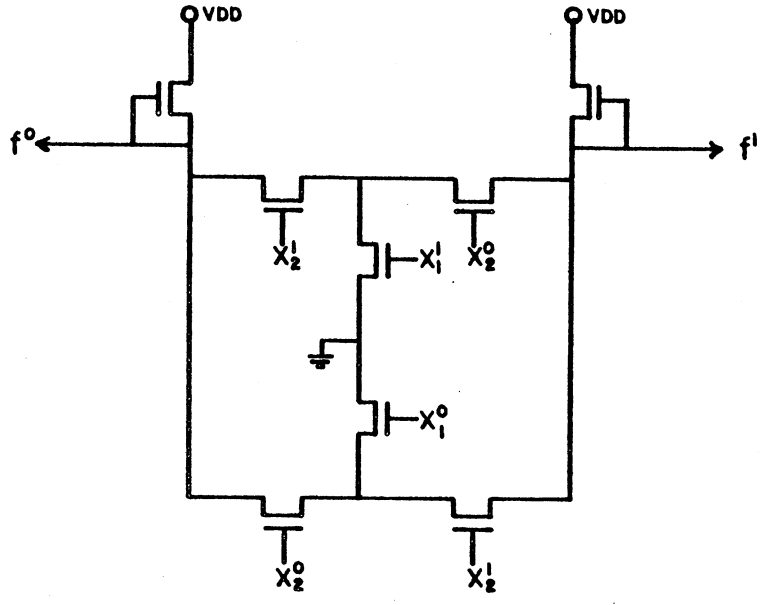


Figure 65 - Schéma électrique pour une cellule du contrôleur double-rail à 8 MOS.

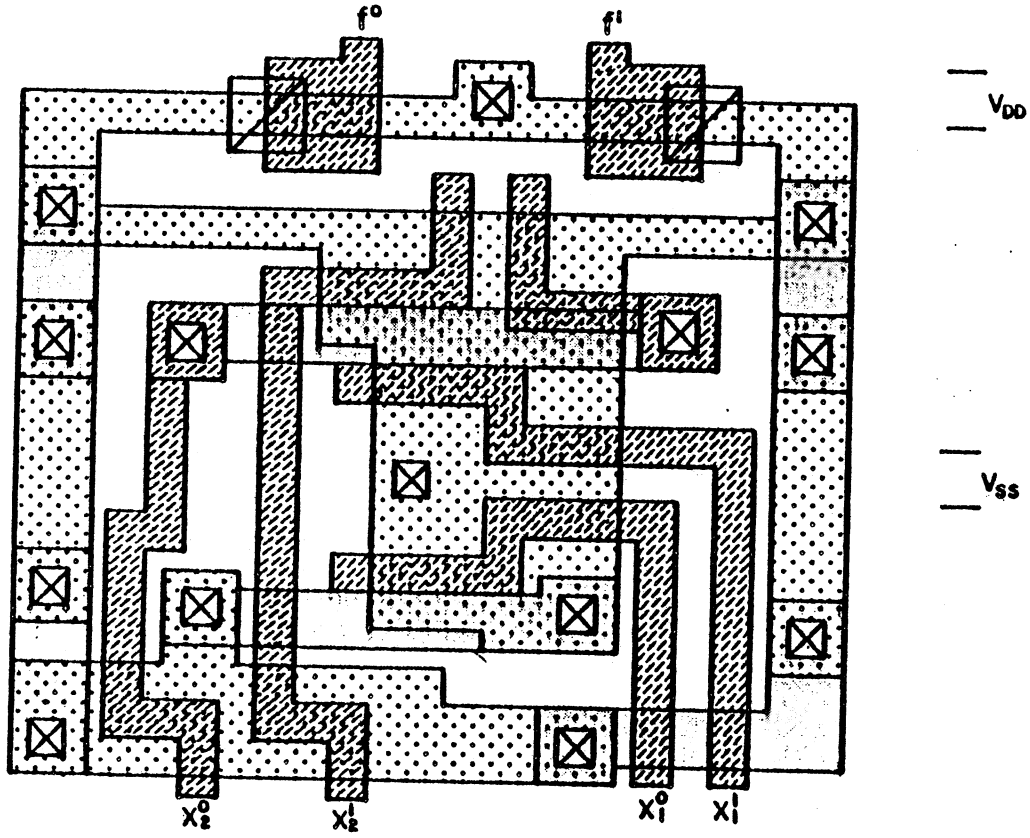


Figure 66 - Implémentation SCD pour le schéma de la figure 65.

VI.2.2. Conception SCD pour la réalisation cellulaire des contrôleurs double-rail

On peut démontrer qu'un contrôleur réalisé avec des cellules SCD et tel que la détection des défauts des interconnexions est assurée , est un contrôleur SCD. La démonstration est basée sur la constatation suivante: chaque défaut non détectable, peut affecter une seule cellule (les défauts des interconnexions étant détectables). La démonstration se fait ensuite en utilisant la propriété SCD de chaque cellule.

Par conséquent, pour la réalisation cellulaire des contrôleurs double-rail, on peut utiliser une implémentation SCD pour la cellule de base et ensuite on doit s'assurer que chaque cellule reçoit les quatre vecteurs du code et que tous les défauts des interconnexions sont détectés en fonctionnement normal.

Pour un contrôleur double-rail qui reçoit en fonctionnement normal tous les vecteurs du code, on peut vérifier que chaque cellule reçoit les quatre vecteurs du code et que tous les défauts des interconnexions sont détectés. Dans ce cas le contrôleur sera SCD.

Dans le cas où le contrôleur ne reçoit pas en fonctionnement normal tous les vecteurs du code, il est possible que les 4 vecteurs ne soient pas appliqués sur chaque cellule, ou que les défauts des interconnexions ne soient pas détectables.

La première difficulté peut être surmontée en utilisant un algorithme proposé dans [KHA 83].

Cet algorithme permet de dessiner des arbres de cellules X-OR, de façont à ce que chaque cellule reçoive les quatre configurations d'entrées possibles, lorsque l'arbre reçoit en entrée 4 configurations qui forment un type de matrice binaire déterminé dans [KHA 83].

Cet algorithme peut aussi être utilisé pour la réalisation cellulaire d'un contrôleur double-rail car chaque arbre des cellules de contrôle double-rail est associé cellule par cellule à un arbre X-OR [WAK 78].

Le deuxième problème à surmonter pour les contrôleurs double-rail qui ne reçoivent pas en fonctionnement normal toutes les configurations du code, est le problème des interconnexions. L'utilisation de l'implémentation donnée à la figure 66 permet de réduire les défauts des interconnexions en certains types de défauts facilement détectables. Dans la figure 67 on donne certains défauts qui peuvent représenter les défauts des interconnexions du type 3 données au début de la section VI.2.

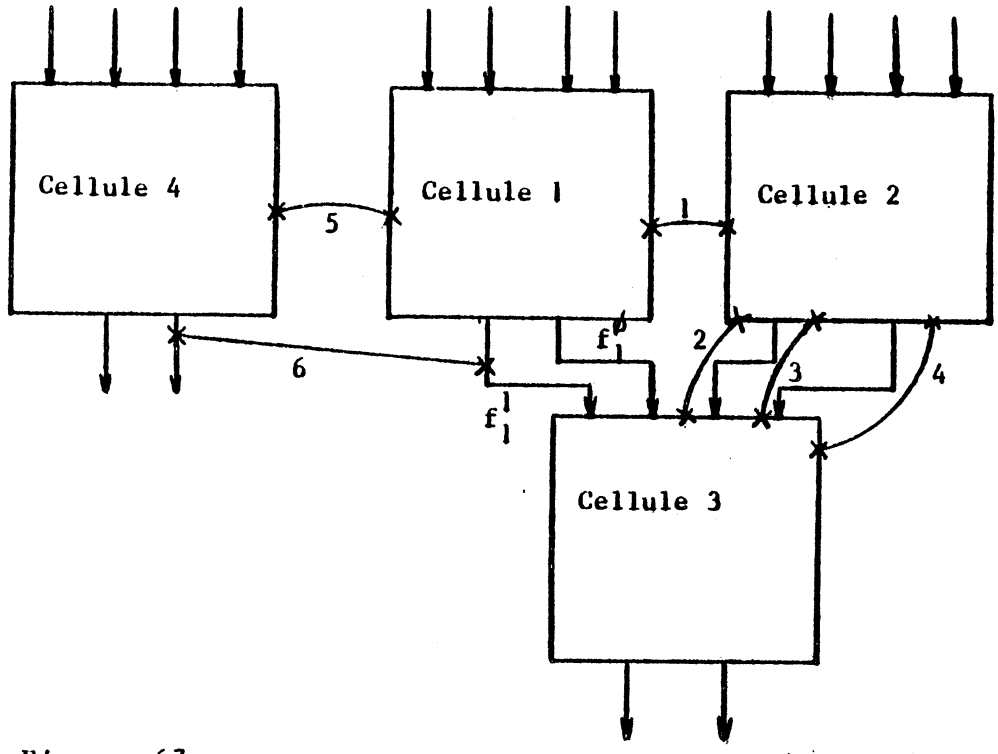


Figure 67

Si on utilise la cellule de la figure 66:

- les défauts 1, 2, 3 et 4 sont couverts par le test de la cellule3 car le défaut 1 est équivalent au court-circuit entre deux entrées de la cellule 3, le défaut 2 est équivalent au court-circuit entreune entrée et une ligne interne de la cellule 3, le défaut 3 est équivalent au court-circuit entre une ligne de la cellule 3 et une ligne VDD, et le défaut 4 est équivalent entre une entrée et une sortie de la cellule 3.

- le défaut 5 est équivalent au défaut 6 (cout-circuit entre une sortie de la cellule 1 et une sortie de la cellule 4). Ces défauts ne sont pas couverts par le test des cellules, l'application des 4 vecteurs de code à chaque cellule n'implique pas la détection de ces défauts. S'il existe des défauts du type 5 et 6 qui sont indétectables en fonctionnement normal, une modification simple peut résoudre ce problème. En échangeant les entrées X_2^0 et X_2^1 de la cellule 1, on échange aussi les valeurs des sorties f_1^0 et f_1^1; grâce à cet échange, les défauts 5 et 6 deviennent

automatiquement détectables car $f_1^0 = \overline{f_1^1}$.

VI.3. Contrôleurs SCD pour des codes systématiques

Définitions:

On donne d'abord quelques définitions nécessaires prises par [AND 71] et [ASH 76].

Définition D31

Un code est systématique si les bits des vecteurs du code sont partagés en deux groupes, l'un des groupes contient les bits arbitraires d'information et l'autre groupe contient les bits redondants du contrôle. Dans un code systématique, le nombre des bits d'information sera désigné par "t" et le nombre des bits de contrôle sera désigné par "r". Alors la longueur des vecteurs du code est n=t+r.

Définition D32

Un ensemble C des n-uplets binaires est un code séparable si:

i/ l'ensemble C contient 2^t n-tuples, $0 \leq t \leq n$ et

ii/ $C = \{X : X = I_x P_x\}$, où I_x est un t-uplet binaire et P_x est un (n-t)-uplet binaire, qui sont appelés respectivement les bits d'information et les bits de contrôle, et chacun des 2^t t-uplets d'information apparaît dans un des n-uplets du code.

Définition D33

Un code séparable est un code séparable complet si et seulement si chacun des $2^{(n-t)}$ (n-t)-uplets de contrôle apparaît dans un des n-uplets du code. Si cette condition n'est pas vérifiée le code séparable est un code séparable incomplet.

VI.3.1. Conception des contrôleurs pour les codes systématiques

Une réalisation possible pour les contrôleurs des codes systématiques est donnée figure 68. Dans cette figure, le générateur des bits de contrôle donne à ces sorties le complément des bits de contrôle.

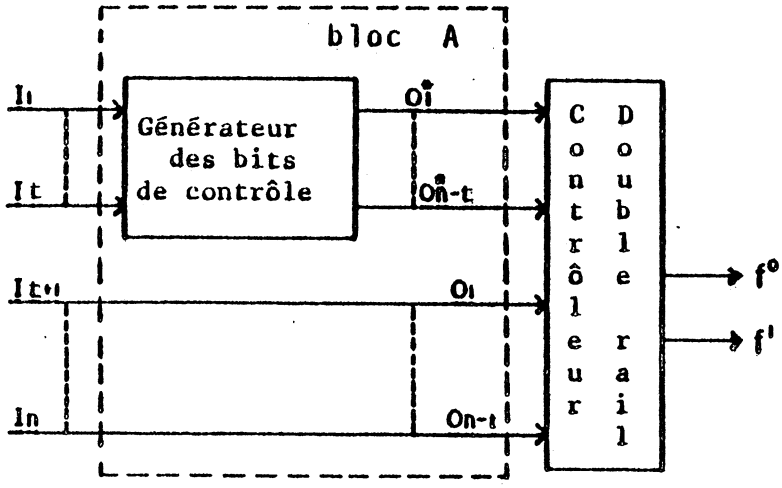


Figure 68 - Contrôleur pour les codes systématiques

Une telle structure est proposée dans [WAK 74] pour la conception des contrôleurs totalement autotestables (TSC checkers).

Dans [ASH 76] il est noté que le contrôleur double-rail ne peut pas être "TSC checker" si le code est un code séparable incomplet; dans la même étude la condition d'avoir un code séparable complet est donné comme une condition suffisante pour avoir un contrôleur double-rail totalement autotestable.

Dans [WAK 74] et [ASH 76] il est exigé que le générateur des bits de contrôle ne soit pas redondant (contrôleur du type 1 [ASH 76]). Dans ces études ([WAK 74] et [ASH 76]), la définition des "TSC checkers" est utilisée et le modèle de collage logique est

considéré. Pour ce modèle les redondances peuvent toujours être éliminées et l'exigence de non-redondance est acceptable. Mais pour des hypothèses de pannes de niveau bas, une telle exigence n'est pas acceptable et la définition des contrôleurs SCD doit être utilisée.

On peut vérifier qu'une condition nécessaire et suffisante pour que le contrôleur de la figure 68 soit SCD est que le bloc A soit SCD et que le contrôleur double-rail soit SCD. La conception des contrôleurs double-rail ayant déjà été discutée, il est à étudier la conception du générateur des bits de contrôle.

VI.3.1.1. Le générateur des bits de contrôle pour les codes séparables

Proposition P26
Pour chaque code séparable, et pour chaque modèle de défauts, affectant le générateur des bits de contrôle, le bloc A (figure 68) est SCD.

Preuve
Soit A(b,0) la fonction qui réalise le bloc A avant l'occurrence de défauts; $b \in B$, B et C sont respectivement le code d'entrée et le code de sortie du contrôleur de la figure 68. Pour un code séparable, tous les 2^t t-uplets de bits d'information I1,I2,...,It sont appliqués en fonctionnement normal.

Soit <f1,f2,...,fi,...,fn> une séquence de défauts affectant le générateur des bits de contrôle (n'importe quel type de défauts peut appartenir à cette séquence, même des défauts multiples).

Soit k l'entier déterminé dans la définition D24, pour cet entier il est évident que:

$$\forall\, b \in B,\ \forall\, m \in \{1,2,\ldots,k-1\} \ /\ A\left(b, \bigsqcup_{j=1}^{m} fj\right) \in C \quad (1)$$

Etant donné que les défauts de la séquence <f1,f2,...,f k-1>

affectent le générateur des bits de contrôle on peut vérifier que les lignes $01,\ldots 0n-t$ prennent des valeurs correctes ; la relation (1) implique que les lignes $01^*, 02^*, \ldots 0n-t^*$ prennent des valeurs correctes.

Soit Bl l'ensemble des vecteurs des bits de contrôle, alors:

$$\forall\, bl \in Bl,\ \forall m \in \{1,2,\ldots k-1\},\ \forall\, i \in \{1,2,\ldots,n-t\} \ /\ 0i^*(bl, \bigcup_{j=1}^{m} fj) = 0i^*(bl,0)$$

Le code étant séparable, i.e. l'ensemble Bl des vecteurs des bits de contrôle étant complet, la relation précédente est réduite à:

$$\forall\, bl,\ \forall m \in \{1,2,,,k-1\},\ \forall i \in \{1,2,\ldots,n-t\}$$
$$0i^*(bl, \bigcup_{j=1}^{m} fj) = 0i(bl,0) \qquad (2)$$

D'autre part:

$\forall\, b \notin B,\ A(b,0) \notin C$, i.e.

$$\forall\, b \notin B,\ \exists j \in \{1,2,\ldots,n-t\} \ /\ 0j^*(bl,0) = 0j \qquad (3)$$

avec $b = bl\ b2$

De (2) et (3) on trouve:

$$\forall\, b \notin B,\ \forall\, m \in \{1,2,\ldots,k-1\},\ \exists j \in \{1,2,\ldots,n-t\} / \ 0j^*(bl, \bigcup_{j=1}^{m} fj) = 0j$$

i.e,

$$\forall\, b \notin B,\ \forall m \in \{1,2,\ldots,k-1\},\ A(b, \bigcup_{j=1}^{m} fj) \notin C$$

Selon les définitions D24 et D25, le bloc est SCD. La conséquence de la proposition P26 est que n'importe quelle réalisation du générateur des bits de contrôle donne un bloc A SCD pour n'importe quel type d'hypothèses de pannes. L'exigence d'un générateur des bits de contrôle irredondant n'est pas nécessaire.

VI.3.1.2. Le générateur des bits de contrôle pour un code systématique mais non séparable.

Pour un tel code, seulement un sous-ensemble B des vecteurs

d'entrée est appliqué au contrôleur en fonctionnement normal, c'est-à-dire que l'ensemble Bl des vecteurs des bits d'information n'est pas complet. Dans ce cas, la proposition P27 donne une condition nécessaire et suffisante pour que le bloc A soit SCD pour un ensemble F de défauts affectant le générateur des bits de contrôle.

Proposition P27

Pour chaque séquence $\langle f1,f2,\ldots,fn\rangle$ de défauts $fi \in F$ affectant le générateur des bits de contrôle telle que:

- le bloc A est "strongly redundant" pour la séquence $\langle f1,f2,\ldots f_{m-1}\rangle$
- $\exists\ b \in B/A(b,\langle f1,f2,\ldots,fm\rangle \notin C$ (b étant un vecteur non appliqué en fonctionnement normal).

S'il existe un vecteur $b' \in B/A$ $(b', \langle f1,f2,\ldots,fm\rangle) \notin C$ (b' étant appliqué en fonctionnement normal), alors le bloc A est SCD.

Preuve

Si la proposition P27 est vraie, alors, en utilisant une analyse similaire à la démonstration de la proposition P26, on peut vérifier que le bloc A est SCD.

Si la proposition P2 n'est pas vraie, alors il existe une séquence $\langle f1,f2,\ldots fm\rangle$ de défauts de l'ensemble F telle que:

- le bloc A est "strongly redundant" pour la séquence $\langle f1,f2,\ldots f_{m-1}\rangle$,
- il existe un veccteur b non appliqué en fonctionnement normal tel que $A(b,\langle f1,f2,\ldots,fm\rangle \notin C$ (1)
- pour tous les vecteurs b' appliqués en fonctionnement normal, $A(b', \langle f1,f2,\ldots,fm\rangle) \in C$, la séquence $\langle f1,f2,\ldots,fm\rangle$ n'est pas détectée en fonctionnement normal.

On pose b=blb2 (avec bl bits d'information et b2 bits de contrôle).

Soit cl tel que:

$A(b1\ b2, \langle f1,f2,\ldots,fm\rangle) = c1\ b2$, la relation (1) implique que $\overline{c1} \neq b2$.

Soit le vecteur b1 $\overline{c1}$; ce vecteur n'appartient pas au code d'entrée B, car $\overline{c1} \neq b2$ et b1 b2 appartient au code d'entrée B. Pour ce vecteur on a:
$A(b1\ \overline{c1}, \langle f1,f2,\ldots,fm\rangle = c1\ \overline{c1} \in C$.

Alors $\exists b = b1\ \overline{c1} \notin B$ / $A(b,\langle f1,f2,\ldots,fm\rangle) \in C$ et aucune sous-séquence de la séquence $\langle f1,f2,\ldots,fm\rangle$ n'est détectable. Selon la définition D24 le bloc A n'est pas SCD.

Etant donné que les contrôleurs SCD sont la plus large classe de contrôleurs avec laquelle un système peut accomplir le "TSC goal", la proposition P2 est une condition nécessaire et suffisante pour la conception du générateur de bits de contrôle.

Notons que la détection, par les vecteurs appliqués en fonctionnement normal, de tous les défauts détectables par les vecteurs non appliqués, n'est pas une condition suffisante.

VI.4. CONTROLEUR SPECIFIQUE

Cette section est consacrée aux contrôleurs qui ne sont pas SCD car l'ensemble des vecteurs qui sont appliqués en fonctionnement normal est insuffisant.

De tels contrôleurs peuvent être les contrôleurs de codes classiques (code double-rail, code de parité, code de BERGER, etc...) qui reçoivent en fonctionnement normal un sous-ensemble des vecteurs du code qui ne suffit pas pour assurer les propriétés nécessaires. Ils peuvent être aussi les contrôleurs des codes arbitraires, comme par exemple un contrôleur des codes invalides d'un microprocesseur, pour de tels codes il est fort posible que le contrôleur ne puisse pas être SCD même s'il reçoit tous les vecteurs du code d'entrée.

Par conséquent, la seule méthode pour surmonter cette situation est

l'application sur le contrôleur de vecteurs supplémentaires pendant une phase de test.

Dans la suite, on dénote par B' les sous-ensembles de vecteurs du code d'entrée B qui sont appliqués en fonctionnement normal.

La figure 69 présente une architecture possible pour les contrôleurs spécifiques. Pendant la validation du signal TEST, des vecteurs d'entrée appartenant à l'ensemble B-B' sont générés, l'un après l'autre en utilisant le compteur. La phase de test peut être effectuée en plusieurs étapes si nécessaire, pour des raisons de temps réel.

Pendant la phase de test on peut aussi générer des vecteurs qui n'appartiennent pas au code d'entrée. Ceci est d'ailleurs nécessaire dans le cas des codes pour lesquels l'application en fonctionnement normal de tous les vecteurs du code ne suffit pas pour avoir une protection suffisante (cas de certains codes arbitraires par exemple). L'application en fonctionnement correct, d'un vecteur qui n'appartient pas au code d'entrée B, produira un signal d'erreur (00 ou 11) aux sorties du contrôleur (entrées du contrôleur double-rail).

Ceci étant indésirable, on doit inverser, pendant la génération des vecteurs en dehors du code, l'une des deux sorties du contrôleur. La figure 69 présente le cas où tous les vecteurs générés appartiennent au code (non renversement de sortie) ou le cas où tous les vecteurs générés n'appartiennent pas au code (inversement d'une sortie du contrôleur). Le cas de génération, pendant la phase du test, à la fois des vecteurs qui appartiennent au code d'entrée et des vecteurs qui n'appartiennent pas au code d'entrée n'est pas représenté par la figure 69. Ceci nécessite un signal qui permet de distinguer la phase de test en phase de génération des vecteurs n'appartenant pas au code d'entrée (inversement à une sortie) et en phase de génération des vecteurs appartenant au code d'entrée (non inversement de sortie).

Pour définir les contrôleurs spécifiques SCD on dénote par TVB l'ensemble des vecteurs du code d'entrée B qui sont générés pendant la phase du test et par TV∉B l'ensemble de vecteurs en dehors du code B qui sont générés pendant la phase du test.

Définition D34
Dans un contrôleur spécial donné figure 69 existe un circuit appelé "contrôleur". Avant l'occurrence de défauts le "contrôleur" est à codes disjoints. Pour une séquence <f1,f2,...,fn> affectant le "contrôleur", soit k le plus petit entier pour lequel il existe:

soit un $b \in B'$ tel que $G(b, \bigcup_{j=1}^{k} fj) \notin C$

soit un $b \in TVB$ tel que $G(b, \bigcup_{j=1}^{k} fj) \notin C$

soit un $y \in TV\notin B$ tel que $G(y, \bigcup_{j=1}^{k} fj) \in C$
(C étant le double-rail code de sortie du "contrôleur").

S'il n'existe tel k, posons k = n+1. Alors la partie "contrôleur" est "strongly code disjoint" pour la séquence <f1,f2,...,fn> si:
$\forall b \notin B$, $\forall m \in \{1,2,\ldots..,k-1\}$, $G(b, \bigcup_{j=1}^{m} fj) \notin C$.

Définition D35
La partie "contrôleur" du contrôleur spécifique présenté à la figure 69 est "strongly code disjoint" (SCD) pour un ensemble F, s'il est SCD pour toutes les séquences dont les éléments appartiennent à l'ensemble F.

Une définition plus forte ainsi qu'une définition pour une classe d'hypothèses de pannes peuvent être données, ceci n'est pas détaillé ici.

Un circuit ayant des propriétés intéressantes peut être dessiné lorsqu'aucun vecteur appartenant au code d'entrée n'est pas implémenté dans le PLA et que tous les vecteurs qui n'appartiennent pas au code d'entrée sont implémentés, c'est à dire TVB=0 et TV∉B=Y-B. La proposition P28 résume ces propriétés.

Proposition P28

Le "contrôleur" du contrôleur spécifique présenté à la figure 69 avec TVB=0 et TV∉B=Y-B est SCD:

- pour n'importe quel type de défaut affectant le "contrôleur",
- pour n'importe quel dessin du "contrôleur",
- et pour n'importe quel ensemble de vecteurs d'entrée appliqué en fonctionnement normal.

Preuve

Soit <f1,f2,...,fn> une séquence composée de défauts de n'importe quel type. Soit k l'entier déterminé dans la définition D34 ; alors par cette définition on a:
$\forall y \in TV\notin B$, $\forall m \in \{1,2,\ldots,k-1\}$, $G(y, \bigcup_{j=1}^{m} fj) \notin C$, c'est à dire $\forall b \notin B$, $\forall m \in \{1,2,\ldots,k-1\}$, $G(b, \bigcup_{j=1}^{m} fj) \notin C$ (TV∉B étant égal à Y-B). La partie "contrôleur" du contrôleur spécifique est alors SCD.

Les avantages d'un tel circuit sont les suivants:

- il n'y a pas de restriction concernant le modèle de défauts considéré,
- il n'y a pas de restriction concernant la conception du circuit,
- il n'y a pas de condition concernant l'ensemble des vecteurs qui doivent être appliqués en fonctionnement normal. Dans tous les cas le circuit conserve ses propriétés. Ainsi le circuit peut être utilisé sans problème pour des applications diverses.
- il n'y a pas de calculs à faire pour déterminer les vecteurs à stocker dans le PLA.

Notons que ces résultats sont dûs à la propriété SCD. Pour concevoir un "contrôleur" TSC qui assure les mêmes performances, il est nécessaire de stocker dans les PLAs l'ensemble Y de tous les vecteurs binaires d'entrée possibles. Par conséquent le temps nécessaire pour le test augmente la surface du circuit augmente et un signal qui distingue la phase du test, en phase de génération de vecteurs du code d'entrée et en phase de génération des vecteurs en dehors du code d'entrée, est nécessaire pour inverser ou non une sortie du "contrôleur".

Finalement, un tel circuit peut être utilisé pour les cas suivants:
- contrôleur des codes standards, pour lesquels il existe des contrôleurs SCD (ou TSC) mais les vecteurs appliqués en fonctionnement normal ne peuvent pas assurer ces propriétés ;
- contrôleur pour des codes arbitraires pour lesquels l'application de tous les vecteurs du code ne suffit pas pour assurer les propriétés nécessaires. Dans ce cas, la génération des vecteurs qui n'appartiennent pas au code, est nécessaire.

Notons que dans la figure 69, la structure double-rail du PLA est utilisée pour assurer au circuit de génération des vecteurs de test une protection similaire à celle assurée pour le "contrôleur" de la proposition P28. Bien sûr pour des hypothèses de pannes moins fortes un PLA SFS peut être utilisé. Il est aussi évident qu'un circuit SFS en logique anarchique peut remplacer les PLAs.

Notons aussi que la propriété nécessaire pour le circuit de génération des vecteurs de test est moins stricte que la propriété SFS.

Une panne dans le circuit de génération des vecteurs de test est détectée si:
a/ soit la panne provoque une sortie en dehors du code du circuit de génération des vecteurs de test (code double-rail pour les deux PLAs de la figure 69 ou un autre code si on utilise un PLA SFS).
b/ soit la panne provoque un vecteur en dehors du code d'entrée du "contrôleur" pendant la phase de génération des vecteurs dans le code.
c/ soit la panne provoque un vecteur dans le code d'entrée du "contrôleur" pendant la phase de génération des vecteurs en dehors du code.

La propriété nécessaire du circuit de génération des vecteurs de test est donnée dans la définition suivante:

Définition D36:

Soit <f1,f2,...,fn> une séquence de défauts affectant le circuit de génération des vecteurs de test ; soit k le plus petit entier tel que le circuit en présence de la séquence <f1,f2,...,fk> ne génère pas tous les vecteurs prévus, alors le circuit de génération des vecteurs de test est "strongly fault secure" pour la séquence <f1,f2,...,fn> s'il n'existe pas un tel k ou si un tel k existe mais le circuit en présence de la séquence <f1,f2,...,fk> produit une des situations a/, b/ ou c/.

La définition D36 est intéressante car il est possible pour des applications précises et des hypothèses de pannes précises, de concevoir des circuits de génération des vecteurs de test de façon ordinaire, c'est à dire sans utiliser des méthodes d'autotest (codage des sorties, etc...), et donc sans augmenter la taille du circuit. Cette possibilité n'a pas été examinée en détail.

Finalement, l'hypothèse concernant l'occurrence de défauts dans le système global, c'est à dire bloc fonctionnel et contrôleur spécifique, doit être donnée.

Hypothèses H3

- Après l'occurrence d'un défaut dans le bloc fonctionnel, il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée du bloc fonctionnel soient appliqués avant qu'un deuxième défaut survienne dans le bloc fonctionnel ou qu'un défaut survienne dans la partie "contrôleur" du contrôleur spécifique.

- Aprés l'occurrence d'un défaut dans la partie "contrôleur", il s'écoule un laps de temps suffisant pour que tous les vecteurs du code d'entrée B' du partie "contrôleur" et tous les vecteurs des ensembles TVB, TV∉B soient appliqués à la partie "contrôleur" (dans le cas de TV∉B = Y-B, il suffit que tous les vecteurs de l'ensemble TV∉B soient appliqués), avant qu'un deuxième défaut survienne dans la partie "contrôleur" ou qu'un défaut survienne dans la partie "générateur des vecteurs de test" ou dans le bloc fonctionnel.

- Après l'occurrence d'un défaut dans la partie "générateur des vecteurs de test", il s'écoule un laps de temps suffisant pour qu'une des trois situations a/, b/ ou c/ prevues dans la définition D36 soit produite, avant qu'un deuxième défaut survienne dans la "partie générateur des vecteurs de test" ou qu'un défaur survienne dans la partie "contrôleur".

En prenant en compte l'hypothèse H3; on peut vérifier qu'un système composé d'un bloc fonctionnel SFS et d'un contrôleur spécifique, composé d'une partie "contrôleur" SCD (définition D35) et d'une partie "générateur des vecteurs de test" SFS (définition D36), accomplit le "TSC goal".

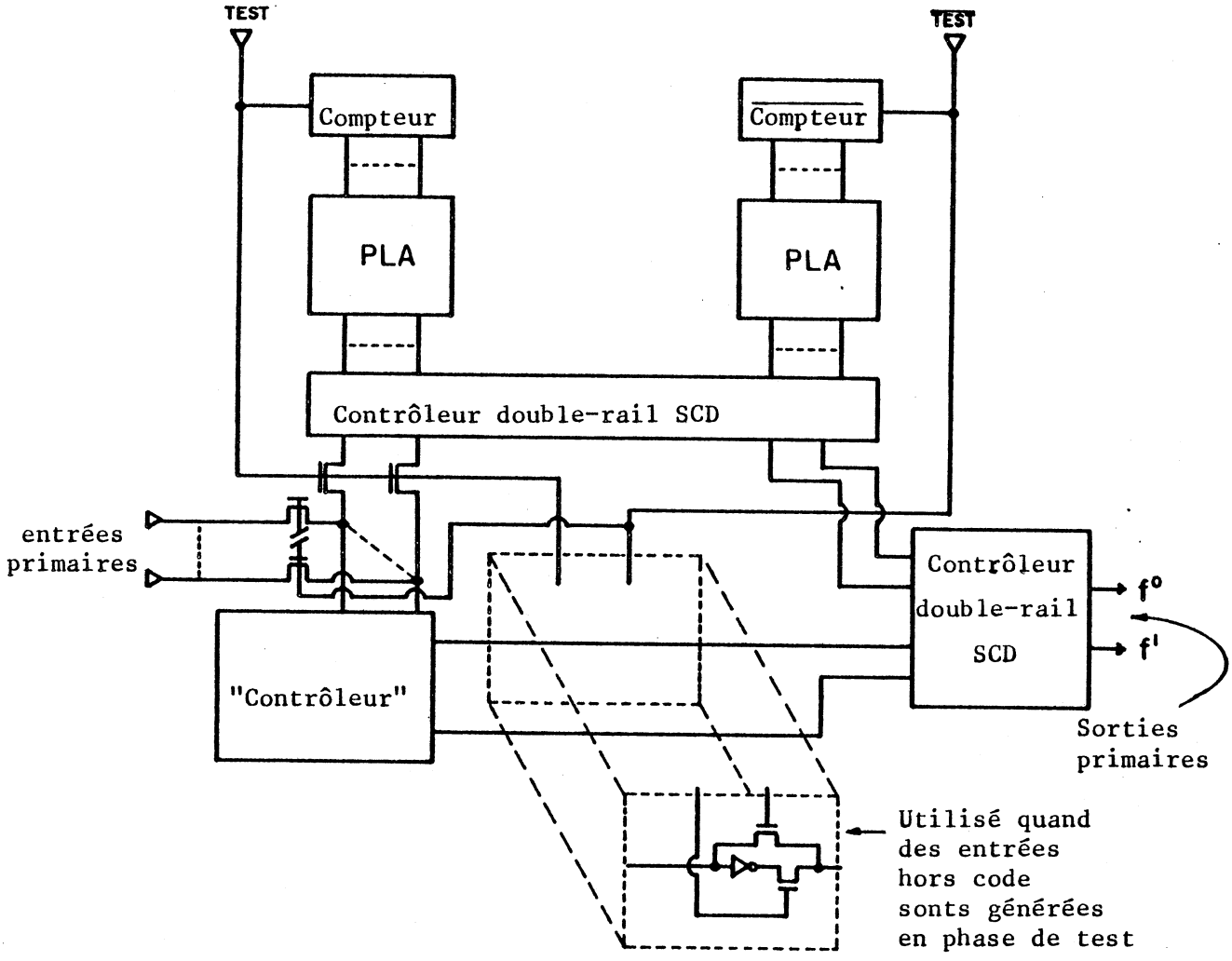


Figure 69 - Contrôleur spécifique.

CONCLUSION

La plus large classe de contrôleurs avec laquelle un système peut accomplir le "TSC goal" est définie dans ce chapître ; il s'agit des contrôleurs "Strongly code disjoint" (SCD).

La conception des contrôleurs double-rail SCD pour la classe I est discutée. Dans le cas où le contrôleur reçoit en fonctionnement normal tous les vecteurs du code d'entrée, la propriété SCD peut être assurée. Dans le cas où le contrôleur ne reçoit pas en fonctionnement normal tous les vecteurs du code d'entrée, la procédure donnée en [KHA 82] est très utile pour assurer la propriété SCD. Si cette procédure n'est pas suffisante, il reste à vérifier s'il existe des cellules de base pour ce contrôleur pour lesquelles les quatre configurations d'entrée ne sont pas nécessaires.

La conception des contrôleurs pour des codes systématiques est aussi discutée. On a montré que pour les codes séparables tous les dessins du générateur des bits de contrôle sont valables et que la condition de non redondance exigée pour avoir un contrôleur TSC (type I checker [ASH 76]) n'est pas nécessaire pour les contrôleurs SCD.

Pour le contrôleur double-rail de la figure 68, on peut utiliser l'analyse donnée pour les contrôleurs double-rail. En particulier pour les codes systématiques non complets, la procédure donnée dans [KHA 82] doit être utilisée pour la conception de la partie "contrôleur double-rail".

Les contrôleurs pour les codes de parité ne sont pas discutés mais les résultats de cette étude peuvent être utilisés pour la conception de tels contrôleurs. Pour un code de parité séparable, n'importe quel dessin du contrôleur est valable. Pour un code de parité non séparable, la procédure donnée dans [KHA 82] doit être utilisée pour assurer la propriété SCD de chaque cellule du

contrôleur.

Finalement, une proposition des contrôleurs spécifique est faite pour assurer la protection des contrôleurs en utilisant la génération des vecteurs de test. Ces contrôleurs peuvent être utilisés dans le cas des contrôleurs des codes standards qui ne reçoivent pas en fonctionnement normal un ensemble des vecteurs d'entrée suffisant, ou dans le cas des contrôleurs des codes arbitraires pour lesquels la propriété SCD n'est pas assurée même s'ils reçoivent tous les vecteurs du code d'entrée.

REFERENCES

[AND 71] ANDERSON D.A.
Design of self-checking digital networks using coding techniques.
Coordinated Science laboratory, report R/527
University of Illinois, Urbana, September 1971.

[ASH 76] ASHJAEE J.M., REDDY S.M.
On totally self-checking checkers for separable codes.
The 6th Annual conf. on fault-tolerant comp., Pittsburgh, June 21/23, 1976.
Digest of papers, New-York, IEEE, 1976, pp. 151/156.

[BAI 83] BAILLE G., BERGHER L., COURTOIS B.
LAURENT J., RUBAT DU MERAC C.,
Testing for failure analysis: new tools and new methods.
Fault tolerant comp. symp., Milano, June 1983, Italy.

[BEH 82] BEH C.C. ET AL.
Do stuck models affect manufacturing defects?
1982 Test conference, Philadelphia, USA, November 1982.

[BOS 82] BOSE A.K. ET AL.
A fault simulator for MOS LSI circuits
19th Design automation conf., Las Vegas, June 82, USA.

[BRE 76] BREUER M.A., FRIEDMAN A.D.
Diagnosis and reliable design of digital systems
Pitman Pub. Ltd., 1976.

[CAR 68] CARTER W.C., SCHNEIDER P.R.
Design of dynamically checked computers
IFIP Congres, Edinburgh 1968, Inf. Proc. 68,
Amsterdam, North Holland, 1969, pp. 878/883.

[CHI 82] CHIANG K.W.
Test generation for MOS complex gate networks
12th Fault tolerant computing symposium
Santa Monica, June 82, USA.

[COU 81] COURTOIS B.
Failure mechanisms, fault hypotheses and analytical testing of LSI-NMOS (HMOS) circuits
VLSI 81, University of Edinburgh, August 18/21 1981
United Kingdom, Academic Press.

[CRO 78] CROUZET Y.
Conception de circuits à large échelle d'intégration totalement autotestables
Thèse de docteur-ingénieur, Toulouse, novembre 1978.

[DON 81] DONG H., MCCLUSKEY E.J.
Matrix representation of PLAs and an application to characterize errors
CRC Technical report n.81-11, Standford University, September 1981.

[DON 82] DONG H.
Modified BERGER codes for detection of unidirectional errors
Proc. of 12th Fault tolerant comp. systems,
Santa Monica, June 1982.

[ELZ 79] ELZIQ Y.M.
Testing of MOS combinatorial networks. A procedure for efficient fault simulation and test generation
16th Design automation conf., June 1979, USA.

[ELZ 81] ELZIQ Y.M.
Automatic test generation for stuck-open fault in CMOS LSI
18th Design automation conf., June 1981, USA.

[FRI 67] FRIEDMAN A.D.
Fault detection in redundant circuits
IEEE Trans. on comp., February 1967.

[GAL 80] GALIAY J., CROUZET Y., VERGNIAULT M.
Physical versus logical fault models MOS LSI circuits: impact on their testability
IEEE Trans. on comp., vol. C-29, n.6, June 1980.

[GON 82] GONCALVES N.F., DEMAN H.
NP-CMOS: a racefree dynamic CMOS technique for pipelined logic structures
ESSCIRC 1982.

[HAR 65] HARRISON M.A.
Introduction to switching and automate theory
New-York, Mc Graw Hill, 1965.

[HLA 76] HLAVICKA J., KOTTEK E.
Fault model for TTL circuits
Digital processes, 2, 1976.

[JAN 82] JANSCH I., COURTOIS B.
Design of checkers based on analytical fault hypothesis (preliminary report)
IMAG, University of Grenoble, report RR 379
March 1983.

[KHA 82] KHABAZ J.P., MCCLUSKEY E.J.
Self testing embedded parity checkers exhaustive XOR gate testing
Standford University, Center for reliable comp., June 1982 (CRC report n.82-10/CSL TN 207)

[KHA 82A] KHARBAZ J.
Self testing embedded parity trees
12th Annual Int. Symp. on fault tolerant comp.
Santa Monica, June 22/24 1982, Digest of papers,
New York , IEEE, 1982, pp. 109/116.

[KRA 82] KRAMBECK R.H. ET AL.
High speed compact circuits with CMOS
IEEE Journal of solid-state circuits, n.3, June 1982.

[KUN 72] KUNTZMANN J.
Théorie des réseaux graphes
Dunod 1972.

[MAG 73] MAGO G.
Monotone functions in sequential circuits
IEEE Trans. on comp., vol. C-22, n.10, October 1973.

[MAK 82] MAK G.P., ABRAHAM J.A., DAVIDSON E.S.
The design of PLAs with concurrent error detection
Proc. of 12th Fault tolerant comp. symp.,
Santa Monica, USA, June 1982.

[MAL 82] MALAIYA Y.K., SU S.Y.H.
A new fault model and testing technique
for CMOS devices
1982 Test conf., Philadelphia, USA, November 1982.

[NIC 83] NICOLAIDIS M., COURTOIS B.
Design of self-checking systems bases on
analytical fault hypotheses
IMAG, RR 353, March 1983.

[SMI 77] SMITH J.E., METZE G.
The design of totally self checking combinatorial circuits
Proc. of 7th Fault tolerant comp. symp., Los Angeles, USA, June 1977.

[SMI 78] SMITH J.E., METZE G.
Strongly fault secure logic networks
IEEE Trans. on comp., vol. C 27, n.6, June 1978.

[SON 81] SON K.R., PRADHAM D.K.
Completely self checking checkers in PLAs
1981 IEEE Test conf., 1981 Proceedings, Philadelphia, pp. 231/237.

[SUZ 73] SUZUKI Y. ET AL.
Clocked CMOS calculator circuitry
IEEE Journal of solid state circuits, n.6, December 1973.

[TSA 81] TSAO M., WILSON A., MCCARTY R., TSENG C.J., SIEWIOREK D.
C Fast: a fault tolerant and self testing microprocessor
Proc. of the CMU conf. on VLSI systems and computations, Pittsburgh, USA, October 1981.

[VIA 80] VIAUD J., DAVID R.
Sequentially self-checking circuits
Proc. of 10th Fault tolerant computing symp., Kyoto, Japan, October 1980.

[WAD 78] WADSACK R.L.
Fault modelling and logic simulation of CMOS and MOS integrated circuits
The BELL system technical journal, May/June 1978.

[WAK 74] WAKERLY J.F.
Partially self checking circuits and their use in performing logical operations
IEEE Trans. comp., New York, July 1974, pp.658/667.

DEUXIEME PARTIE

ETUDE D'UN MICROPROCESSEUR MC68000 AUTOTESTABLE

CHAPITRE I

EVALUATION D'UNE PARTIE OPERATIVE AUTOTESTABLE POUR LE MC 68000

RESUME

Ce chapître présente l'étude d'une partie opérative (PO) autotestable pour le microprocesseur MC 68000.

La présentation générale de la partie opérative est d'abord donnée, ensuite chaque bloc de cette partie est présenté et un bloc modifié est proposé pour assurer l'autotestabilité du circuit. Les circuits modifiés sont obtenus en utilisant les méthodes présentées Dans la 1er partie de cette etude; le "Totaly self checking goal" de la partie opérative est ainsi garanti pour une couverture de 100% des pannes de la classe C1 des hypothèses de pannes [COU 81].

L'augmentation globale de la PO ainsi obtenue est de l'ordre de 69%. L'augmentation en surface de toute la partie du MC 68000 contenant la partie opérative, les plots de données, les plots d'adresses et une zone de routage est de l'ordre de 45,5%.

SOMMAIRE

III - SCHEMA GENERAL ET EVALUATION DE LA SURFACE DE LA PARTIE OPERATIVE AUTOTESTABLE.

I - PRESENTATION GENERALE DE LA PARTIE OPERATIVE DU MC 68000

La partie opérative occupe un bloc rectangulaire dont la longueur détermine une des dimensions de la puce (figure 1 a, b, c). Elle est organisée autour de 2 bus complémentés, bus adresses et bus données, de 16 bits, segmentables en trois sections par des commutateurs commandés par la partie contrôle. Ceci permet d'effectuer des transferts et des opérations simultanément sur les trois morceaux de la partie opérative.

Cette partie opérative compte 48 registres de 16 bits, dont 36 accessibles à l'utilisateur.

Poids forts (16 bits)

- 8 registres données (mémoire)
- 7 registres adresses (mémoire)
- 1 registre temporaire donnée (mémoire)
- 1 registre temporaire adresse (mémoire)
- 1 pointeur de pile utilisateur (mémoire)
- 1 pointeur de pile superviseur (mémoire)
- 1 compteur ordinal (mémoire)
- 1 buffer de sortie d'adresses (registre)
- 1 registre de AUH (registre)

Poids faibles (16 bits)

- 8 registres données (mémoire)
- 7 registres adresses (mémoire)
- 1 registre temporaire donnée (mémoire)
- 1 registre temporaire adresse (mémoire)
- 1 pointeur de pile utilisateur (mémoire)
- 1 pointeur de pile superviseur (mémoire)
- 1 compteur ordinal (mémoire)
- 1 buffer de sortie d'adresses (registre)
- 1 registre d'AUL (registre)
- 1 buffer d'entrée de données (registre)
- 1 buffer de sortie de données (registre)
- 1 registre de l'ALU (registre)

1 registre Field Translator Unit (mémoire).

Les 8 registres données et les 7 registres adresses sont organisés en RAM.

Les opérateurs ALUE, PREN et DCR contiennent aussi des circuits de mémorisation.
La notation mémoire ou registre est utilisée pour distinguer la réalisation d'un point mémoire avec ou sans resistance,l'écriture dans le point mémoire est alors possible à partir d'un chemin unique ou de deux chemins complémentaires (figure 2).

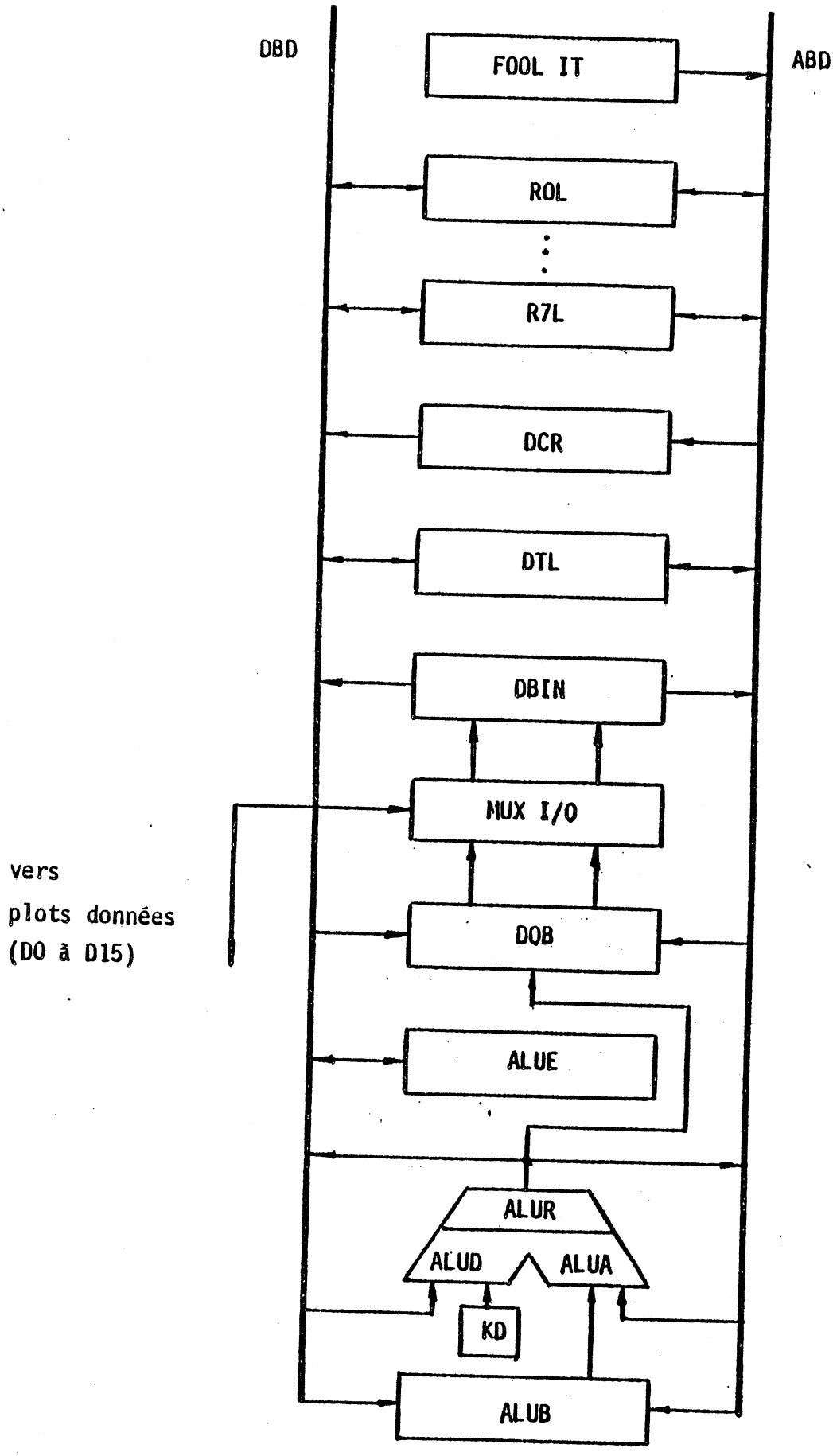


Figure 1a Schéma fonctionnel de la partie opérative du MC 68000 (suite)

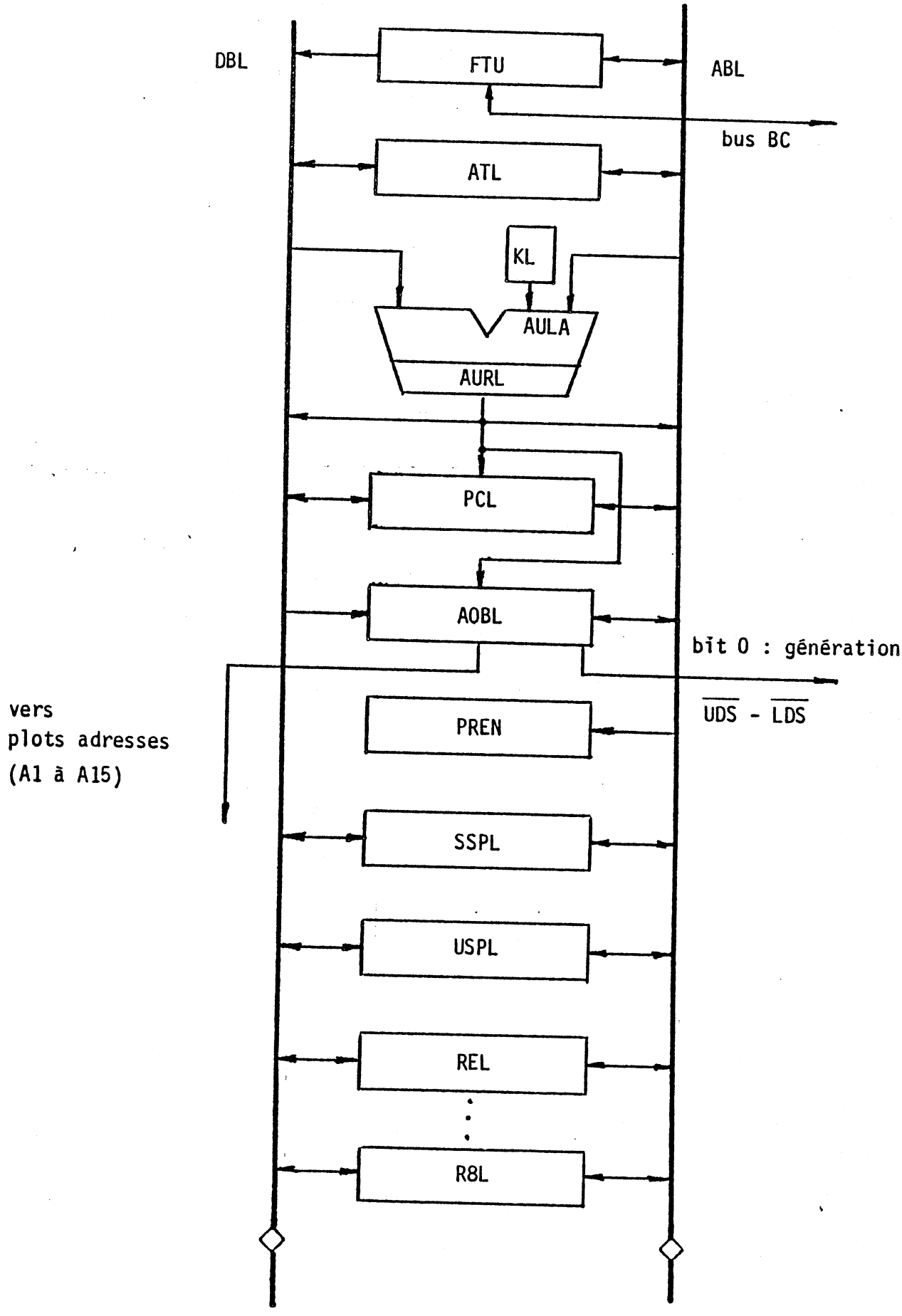


Figure 1b Schéma fonctionnel de la partie opérative du MC 68000 (suite)

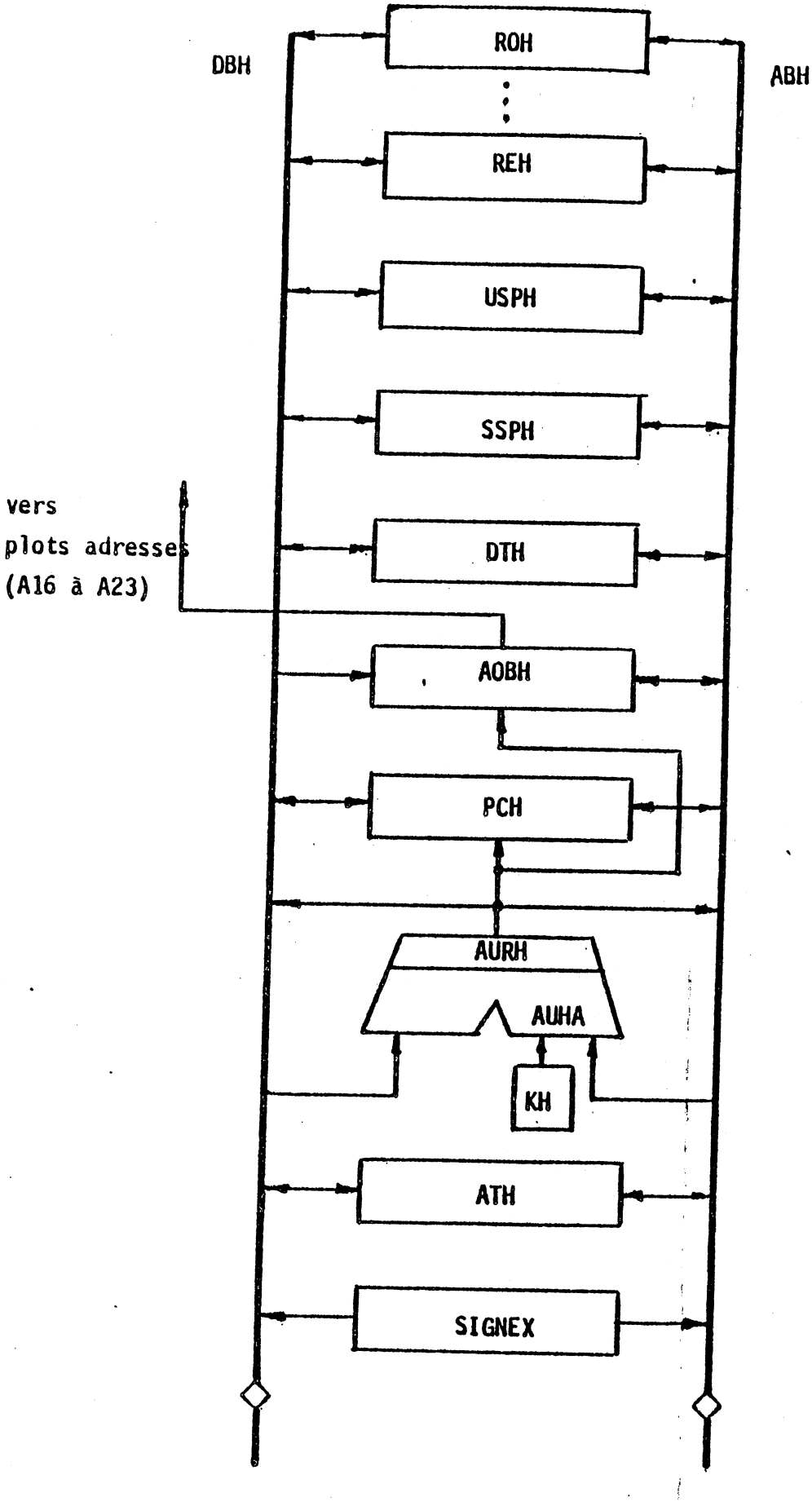
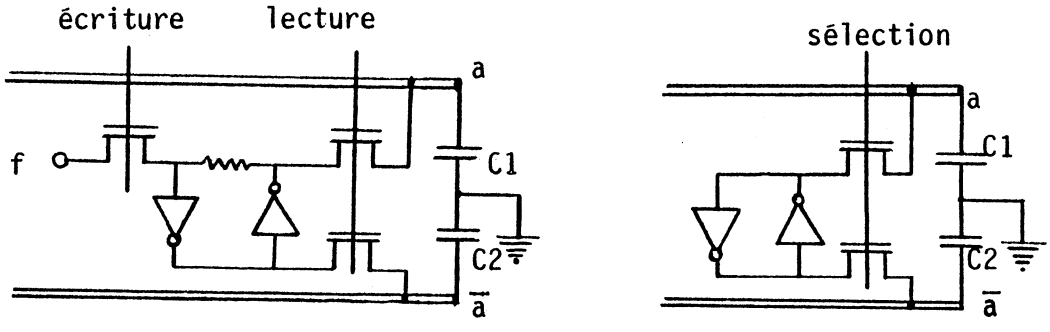


Figure 1c Schéma fonctionnel de la partie opérative du MC 68000



(a) branchement du registre (b) branchement de la mémoire

Figure 2

La partie opérative contient aussi 10 opérateurs nécessaires au traitement des opérations de son jeu d'instructions:

- AUh: unité arithmétique poids fort (adresses).
- AUl: unité arithmétique poids faible (adresses).
  (pour les incrémentations, décrémentations et additions d'adresses).
- ALU: unité arithmétique et logique (données) - Pour les opérations arithmétiques et logiques ainsi que pour les décalages.
- ALUE: extension de l'ALU - Pour les décalages sur 32 bits.
- SIGNEX: extension de signe - Pour extension du signe des 16 bits à 32 bits.
- PREN: registre encodeur de priorité - Pour coder en binaire le numéro de la ligne du bus qui est à "1", utilisé pour le traitement du masque de l'instruction MOVEM.
- FOOL-IT: pour écrire dans l'octet poids faible d'un registre sans modifier l'octet poids fort.
- DCR: pour encoder en un code 1 parmi 16 le numéro du bit à

manipuler qui est codé en 5 bits.
- MUX I/O: manipulation des opérandes de taille 1 octet en entrée ou en sortie.

II - ETUDE D'UNE PARTIE OPERATIVE AUTOTESTABLE DU MC 68000

Deux techniques seront utilisées pour réaliser la partie opérative autotestable:
- technique de détection des erreurs simples avec un code de parité,
- technique de détection des erreurs multiples avec duplications des blocs à tester.

La technique de détection des erreurs simples étant plus économique, elle est choisie si elle est utilisable pour l'autotest d'un bloc. Sinon la technique des erreurs multiples est utilisée.

En effet il est montré dans la section IV.2 de la 1er partie, que l'on peut utiliser la détection des erreurs unidirectionnelles pour réaliser n'importe quel circuit autotestable. Toutefois cette technique n' est pas utilisée, car lorsque la détection des erreurs simples n'est pas utilisable, la technique de détection des erreurs unidirectionnelles demande une augmentation de surface comparable à celle demandée pour la duplication, en raisons de la nature des circuits considerés, cette dernière est alors preferée.

En génèral, chaque commande de la partie opérative, descendant de la partie contrôle est utilisée pour commander les 16 bits de chaque bloc. Ainsi une erreur sur une commande peut provoquer des erreurs multiples sur le bloc commandé. Pour cette raison, toutes les commandes de blocs testés par un code de parité seront contrôlées après avoir traversée la partie opérative.

Pour les blocs dupliqués, un seul contrôleur est suffisant pour

l'ensemble bloc-commandes.

II.1. BUS ET REGISTRES

Les deux bus AB et DB traversant la P.O sont des bus différentiels (bifilaires); c'est à dire que deux lignes sont disponibles par bit, ces deux lignes portent respectivement la valeur directe et la valeur complémentée de l'information.

Chacun des bus AB et DB est divisé en trois parties interconnectées entre elles avec deux groupes de 32 commutateurs.

Chaque bus est pourvu de ses propres circuits de gestion:
- un circuit de précharge, qui précharge systématiquement sur la phase T4 les lignes du bus direct et du bus complémenté.
- un groupe de 16 amplificateurs qui amplifient systématiquement en T1+10 nanosecondes, T2, T3 les valeurs écrites au bus en T1.
- un groupe de 32 résistances servant à compenser les fuites du bus.

Le schéma des circuits de gestion du bus est donné dans la figure 3.

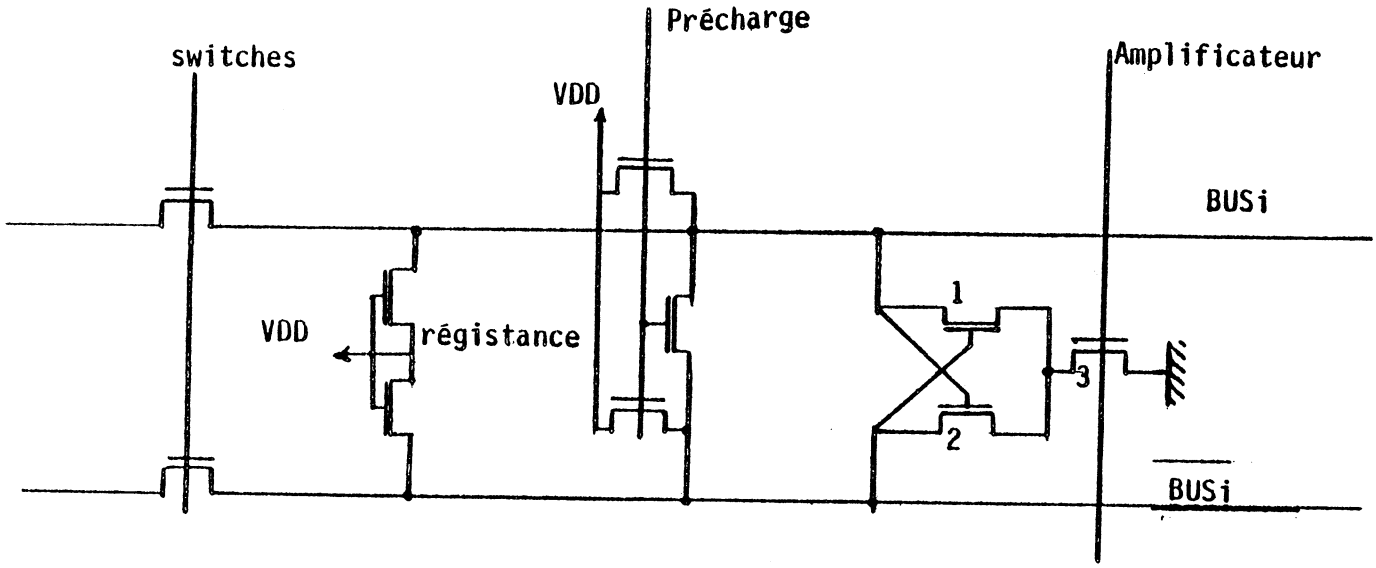


Figure 3 - Circuits de gestion du bus

Les deux schémas possibles de connexion d'un registre sur un bus bifilaire sont donnés en figure 2.

Le fonctionnement du bus est synchronisé par les quatre phases T1, T2, T3 et T4 d'horloge selon la figure 4.

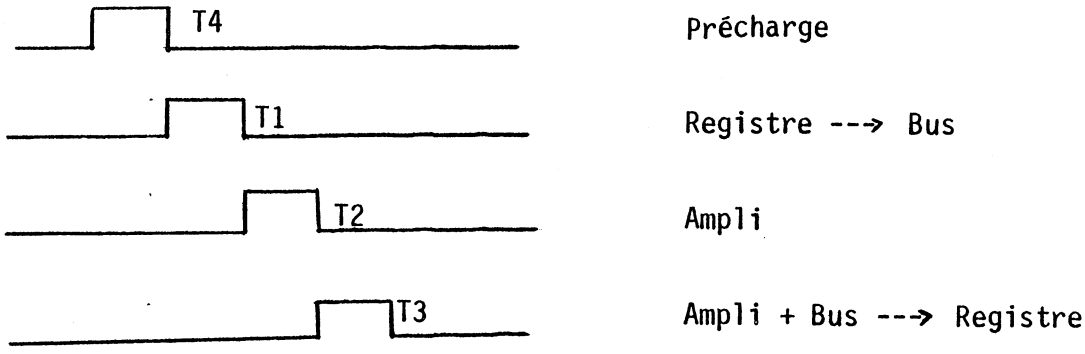


Figure 4 - Le cycle du transfert Registre ---> Bus ---> Registre

II.1.1. Autotest du bus et des registres

Les lignes de contrôle etant testées à leur sortie de la PO, après avoir commandé tous les bits, on peut vérifier que en utilisant une technique de détection des erreurs simples, le système bus-registres est Stongly Fault Secure (SFS) pour la classe I des hypothèses de pannes (section III.1 de la 1er partie).

Les règles de dessin des circuits SFS pour un code de detection des erreurs simples, données dans la section III.1 dela 1er partie sont examinées ci-dessous.

Il est évident que la règle R1 est vérifiée. La règle R2' est vérifiée par les lignes Vss; en effet chaque ligne Vss est utilisée par un seul bit, chaque ligne Vdd est utilisée par deux bits, la règle R2' n'est donc pas respectée par ces lignes mais la règle R2" est respectée car les contrôleurs "double rail" de l'ALU sont

alimentés par les extrémités de ces lignes. La règle R4 est vérifiée par les lignes Vss et par les lignes Vdd car celles-ci sont implantées alternativement.
Etant donné que tous les vecteurs possibles sont appliqués aux registres et aux bus pendant le fonctionnement normal du circuit, on peut vérifier qu'il ne peut pas exister de défauts du type B2 qui ne soit détectés.Par consequent, le test du bus et des registres sera fait en utilisant le code de parité.

L'existence des opérations de la taille d'un octet impose de coder séparement les deux octets du bus ABD, du bus DBD et des registres connectés sur ces bus. Le même codage sera aussi utilisé pour les bus ABL, DBL et les bus ABH, DBH, car ces bus sont parfois connectés avec les bus ABD et DBD.

L'existence du cycle de précharge implique l'utilisation du code de parité impaire, sinon on aurait la détection de défauts non existants, pendant le cycle de précharge (9 bits à 1).

Une dernière condition pour le fonctionnement correct du circuit est que les amplificateurs ne doivent pas être connectés sur les bus pendant les cycles où il n'y a pas de transfert, sinon, pendant de tels cycles, on aura: T4: précharge (BUS=5V, $\overline{\text{BUS}}$=5V) ; T1: rien ; T2, T3: connexion des amplificateurs (BUS:0,8v, $\overline{\text{BUS}}$:0,8v) ; cet état peut provoquer une détection des défauts non existant.
Cette condition sera respectée soit en générant la commande des amplificateurs seulement pendant les cycles où il y a un transfert sur le bus, soit en connectant systématiquement un registre sur le bus pendant les cycles sans transfert, cette dernière solution est facile à réaliser par le circuit de sélection des registres.

En fait, on ajoute sur les bus et sur les registres deux bits de parité, le bit 7' pour l'octet poids faible et le bit 15' pour l'octet poids fort.

Par suite de cette solution, toutes les données entrant sur le bus doivent être codées en parité impaire, ceci implique l'utilisation

des générateurs de parité à la sortie de quelques circuits.

Pour chacune des trois sections de la partie opérative , deux controleurs - un pour le bus AB et un pour le bus DB - seront utilisés.

On doit remarquer que le codage en parité garantit aussi l'autotest des circuits de précharge, des amplificateurs et des commutateurs du bus, étant donné que leur structure du point de vue propagation des erreurs est identique à celle des registres (structure bit slice).

Il est évident que cette solution est valable seulement si les commandes de tous ces circuits sont controlées après avoir commandé le circuit respectif comme il était noté à la section II.

II.1.2. Contrôleur du bus

Puisque en fonctionnement normal, le bus direct et le bus complémenté portent respectivement la valeur normale et la valeur complémentée de l' information, on admet la solution de connecter les controleurs sur le bus direct seulement.

Cette solution permet de controler les erreurs détectées par la technique choisie pour lesquelles les valeurs sur le bus sont normales, c'est à dire du type '0', '1' ou '1', '0', il peut également détecter des erreurs affectant le bus direct seulement qui donnent alors sur le bus des valeurs du type '0', '0' ou '1' '1', tandis que les erreurs du même type mais affectant le bus complémenté cette fois, ne sont pas détectées. C'est le cas par exemple des erreurs provoquées par une coupure du bus complémenté, par une panne de précharge ou d'un amplificateur du bus complémenté, etc... Malgré l'existence de ces erreurs indétectables, un tel système peut garantir le "totaly self checking goal", si on ne se fixe pas sur les erreurs de

l'information qui circule sur le bus, mais sur l'influence de ces erreurs à la destination de l'information.

Chaque information qui passe par le bus peut avoir comme destination soit un registre pour être stocké, soit un opérateur pour être traité, soit une sortie de la partie opérative.

Alors les conditions suivantes peuvent garantir le "totaly self checking goal" d'un tel système.

Pour chacun des opérateurs ou des sorties de partie opérative:

a/ soit les erreurs référencées précédemment seront détectées par le mécanisme du test de l'opérateur ou des sorties.

b/ soit la sortie ou l' operateur est connecté seulement avec le bus controlé, et dans notre cas avec le "bus direct".

Cette condition étant respectée, toutes les informations destinées aux opérateurs ou aux sorties de la partie opérative seront soit correctes, soit erronées, mais détectées.

Les informations erronées, non détectées, destinées à un registre, prendront une valeur normale, correcte ou erronée du type '0','1' ou '1', '0', après la déconnexion du registre par le bus.

Cette information, pour être utilisée, doit passer obligatoirement par le bus, et le cycle recommence.

Sur le diagramme ci-après il est facilement visible que le "totaly self checking goal" d'un tel système est garanti.

IC = information correcte
ED = erreur détectée
UPC = utilisation seulement de la partie correcte de l'information.

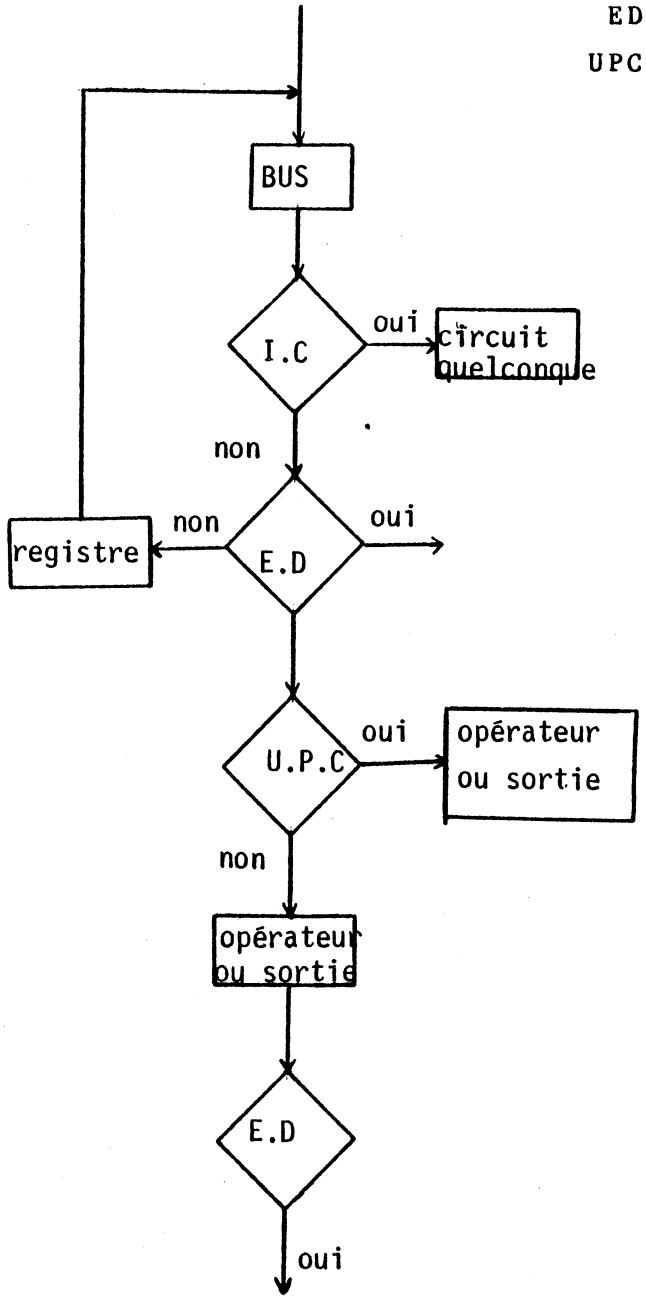


Figure 5

On doit maintenant remarquer que la nature des circuits concernés par les conditions proposées permet facilement le respect de ces conditions.

En general les opérateurs sont connectés en entrée soit avec le bus direct, soit avec le bus complémenté.
Dans le MC 68000, l'ALU, le DCR, le PREN, l'AUL et l'AUH sont connectés soit avec le bus direct, soit avec le bus complémenté, et seul l'ALUE est connecté avec les deux.

Les buffers de sortie disposent en général la résistance nécessaire pour être connectés avec un seul coté du bus.
Dans le MC 68000, DOB, AOBL, et AOBH sont tous les trois connectés au bus complémenté seulement. Donc pour ces circuits on peut appliquer facilement la condition b/ en les connectant avec le bus direct, sans pourtant augmenter sensiblement la surface du circuit. La figure 6 donne une telle modification pour le DOB.

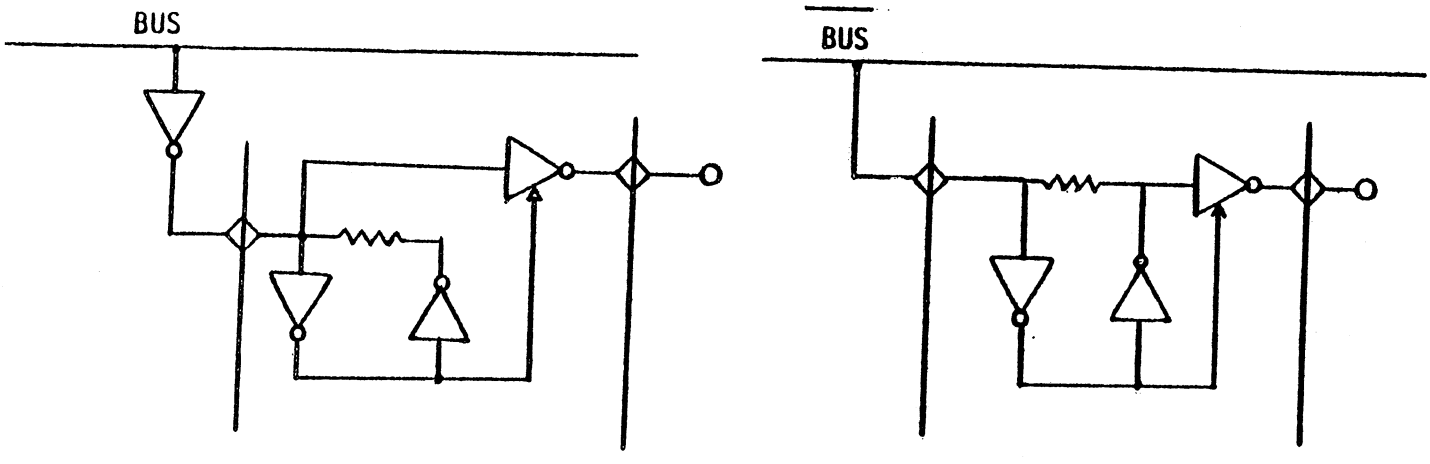


Figure 6 - DOB: (a) Version originale, (b) Version modifiée

La condition a/ peut aussi être appliquée facilement dans la

plupart des cas.
La structure "double rail" est utilisée dans la majorité des cas pour l'autotest des opérateurs et les connexions du circuit dual avec les deux cotés du bus sont inversées.
Par cette structure, les erreurs affectant l' un des deux bus (bus direct ou bus complementé) sont détectées.
D' autre part, le mécanisme de détection des pannes de buffers de sortie détecte ce type d' erreur; donc la condition a/ est respectée par les buffers de sortie.

La figure 7 donne la connexion des opérateurs et des buffers de sortie avec les bus directs et le bus complémenté des trois sections de la partie opérative du MC 68000.

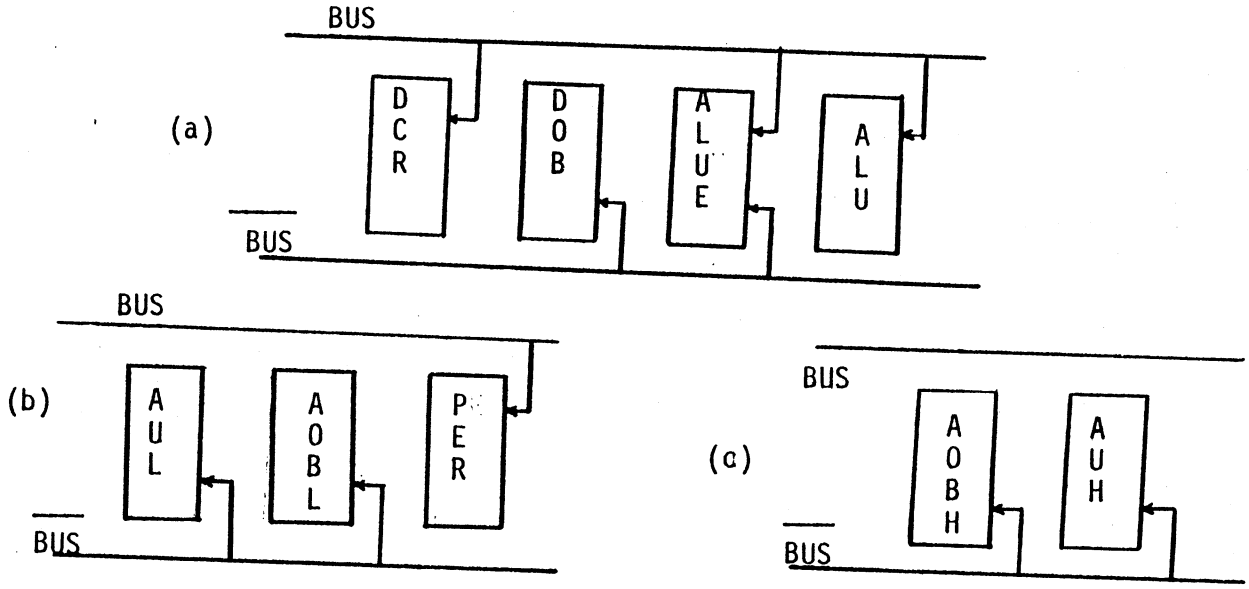


Figure 7 - Connexion des opérateurs sur le bus
a/ section droite
b/ section médiane
c/ section gauche.

On trouve alors que la condition b/ est respectée pour le PREN.
La condition b/ est respectée pour l'ALU,de plus, la structure "double rail" utilisée pour l'autotest de ces circuits détecte

aussi ce type d'erreurs donc la condition a/ est aussi respectée.

La condition a/ est respectée dans le cas de l'ALUE puisque la structure "double rail" est utilisée pour son autotest.

La condition b/ est respectée pour le circuit DCR.

La condition a/ est respectée pour les circuits AUH, AUL, AOBH, AOBL, DOB, car les erreurs considérées seront détectées par la structure "double rail" dans le cas de l'AUH et l'AUL et par le codage de parité dans le cas de l'AOBH, AOBL et DOB.

On doit remarquer que dans le cas d'utilisation d'autres mécanismes d' autotest, pour lesquels la condition a/ n'est pas respectée, on peut modifier facilement la connection des circuits avec le bus, de façon à ce que la condition b/ soit respectée. L'exemple d'une telle modification est donnée à la figure 6 pour le circuit DOB.

Dans cette analyse l'une de deux conditions a/ ou b/ doit etre respectée par les opérateurs, car une erreur non détectée sera masquée après le traitement de l'information par un operateur. Mais les circuits SIGNEXT et FOOL IT considérés en général comme opérateurs ne sont pas pris en compte , car le SIGNEXT est utilisé pour transférer la valeur du bit 15 des bus aux bits 16 à 31 et le FOOL IT est utilisé pour écrire dans l'octet poids faible d'un registre.Ils ne masquent donc pas les erreurs considérées.

La solution envisagée permet donc de diminuer le nombre des contrôleurs du bus tout en garantissant le "totaly self checking goal" du système.

Dans la partie opérative du MC 68000 deux bus sont disposés, chacun étant coupé en trois parties.

Cette solution permet de gagner six controleurs de parité d'un nombre de 648 transistors au total ou une surface de 1mm2 environ.

II.2. SECTION DROITE DE LA PARTIE OPERATIVE (Traitement des données)

Le schéma fonctionnel de cette section est donné par la figure 1c. Les mécanismes du test de cette section sont les suivants:
- ALU et tous les circuits de son environnement sont doublés (structure "double rail").
- ALUE doublé (structure "double rail").
- DOB, MUX I/O, DBIN, code de parité.
- DCR, code de parité.
- FOOL IT, code de parité.
- DTL, R0LD,...,R7LD, code de parité.
- BUS ABD, BUS DBD, amplis des bus, précharge des bus, commutateurs, code de parité.

Par cette solution, la taille des registres et des bus devient 18 bits: 16 bits d'information et 2 bits de parité (1 par octet).

Les circuits de contrôle et de génération de parité sont les suivants:
- un controleur "double rail" 16 bits est utilisé pour le test d'ALU et ALUE.
- deux controleurs de parité 18 bits sont utilisés pour le contrôle des circuits testés par la parité, l'un est connecté à l'extrémité droite du bus ABD , l'autre à l'extrémité droite du bus DBD.
- un générateur de deux bits de parité est connecté au plot des données pour coder les données entrant dans la partie opérative.

Deux bits de parité sont générés aux sorties de tous les circuits modifiant l'information, donc:
- un générateur de deux bits de parité est connecté aux 16 sorties de l'ALU.
- deux cellules de calcul de bit de parité sont ajoutées à l'ALUE.

II.2.1. Le mécanisme des entrées/sorties des données

Ce bloc, présenté à la figure 1b est constitué par les circuits:
- MUX I/O - multiplexeur des entrées-sorties,
- DBIN - Data Input Buffer,
- DOB - Data Output Buffer.

a/ DOB:
Le buffer de sortie des données est présenté dans la figure 8.

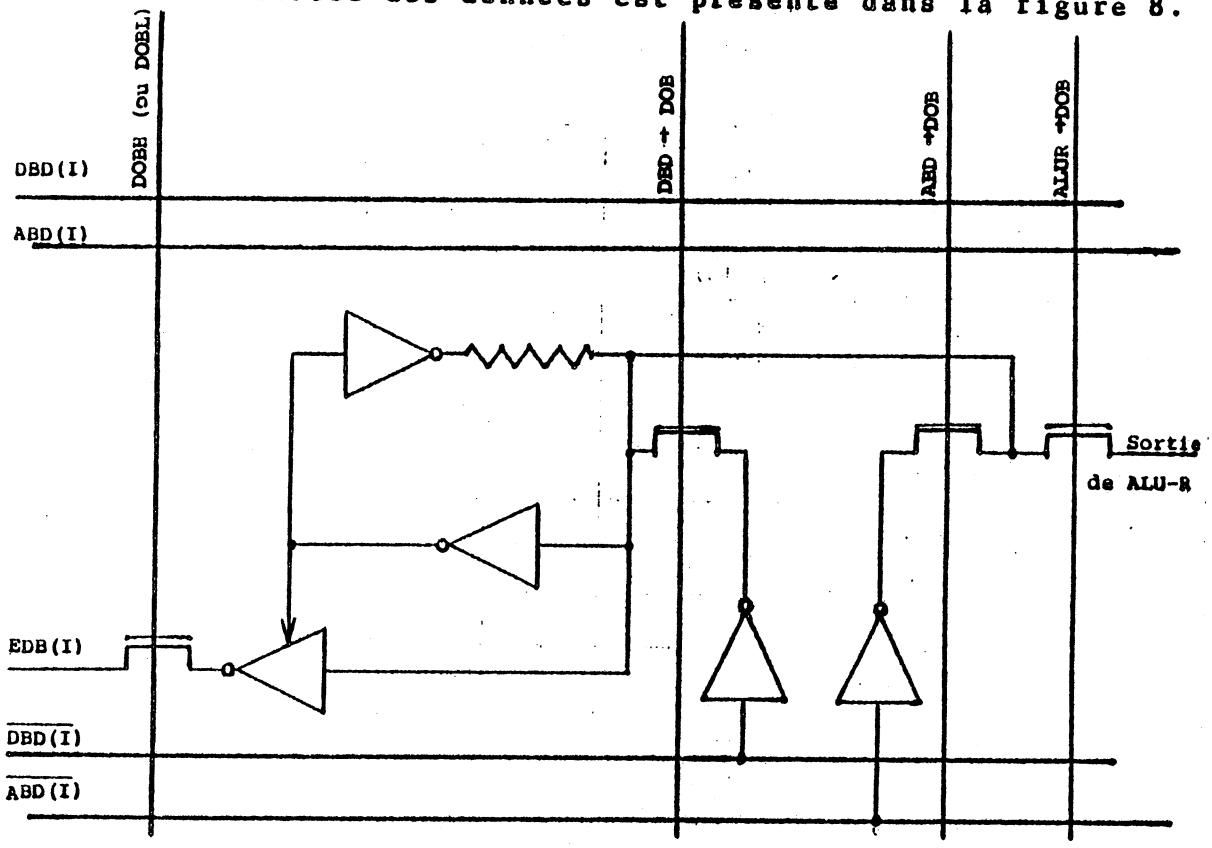


Figure 8- Buffer de Sortie de Données (DOB).

Pendant un cycle d'écriture mémoire, les commandes DOBL et DOBH permettent la sortie aux plots soit du contenu de l'octet poids faible de DOB dans le cas d'opérande de taille octet, soit de tous les 16 bits de DOB dans le cas d'un opérande de taille mot.

Les commandes ABD->DOB, DBD->DOB ou ALU->DOB permettent l'écriture en DOB du contenu du bus ABD, du bus DBD ou d'ALUR respectivement.

b/ DBIN:

Le buffer d'entrée des données est présenté dans la figure 9.

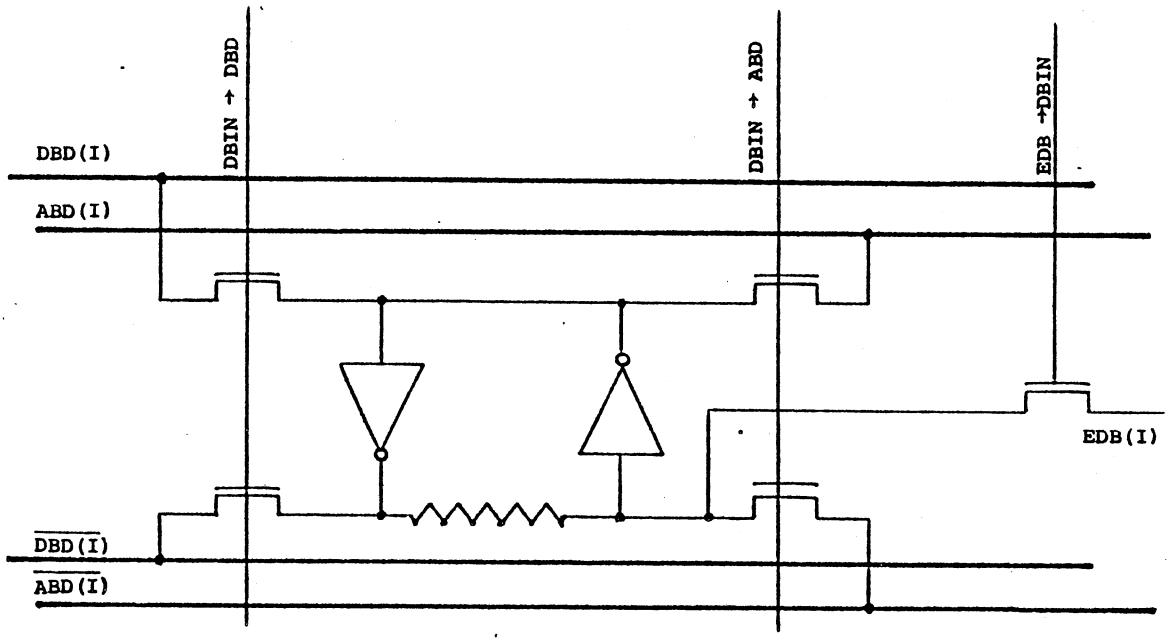


Figure 9 - Buffer d'Entrée de Données (DBIN)

Les commandes DBIN->ABD et DBIN->DBD permettent l'écriture du contenu de DBIN sur le bus ABD ou le bus DBD respectivement.

Les commandes EDB->DBINL et EDB->DBINH permettent d'écrire le contenu de EDB poids faible sur DOB poids faible ou du EDB poids fort sur DBIN poids fort, ou le contenu du EDB (16 bits) sur DBIN (16 bits).

Les commandes I/OH et I/OL (figure 1.b) sont activées seulement en cycle de lecture et elles commandent l'entrée dans EDB d'opérandes présents sur les plots.

Pour les opérandes de taille octet, seule la commande I/OL est activée pour une adresse de parité paire ou seule la commande I/OH est activée pour une adresse de parité impaire.

c/ MUX I/O:
Ce circuit fait la commutation entre les poids faibles et les poids forts du bus EDB (figure 10).

Il permet la manipulation d'opérandes de taille octet en recopiant l'octet présent sur les lignes EDB0 à EDB7 aux lignes EDB8 à EDBF et l'inverse.

Ce mécanisme rend les opérations d'entrées/sorties indépendantes de la parité de l'adresse de l'opérande en mémoire lorsque ce dernier est de taille octet.

II.2.1.1. Mécanisme des entrées/sorties autotestable

La détection des erreurs simples sera utilisée pour l'autotest du circuit, selon la méthode développée dans la section III.1 de la 1er partie. La règle R1 est respectée sous la condition que les lignes de contrôle du circuit soient testées après avoir commandé les MOS du circuit.

La règle R2 est respectée pour les mêmes raisons données à l'analyse du système "registres - bus".

Etant donné que tous les vecteurs possibles sont appliqués au circuit pendant son fonctionnement normal , on peut démontrer que tous les défauts du type B2 sont détectés.

La règle R4 est respectée car les Vdd et Vss power lines sont implantées alternativement.

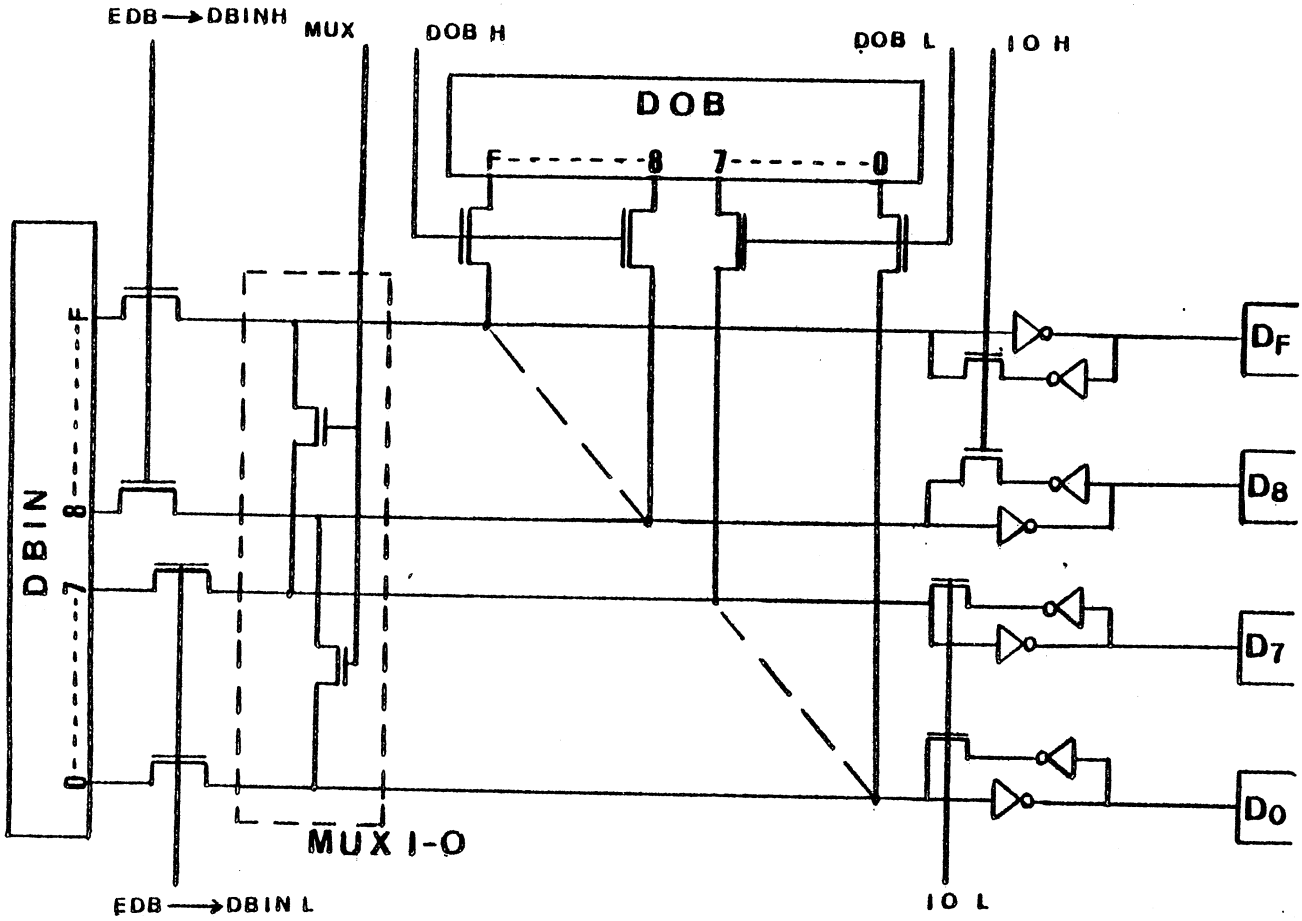
Le circuit augmenté est présenté à la figure 11.

Deux bits de parité (DOB7', DOBF' et DBIN7', DBINF') sont ajoutés sur chacun des buffers DOB et DBIN. Les deux octets du bus des

données sont codés séparément car le transfert des données se fait soit en mot soit en octet, donc deux plots de parité (D7' et DF') sont ajoutés.

Un controleur de parité est ajouté, il contrôle les données sortant et entrant. Ce controleur n'est pas connecté directement aux plots mais à la sortie des buffers d'entrée, pour avoir en lecture "buffering" et protection. En lecture, il détecte les erreurs des données entrant ainsi que les defauts des plots et des buffers d'entrée. En écriture, il détecte les erreurs des données sortant de la partie opérative ainsi que les défauts des DOB, MUX I/O, bus EDB, buffers de sortie et buffers d'entrée.

Figure 10 - Le méchanisme des Entrées-Sorties



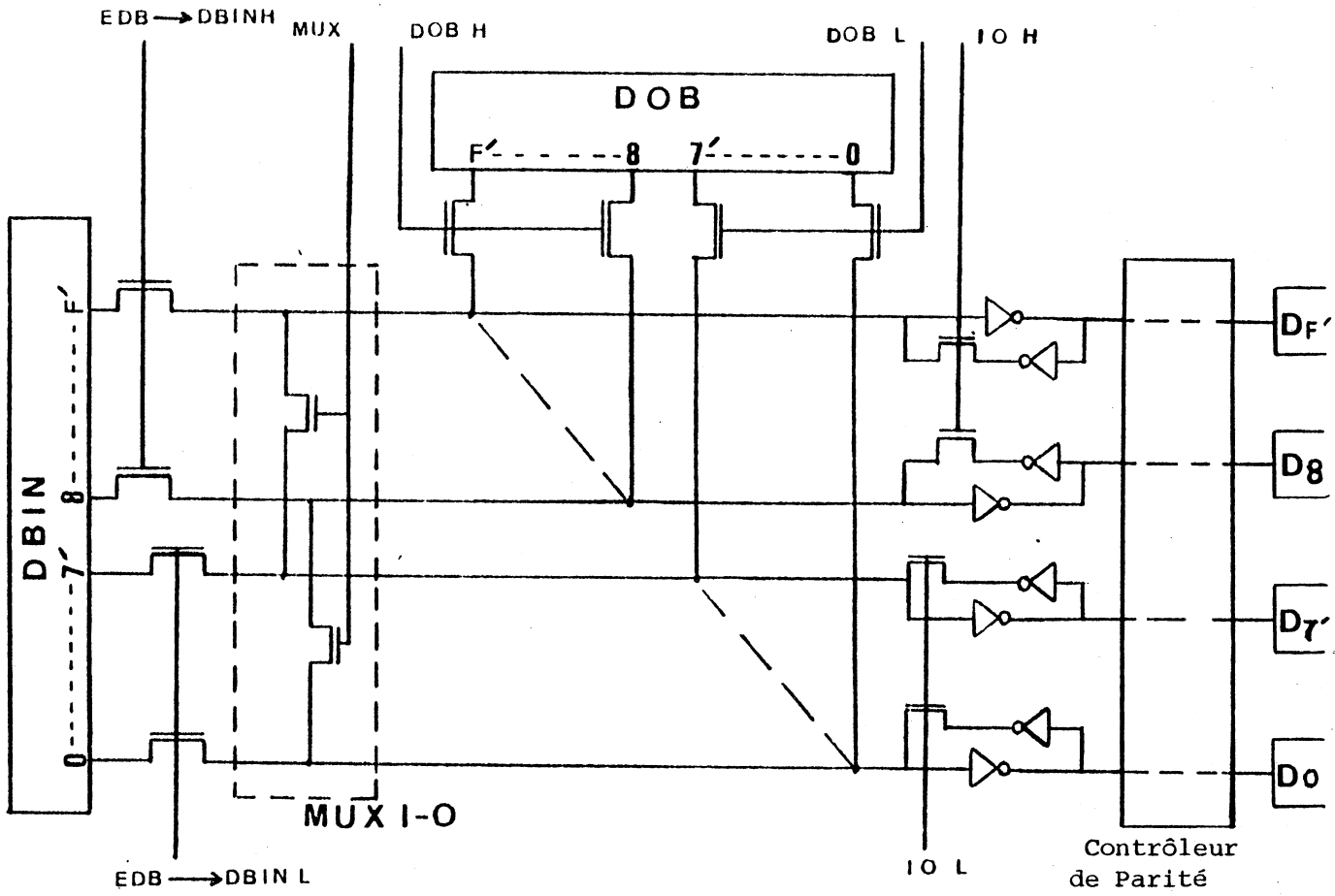


Figure 11 - Le méchanisme des Entrées-Sorties Autotestable

II.2.2. Le système de l'unité arithmétique et logique (ALU)

L'ALU est le circuit principal pour les opérations arithmétiques et logiques. Il s'agit d'un seul circuit à 16 pas identiques, qui permet d'exécuter des opérations arithmétiques, logiques et décalages, en fonction des 4 commandes générées par la Partie contrôle.

Il fait partie d'un système de circuits qui lui permet de réaliser un grand nombre de fonctions.

Ces circuits sont:

- Le circuit "byte correction logique"
Il effectue la correction des résultats de l'ALU dans le cas des opérations arithmétiques décimales.

- Le circuit d'acquisition des opérandes
Ce circuit permet la connexion de l'ALU avec les sources suivantes des opérandes: bus ABD, bus DBD, "byte correction logique", constante k, ALU buffer.
Il sert aussi à compléter un opérande en entrée, pour effectuer un décalage à droite, pour l'extension de signe du bit 7 aux 8 bits poids fort.

- Le buffer de l'ALU (ALU-B)
Il est utilisé pour le stockage provisoire des données si nécessaire, avant qu'elles ne soient utilisées par l'ALU. Par exemple dans le cas de multiplication, division, etc...

- Le registre d'ALU (ALU-R)
Dans ce registre sont stockés systématiquement les résultats de l'ALU avant d'être transférés aux bus ou au DOB.

- Deux circuits pour la génération des BV et WV
et

- Deux circuits pour la génération des LBZ, HBZ.

- Quatre bascules pour le stockage provisoire des BC, BV, WC, WV.

Un dernier circuit occupant une surface importante fait partie de ce système:

- L'extension de l'ALU (ALUE)
Ce circuit, couplé avec l'ALU, permet d'effectuer le décalage sur 32 bits. Ceci permet entre autres, l'augmentation des performances du circuit en multiplication et division.

II.2.2.1. Systéme de l'ALU autotestable

Tout le système de l'ALU, c'est à dire ALU, ALU-B, circuit d'acquisition des opérandes, circuit "byte correction logique", ALUR, circuit de génération de BV, WV, LBZ, HBZ, bascules de BC, BV, WV, WC, ALUE sera doublé.

La structure "double rail" est envisagée pour cette duplication dont la réalisation est basée sur l'analyse donnée dans le paragraphe III.3.3.1.2 (1er partie).

On doit noter que la structure bifilaire est bien adaptable pour la structure "double rail" car les valeurs directes et complémentaires des données sont disponibles.

a/ Schéma d'un pas de l'ALU

Le schéma d'un pas d'ALU est présenté dans la figure 12a, les quatre commandes $\overline{COR}$, CAND, $\overline{CXOR}$, SL servent à commander les opérations arithmétiques et logiques et le décalage à gauche.

Les deux entrées A et D peuvent être connectées avec différentes

sources des opérandes selon les commandes du circuit d'acquisition des opérandes.

Il est important de remarquer que le circuit de génération de $\overline{Ci}$ est un circuit de MOS série, le fonctionnement correct de ce circuit implique que l'état d'un étage ne peut pas influencer l'état de l'étage précédent. Ceci est assuré par les valeurs possibles des commandes $\overline{COR}$, CAND, $\overline{CXOR}$ et SL qui garantissent que l'on n'a jamais Fi=1, Ei=1 et $\overline{Ci-1}$=1.

Sur la figure 12b est présenté le circuit dual de la figure 12a qui est obtenu en appliquant les règles logiques de dualité. Mais on peut remarquer que son fonctionnement n'est pas correct puisque Ci-1 est forcé à '0' si $\overline{Ei}$=0.

Pour éviter cette anomalie, un transistor commandé par Ei était ajouté sur la figure 12c qui réalise cette fois le foncionnement correct.

Mais cette solution n'était pas admise non plus, car le nombre de transistors qui intervient pour la propagation de retenue est doublé, donc le retard de propagation est augmenté.

En calculant Ci sur la figure 12 a, nous avons:

$$Ci = Ei\ Ci-1 + Fi$$

et puisque on n'a jamais Fi=1, Ei=1 et $\overline{Ci-1}$=1

on trouve $Fi=Fi(\overline{Ei} + Ci-1)$

alors

$$Ci = Ei\ Ci-1 + Fi\ \overline{Ei} + Fi\ Ci-1 \Rightarrow Ci = \overline{\overline{Fi}.\overline{Ei} + Ei.\overline{Ci-1}}$$

Par cette expression on obtient le circuit de la figure 12d qui réalise la fonction dual du circuit 12a, sans pour autant augmenter le temps de propagation de la retenue.

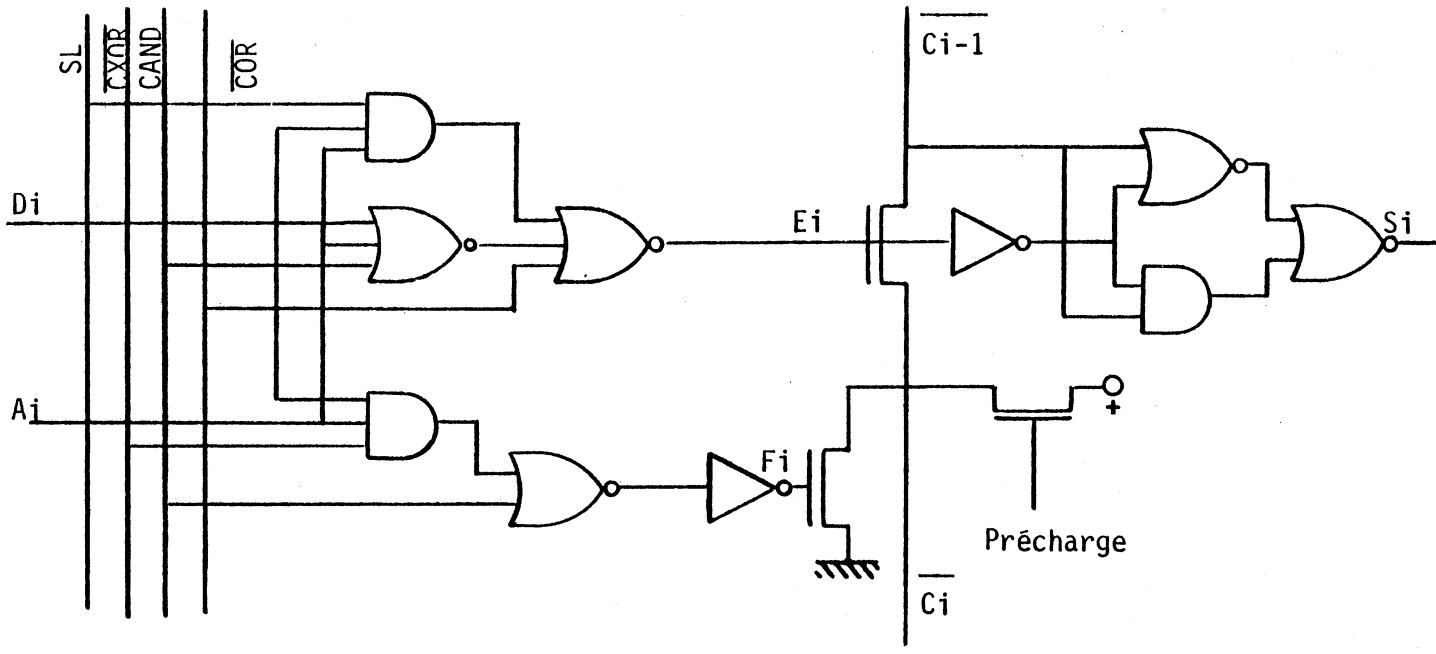


Figure 12a- ALU originale

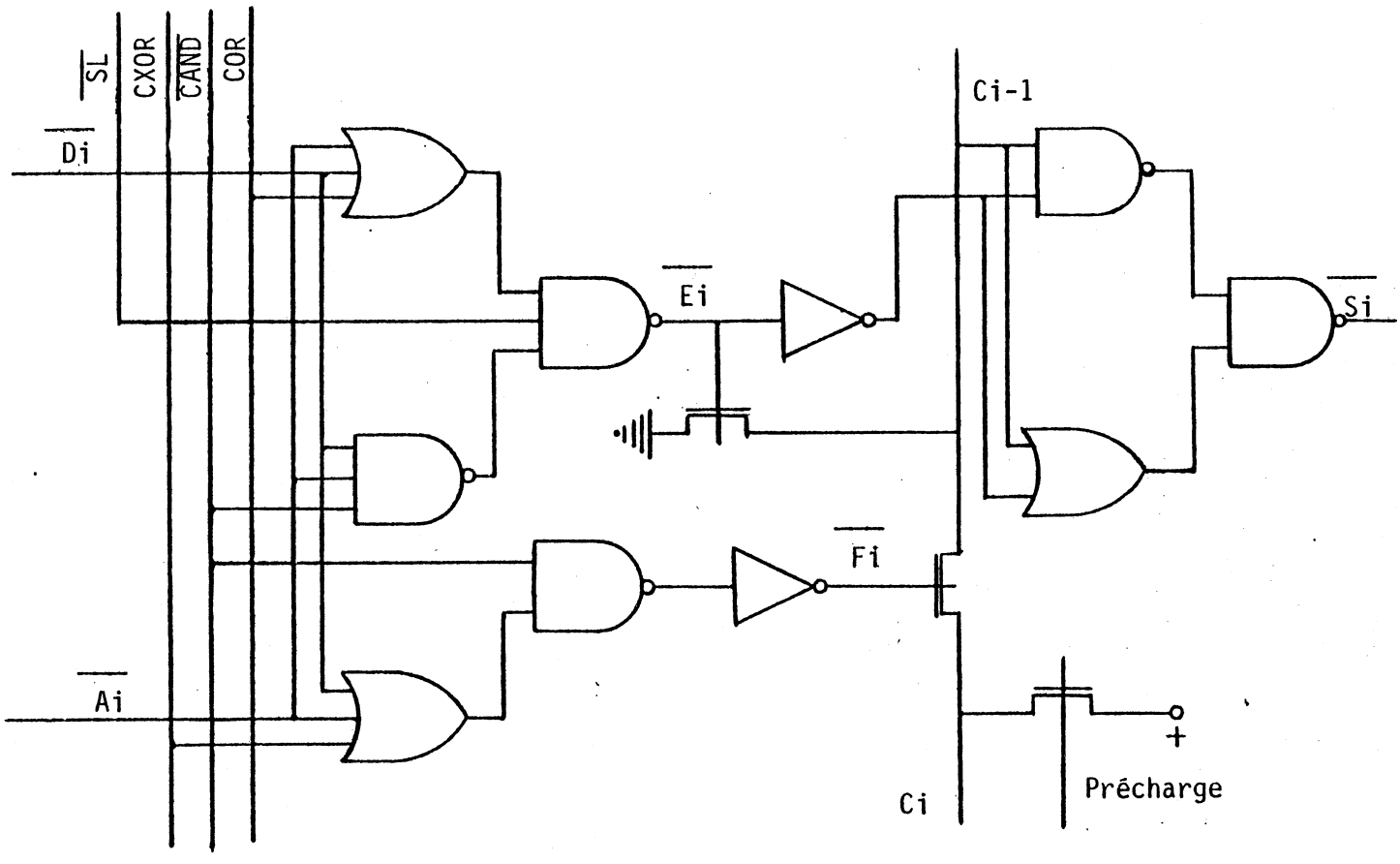


Figure 12b- ALU dual, fonctionnement incorrect.

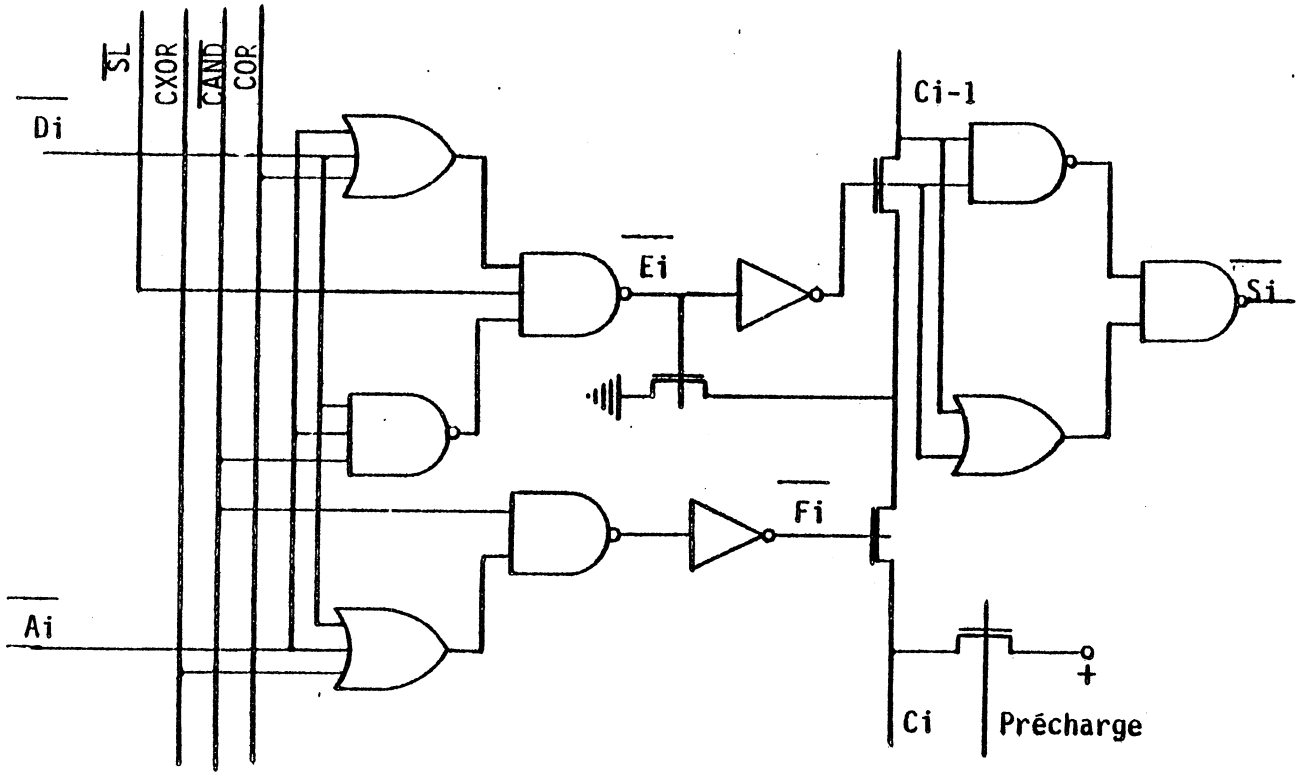


Figure 12c- ALU dual, fonctionnement correct, propagation retardée de la retenue

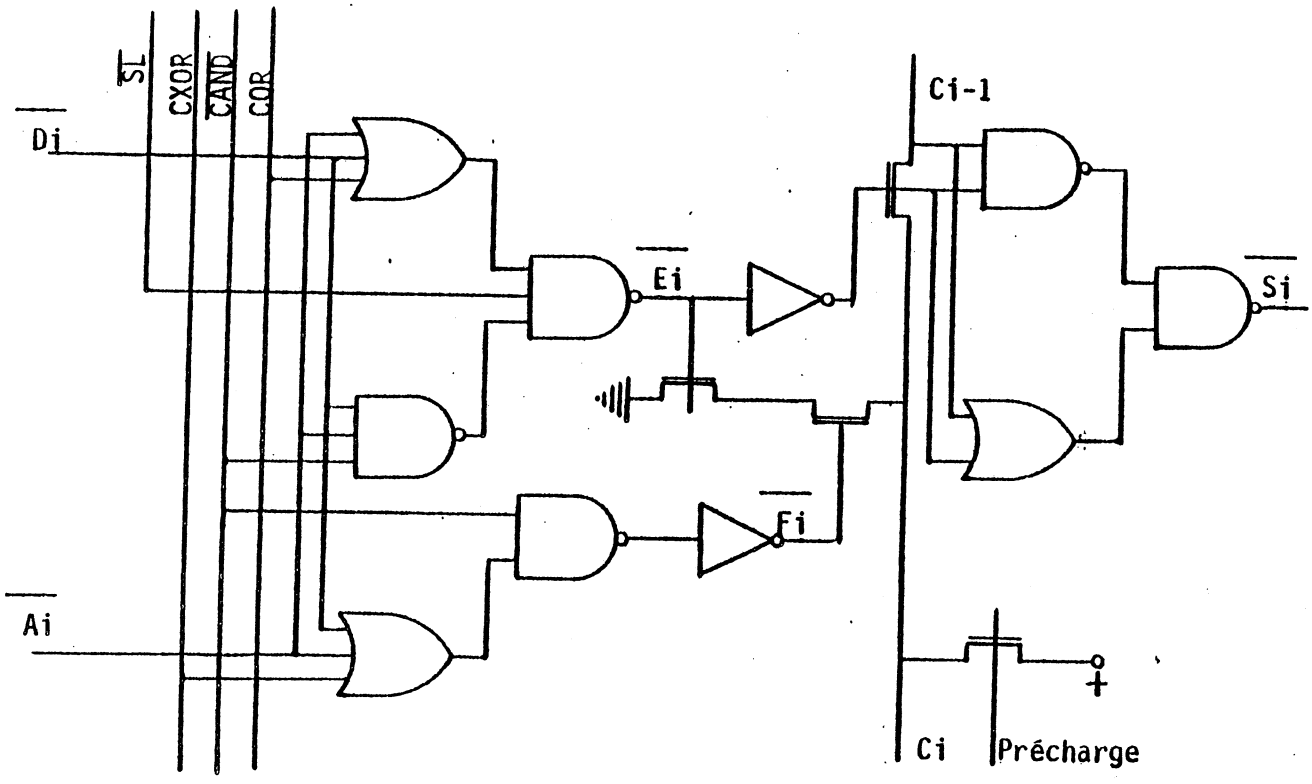


Figure 12d- ALU dual, fonctionnement correct, vitesse de propagation normale de la retenue

b/ Circuit d'anticipation de la retenue

Ce circuit (figure 13) est utilisé pour diminuer le temps de propagation de la retenue.

Il permet la propagation directe de la retenue $\overline{C3out}$ jusqu'au $\overline{C8in}$ ou même jusqu'au $\overline{C12in}$ et aussi la propagation du $\overline{C7out}$ jusqu'au C12in sans passer par les étages intermédiaires, en diminuant ainsi le temps de propagation dans les cas les plus défavorables.

Circuit augmenté:

Dans le système d'ALU complémentaire, un circuit dual pour l'anticipation de la retenue ne peut pas être utilisé, pour des raisons de vitesse (vitesse critique depassée). Alors un circuit identique au circuit direct sera utilisé.

Le couplage de ce circuit, qui n'est pas complémentaire au circuit direct, dans le système de l'ALU complémentaire se fait en tenant compte que: dans l'ALU complémentaire aussi bien que dans l'ALU direct, la propagation de la retenue de l'étage i-1 à l'étage i, est permise quand Ei=1, dans le cas de l'ALU complémentaire, c'est la valeur Ci-1 qui sera propagée tandis que dans l'ALU direct c'est la valeur $\overline{Ci-1}$. Le circuit d'anticipation de l'ALU complémentaire est présenté dans la figure 14.

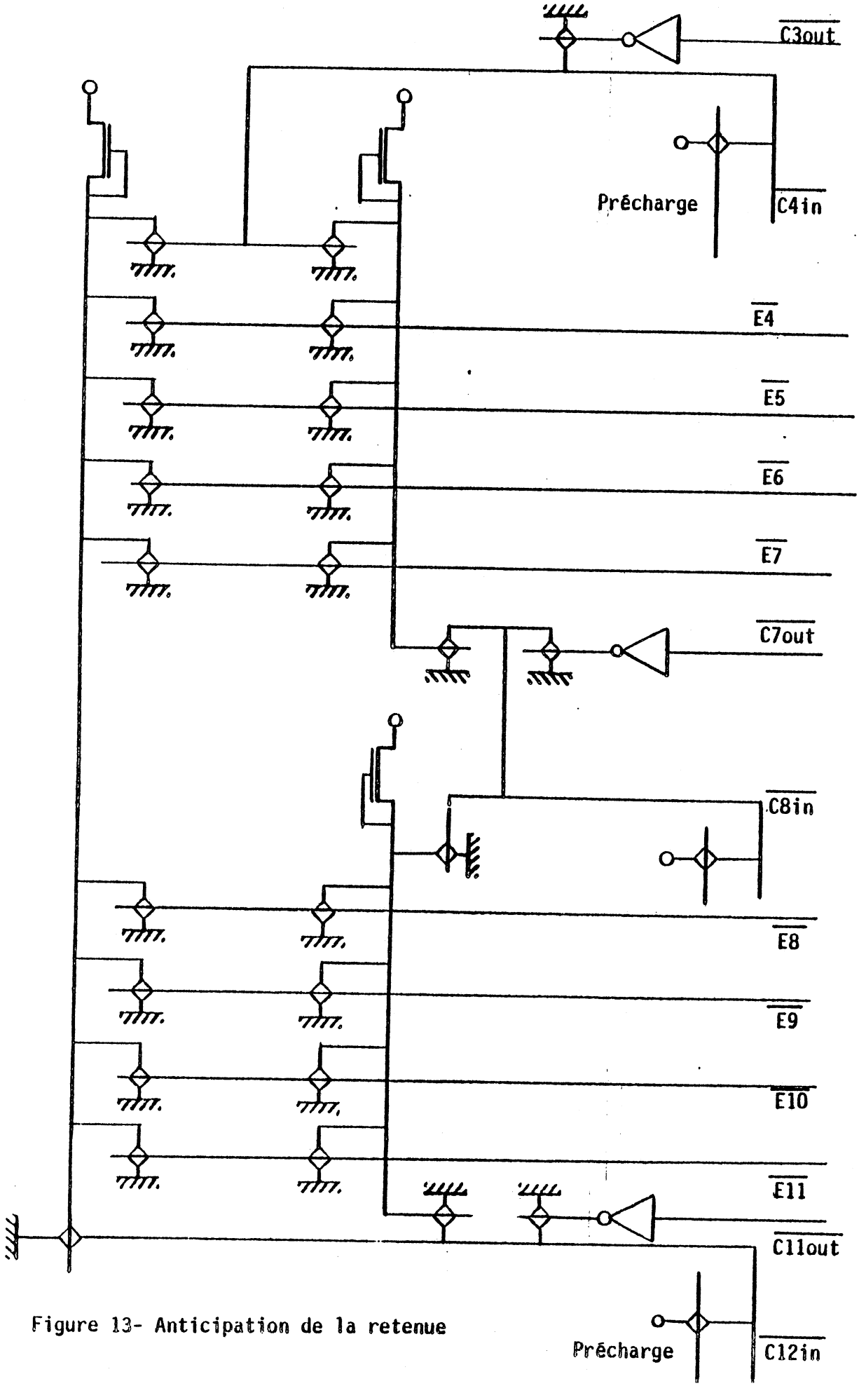


Figure 13- Anticipation de la retenue

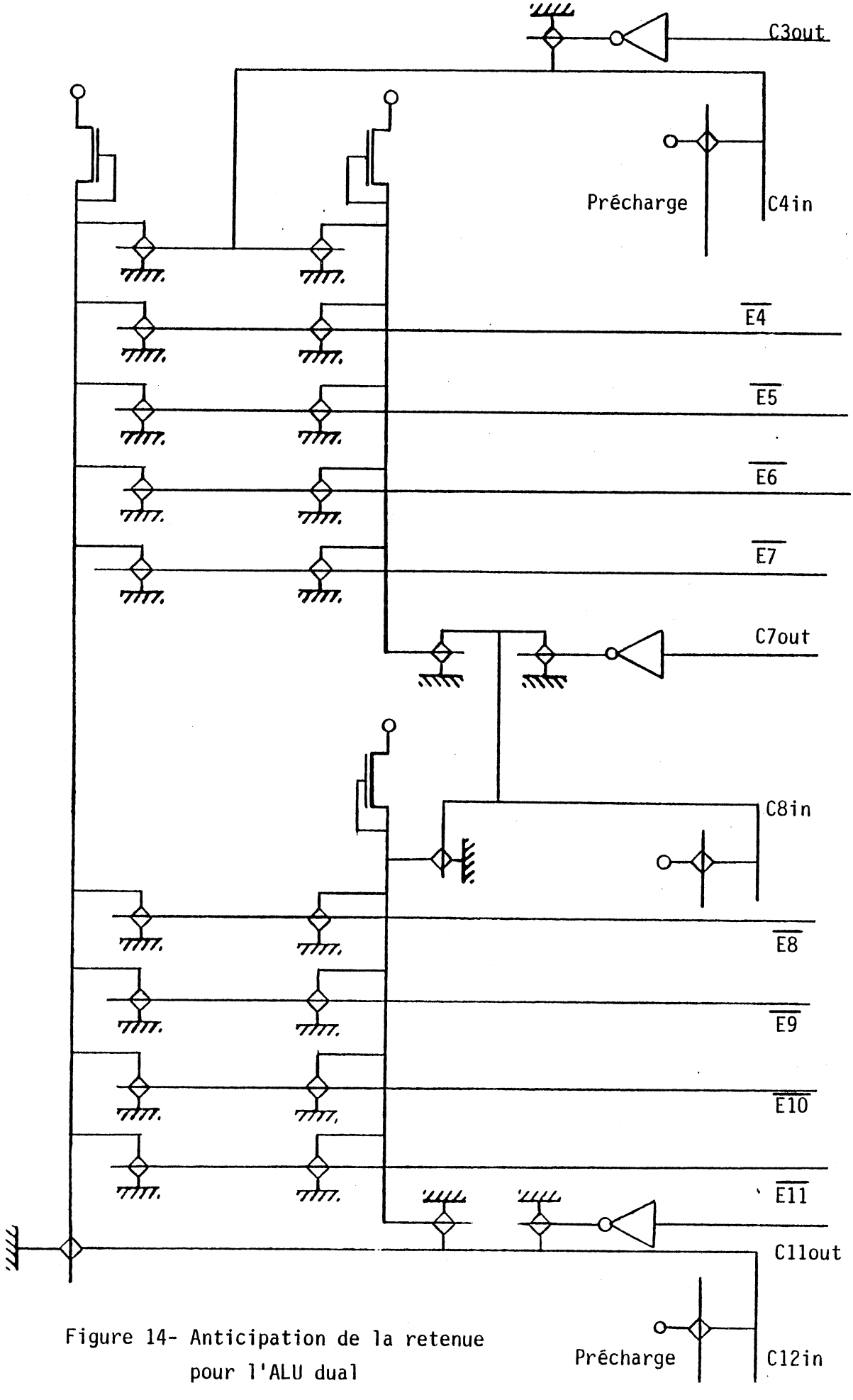


Figure 14- Anticipation de la retenue pour l'ALU dual

c/ "Byte correction logique"

Ce circuit est utilisé pour la correction des résultats de l'ALU, dans le cas des opérations en BCD. Le circuit est présenté dans la figure 15a. La commande CORFA/S permet la génération de deux sorties correspondant à l'addition BCD ou à la soustraction BCD, CORFA/S=1 dans le cas de l'addition, CORFA/S=0 dans le cas de soustraction. La commande CORF ALUD valide le transfert des résultats du circuit, à l'entrée D de l'ALU. La taille des opérations BCD est l'octet. La figure 15b donne le circuit complémentaire ("double rail").

Le circuit 15b est obtenu en utilisant les règles de dualité pour les portes logiques et les règles de dualité pour les MOS de transfert (section III.3.3.1.2,1er partie) , pour les raisons expliquées dans cette section, la commande CORF ALUD est identique dans les deux circuits duaux.

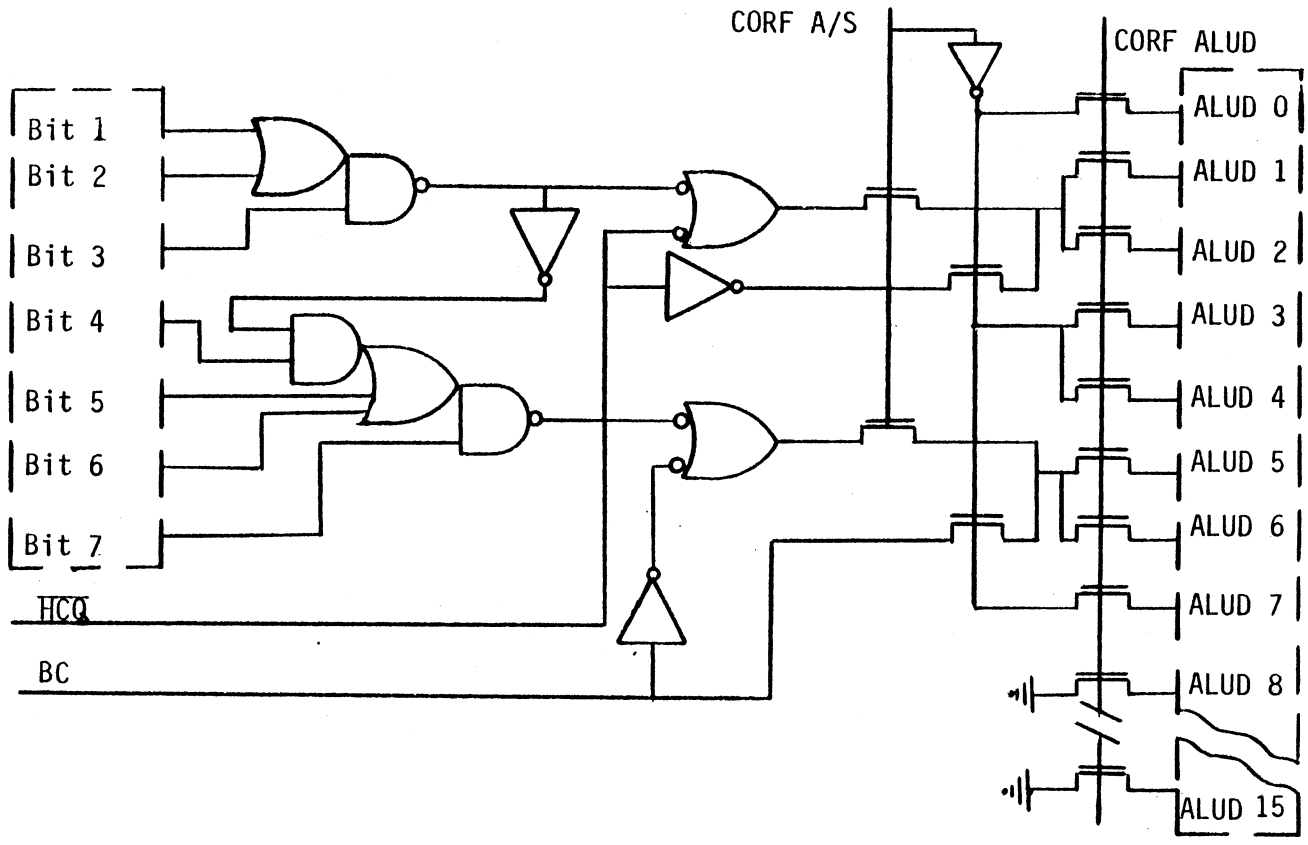


Figure 15a- Circuit de correction pour les opérations sur des nombres decimals

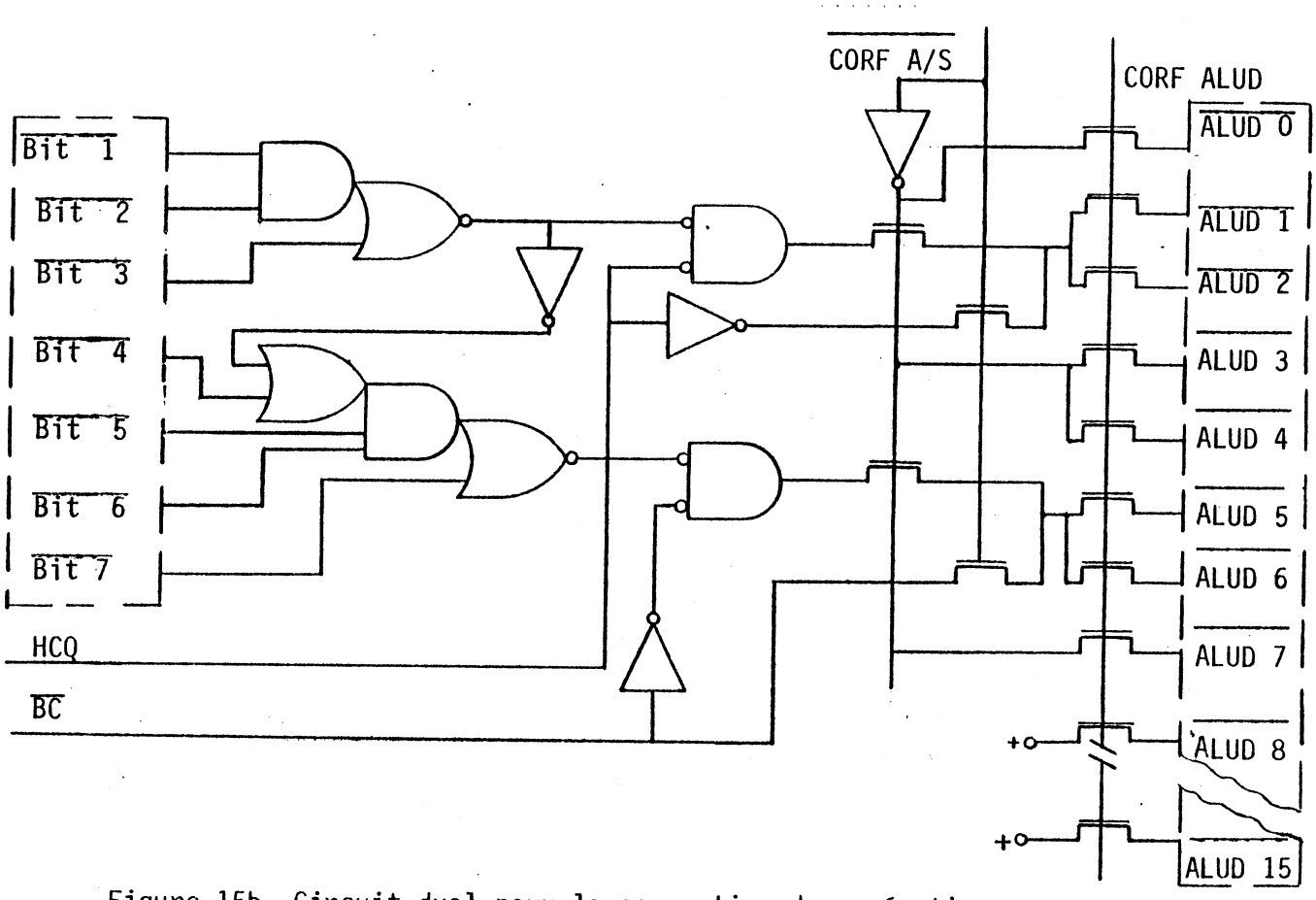


Figure 15b- Circuit dual pour la correction des opérations sur des nombres decimals

d/ Acquisition de l'opérande sur l'entrée D de l'ALU

Le circuit d'acquisition sur l'entrée D de l'ALU est présenté dans la figure 16a. La commande k->ALUD sert à connecter la constante FFFF ou 0000 à l'entrée D.

La commande CORF -> ALUD sert pour la connexion du circuit "byte correction logic" avec l'ALU et la commande DBD -> ALUD effectue la connexion avec le bus DBD.

Le circuit dual est donné dans la figure 16b. Il s'agit d'un circuit identique à l'original pour les raisons données dans la section III.3.3.1.2. (1er partie), concernant la dualité des MOS de transfert.

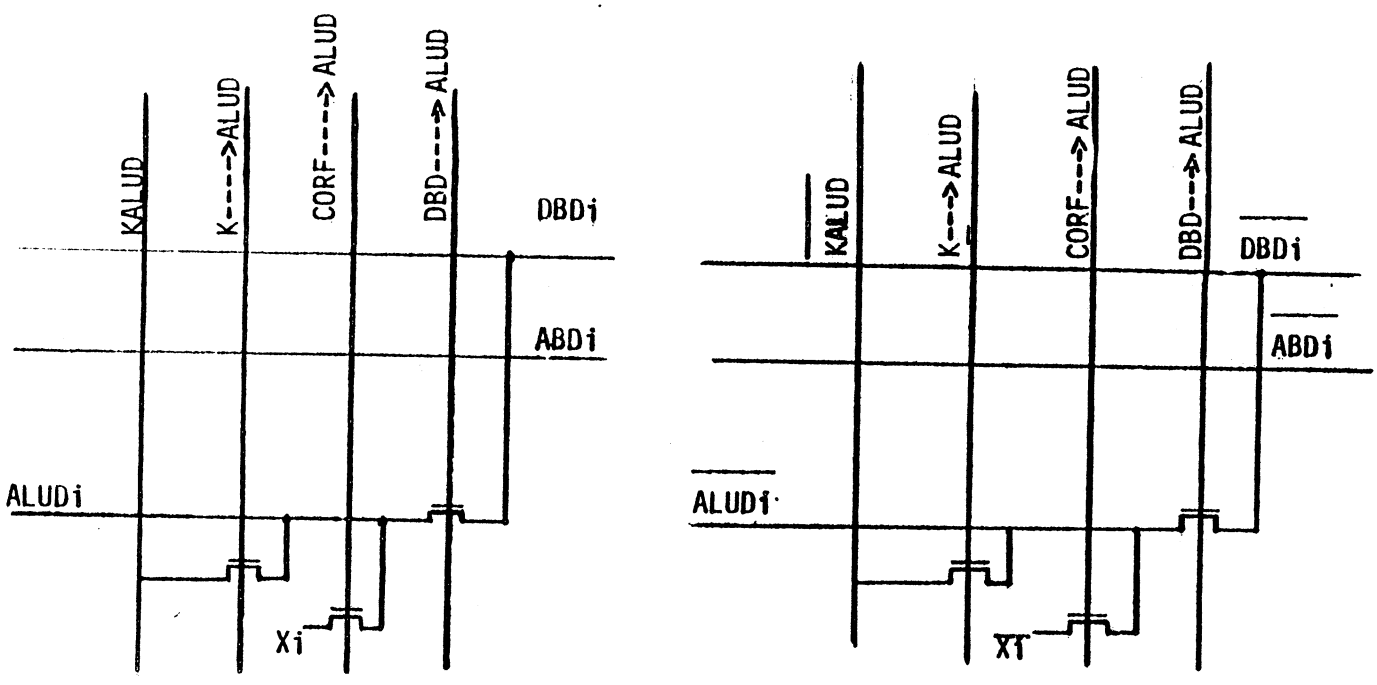


Figure 16- Acquisition de l'opérande sur l'entrée D de l'ALU
a/ circuit original, b/ circuit dual

e/ Acquisition de l'opérande sur l'entrée A de l'ALU

L'acquisition sur l'entrée A de l'ALU s'effectue par le circuit de la figure 17a.

Les acquisitions suivantes peuvent être effectuées par ce circuit:

- bus ABD sur l'entrée A (commande ABD->ALUA)
- buffer de l'ALU sur l'entrée A (commande ALUB->ALUA)
- bus ABD sur le buffer de l'ALU (commande ABD->ALUB)
- bus DBD sur le buffer de l'ALU (commande DBD->ALUB).

Toutes ces acquisitions peuvent se faire sur l'opérande direct (commande TRUE->ALUA activée) ou sur l'opérande complémentaire (commande $\overline{\text{TRUE}}$->ALUA activée).

La commande SR est utilisée enfin pour effectuer le décalage à droite ; elle connecte le bit i+1 du bus ABD à l'entrée A du bit i.

Le circuit dual donné dans la figure 17b est identique au circuit original et ses commandes sont aussi identiques aux commandes du circuit original. Les raisons conduisant à cette solution sont données dans la section III.3.3.1.2 de la 1er partie et concernent la dualité en technologie N-MOS, des inverseurs, des MOS de transfert et des registres.

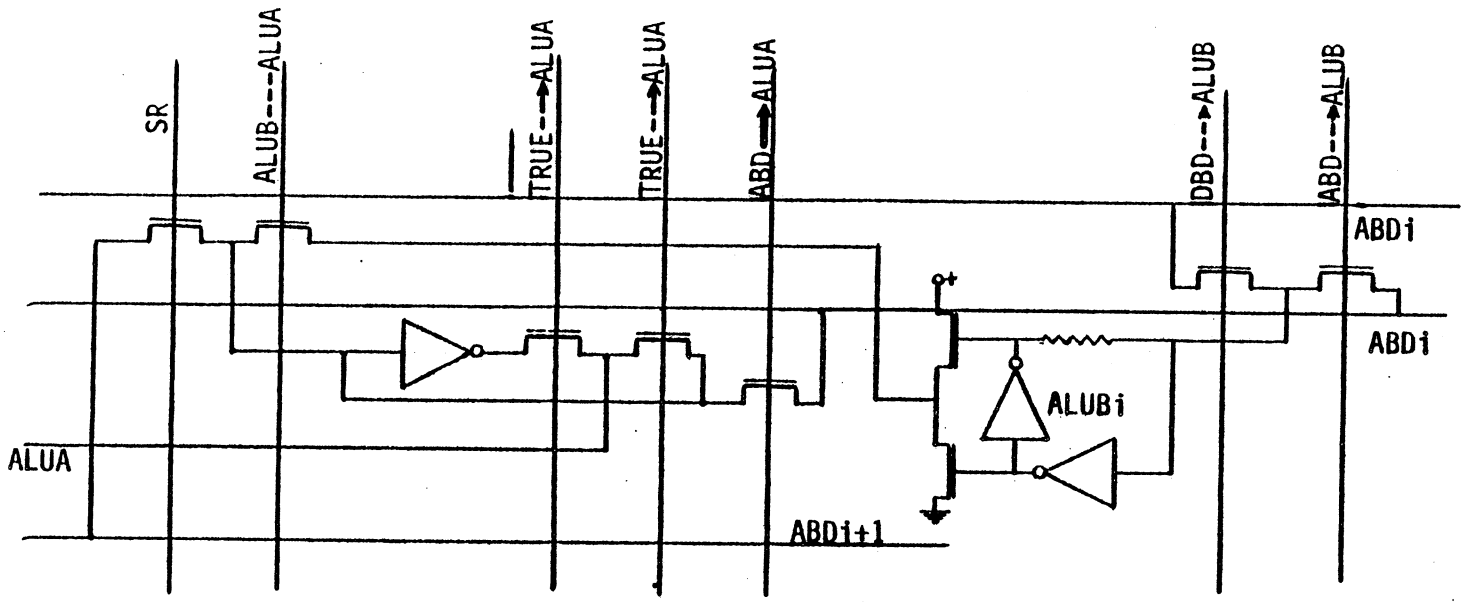


Figure 17a- Circuit d'acquisition de l'opérande sur l'entrée A de l'ALU

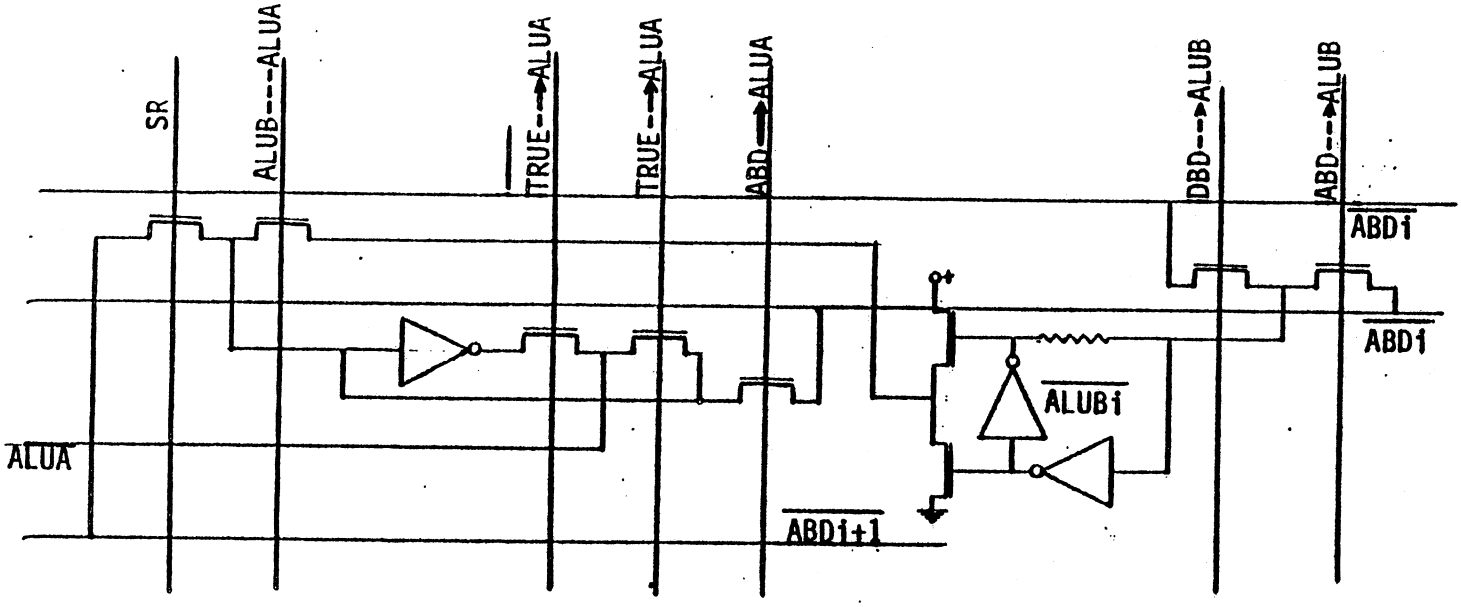


Figure 17b- Circuit dual pour l'acquisition de l'opérande sur l'entrée A de l'ALU

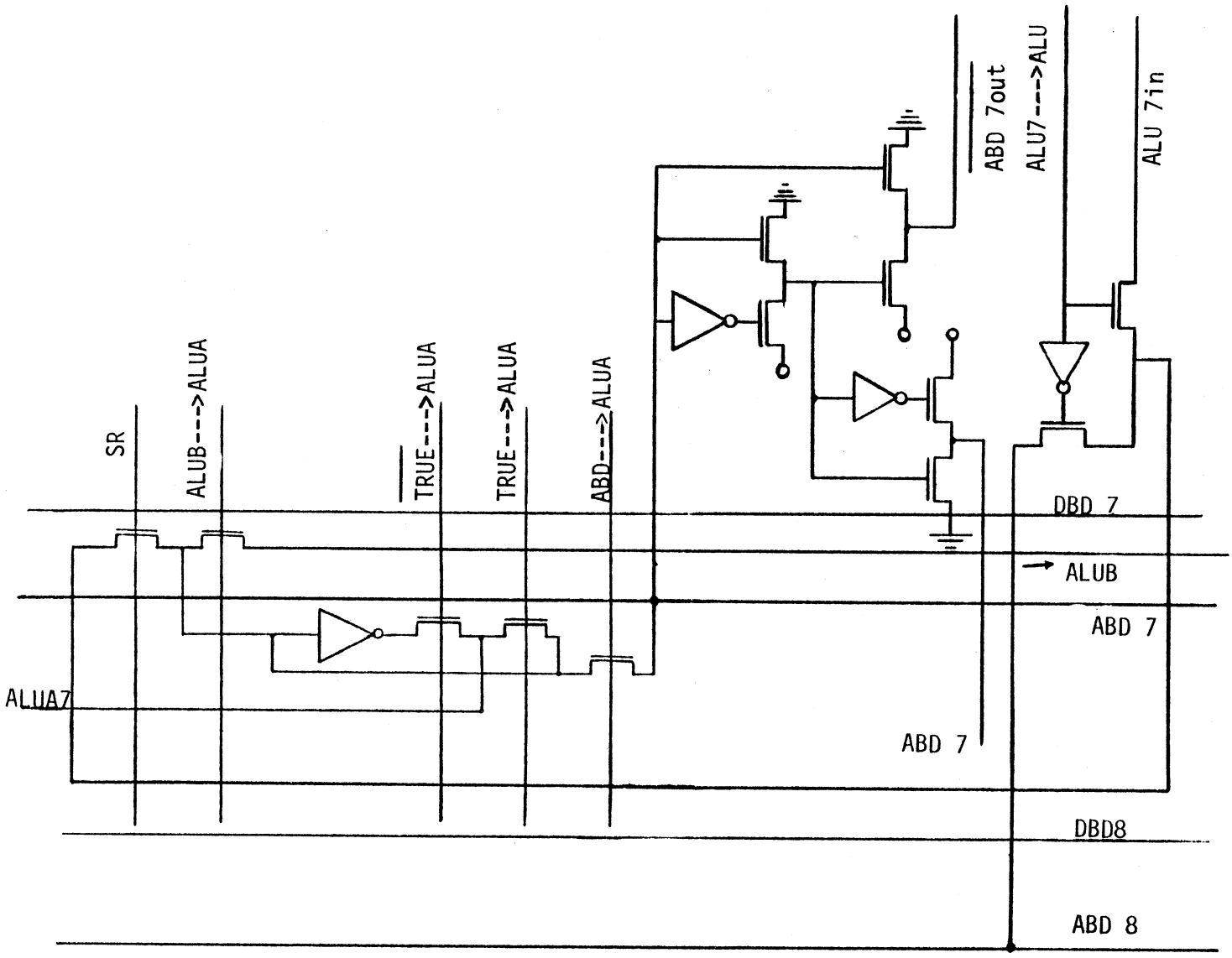


Figure 18- Acquisition de l'opérande sur le bit 7 de l'entrée A de l'ALU

La description d'un pas du circuit d'acquisition sur l'entrée donnée par la figure 17a est valable pour la majorité des bits, sauf quelques cas particuliers.

L'acquisition sur le bit 7 est présentée figure 18, le circuit dual sera identique de même que ses commandes. Les opérandes doivent être complémentaires aux opérandes du circuit original, c'est à dire qu'il sera connecté avec $\overline{ABD7}$, $\overline{ABD8}$ et ALU7IN'=$\overline{ALU7IN}$ au lieu des ABD7, ABD8 et ALU7IN.

Un circuit spécial est présenté figure 19a. Il est utilisé pour faire l'extension du bit 7 de l'opérande présent sur le bus ABD aux bits 8...15 de l'entrée A. Le signal ABDEA est validé dans le cas de l'instruction EXTW (extension du signe de l'octet au mot) ainsi que dans les cas d'adressage "registre indirect indexé" et de l'adressage "compteur de programme indexé". Le circuit dual est donné figure 19b.

Le dernier cas particulier est la connexion de ABDi+1 avec ALUAi, validée par le signal SR, dans le cas du bit 15, la connexion se fait avec ALU15 IN pour l'ALU direct et avec ALU15 IN' pour l'ALU complémentaire.

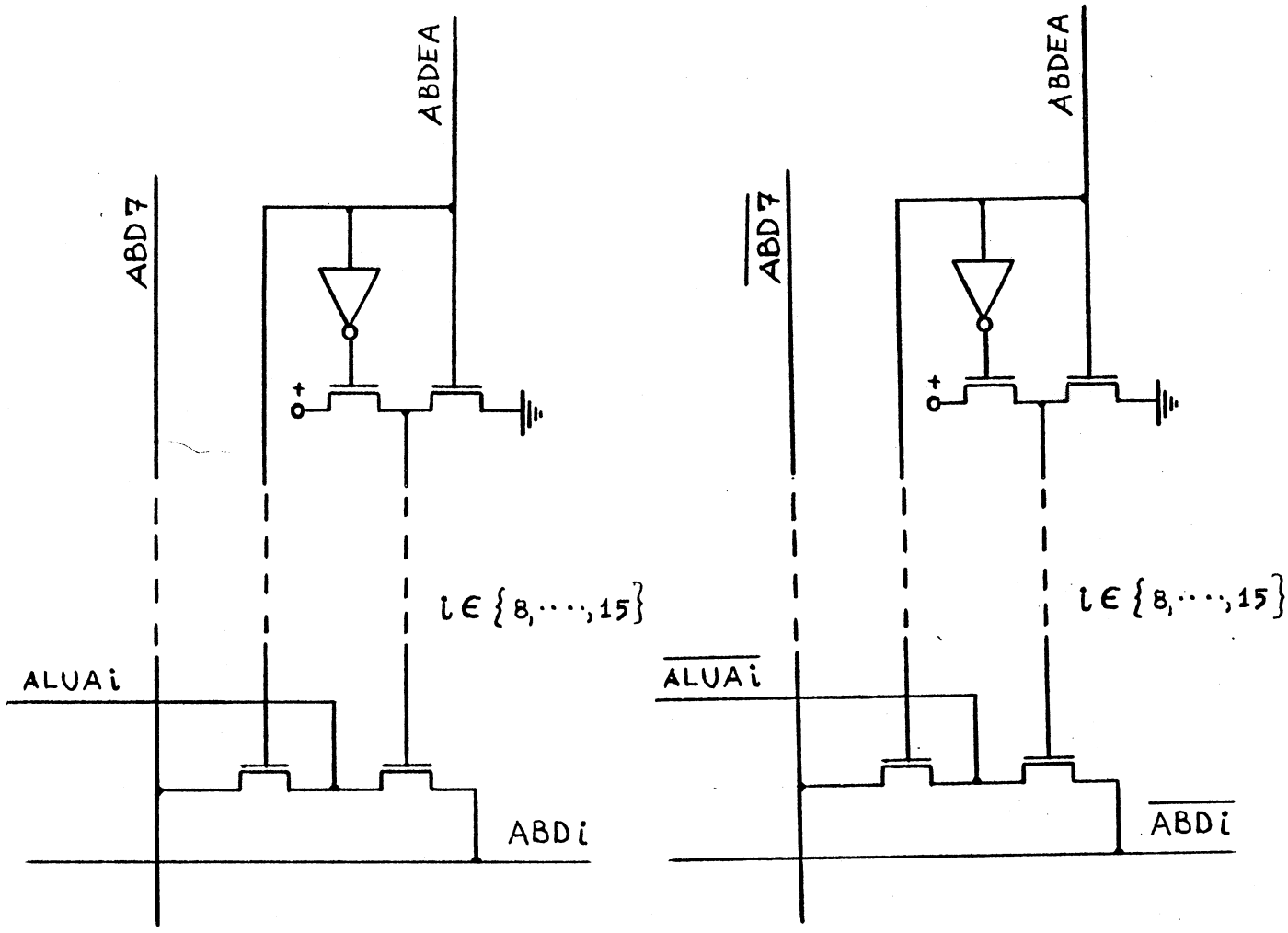


Figure 19- Méchanisme d'extension du signe de l'octet au mot
a/ circuit original, b/ circuit dual

f/ ALUR (registre de l'ALU)

Le registre de l'ALU et ses commandes sont identiques sur le circuit direct et sur le circuit complémentaire (dualité des registres ,section III.3.3.1.2, 1er partie).

Sur les figures 20a et 20b on présente l'ALUR direct et l'ALUR complémentaire pour l'octet poids faible et pour l'octet poids fort, avec les circuits de génération des BN-WN et des LBZ-HBZ.

Les circuits de génération des LBZ-HBZ sont identiques dans les deux circuits duaux pour éviter d'utiliser deux portes NAND à huit entrées. Les entrées de ces circuits sont aussi identiques , donc leur sorties sont égales (figures 20a et 20b).

g/ Autres circuits

Le circuit de génération des BV et WV est présenté à la figure 21a. le circuit dual est présenté à la figure 21b.

La figure 22 donne la bascule pour le stockage provisoire des BC, BV et WC, WV. Un circuit identique sera utilisé dans le système complémentaire.

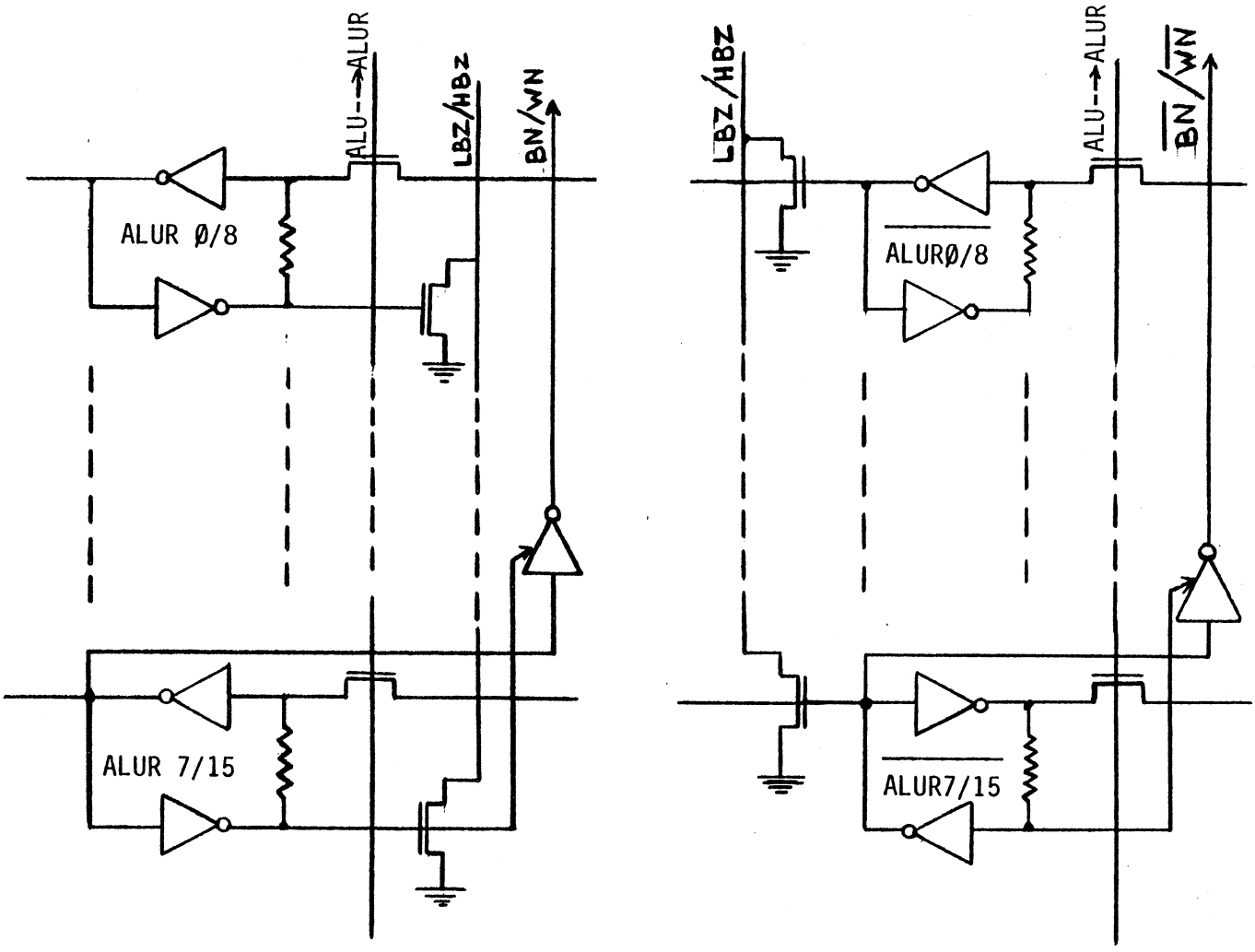


Figure 20- Registre de l'ALU et géneration des prédicats LBZ/HBZ, BN
a/ circuit original, b/ circuit dual

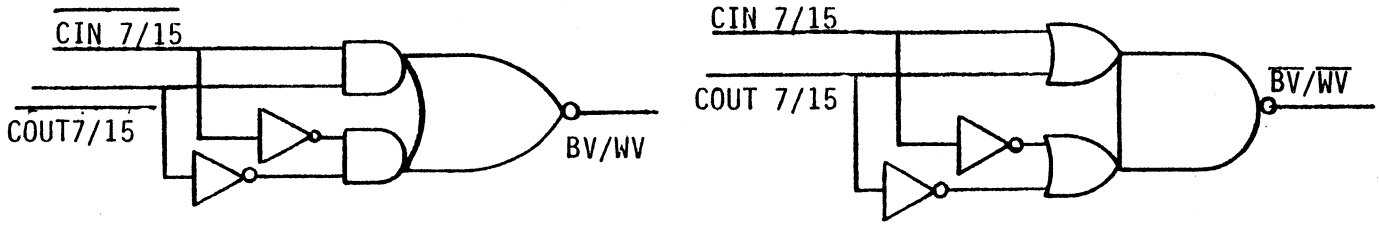
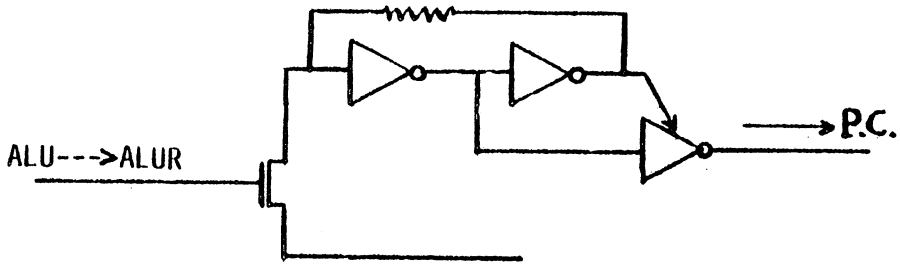


Figure 21- Géneration des prédicats BV/WB
a/ circuit original, b/ circuit dual

Figure 22- Bascule de stockage pour les prédicats BC/BV/WC/WV



h/ ALUE (ALU EXTENDER)

L'ALUE est utilisé en couplage avec l'ALU pour effectuer le décalage sur 32 bits.

Le circuit de l'ALUE est présenté figure 23. Son fonctionnement peut être consideré dynamique car la valeur du bit précédent est mémorisé dans les capacité de grille des transistors signal des portes NAND.

Le circuit dual de l'ALUE sera identique au circuit de l'ALUE direct car la mémorisation et le décalage sont des fonctions autoduales.

Les connexions de l'ALUE complémentaire avec le bus DBD seront échangées pour avoir en entrée de l'ALUE complémentaire des données complémentaires par rapport aux données entrant dans l'ALUE, de même pour les entrées $\overline{\text{B15 IN ALUE}}$ et B0 IN ALUE de l'ALUE direct, on aura les signaux B15 IN ALUE' = B15 IN ALUE et B0 IN ALUE' = $\overline{\text{B0 IN ALUE}}$ pour l'ALUE complémentaire.

Dans ces conditions, les résultats de l'ALUE et de l'ALUE dual sont complémentaires.

Les résultats de l'ALUE doivent être codés en parité avant leur sortie sur les bus.

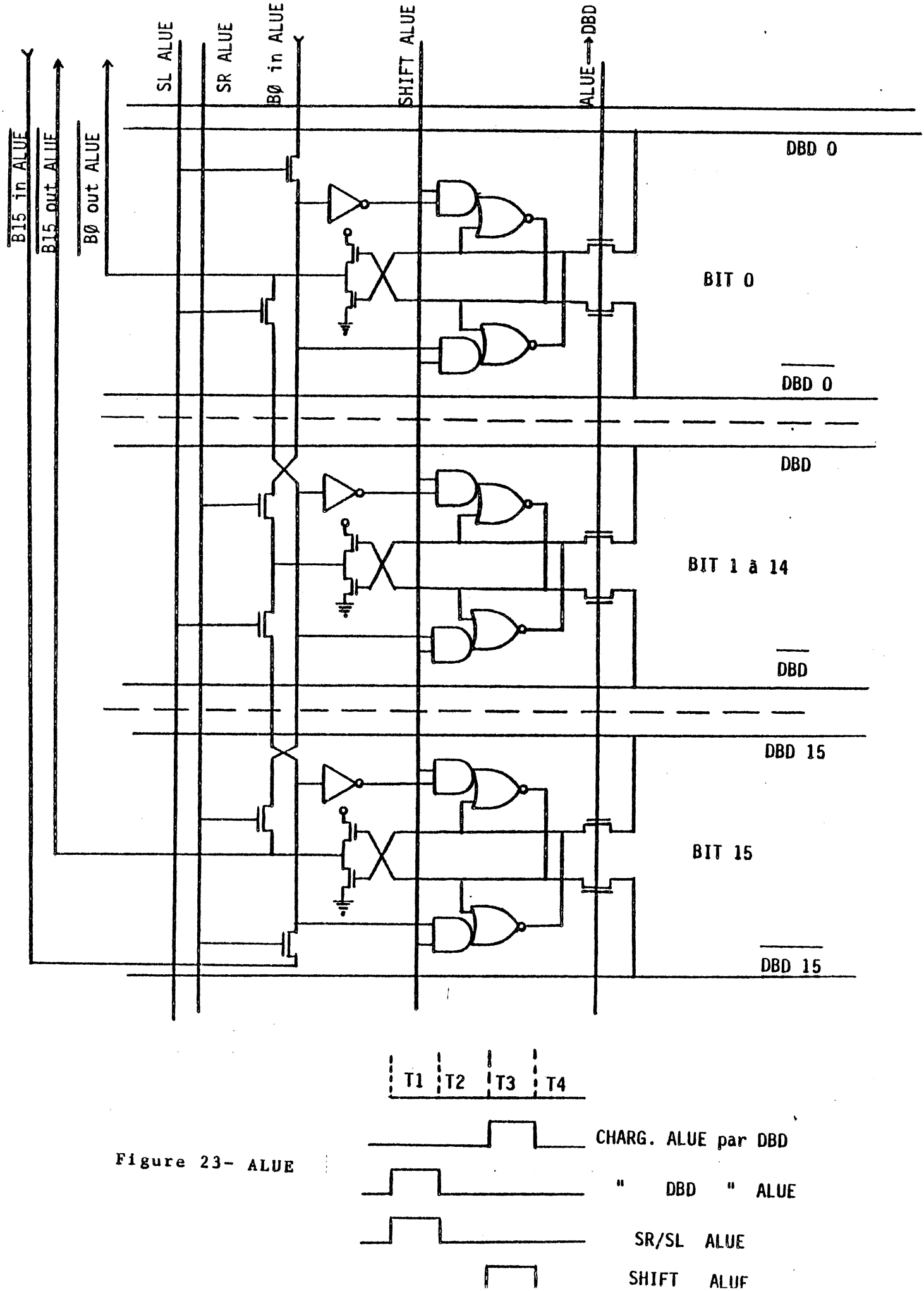


Figure 23- ALUE

Les bits de parité des résultats de l'ALUE au lieu d'être générés chaque fois à partir des résultats de l'ALUE seront calculés pendant le fonctionnement de l'ALUE.

Cette solution permet d'une part, d'utiliser l'espace vide des bits de parité pour placer les circuits nécessaires tandis que pour un générateur des bits de parité on devait occuper un espace à coté de l'ALUE, d'autre part, cette solution permet de supprimer l'ALUE complémenté dans une version du MC 68000 partiellement autotestable, puisque cette solution peut détecter toutes les erreurs impaires, elle peut alors détecter la quasi totalité des défauts mais sans assurer le "totaly self checking goal" du circuit.

Le circuit de calcul de bit de parité est donné figure 24. Deux circuits identiques seront utilisés, un pour chaque octet de l'ALUE.

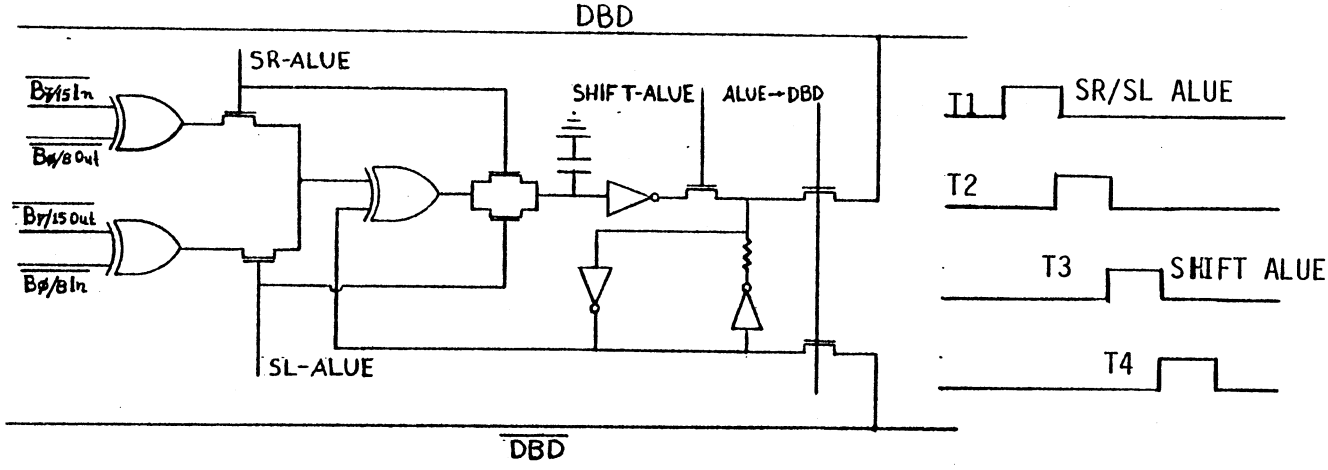


Figure 24- Calcul du bit de parité, octet poids faible/octet poids fort

Les bits de parité du bus sont mémorisés dans le registre de ce circuit. A chaque décalage le calcul du nouveau bit de parité de chaque octet de l'ALUE se fait à partir du bit entrant, du bit sortant et du bit de parité précédant de l'octet.

Le fonctionnement de ce circuit est dynamique en respect du fonctionnement de l'ALUE, le nouveau bit de parité est calculé pendant T1, SR-ALUE ou SL-ALUE étant activé. Les mêmes commandes sont utilisées pour déconnecter le circuit de calcul après T1, pendant T2 le résultat est stocké dans la capa désignée dans la figure 24, pendant T3 la commande SHIFT-ALUE est activée et les résultats sont stockés dans le registre.

II.2.2.2. Circuits de contrôle et Générateurs de parité.

Pour chacun des circuits doubles (ALU et ALUE) on doit utiliser un controleur "double rail".

On doit également utiliser un générateur de parité pour chacun des deux circuits pour coder les résultats avant de les envoyer aux bus ou au DOB.

En fait, un seul controleur "double rail" est suffisant pour les deux circuits , ceci est possible en utilisant la technique suivante:

Les deux circuits (ALU et ALUE) fonctionnent simultanément, mais on peut les distinguer et les controler avec un seul controleur en utilisant le signal ALUE -> DBD. ALUE étant connecté uniquement avec le bus DBD sera controlé chaque fois que ses résultats sortent dans le bus DBD, alors sa connexion avec le controleur sera validée par le signal ALUE -> DBD, ceci introduit aussi un contrôle pendant la phase d'écriture du bus DBD dans ALUE qui n'est pas nécessaire mais qui n'introduit pas d'anomalie.

Le reste du temps le registre de l'ALU (ALUR) est connecté au controleur.

Sur le chronogramme de fonctionnement des deux circuits (figure 25), on peut vérifier que tous les résultats sortant des deux circuits sont controlés avec cette méthode.

Le codage en parité des résultats se fait par un générateur de parité pour l'ALU et par un circuit de calcul de parité pour l'ALUE décrits précédemment.

La figure 26 présente le schéma général du système ALU-ALUE doublé, les controleurs de parité des bus ABD et DBD sont également présentés dans cette même figure.

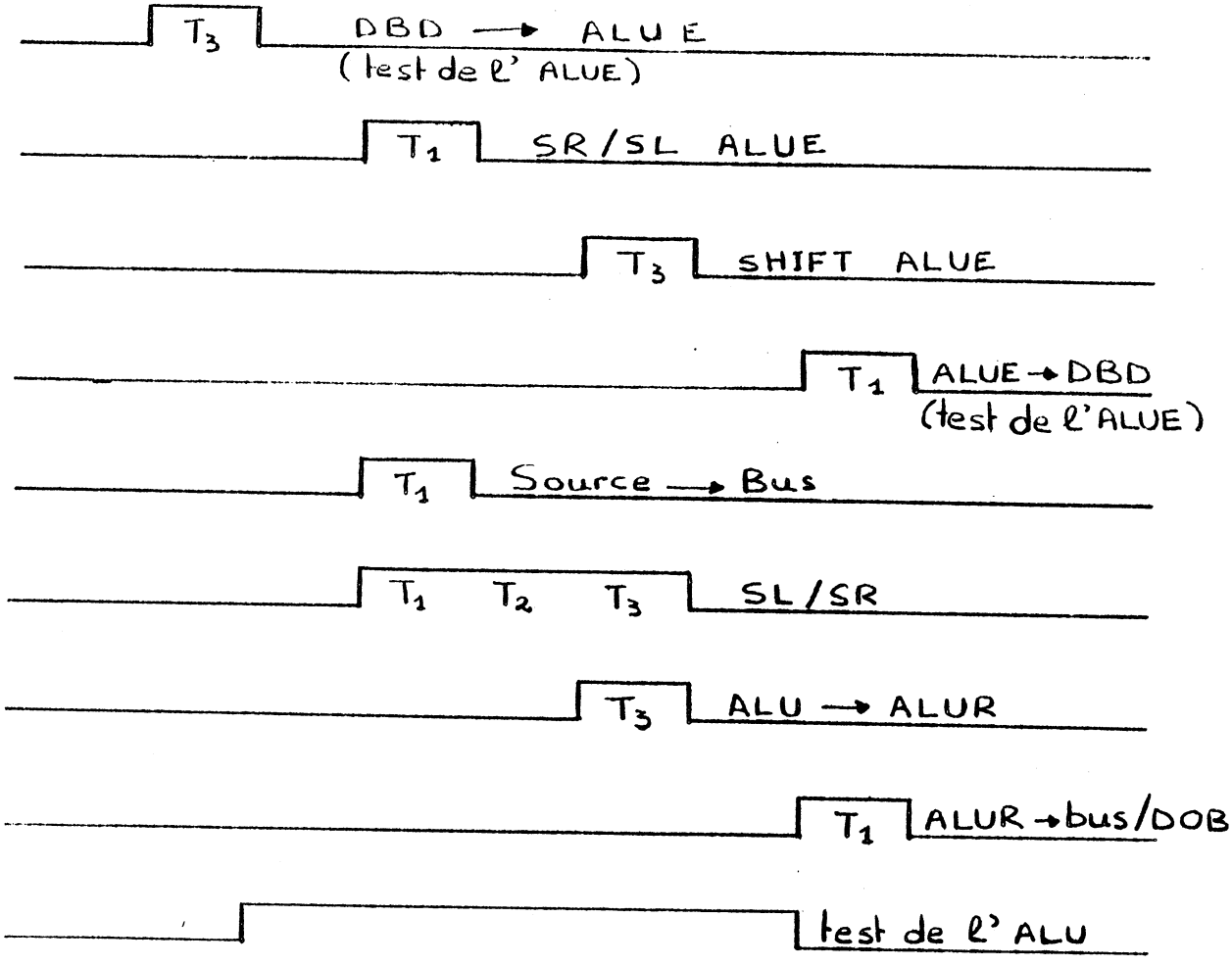

T3
DBD → ALUE
(test de l' ALUE)
T1
SR/SL ALUE
T3
SHIFT ALUE
T1
ALUE → DBD
(test de l'ALUE)
T1
Source → Bus
T1 T2 T3
SL/SR
T3
ALU → ALUR
T1
ALUR → bus/DOB
test de l' ALU

Figure 25:

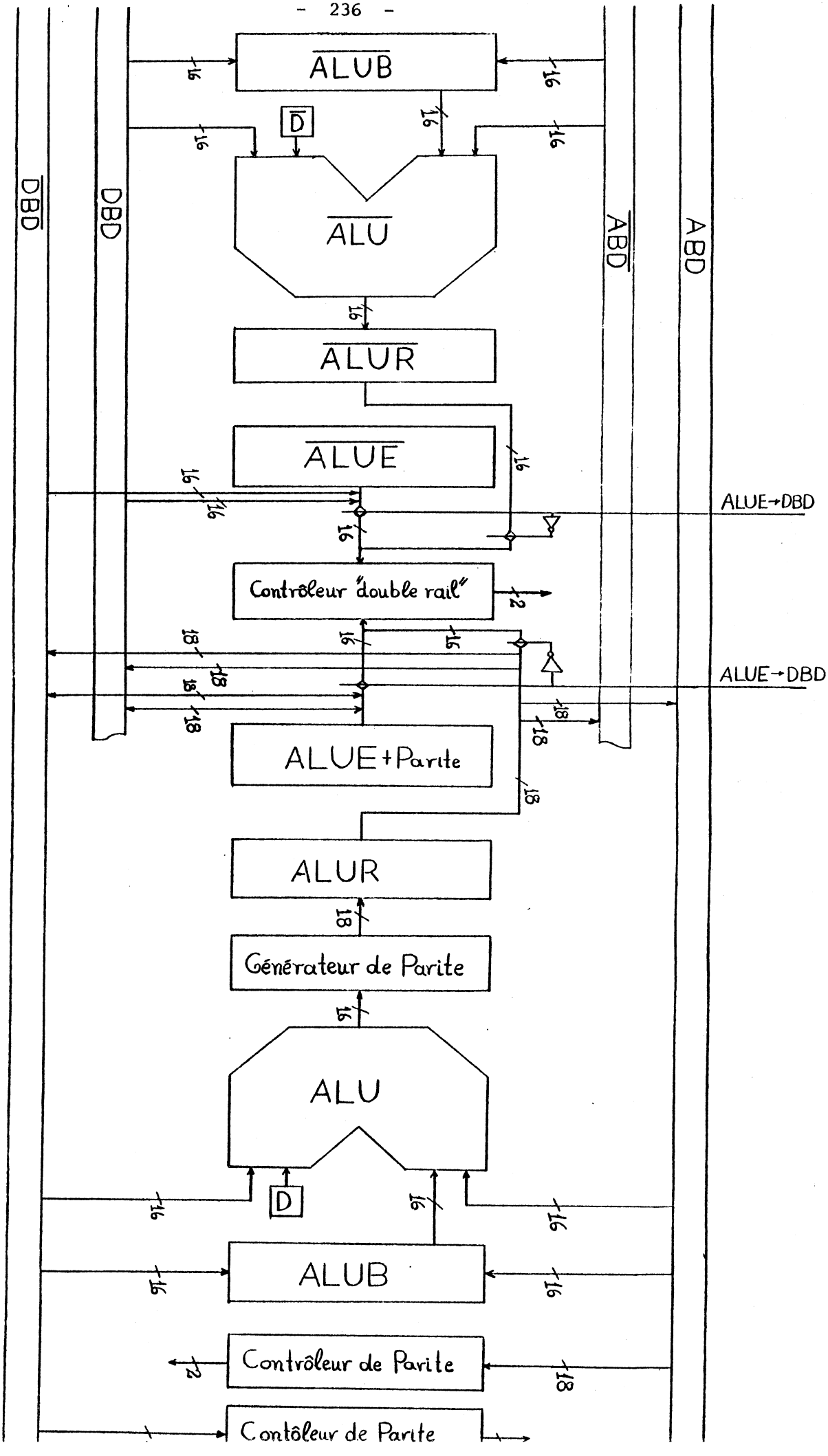

ALUB
D
ALU
ALUR
ALUE
ALUE→DBD
Contrôleur "double rail"
ALUE→DBD
ALUE+Parite
ALUR
Générateur de Parite
ALU
D
ALUB
Contrôleur de Parite
Contôleur de Parite
DBD
DBD
ABD
ABD
16
18
2

Figure 26:

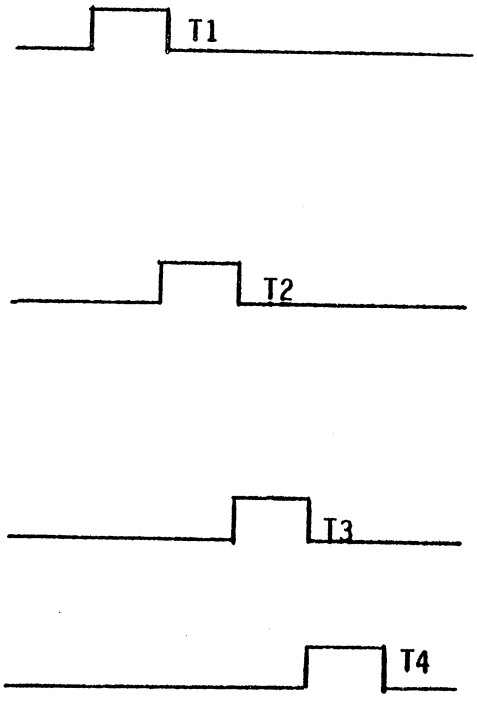
II.2.3. FOOL IT

Ce circuit est destiné à effectuer la concaténation des huit bits poids faible du bus ABD et des huit bits poids fort d'un registre de données, nécessaires pour les opérations de taille octet dont la destination est un registre données.

Le circuit du FOOL IT est identique au circuit de précharge du bus mais il affecte seulement les huit bits poids fort du bus ABD.

La particularité du bus ABD qui possède une commande LBYTE ABDSA pour les amplis des huit bits poids faible et une commande HBYTE ABDSA pour les amplis des huit bits poids fort, est destinée à permettre la concaténation.

Le timing d'une concaténation est donné figure 27.



ALUR->ABD (16 bits)
FOOL IT ('1' 8 bits ABD poids fort,
'1' 8 bits $\overline{ABD}$ poids fort)

FOOL IT
LBYTE ABDSA (information aux 8 bits
poids faible du bus)
non HBYTE ABDSA('1' 8 bits ABD poids fort,
'1' 8 bits $\overline{ABD}$ poids fort)

FOOL IT, LBYTE ABDSA
ABD->Registre données (concaténation)

FOOL IT, précharge

Figure 27-Cycle d'écriture sur l'octet poids faible d'un registre

On voit alors que l'écriture sur le poids faible d'un registre se fait de façon normale et que le poids fort du bus ABD est préchargé et non amplifié, donc le poids fort du registre n'est pas changé.

Le circuit d'un bit de FOOL IT et des amplis du bus ABD poids fort est donné figure 28.

Pour l'autotest du circuit, on peut remarquer que FOOL IT étant identique au circuit de précharge, son autotest est assuré par le bit 15' du code de parité.

La commande FOOL IT doit être testée à la sortie de la partie opérative.

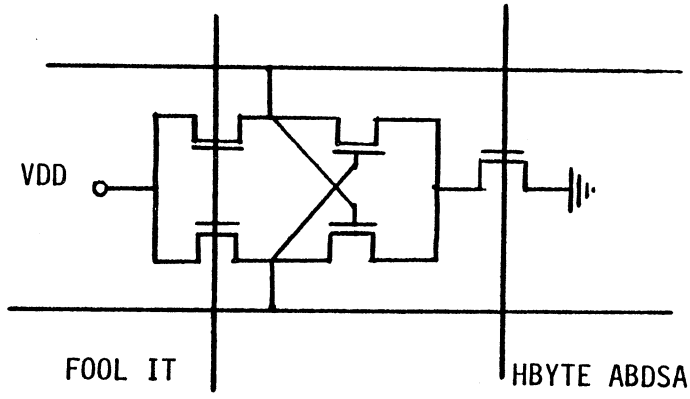


Figure 28 - FOOL IT et amplificateurs du bus ABD poids fort. (bits 8 à 15)

II.2.4. DCR

Le circuit DCR est utilisé dans le cas des opérations de manipulations de bits.

Il fait le décodage du numéro du bit à manipuler, ce numéro est codé en 5 bits, dans le champ "bit number" de mot d'extension des opérations de manipulation de bits.

Le schéma du circuit DCR est présenté figure 29. Il est constitué d'une partie registre et d'une partie décodeur.

Le champ "bit number" est présenté sur le bus ABD, en T3 les 5 bits significatifs ABD0, ABD1, ABD2, ABD3, ABD4 sont écrits (commande ABD->DCR) dans le registre DCR.

Le bit DCR4 est utilisé dans le PLA de branchement pour choisir le poids faible ou le poids fort du registre des données.

Les bits DCR0, DCR1, DCR2, DCR3 sont décodés par la partie décodeur qui donne à sa sortie le numéro du bit à manipuler en code 1/16.

Les sorties du décodeur sont directes et complémentaires pour avoir accés au bus bifilaire DBD.

Dans ce circuit, quelques MOS sont fortement déplétés et ils ne seront pas pris en compte pour le fonctionnement logique du circuit.

Le réseau des MOS signal est réalisé par deux arbres binaires identiques (figure 29), la figure 30 donne l'un de ces arbres. Sur cette figure, les MOS fortement déplétés n'apparaissent pas.

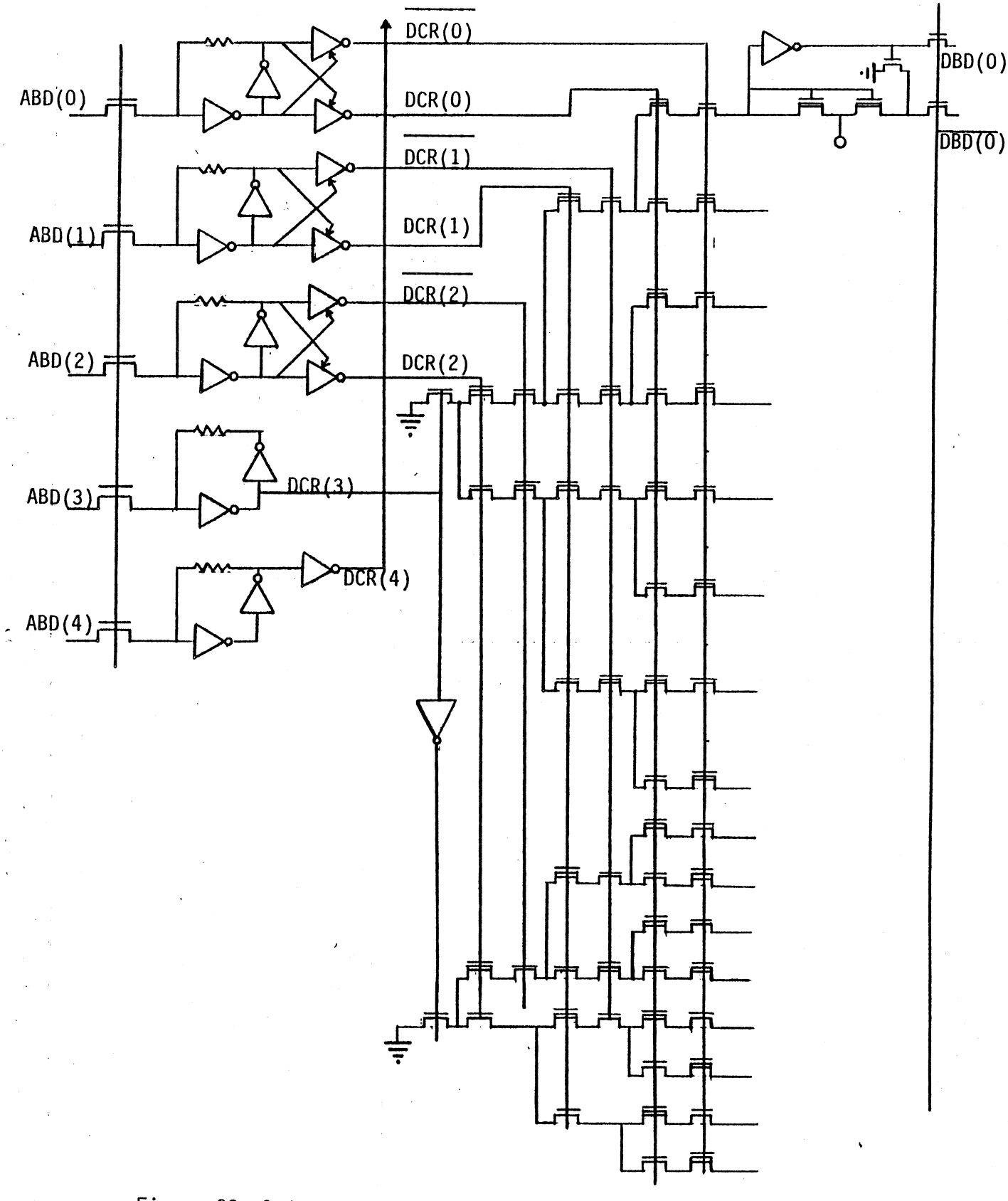


Figure 29- Opérateur DCR

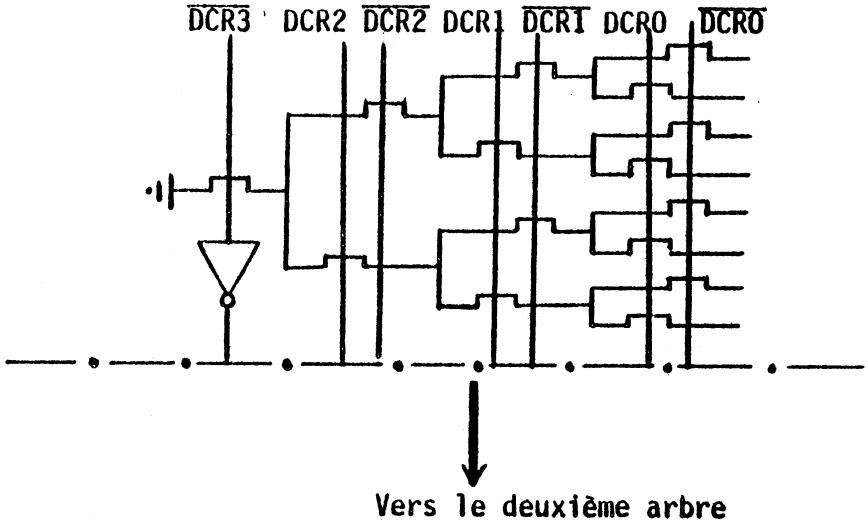


Figure 30- L'arbre de transistors MOS,partie decodeur de l' opérateur DCR

II.2.4.1. Circuit augmenté

L'etude du circuit est faite dans le paragraphe V.1 de la 1er partie.
Le circuit augmenté est présenté à la figure 31.

Le module registre est testé avec le code de parité. Deux bits de parité, un pour les 8 bits poids faible et un pour les 8 bits poids fort, sont générés très facilement à partir du bit DCR3. Avec ce codage, la partie décodeur est SFS, son test est effectué par les controleurs de parité du bus DBD.

On peut observer que la solution obtenue en utilisant la notion ''degré de divergence effective'' (section III.1.1, 1er partie) est très peu coûteuse en surface.

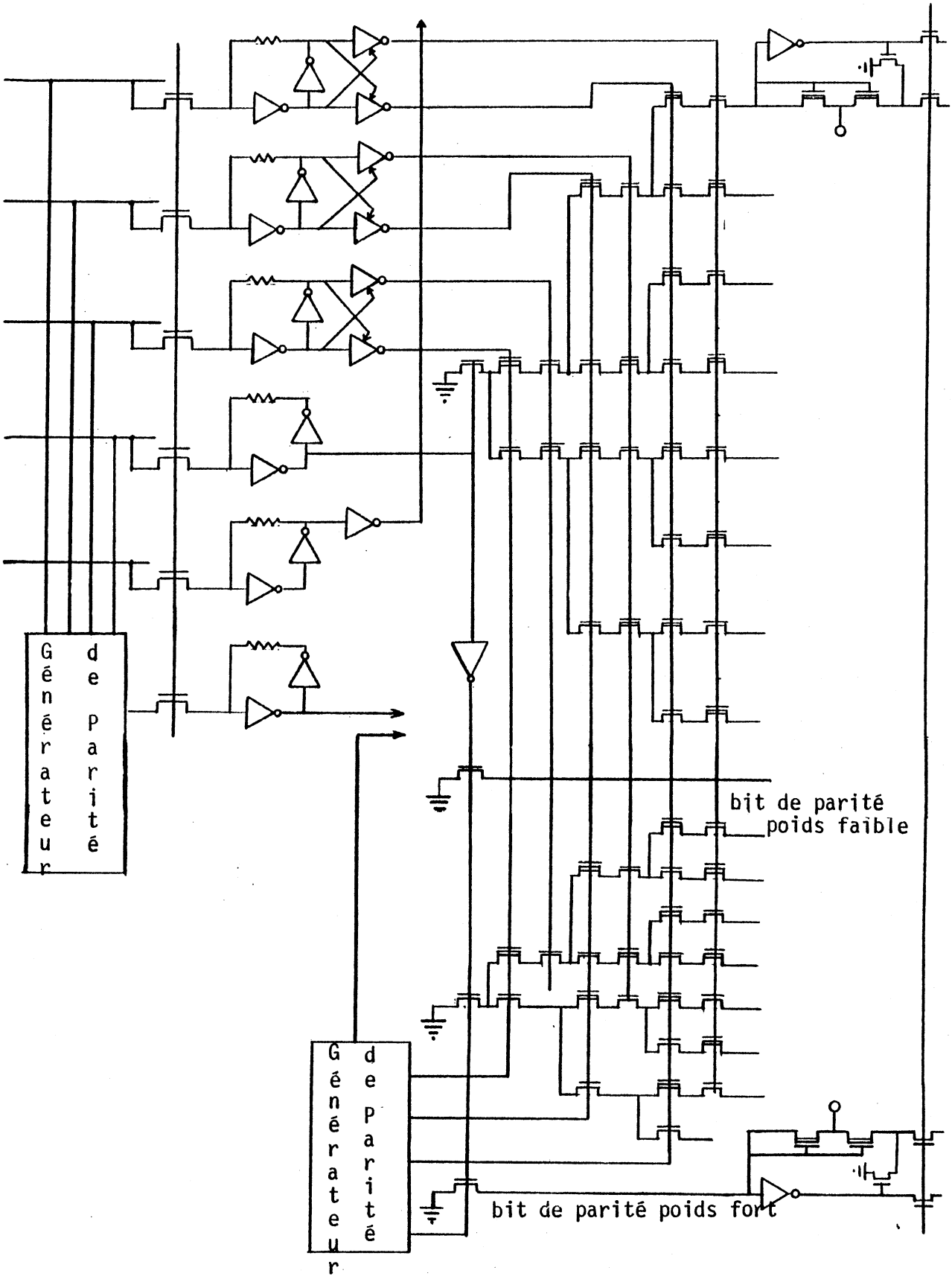


Figure 31- Opérateur DCR autotestable

II.2.4.2. Autres versions du DCR

Dans d'autres versions du MC 68000 on peut trouver le circuit DCR présenté figure 32.

Le signal FOOL IT est utilisé pour valider les 8 sorties poids faible et invalider les 8 sorties poids fort du circuit DCR, dans le cas de manipulation de bits d'une case mémoire. La taille de l'opérande d'une telle opération est l'octet, normalement le quatrième bit du champ "bit number" est a la valeur '1', le signal FOOL IT sert à corriger une erreur de software dans le cas où ce bit est posé par erreur à la valeur '0'.

La même solution proposée précédemment sera utilisée pour cette version mais le circuit supplémentaire relatif au FOOL IT sera doublé.

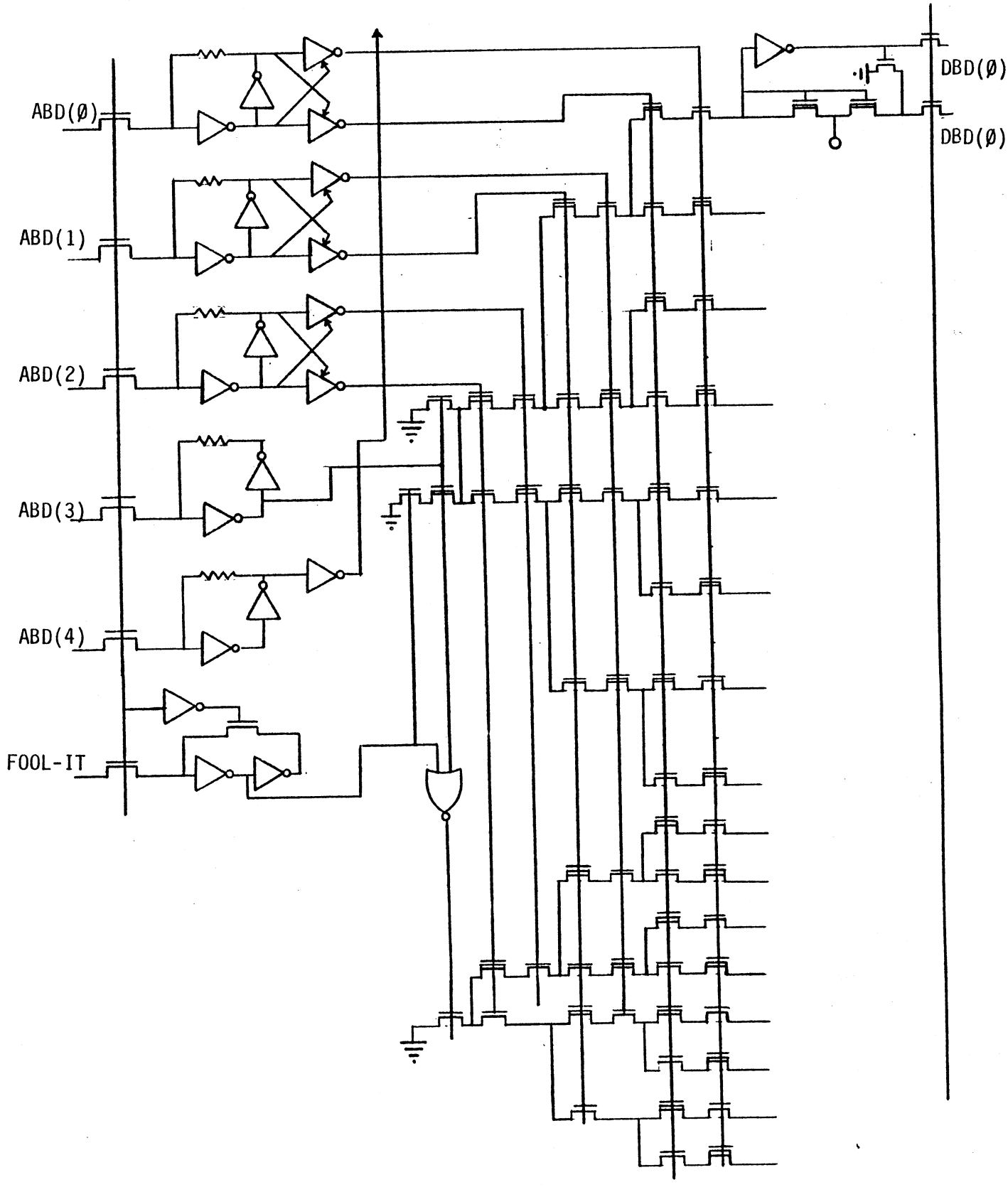


Figure 32- Version du DCR permetant la correction de certains erreurs du logiciel.

II.3. LA SECTION MEDIANE ET LA SECTION GAUCHE DE LA PARTIE OPERATIVE (TRAITEMENT DES ADRESSES)

Le schéma fonctionnel de ces deux sections est donné figure 1.

Les mécanismes du test de ces sections sont les suivants:

- AUH, AUL et leur environnement sont doublés ("double rail"),
- PREN doublé (duplication pure),
- SIGNEX, code de parité,
- Registre FTU, code de parité,
- R8LD,...,RELD, code de parité,
- ROH,...,REH, code de parité,
- USPL, SSPL, AOBL, PCL, ATL, code de parité,
- USPH, SSPH, DTH, AOBH, PCH, ATH, code de parité,
- bus ABL, bus DBL, bus ABH, bus DBH, les amplis des bus, circuit de précharge et commutateurs, code de parité.

Par cette solution, la taille des registres et des bus devient 18 bits, 16 bits d'information et deux bits de codage.

Les circuits de contrôle et de génération de parité sont les suivants:

- deux contrôleurs "double rail" 16 bits, un pour AUL et un pour AUH,
- un contrôleur "double rail" 4 bits pour PREN,
- quatre contrôleurs de parité 18 bits, chacun est placé à l'extrémité gauche d'un des bus ABL, DBL, ABH, DBH,
- un contrôleur de parité 6 bits est utilisé pour le test du circuit SIGN-EXT.

II.3.1. Unité Arithmétique (calcul d'adresse)

Cet opérateur exécute généralement le calcul des adresses, bien que parfois il soit chargé (par l'intermédiaire du registre FTU) de compter les nombres de décalages effectués par l'ALU et l'ALUE et bien que parfois l'ALU soit utilisé pour les calculs d'adresse (cas d'adressage "registre indirect indexé" et "compteur de programme indexé")

L'AU est divisé en AUL placé dans la partie médiane du PO et en AUH placé dans la partie gauche du PO.

Les deux parties AUH et AUL ne peuvent pas être utilisées indépendamment. Elles sont utilisées pour effectuer en un seul cycle les calculs d'adresse sur 32 bits, dont les 24 premiers sont significatifs pour l'adresse finale. Pour ce couplage, la retenue entrante AUH est egale à la retenue sortante AUL.

Le schéma logique d'un pas d'AU est donné figure 33.

On trouve:

$$Si = \overline{a} \oplus \overline{b} \oplus \overline{Ci-1}$$
$$Ci = \overline{\overline{a}.\overline{b} + \overline{a}.Ci-1 + \overline{b}.Ci-1}$$

Alors il effectue l'addition arithmétique en logique complémentée.

L'entrée a étant connectée en permanence avec $\overline{DB}$ et l'entrée b étant connectée soit avec $\overline{AB}$, soit avec une constante k, le circuit donne en sortie complémentée:

- soit l'addition des valeurs circulant sur les bus AB et DB - addition des opérandes pour le calcul d'adresses.
- soit l'addition de la valeur du bus DB et du complément de la constante k -la constante k étant composée par k1 connecté en b0, k2 connecté en b1 et k3 connecté en b2,b3...b31, permet d'effectuer les incrémentations décrémentations +1, +2, +4, -1, -2, -4.

Le circuit dual de l'AU est donné figure 33b ; il est obtenu en utilisant une analyse identique à celle utilisée dans le cas du pas de l'ALU.

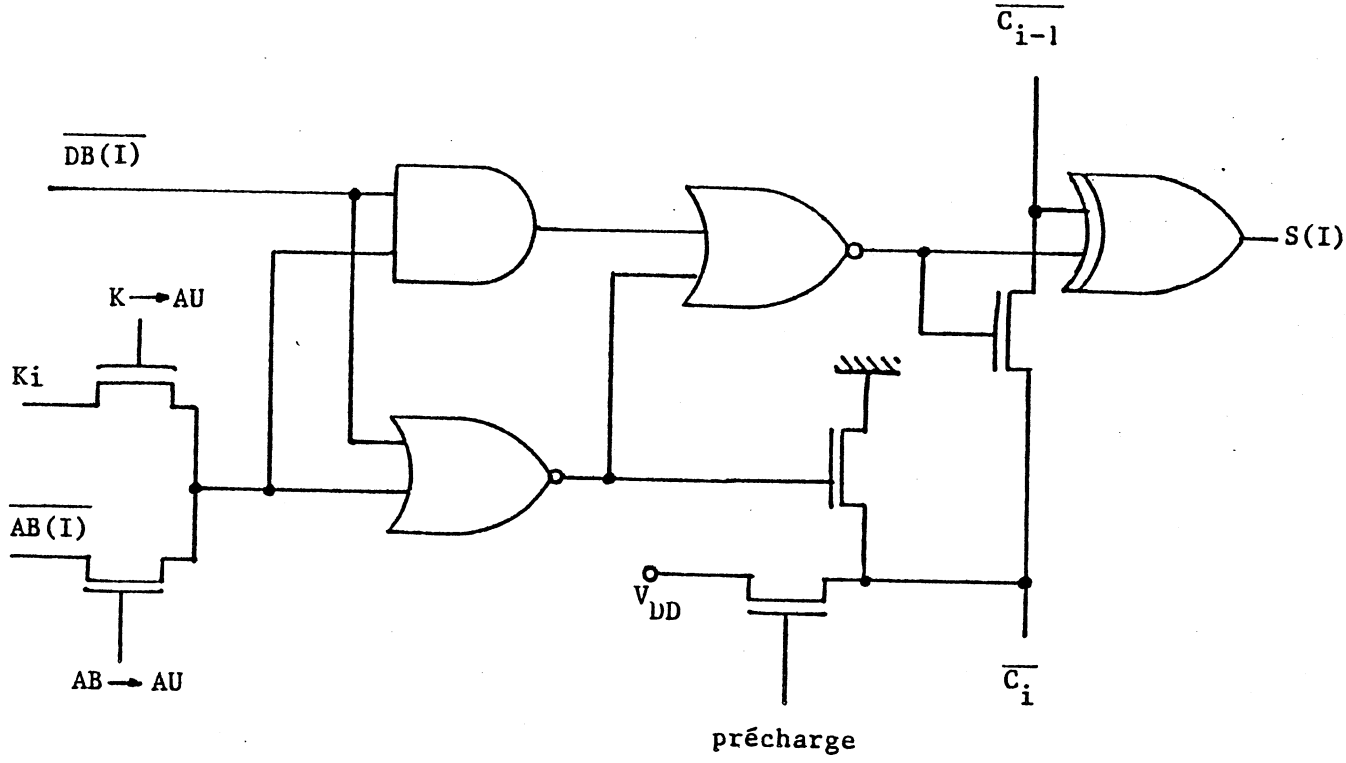


Figure 33a-Unité arithmétique (AU)

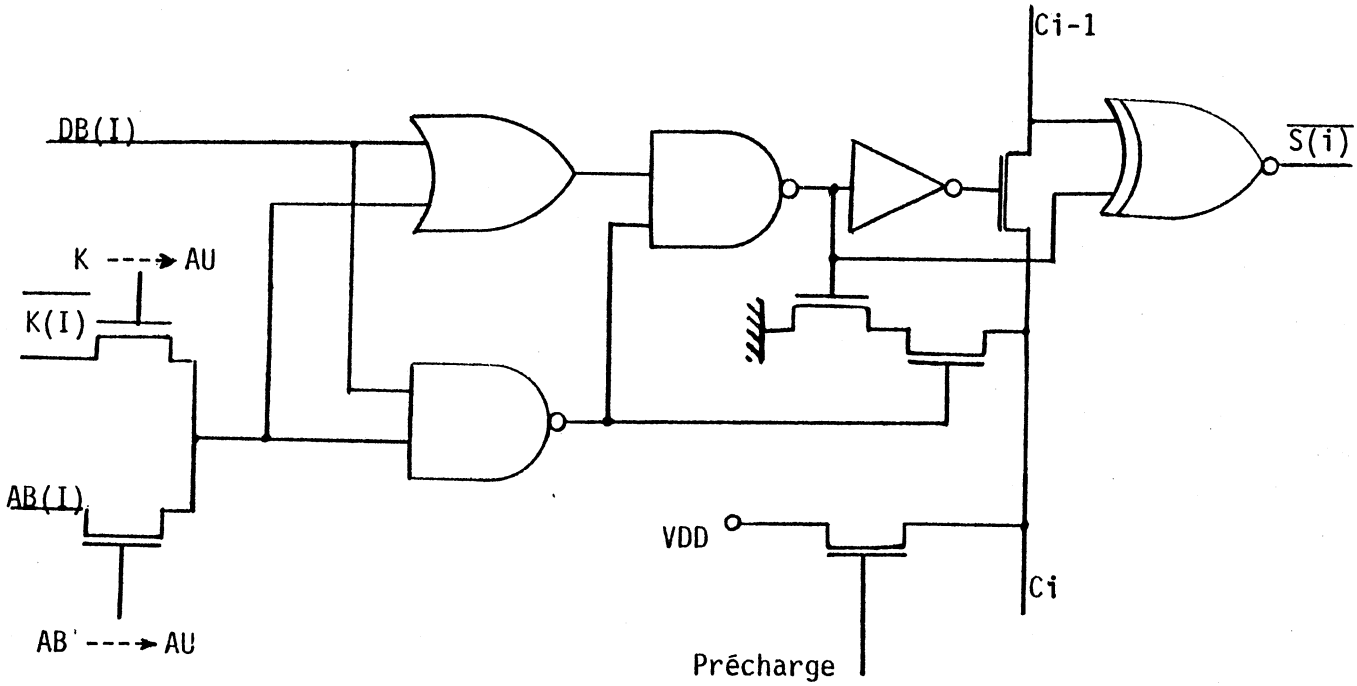


Figure 33b-Unité arithmétique,circuit dual

II.3.1.1. Circuit d'anticipation de la retenue

Le mécanisme d'anticipation de la retenue de l'AU est identique au mécanisme de l'anticipation de la retenue de l'ALU, en tenant compte bien sûr que cette fois il s'agit de la propagation de la retenue sur 32 bits et non sur 16 bits.

Un circuit identique au circuit direct sera utilisé pour l'anticipation de la retenue de l'AU complémentaire, pour les mêmes raisons données dans le cas du circuit d'anticipation de la retenue de l'ALU.

II.3.2. PREN

Cet opérateur est utilisé pour le traitement de l'instruction MOVEM.

Il donne à la sortie, en code binaire, une séquence des numéros des registres affectés par l'instruction MOVEM.

Le format de cette instruction est le suivant:

0	1	0	0	1	dr	0	0	1	Sz	Effective Address					
Register List Mask															

Le circuit PREN est constitué de deux parties:

- une partie séquentielle présentée figure 34, qui donne en code 1 parmi 16, une séquence des numéros des registres affectés par l'instruction MOVEM (bits à '1' du "register list mask").

- une deuxième partie (encodeur) présentée figure 35, qui donne en quatre bits, la même séquence, mais en code binaire cette fois.

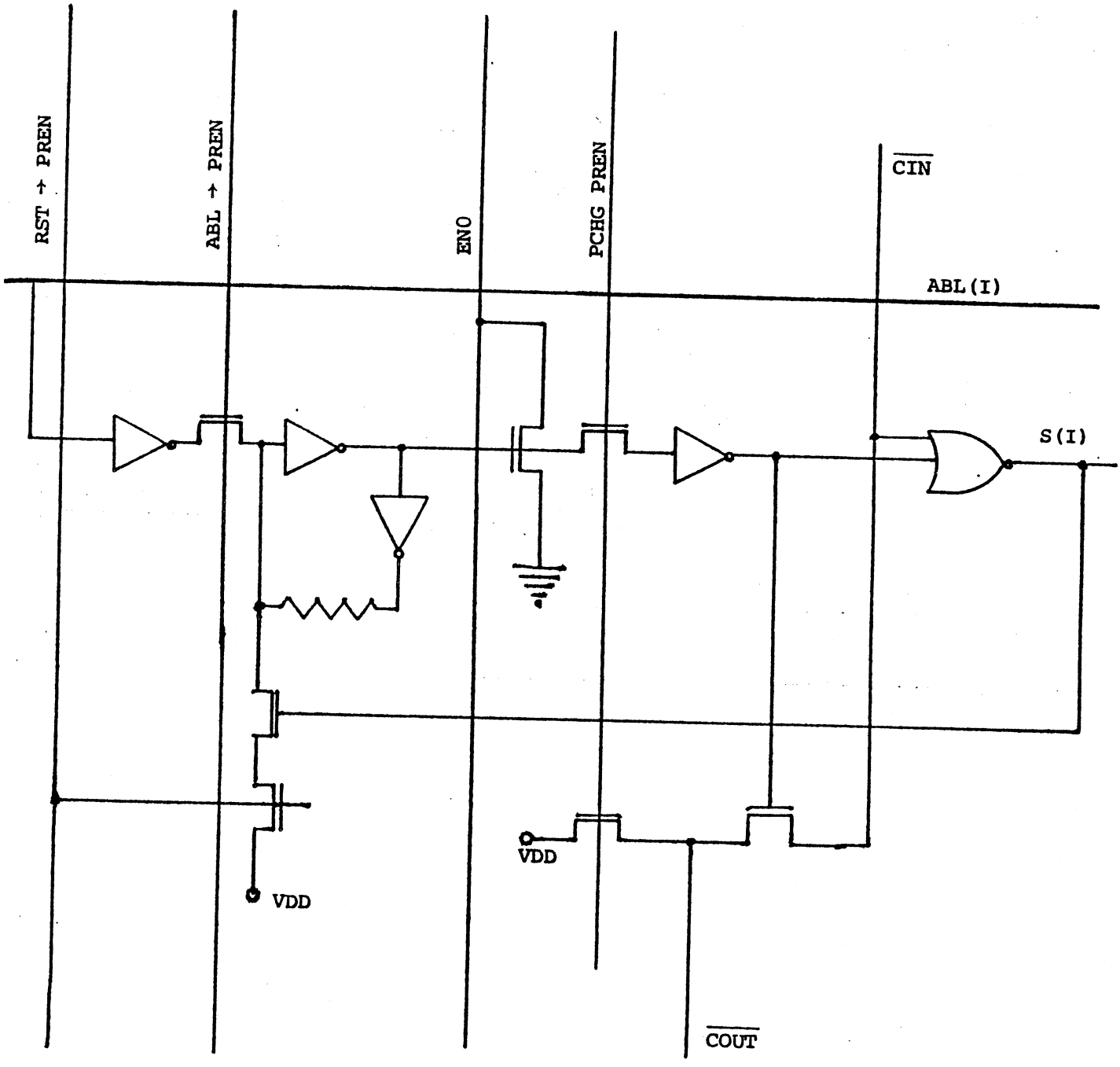


Figure 34- Pas de l'opérateur PREN du MC 68000 (coté registre).

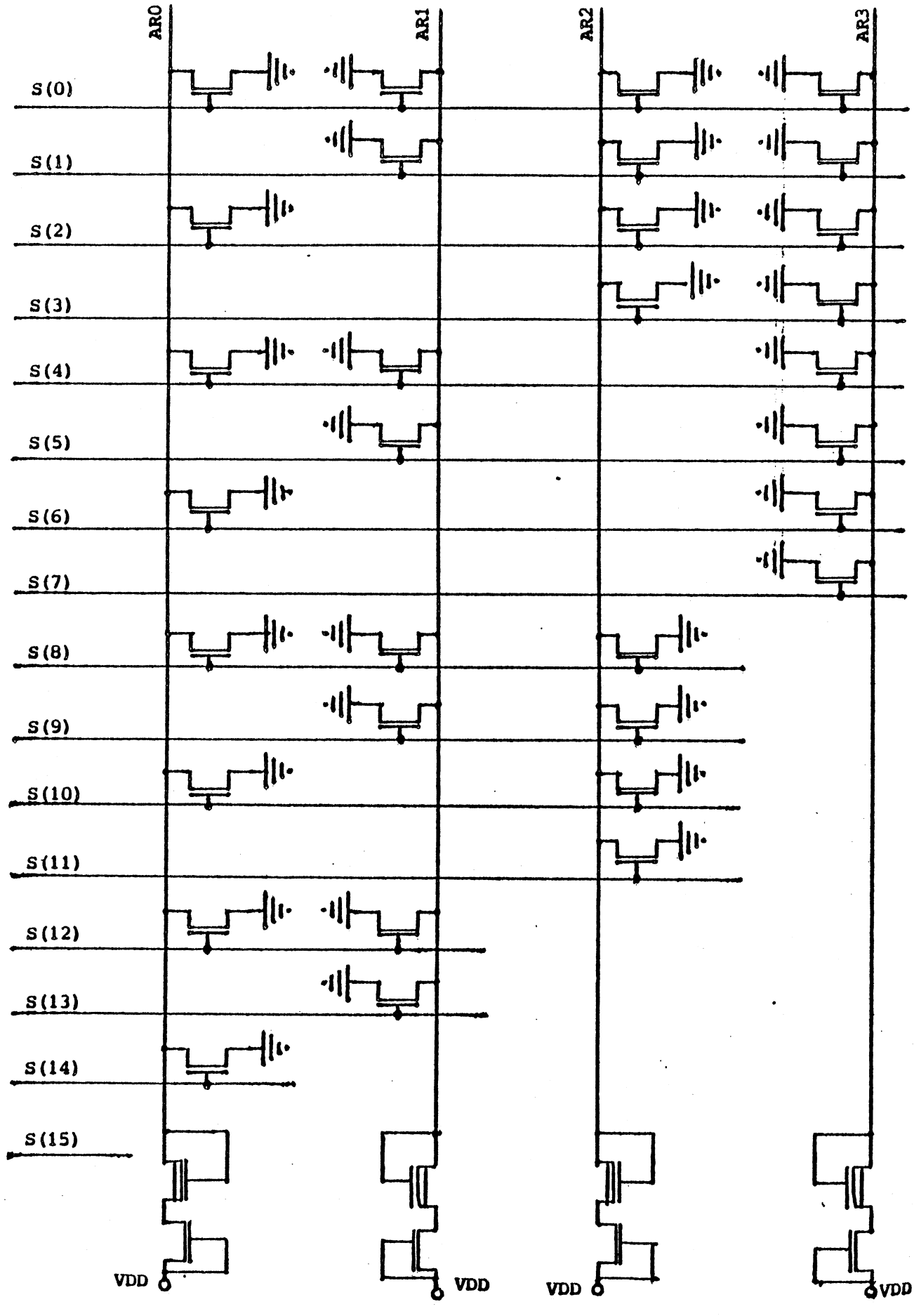


Figure 35 - Opérateur PREN du MC 68000 (coté encodeur).

Les quatre sorties du circuit sont utilisées dans le circuit de sélection des registres (PO) pour indiquer le numéro du registre à sélectionner.

La ligne du compte-rendu "en zéro" sert à indiquer la fin du traitement de l'instruction, i.e. tous les bits à '1' du champ "register list mask" ont été décodés.

Le fonctionnement de la partie séquentielle est éclairci par les explications suivantes:

La commande ABL->PREN sert à charger en T1 le registre PREN par le bus ABL.

Le circuit de retenue sert à valider à chaque cycle à la sortie du circuit séquentiel le premier bit (coté poids faible) du registre PREN qui est à l'état '1'. Le circuit de retenue est systématiquement préchargé en T1 par la commande PCHG PREN.

La commande RST->PREN sert à forcer à '0' à chaque cycle le bit du registre qui est validé.

Au moment où tous les bits du registre sont à '0' le signal "en zéro" est activé et la fin de l'instruction MOVEM est décodé au niveau du PLA de branchement.

Ainsi, le nombre de cycles du décodage des registres par le PREN est égal au nombre de bits du champ "register list mask" positionnés à '1'.

La figure 36 donne le timing du circuit.

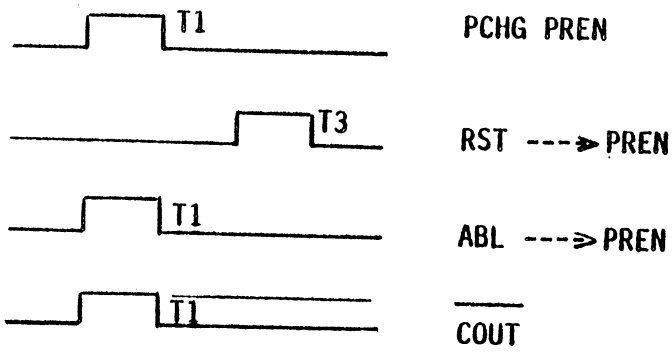


Figure 36- Timing du PREN

II.3.2.1. Circuit augmenté

L'autotest du circuit PREN sera assuré en utilisant un circuit identique (duplication pure).

Un contrôleur de "double rail" est utilisé pour contrôler les quatre sorties doublées, le contrôle sera fait après avoir commander les MOS du circuit de sélection des registres (PC), pour assurer aussi le test des interconnexions.

Le signal "en zero" doublé sera contrôlé après avoir commander les MOS du PLA de branchement.

II.3.3. SIGN EXT

La figure 37 présente l'opérateur SIGNEXT.

Cet opérateur effectue l'extension du signe du mot (16 bits) au long mot (32 bits).

Quatre cellules identiques sont utilisées dans ce circuit. La fonction logique de chaque cellule est équivalente à une porte NOR à deux entrées.

Deux cellules sont utilisées pour le bus DBH, l'une entrée de chaque cellule est la commande $\overline{\text{SIGNEX -> DBH}}$, tandis que l'autre est DBL15 pour une cellule et $\overline{\text{DBL15}}$ pour l'autre.

La sortie d'une cellule permet de connecter à la masse tous les bits DBH, tandis que la sortie de l'autre cellule connecte à la masse tous les bits $\overline{\text{DBH}}$.

Les deux autres cellules sont utilisées de la même façon pour l'extension du signe sur le bus AB.

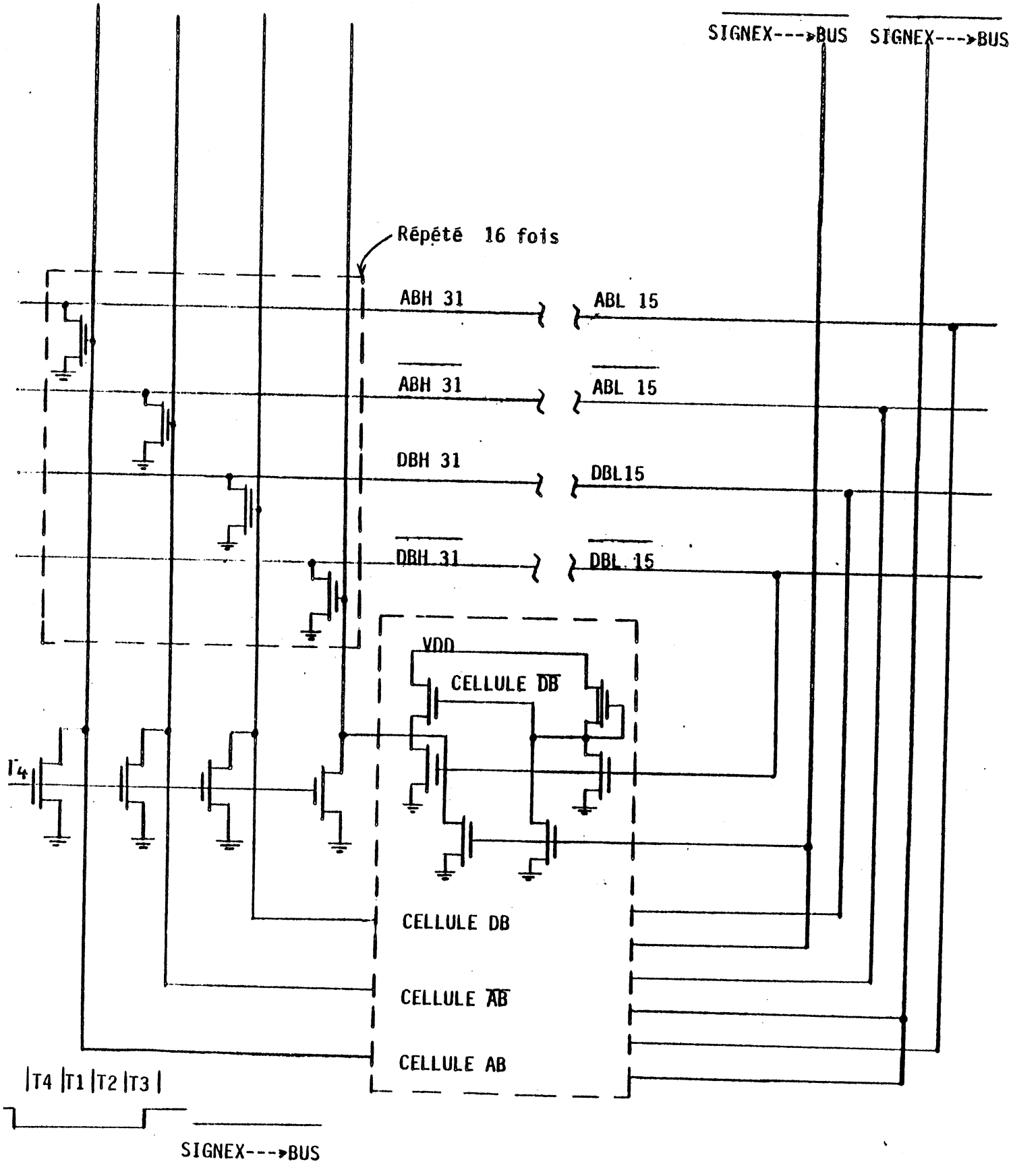


Figure 37- Opérateur SIGNEX

Etant donné que les lignes des commandes $\overline{\text{SIGNEX -> DBH}}$ et $\overline{\text{SIGNEX -> ABH}}$ sont testées après la prise de contact avec les quatre cellules le degré de divergence maximal du circuit est égal à 1 (figure 20). Alors la règle R1 (paragraphe III.1, 1er. partie) est respectée.

Ensuite on divise le circuit en deux blocs, l'un qui fait l'extension du signe sur le bus AB et l'autre sur le bus DB. La figure 38 donne la prise de contact de l'alimentation sur les circuits MASGXD.

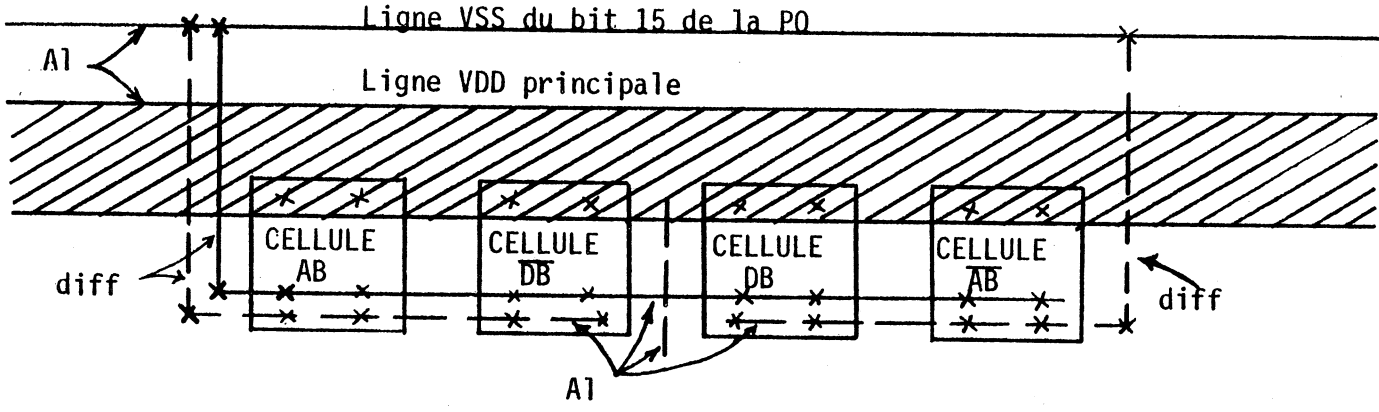


Figure 38

Dans cette figure, l'emplacement des circuits MASGXD et la prise de contact des alimentations est ceci du circuit original. Avec les deux lignes en pointillé on a désigné la modification qui doit être faite sur la prise de contact avec la ligne Vss dans le circuit autotestable. Une troisième ligne en pointillé connectée avec la ligne Vdd est utilisée pour enlever les courts-circuits redondants entre les deux lignes en pointillé Vss.

Avec ces modifications, chacun des deux blocs considérés respecte les règles R2 et R4 (paragraphe III.1, 1er. partie).

Ensuite il reste à examiner pour chacun des deux blocs les courts-circuits du type B2.

On peut vérifier qu'il n'existe pas de courts-circuits entre les cellules AB et la cellule $\overline{AB}$ car ces cellules sont éloignées entre elles.

On peut également éliminer les courts-circuits entre la cellule DB et la cellule DB en plaçant les quatre cellules dans l'ordre: cellule AB, cellule $\overline{DB}$, cellule $\overline{AB}$, cellule DB.

Avec ces modifications, chacun des blocs considérés devient SFS pour une technique de détection des erreurs simples. On peut alors utiliser le code de parité pour tester le circuit d'extension du signe.

Circuit augmenté:

Chacun des deux blocs "extension du signe sur le bus AB" et "extension du signe sur le bus DB" sera testé avec contrôle de parité de ces sorties.

La génération des bits de contrôle est simple puisque la commande $\overline{\text{SIGNEX -> ABH}}$ est le bit de parité impaire du premier bloc et la commande $\overline{\text{SIGNEX -> DBH}}$ est le bit de parité impaire du deuxième bloc.

Alors le seuls circuits complémentaires nécessités sont deux controleurs de parité impaire de nombre de bits d'information NI=2. Les deux contrôleurs seront placés en-dessus de la partie opérative pour tester les sorties du circuit après la prise de contact avec les 16 bits de chaque bus.

Le fonctionnement du SIGNEXT ne doit pas provoquer d'erreur de parité sur les bus ABH et DBH. Cette condition implique une valeur 1 pour le bit de parité du bus ABH ou DBH chaque fois que l'extension du signe sur ces bus est validée, étant donné que les bus sont codés en parité impaire, donc la solution de ce problème est donnée figure 39.

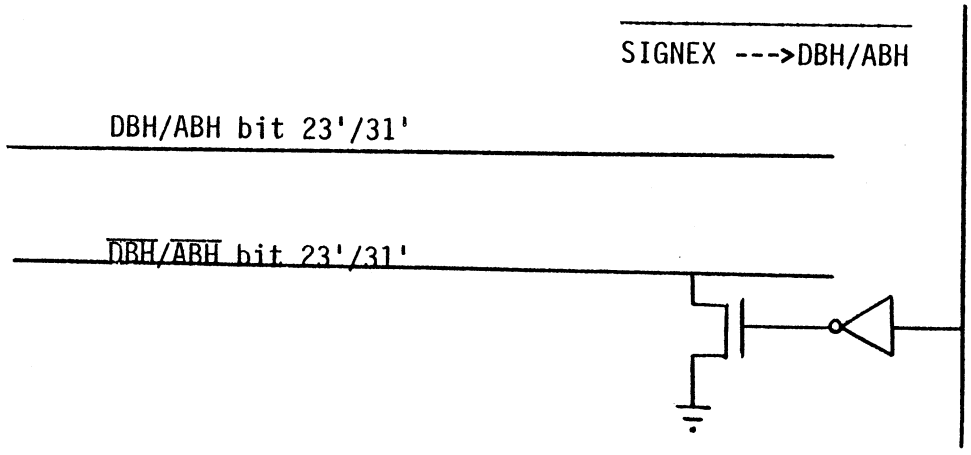


Figure 39-La génération de parité des Bus ABH et DBH en cas d' extention du signe

III - SCHEMA GENERAL ET EVALUATION DE LA SURFACE DE LA PO AUTOTESTABLE

Le schéma fonctionnel de la PO autotestable du MC 68000 est présenté ici. Ensuite l'évaluation de la surface et la comparaison avec la surface de départ sont réalisées. Les résultats obtenus sont comparés avec les résultats des solutions données en [DISP 81].

Le schéma fonctionnel de la "partie opérative autotestable" est présenté figure 40. La technique utilisée pour l'autotest de chaque bloc est aussi présentée.

La surface occupée par la partie opérative du MC 68000 plus les registres IR, IRC est donnée figure 41a (rectangle de 4,55mm x 1,5mm). Dans la même figure on présente la surface occupée par les buffers et les plots d'adresses et de données de même que la surface d'une zone autour de la PO qui contient les lignes principales d'alimentation et quelques lignes de transfert.

La surface occupée par le circuit autotestable est présentée à la figure 41b.

Surface des blocs de la figure 41a:

- partie opérative: S1 = 4,97mm x 1,62mm = 8,05mm2
- zone de routage: S2 = (a2+a3) x H + (L+a2+a3)a1 = 2,45mm2
- zone des buffers et des plots de données: S3 = 2,13mm2
- zone des buffers et des plots d'adresses: S4 = 2,84mm2

Surface des blocs de la figure 41b (circuit autotestable):

- la hauteur de la PO est augmentée de deux bits (x1 2/16) :

  H' = H x 1 2/16 = 1,83mm

L'augmentation de surface des circuits doublés est de 100% (x2). On doit ajouter ici deux bits de surface vide (x 2/16) en raison de l'augmentation de la hauteur de la PO.

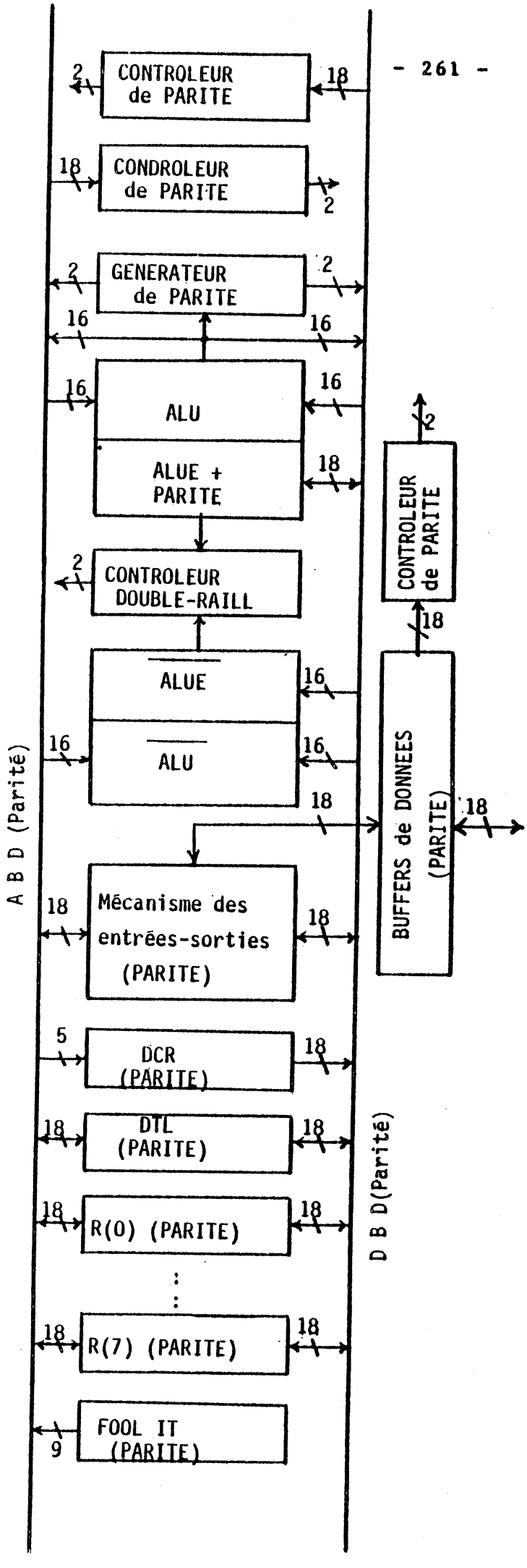


Figure 40- Schéma générale dela partie opérative autotestable.

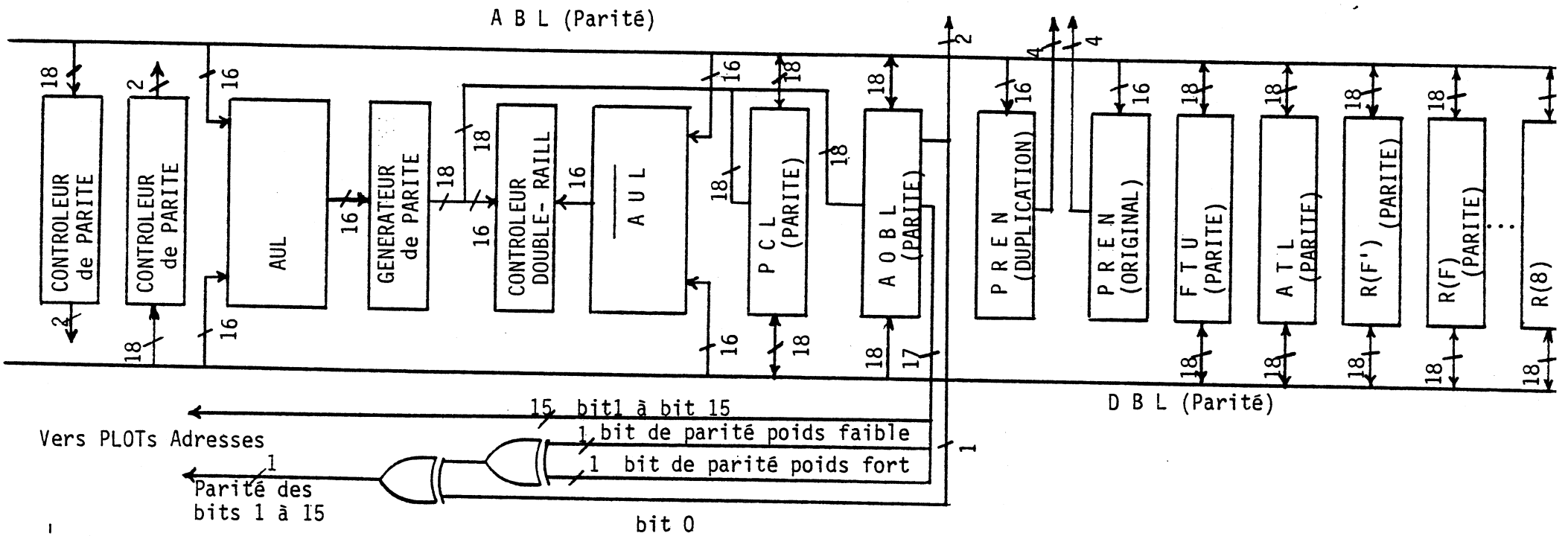


Figure 40- Schéma générale de la PO autotestable (swite)

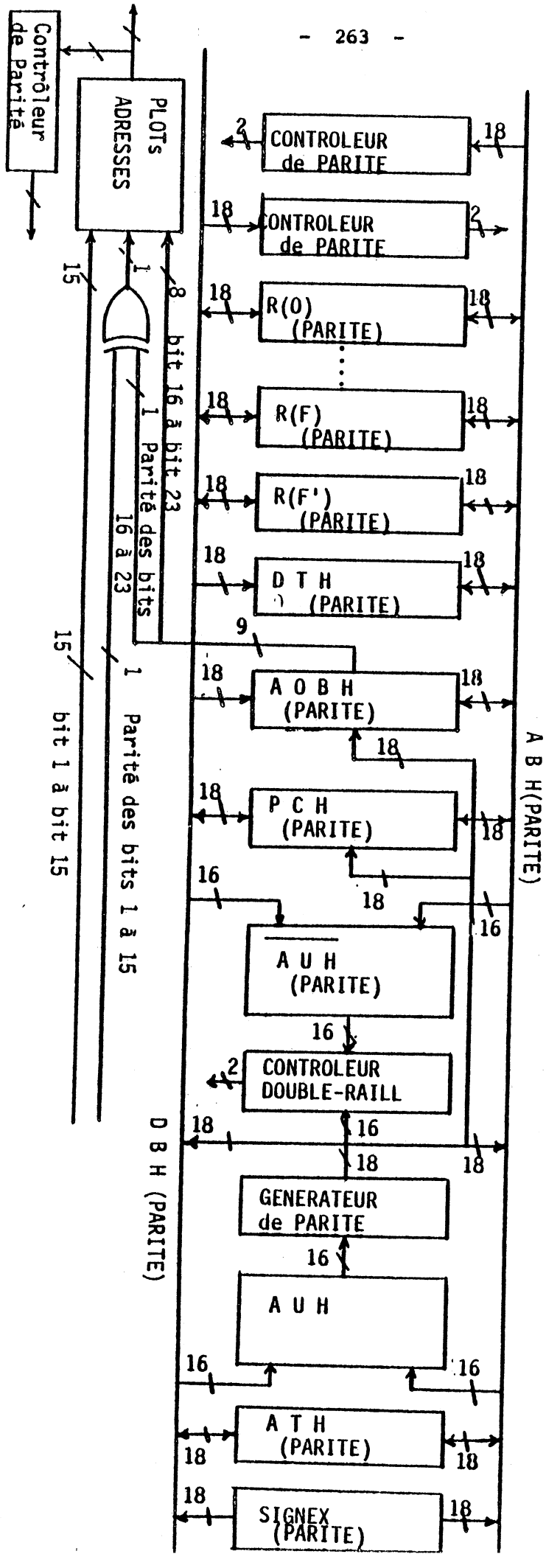


Figure 40 - Schéma général de la PO autotestable (swite)

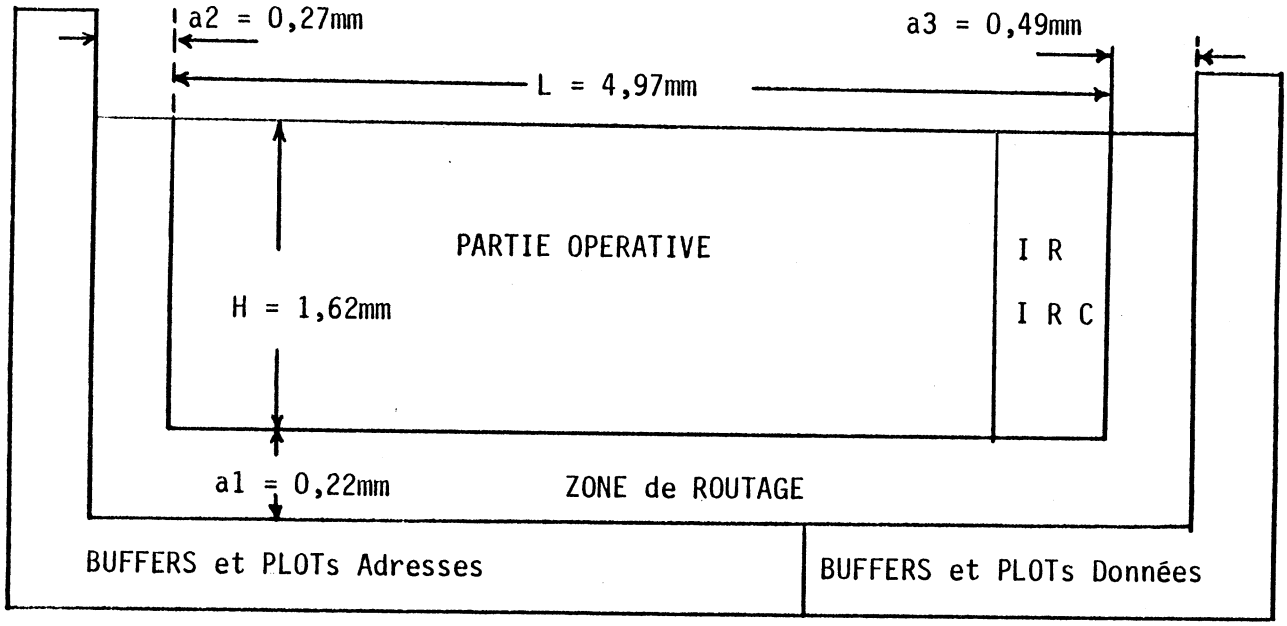


Figure 41(a)

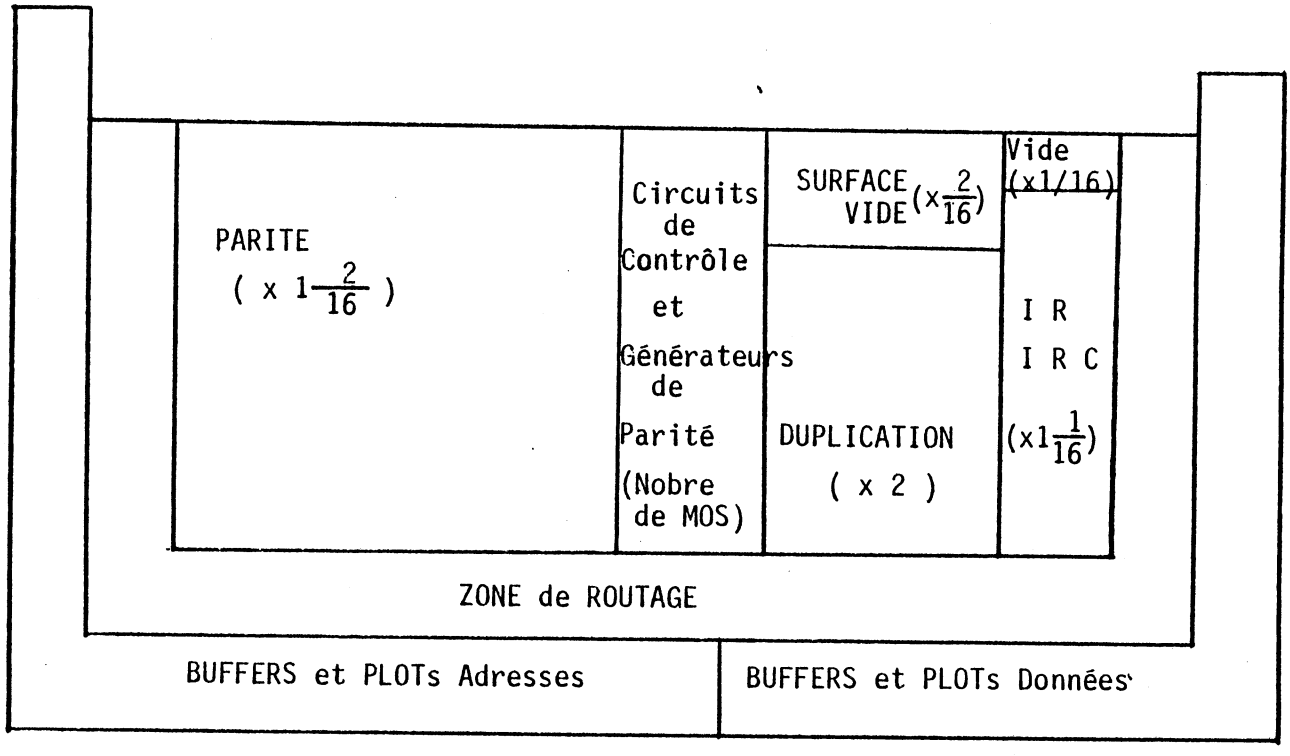


Figure 41(b)

Circuits doublés

ALU + ALUE : surface 1,62mm x 0,7mm = 1,13mm2
DER : surface 1,62mm x 0,22mm = 0,35mm2
AUL : surface 1,62mm x 0,22mm = 0,35mm2
AUH : surface 1,62mm x 0,22mm = 0,35mm2

surface totale .....= 2,18mm2

Surface de circuits doublés dans le circuit autotestable:
2,18mm2 x 2 x 1 2/16 = 4,9mm2

Circuits testés par la parité:

L'augmentation de surface de ces circuits est 2/16.

Surface de ces blocs = surface totale - Surface des circuits doublés
= 8,05mm2 - 2,19mm2 = 5,87mm2
Surface des circuits testés en parité dans le circuit autotestable
5,87mm2 x 1 2/16 = 6,60mm2

Générateurs de parité et controleurs

La surface de ces blocs sera calculée en utilisant le nombre de transistors, l'équation suivante est utilisée:
surface = k x NT x 13 x PP x PM [REI 83]

- k est le coefficient d'optimisation, il se situe dans l'intervalle:
k = 0,9 - conception optimisée
k = 1,1 - conception lache
on choisit :
k = 1.
- NT est le nombre de transistors (de signal et de charge).

- 13 est le nombre de pas de poly multiplié par le nombre de pas de métal par transistor, donné en [REI 83] comme résultat de statistique sur la logique aléatoire.
- PP est le pas de poly, PP = 9,25 micr. pour la PO de MC 68000
- PM est le pas de métal, PM = 12 micr. pour la PO de MC 68000.

Avec ces données, on obtient l'équation suivante:

surface = 1443 x NT micr.2

Le nombre de transistors des générateurs de parité, des circuits de contrôle de parité et des circuits de contrôle "double rail" (solution cellulaire) est donné en [CRO 80].

Selon la figure 40, on utilise :

- 3 contrôleurs "double rail" de 16 bits:
Le nombre de transistors de chaque contrôleur est NT = 10(16-1) = 150, donc au total NT1 = 3x150 = 450 transistors pour les 3 contrôleurs.

- 6 contrôleurs de parité (2 adjacent code, 16 bits d'information).
Le nombre de transistors d'un tel contrôleur est 7 x 16 + 3x2-10 = 108, donc au total NT2 = 6x108 = 648 transistors pour les 6 contrôleurs.

- 3 générateurs de parité (2 adjacent code, 16 bits d'information)
Le nombre de transistors d'un tel générateur est 7(16-2) = 98, donc au total NT3 = 3x98 = 294 transistors pour les 3 générateurs.

On doit encore ajouter:

- un générateur de parité (4 bits d'information), placé à l'entrée du registre du circuit DCR, nombre de transistors NT4 = 7(4-1) = 21 transistors. Les deux bits ajoutés au registre du DCR et le contrôleur de parité du registre du DCR ne sont pas considérés puisqu'ils sont placés dans l'espace vide, au-dessous du registre

du DCR (figure 31).

- un contrôleur "double rail" (4 bits d'information) placé dans la parité contrôle qui teste le circuit PREN. Nombre de transistors NT5 = 10(4-1) = 30.

- un contrôleur de parité (code 2 adjacent, 4 bits d'information). Pour le circuit SIGNEXT. Nombre de transistors NT6 = 7 x 4 + 3x2-10 = 24 transistors.

Alors le nombre total de transistors est:

NT = NT1 + NT2 + NT3 + NT4 + NT5 + NT6 = 1467

Surface = 1443 x 1467 micr2 = 2,1mm2

Surface totale de la PO autotestable :

S1' = (4,9+6,95+2,1)mm2 = 13,6mm2

L'évaluation de la surface du circuit de départ et du circuit autotestable est donnée dans la Table I.

Augmentation en surface :

$$\frac{S1' - S1}{S1} = 69\%$$

Si on compare ce résultat avec l'application faite par C. DISPARTE en [DIS 81] on trouve une différence de 40%. La raison en est que dans cette application, la PO est entièrement doublée, ce qui donne une augmentation en surface de 100%, plus 12% pour les contrôleurs et les encodeurs du bus adresse et du bus donnée, on obtient une augmentation de surface de 112%.

La longueur de la PO autotestable est :

$$L' = \frac{S1'}{H'} = \frac{13,60mm2}{1,83mm} = 7,43mm$$

- zone de routage (circuit autotestable):

$$S2' = (a2+a3) \times H' + (L'+a2+a3)a1 = 3,19mm2$$

- zone des buffers et des plots de données (circuit autotestable)
Ici on doit considérer la surface des buffers et des plots qui est égale à S3 x 1 2/16 = 2,39mm2, plus la surface de controleur de parité (2 adjacent code, 16 bits d'information), le nombre de transistors du contrôleur est 98 et sa surface est 1443 x 98micr.2 = 0,14mm2. Alors S3' = 2,39 + 0,14mm2 = 2,53mm2.

- zone des buffers et des plots des adresses (circuit autotestable)
Surface des buffers et plots: S4 x 1 1/23 = 2,96mm2 (1 bit de parité, 23 bits d'information).
Surface du contrôleur de parité: le nombre des transistors est 7(23-1) = 159 et sa surface est de 1443 x 154 micr.2 = 0,22mm2.
Alors S4' = 2,96 + 0,22mm2 = 3,18mm2.

La solution proposée dans [DIS 81] est plus économique pour la zone des buffers et des plots de données puisqu'un seul bit de parité y est utilisé, mais cette solution ne peut pas être appliquée car le transfert de données ne se fait pas seulement sur les 16 bits mais aussi sur les 8 bits.

Surface totale des blocs de la figure 41a:

$$S = S1 + S2 + S3 + S4 = 15,47mm2$$

Surface totale des blocs de la figure 41b (circuit autotestable)

$$S' = S1' + S2' + S3' + S4' = 22,50mm2$$

L'évaluation de la surface du circuit de départ et du circuit autotestable est donnée dans la Table II.

Augmentation en surface:

$$\frac{S' - S}{S} = 45,5\%$$

REMARQUE:
Au-dessous de la PO seront aussi placés les contrôleurs des

commandes. Le nombre et le type de ces contrôleurs dépendent des méthodes utilisées pour l'autotest de la PC. Ces contrôleurs ne seront pas pris en compte pour l'augmentation de la surface de la PO car leur rôle principal est le contrôle des blocs de la partie contrôle, ils sont placés au-dessous de la PO pour tester aussi les lignes de transfert des commandes.

Table I

	MC68000 Surface en mm2	MC68000 Autotestable Surface en mm2
Blocs doublés	Surface de départ: 2,18	Blocs doublés: 2,18 x 2 = 4,36 Surface vide: 4,36 x 2/16 = 0,54
Blocs testés en parité	Surface de départ: 5,87	Blocs codés: 5,87 x 1 2/16 = 6,6
Générateurs de parité et contrôleurs	---	Nbre. de transistors x1443: 1467 x 1443 m2 = 2,1
	8,05	13,6

Table II

	MC68000 Surface en mm2	MC68000 Autotestable Surface en mm2
Partie Opérative	8,05	13,6
Zone de routage (PO)	2,45	3,19
Buffers et plots données	2,13	2,39
Contrôleur de parité (bus données)	----	0,14
Buffers et plots adresses	2,84	2,96
Contrôleur de parité (bus adresses)	----	0,22
	15,47	22,50

REFERENCES

[COU 81] COURTOIS B.
Failure mechanisms, fault hypotheses and analytical testing of LSI-NMOS (HMOS) circuits
VLSI 81, University of Edinburgh, August 18/21 1981
U.K., Academic press

[CRO 80] CROUZET Y., LANDRAULT C.
Design of self checking MOS LSI circuits, application to a four-bit microprocessor
IEEE Trans. Comp. pp 532/537, June 1980.

[DIS 81] DISPARTE C.P.
A design approach for an electronic engine controler self checking microprocessor
EUROMICRO 80, Paris, September 1981.

[REI 83] REIS DA LUZ R.A.
Evaluateur topologique prédictif pour la génération automatique des plans de masse de circuits VLSI
IMAG, Computer arcitecture group,
Thèse de docteur-ingénieur, janvier 1983.

CHAPITRE II

EVALUATION D'UNE PARTIE CONTROLE AUTOTESTABLE POUR LE MC 68000

RESUME

Ce chapitre présente l'etude d'une partie contrôle (PC) autotestable pour le microprocesseur MC 68000.

La présentation générale de la partie contrôle est d'abord donnée, ensuite la méthode dévelopée dans la première partie de cette étude est utilisée pour assurer l'autotestabilité des différents blocs de la partie contrôle.
La conclusion présente l'augmentation en surface necessaire pour la conception d'un MC 68000 autotestable,cette augmentation est de l'ordre de 48%.

SOMMAIRE

I - PRESENTATION GENERALE DE LA PARTIE CONTROLE DU MC 68000

La partie contrôle du MC 68000 est présentée à la figure 1. Elle occupe presque 70% de la puce. Le séquenceur est la partie la plus importante . Il est conçu senon la principe d'un contrôle microprogrammé, combiné avec le principe des PLAs multiples [OBR 82].
Ce séquenceur est composé de:

- La ROM de contrôle (31 bits, 1/6 de la surface de la puce)- Elle contient le microprogramme décrivant l'algorithme d'interprétation des insttructios sous forme de l'algorithhme de MEALEY,c'est-à-dire les commandes de la partie opérative ne sont pas définies uniquement par l' état du séquenceur (microinstruction) mais aussi par le code opération qui est décodé par le PLA des paramètres (PLA IRD). Ainsi, les microinstructions sont utilisées pour distinguer les classes des opérations à exécuter, ce qui conduit à une économie considérable.

- Les PLAs A1, A2, A3 qui fournissent, à partir du décodage des codes opérations, l'adresse des microprogrammes de traitement.

- Le PLA A0 d'exception, chargé de prendre en compte les signaux d'exceptions, tel qu'une demande de RESET, ou d'interruption, ou encore les cas d'erreurs comme par exemple les erreurs d'adresses, de mode privilégié, de code opération invalide. Ce PLA fournit l'adresse d'initialisation des microprogrammes des exceptions.

- Le PLA de branchement activé par un bit des microinstructions faisant appel à un branchement conditionnel. Il fournit deux bits venant se substituer à deux bits de l'adresse de la microinstruction suivante. Les deux bits de branchement sont calculés à partir de comptes-rendus de la partie opérative, de valeurs du registre d'état et de bits du code opération . Les conditions à tester sont définies par 5 bits de la microinstruction.

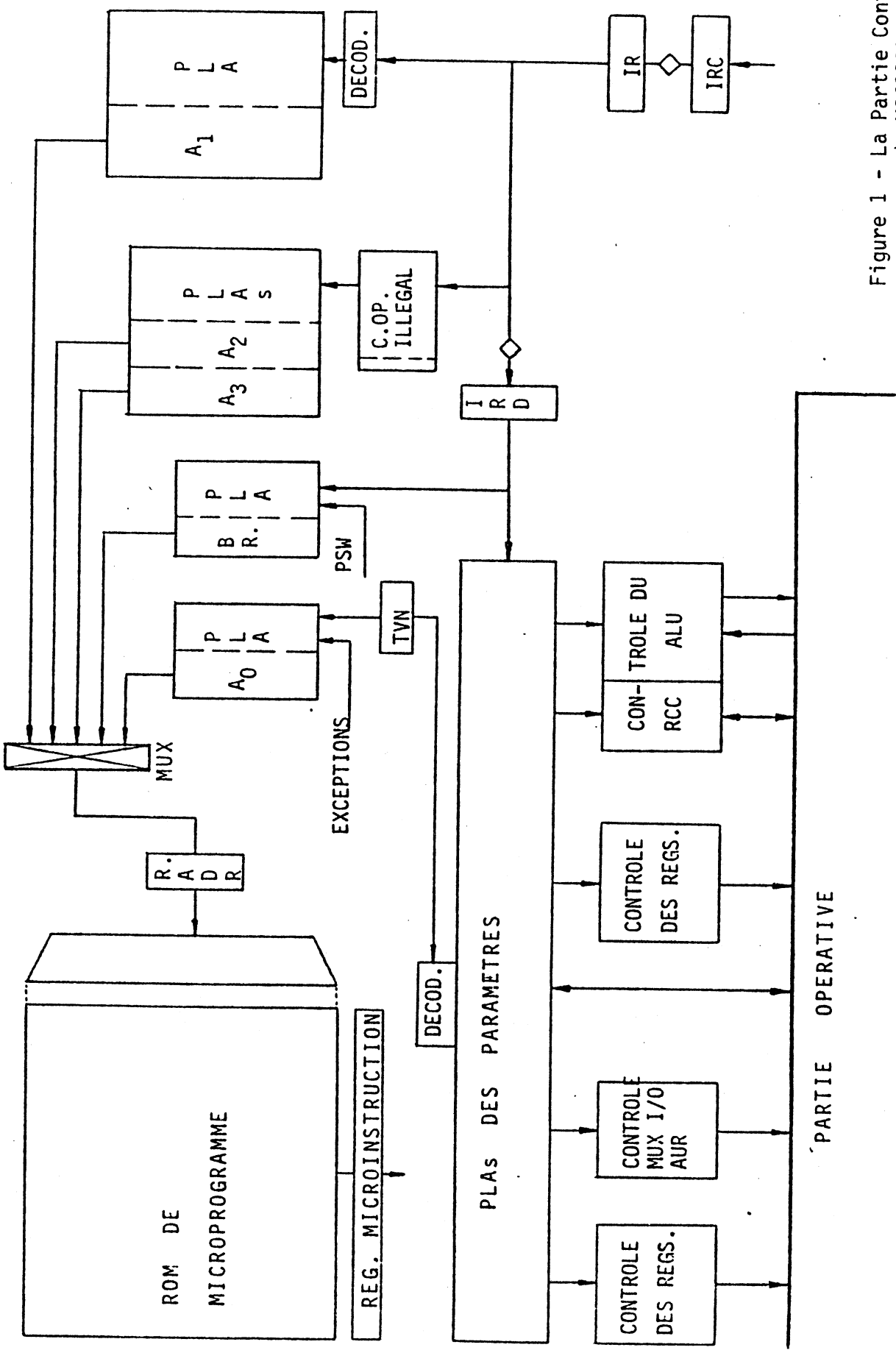


Figure 1 - La Partie Contrôle du MC68000

En plus de ces circuits, la partie contrôle contient aussi le PLA de paramètres IRD. Son rôle est d'extraire directement du code opération les informations concernant les commandes de chaque bloc de la partie opérative. Ces informations sont utilisées pour la génération des commandes précises concernant chaque instruction. On rappelle que le séquenceur (automate de MEALEY) ne peut distinguer seulement que des classes d'instructions. Les sorties de la ROM et les sorties du PLA IRD sont combinées en suite dans les circuits de contrôle des différents blocs de la partie opérative pour générer les commandes de cette dernière.

On peut distinguer les circuits de contrôle suivants: contrôle de l'ALU et contrôle du code condition, contrôle (décodeur) des registres poids faibles, conrôle de l'AU, MUX I/O, FOOL IT, contrôle (decodeur) des registres poids forts.

On peut aussi distinguer:
*Le registre d'état constitué par trois bascules mémorisant le niveau du masque d'interruption (PSW) ; 5 bascules mémorisant les codes condition (PSW) ; 2 bascules mémorisant les bits S (mode superviseur) et T (mode trace) (PSW) ;

*5 bascules superviseurs contenant: les bits de code de fonction FC0, FC1, FC2 définissant le type d'accés demandé (donnée utilisateur, programme utilisateur, donnée superviseur, programme superviseur, reconnaissance d'interruption). Les bits FC0, FC1 sont générés à partir des bits 16, 17 des microinstructions et le bit FC2 recopie la bascule S. Une information sur le sens de transfert des données (R/W), un bit indiquant si on est en traitement d'une exception (E).

*Le PLA TRAP de vectorisation des exceptions.

*Le Field Translate Unit permet, par l'intermédiaire du bus BC de 16 bits, de relier le PLA IRD, le PLA TRAP, les registres d'état d'une part et le registre FTU d'autre part; ce dernier est relié à

son tour aux bus adresses et données de la partie opérative.

Par cette unité sont effectués les transferts vers la partie opérative des données contenues dans le code opération et qui sont nécessaires dans les cas des instructions (ADDQ, SUBQ, MOVEQ, MUL, DIV, SHIFT, TAS), le transfert en partie opérative des vecteurs des exceptions générés par le PLA TRAP et quelques échanges entre le registre d'état et la partie opérative.

Les circuits suivants sont géographiquement localisés au-dessus de la ROM et des PLAs:
- le circuit de logique d'interruption
- le circuit de génération des horloges
- la logique de contrôle du bus, constitué par les blocs:
  - l'automate d'arbitrage d'attribution de bus
  - le circuit de commande de l'automate de contrôle du bus asynchrone
  - l'automate de contrôle de bus asynchrone.

II - UNITE DE CONTROLE MICROPROGRAMMEE

Le circuit de séquencement est présenté à la figure 2 . Il se compose de la ROM de microprogramme des PLAs A1, A2, A3 de décodage des codes opérations, du PLA d'exceptions A0 et du PLA de branchement.

Etant donné la grandeur de la surface occupée par ces circuits, on a intérêt à faire une étude de ces parties plus ou moins détaillée, afin de minimiser l'augmentation en surface nécessaire pour l'autotest de ce bloc. De l'autre coté, la régularité de ce bloc rend cette étude possible sans avoir besoin des détails non accessibles de ces parties (contenu de la ROM et des PLAs), et sans avoir besoin de redessiner ces circuits ce qui demanderait un travail qui ne s'inscrit pas dans le cadre de cette étude.

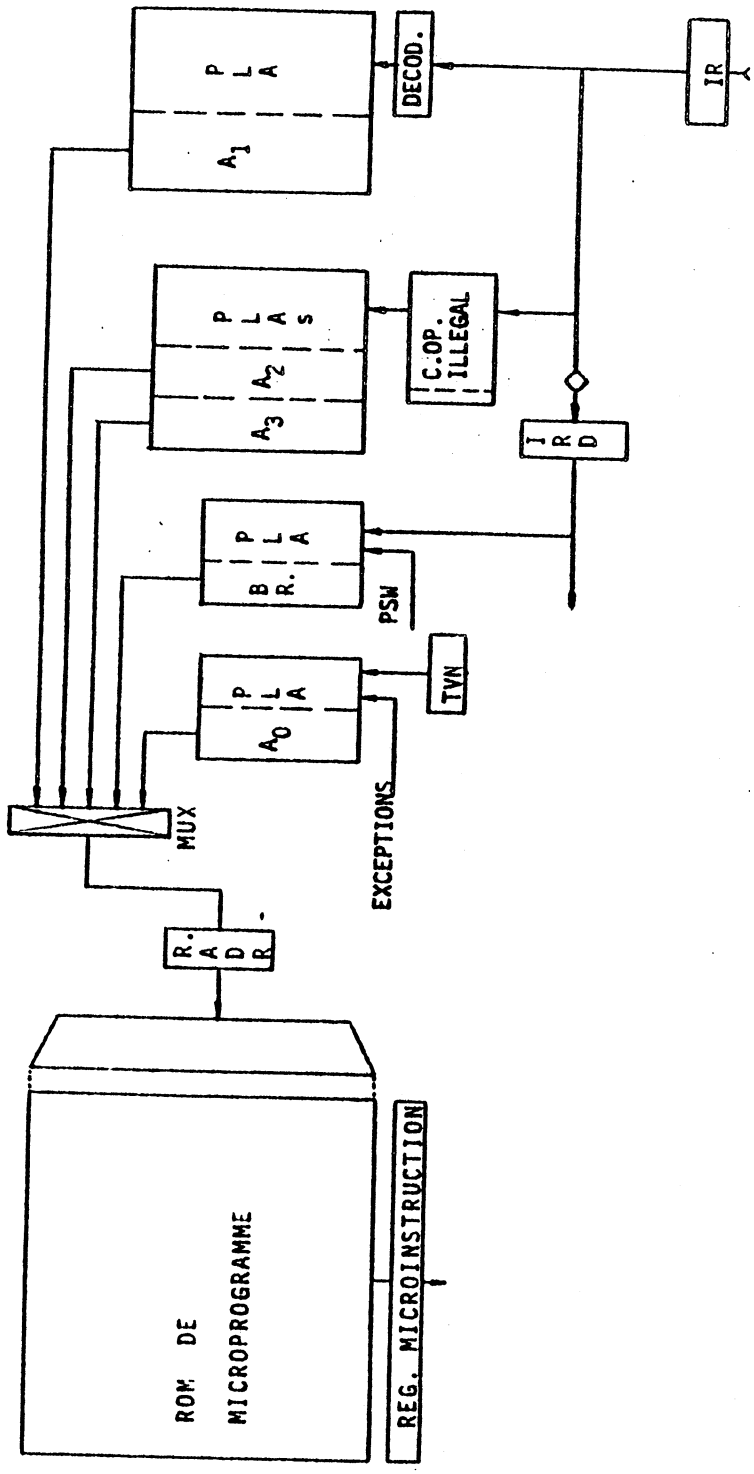


Figure 2- Le Séquenceur du MC68000

II.1. La ROM

Elle contient le microprogramme décrivant l'algorithme d'interprétation des instructions sous la forme de MEALEY. Elle occupe 1/6 de la puce et contient 31 Kbits.

Topologiquement et fonctionnellement, elle est divisée en deux parties:

- la micro-ROM contenant les 16 bits de la partie enchaînement des microinstructions servant au séquencement du microprogramme ; ceux-ci se situent sur la partie haute de la ROM. Il existe 544 micromots d'enchaînement répartis en 34 lignes, en fait seuls 519 de ces micromots sont utilisés.
- la partie action des microinstructions est composée de 68 bits servant à la génération des commandes de la PO ; cette partie action se situe sur la partie basse de la ROM. Il existe 336 mots d'action répartis sur 84 lignes.

Une instruction microprogrammée est constituée par une partie enchaînement et une partie action ; on peut remarquer que le nombre des micromots d'action est bien inférieur au nombre des micromots d'enchaînement, ce qui veut dire qu'à certains micromots d'enchaînement correspondent plusieurs micromots d'action. En fait, parmi les 10 bits d'adressage de la ROM (A0...A9) les bits A2 et/ou A3 peuvent être ignorés dans certains cas pour le décodage lignes de la partie action.

Cette solution permet de diminuer considérablement le nombre des micromots d'action et permet dans la version actuelle un gain de bits d'environ 30%.

Le rôle des bits d'adresses A0 à A9 dans le décodage des lignes et des colonnes est donné ci dessous:

- décodage des colonnes: parties enchaînement
bits A0, A1, 1 colonnes parmi 4
bits A2, A3, 1 colonne parmi 16
- décodage des lignes: parties enchaînement

bits A4, A5, A6, A7, A8, A9, décodage ligne
- décodage des colonnes: parties action
bits A0, A1, 1 colonne parmi 4
- décodage des lignes: parties action
bits A4, A5, A6, A7, A8, A9 décodage ligne
bits A2, A3 décodage ligne mais quelques fois A2 ou A3
ou les deux sont ignorés.

Le schéma de la ROM est donné à la figure 3. La figure 4 présente le circuit de decodage des colonnes ; trois circuits sont utilisés identiques à celui-ci: un pour le decodage colonnes de la partie action et deux pour le decodage colonnes de la partie enchaînement.

Seules les colonnes selectées (68 colonnes) de la nano-ROM sont préchargées et connectées sur les sorties (colonnes sélectées par A0, A1). 68 colonnes de la micro-ROM sont préchargées et seules les colonnes sélectées par A0, A1, A2, A3 (17 colonnes) sont connectées aux sorties de la micro-ROM.

Le circuit de décodage des lignes est donné figure 5. Les bits A2 et A3 sont utilisés uniquement pour le décodage ligne de partie action (et pas toujours).

Notons que les circuits de décodage de lignes et des colonnes de la partie action et de la partie enchaînement ne sont pas les mêmes ; ce détail est important pour l'autotest du circuit.

Le microprogramme est stocké dans la ROM où la présence d'un MOS sur une colonne tire la colonne à la masse (valeur zero) et l'absence d'un MOS laisse la colonne préchargée (valeur '1'). Cette structure est présentée figure 6.

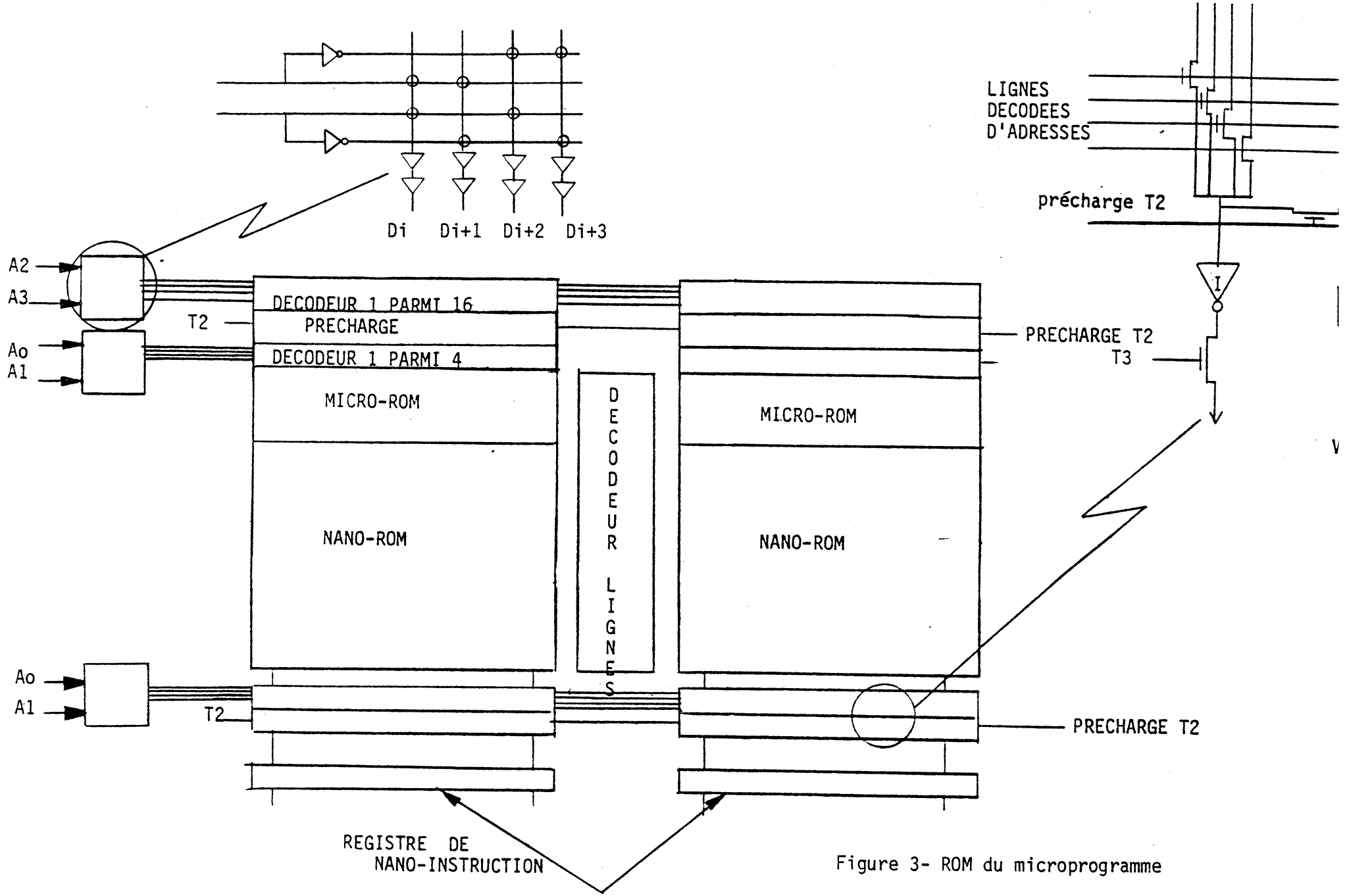


Figure 3- ROM du microprogramme

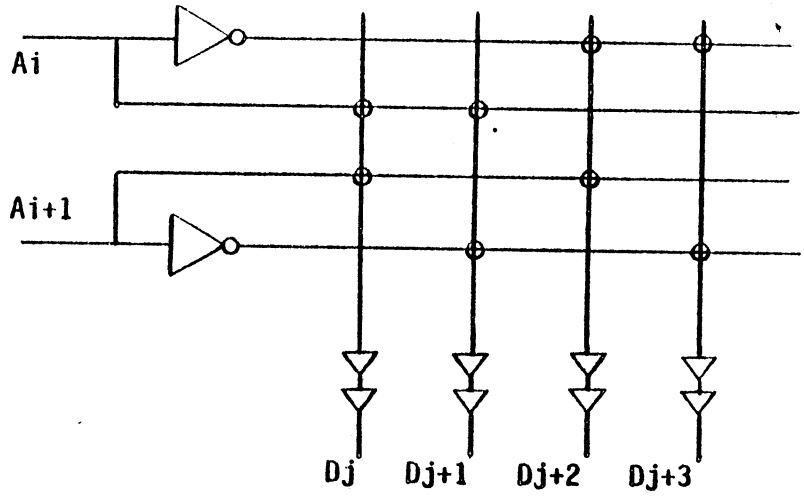


Figure 4:

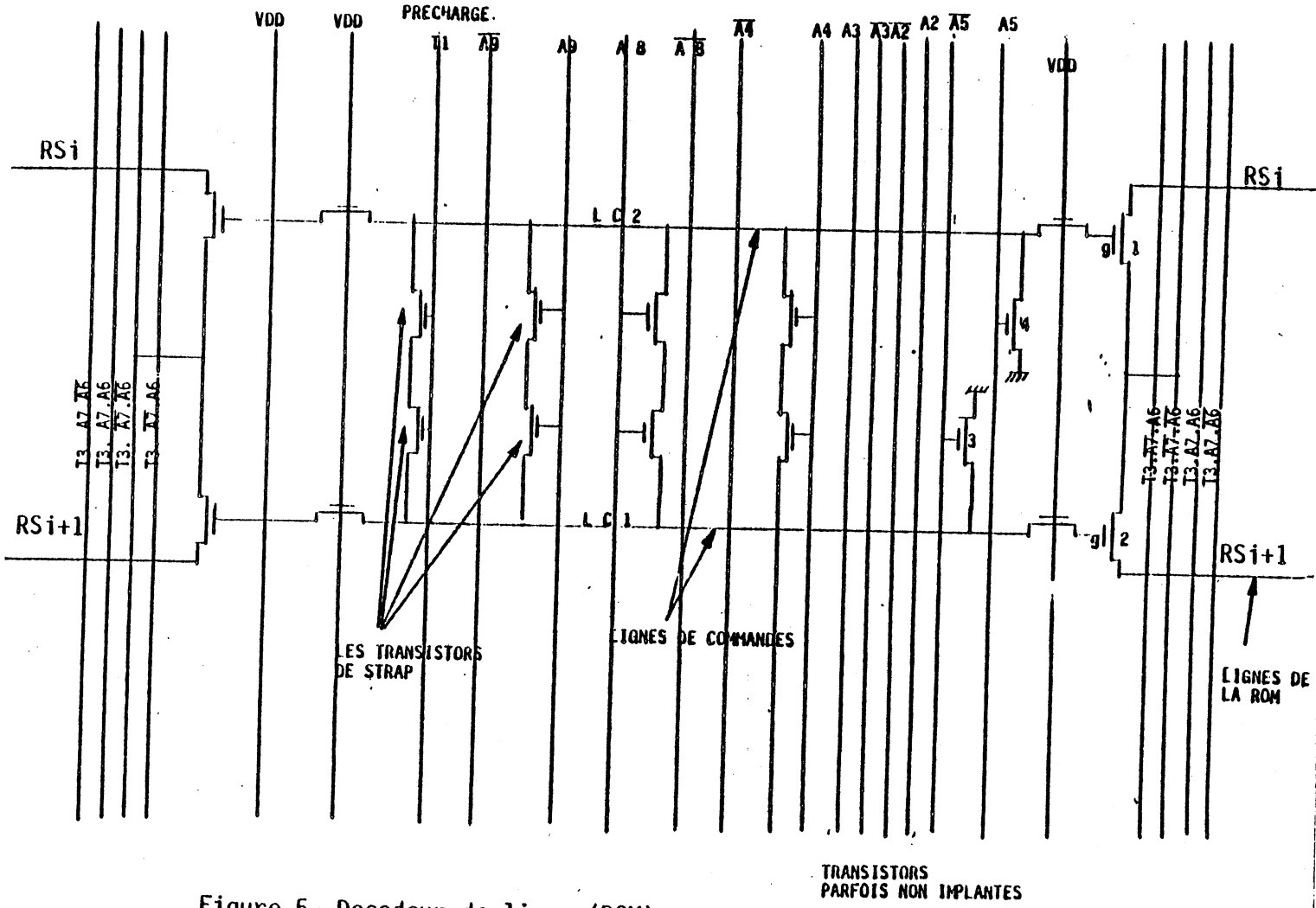


Figure 5- Decodeur de ligne (ROM)

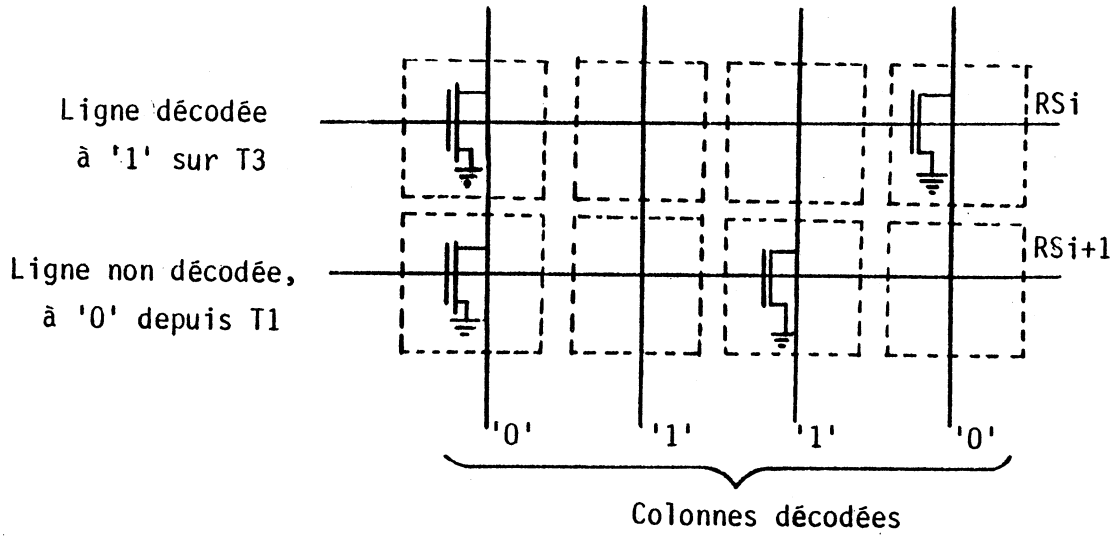


Figure 6 - Stockage des micromots dans la ROM

II.2. Unité de contrôle augmenté

Plusieurs propositions existantes dans la littérature et concernant la réalisation autotestable des unités de contrôle microprogrammées, sont examinées afin de choisir la solution optimale du point de vue recouvrement des pannes et augmentation en surface.

Ces propositions étant générales, sont basées sur des représentations plus ou moins fonctionnelles des unités de contrôle microprogrammées. L'analyse de ces propositions a donné des résultats concernant le séquenceur du MC 68000 qui ne sont pas satisfaisants.

Etant donné l'importance du séquenceur du point de vue surface occupée et du point de vue fonctionnement du microprocesseur, une autre solution a été recherchée afin de minimiser l'augmentation en surface et obtenir un recouvrement de 100% des pannes de la classe I. Une solution très satisfaisante a été obtenue, basée sur les règles données dans la première partie de cette étude, mais qui nécessite une connaissance détaillée de la structure du circuit.

II.2.1. Analyse des défaillances de la ROM de contrôle

a/ Bloc principal

Le bloc examiné ici est composé des blocs suivants (Figure 3): la partie de stockage des micromots (enchaînement et actions), les circuits de sélection '1 colonne parmi 4', les circuits de précharge et les registres (micro-instruction, nano-instruction). Le stockage des micromots est basé sur le principe detaillé dans la Figure 6.

L'analyse de parité d'inversion sur les chemins des blocs indiqués ci-dessus nous permet de vérifier que la règle R5 (section III.2.1, lère. partie) est respectée par les lignes du bloc, on peut alors vérifier qu'une erreur simple sur une ligne Di ne provoque que des

erreurs unidirectionnelles aux sorties de la ROM. On peut vérifier aussi qu'une erreur unidirectionnelle aux lignes Di ne provoque que des erreurs unidirectionnelles aux sorties de la ROM. Notons aussi qu'une erreur unidirectionnelle aux lignes RSi ne provoque que des erreurs unidirectionnelles aux sorties de la ROM.

La règle R6' (section III.2, lère. partie) est respectée par les lignes d'alimentation du bloc (lignes VSS des MOS de stockage, lignes VDD de précharge, lignes VSS et VDD des latches).

Donc les erreurs provoquées aux sorties de la ROM par les pannes de la classe I sont unidirectionnelles.

b/ décodeur colonnes

Le circuit de décodage des colonnes est donné figure 4. Un tel circuit est utilisé pour la partie action et deux tels circuits sont utilisés pour la parties enchaînement de la ROM. Une coupure sur une ligne ou un court-circuit entre les deux lignes d'entrée peut provoquer l'adressage d'un micromot d'enchaînement ou d'un micromot d'action qui ne correspond pas à l'adresse correcte.

Pour toutes les autres lignes du circuit, une erreur simple sur une ligne ne sera propagée qu'en une erreur unidirectionnelle aux sorties Dj, Dj+1, Dj+2, Dj+3.

A leur tour ces erreurs sont propagées en erreurs unidirectionnelles aux sorties de la ROM comme remarqué précédemment dans l'analyse du bloc principal.

D'autre part, la règle R6' est facilement verifiable.

Donc les pannes de la classe I affectant ce bloc ne peuvent provoquer que des erreurs unidirectionnelles aux sorties de la ROM, exception faite des pannes affectant les deux entrées primaires du bloc.

c/ décodeur lignes

L'analyse du decodeur lignes (figure 5) montre qu'une erreur sur les lignes d'entrée (A6, A7, A2, A3, A4, A5, A8, A9) peut resulter en l'adressage d'un micromot (partie enchaînement et/ou partie action) ne correspondant pas à l'adresse correcte.

Pour toutes les autres lignes du circuit on peut vérifier qu'une erreur simple sur une ligne ne peut provoquer qu'une erreur unidirectionnelle aux sorties RS.

A son tour, une erreur unidirectionnelle sur les lignes RS est propagée en erreur unidirectionnelle aux sorties de la ROM.

Donc les pannes de la classe Cl affectant ce bloc ne peuvent provoquer que des erreurs unidirectionnelles aux sorties de la ROM, exception faite des pannes affectant les entrées primaires du bloc.

Alors l'analyse des défaillances de la ROM de microprogramme montre que les pannes de la classe Cl peuvent provoquer deux types d'erreurs:

- une panne aux entrées du décodeur lignes ou du décodeur colonnes provoque l'adressage d'une microinstruction (partie enchaînement et/ou partie action) qui ne correspond pas à l'adresse correcte ;
- toutes les autres pannes ne peuvent provoquer que des erreurs unidirectionelles aux sorties de la ROM.

II.2.2. séquenceurs autotestables proposés en littérature

Quelques propositions représentatives des séquenceurs autotestables proposées dans la littérature sont présentées. Les raisons pour lesquelles elles ne sont pas adoptées ici sont expliquées.

La technique utilisée par la BELL TELEPHONE LABORATORIES (BTL) pour le No3A Electronic Switching System (ESS) [TOY 78], est présentée à la figure 7.

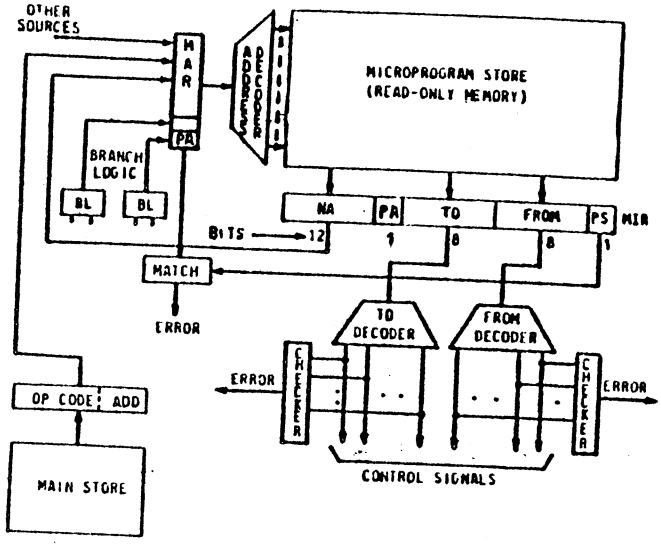


Figure 7-No 3A ESS, sequenceur microprogrammé autotestable

Dans cette proposition, le code m parmi 2m est utilisé pour le test des bits de contrôle (champ 'FROM' et champ 'TO'). Le test des bits d'adresse (champ NA) est effectué par deux bits de parité (PA, PS) ; le bit PA est mémorisé dans MAR (Micro-Adress Register) et il est comparé, dans le controleur MATCH, avec le bit PS de la microinstruction suivante.

La logique de branchement est doublée mais les sorties des deux blocs ne sont pas comparées, l'un des deux blocs fournit au registre MAR le bit de branchement et le second est utilisé pour modifier le bit de parité PA.

Le contrôle de la logique de branchement s'effectue aussi par le controleur MATCH qui compare les bits PA et PS. Le recouvrement des pannes de la classe CI, dans le cas de l'application de la méthode précédente au séquenceur du MC 68000 est mise en évidence par les remarques suivantes:

- La détection des erreurs du champ NA (Next Address) est assurée par les bits PA et PS ; en effet, cette méthode n'est pas suffisante pour la détection des erreurs (unidirectionnelles en majorité) provoquées par les pannes de la classe CI.

- La détection des erreurs concernant l'adressage du champ des bits de contrôle est assurée aussi par les bits PA et PS, ceci est correct pour le No. 3AESS car les champs NA, TO et FROM constituent une microinstruction adressée par un seul décodeur d'adresses, mais dans le cas du MC 68000 le champ 'Next Address' (partie enchaînement) et les champs TO et FROM (partie action) sont stockés dans deux domaines géographiquement séparés de la ROM qui sont adressés par deux décodeurs différents. Donc l'adressage incorrect d'un micromot d'action n'est pas forcément détecté par les bits PA et PS ; de plus, une telle erreur n'est pas détectée par le controleur de la partie actions car une micro-mot d'actions sera adressée à la place d'une autre, bien que toutes deux soient codées correctement.

- Le codage des parties action en code m/2m fournit une bonne protection pour la majorité des pannes affectant ces parties action car les erreurs provoquées sont unidirectionnelles. Dans le cas du MC 68000, il est préférable d'utiliser un code de BERGER qui détecte aussi les erreurs unidirectionnelles, mais qui ne nécessite pas de décodage pour l'extraction de l'information.

- Dans le cas du MC 68000, le PLA de branchement (2 lignes de sortie) le contrôle des bits PA et PS ne fournit pas une protection suffisante, étant donné que les pannes de la classe CI peuvent provoquer des erreurs unidirectionnelles multiples aux sorties d'un PLA (voir section V.2.2, 1ère. partie). De plus, on peut remarquer que cette technique était suffisante pour le No.3A ESS car la logique du branchement fournit 1 bit seulement.

La seconde technique présentée ici est celle utilisée par [DIS 81] et présentée figure 8.

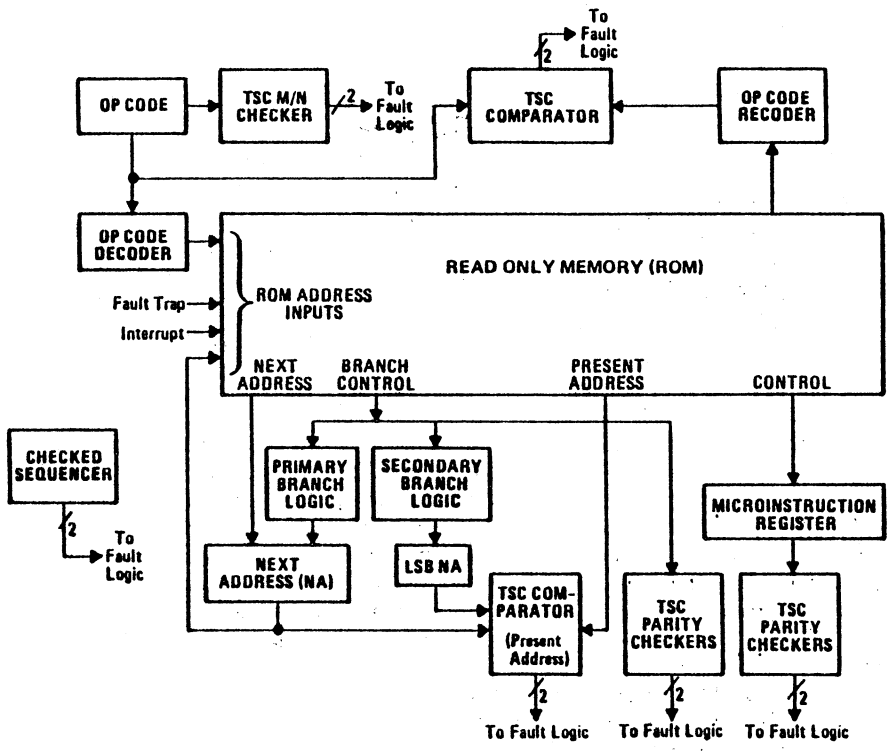


Figure 8- ROM du MC68000 autotestable [DIS 81]

Dans cette proposition un champ "present address" qui contient l'adresse complète de la microinstruction présente est stocké dans la ROM pour controler le champ "next address". Le champ "next address" de la microinstruction précédente et le champ "present address" de la microinstruction présente sont comparés dans un TSC comparator.

Chaque groupe de bits du champ "control" ayant la même destination est codé en parité, i.e. groupe de commande actionnant le même opérateur etc... Ce codage permet à la fois le contrôle du champ "control" et celui des lignes de transfert des bits de controle jusqu'au point de leur utilisation.

La logique du branchement est doublée, un circuit fournit le bit de branchement au registre (NA) et l'autre fournit un deuxième bit de branchement qui est comparé avec le premier dans un 'TSC comparator'.

Le code opération est regénéré à partir du champ "present address" en utilisant un réencodeur , les deux code-opération sont comparés, dans un TSC comparator pour assurer le test du décodeur du code-opération.

Cette proposition appliquée en [DIS 81] sur le MC 68000 implique une augmentation en surface de 41% pour la ROM et de 100% pour le decodeur du code opération.

Le recouvrement des pannes de la classe CI , dans le cas d'application de cette méthode au séquenceur du MC 68000, est mis en évidence par les remarques suivantes:

- La protection, que fournit cette proposition au séquencement des microinstructions, est inacceptable malgré le stockage dans la ROM d'une copie complète des bits d'adressage impliquant une augmentation en surface très élevée. En effet, cette solution fournit une protection complète à l'adressage des microinstructions à partir d'une adresse stockée dans le registre NA ; en revanche il ne fournit aucune protection vis-à-vis des pannes affectant le champ "next address" de la ROM qui représentent la plus grande partie , en surface, des circuits de séquencement.

- Comme on l'avait déjà remarqué dans la proposition de la BELL TELEPHONE [TOY 78], l'adressage du champ "control" dans le cas du MC 68000 n'est pas assuré par cette proposition car le champ enchaînement et le champ action des microinstructions sont stockés dans des régions géographiquement séparées dans la ROM et ils sont adressés par des circuits différents, donc le test d'adressage des micromots d'enchaînement n'implique pas le contrôle d'adressage des micromots des actions.

- Le code de parité utilisé pour tester le champ "control" ne peut pas couvrir les erreurs unidirectionnelles.

- L'utilisation du "decoder" pour regénérer le code opération à partir du champ "present address" n'est pas réaliste sauf dans le cas où chaque microinstruction est décodée au maximum par un code opération ; en effet, la méthode proposée est la méthode reflex [DUR 74] qui nécessite -pour être applicable - l'existence d'une bijection entre les vecteurs d'entrée et les vecteurs de sortie du decodeur.

Dans le cas du MC 68000, cette condition n'est pas respectée car la même microinstruction peut être décodée par plusieurs code-opération (l'injection n'est donc pas surjective) NAMJOO propose en [NAM 82] plusieurs versions.

Le schéma 3 (figure 9) est similaire à la technique utilisée par [TOY 81] dans le système No 3A ESS. Dans ce schéma 3, cf. figure 9, deux "check symbols" sont utilisés, le "check symbol" Cp de la microinstruction présente et le "check symbol" Cs de la microinstruction suivante. Ils correspondent respectivement aux bits de parité PS et PA utilisés dans le No 3AESS.

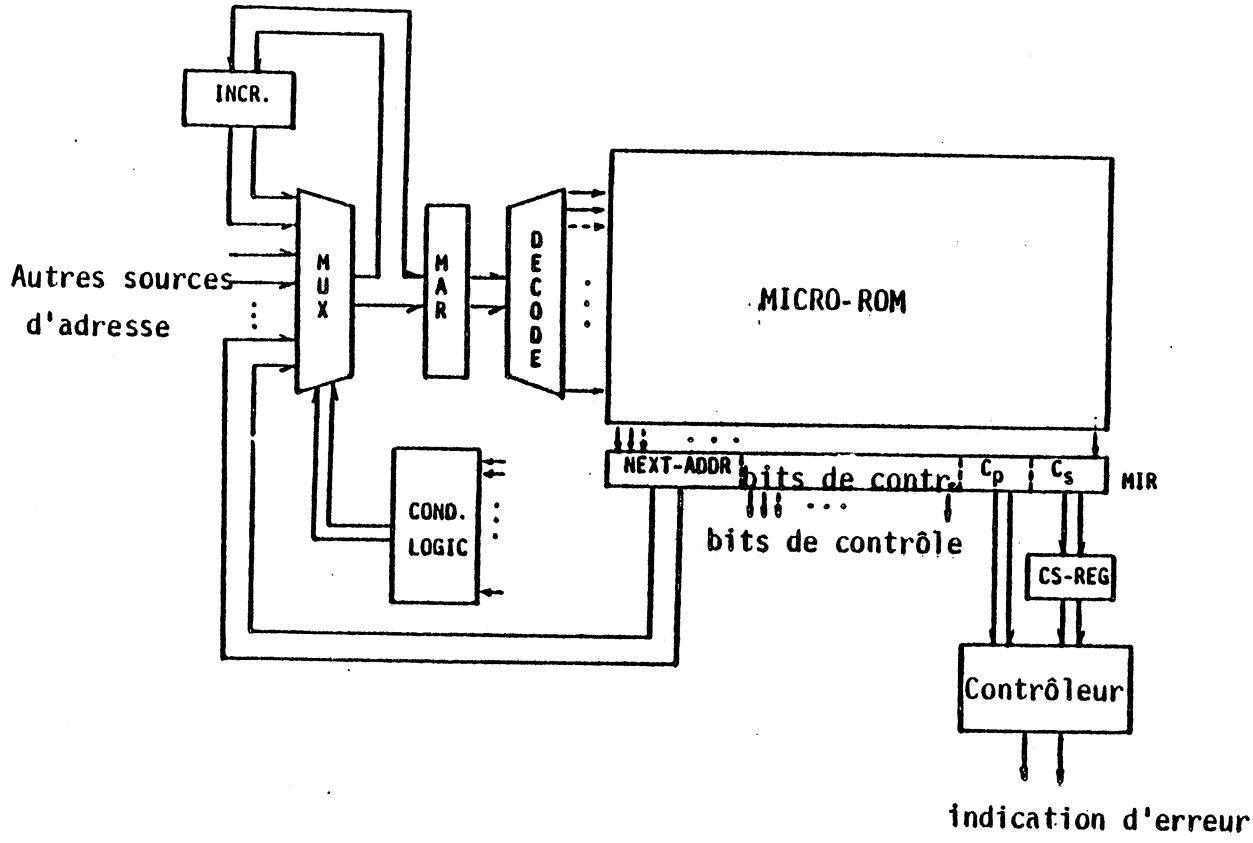


Figure 9- Séquenceur microprogrammé autotestable, Schéma 3

Les deux méthodes différent aux points suivants:

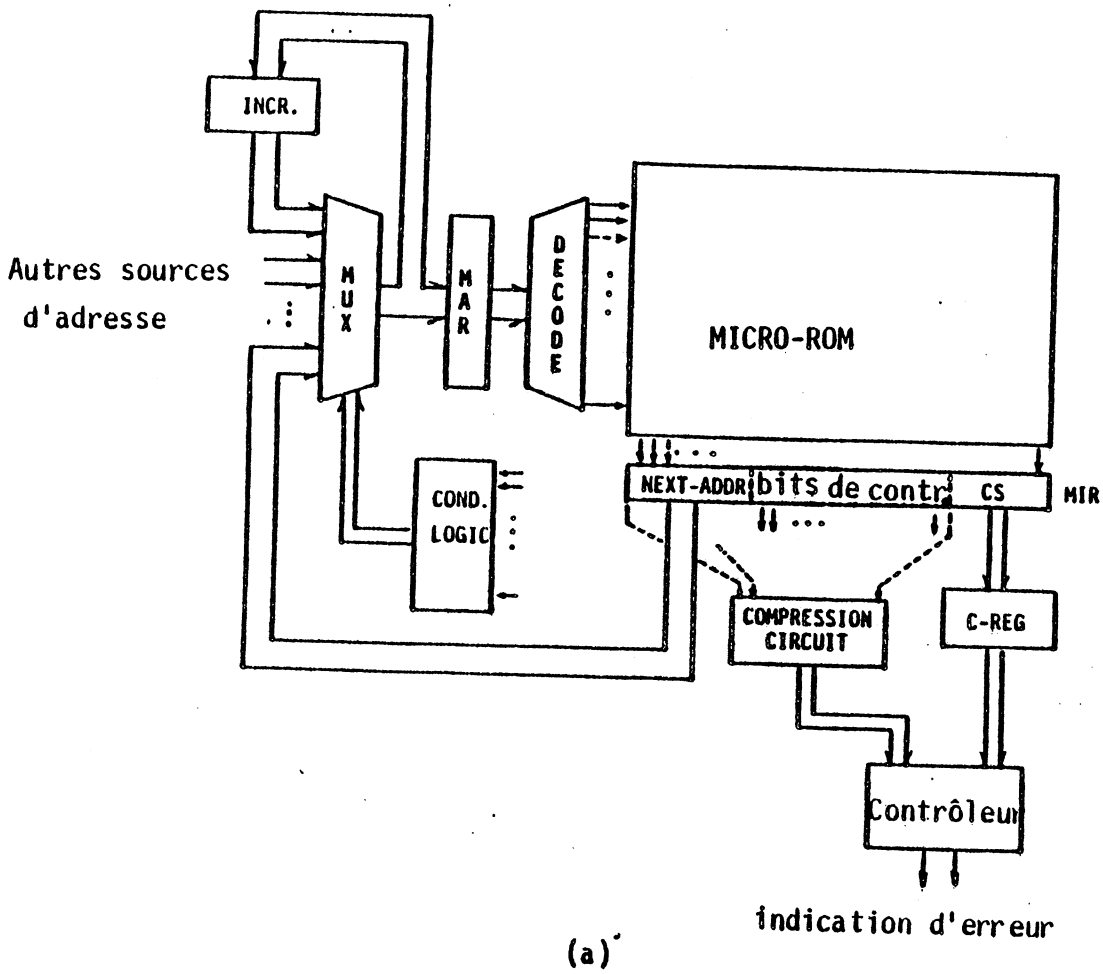
- Les "check symbols" Cp et Cs peuvent être composés de plusieurs bits pour augmenter la sécurité du système.

- Pour le branchement: dans le cas de No 3A ESS le bit de parité PA est modifié par le résultat de la logique de branchement. Dans le cas du schéma 3 aucune modification n'est à effectuer, mais les check symbols Cp doivent être choisis de façon à ce que les check symbols Cp des successeurs de la même microinstruction soient égaux.

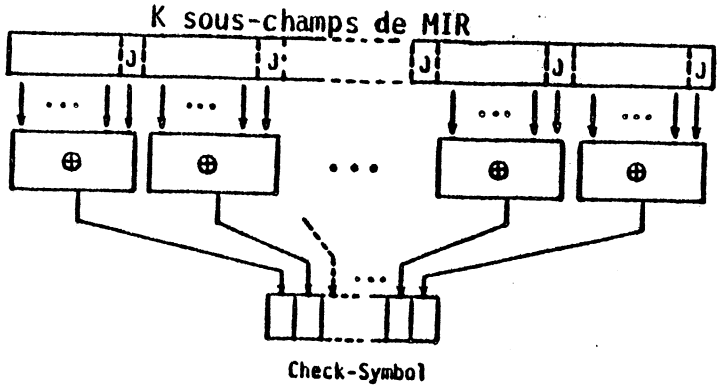
- Codage du champ actions : Dans le cas du schéma 3 le champ actions n'est pas codé, mais ce codage est possible et permet d'augmenter la protection.

Les remarques faites pour le recouvrement des pannes dans le cas du No 3A ESS sont donc valables dans le cas du schéma 3, la seule différence étant que dans le schéma 3 le séquencement du champ d'adressage est mieux protégé si on augmente le nombre des bits de Cp et Cs.

Une protection encore meilleure est assurée par l'intermédiaire du schéma 4 [NAM 82] présenté à la figure 10a. Dans ce cas, seul le champ Cs est ajouté dans les microinstructions tandis que le check symbol Cp est généré par compactage (compression circuit) de tous les bits de la microinstruction (bit d'adressage et bits de contrôle). Etant donné que tous les successeurs de la même microinstruction doivent avoir le même check symbol Cp, il est nécessaire de pouvoir modifier les microinstructions pour obtenir cette propriété. La figure 10b présente cette modification ; dans cette figure la microinstruction est divisée en plusieurs champs et les bits de check symbol Cp sont obtenus en combinant les bits de parité de chaque champ, un bit I est alors ajouté à chaque champ pour obtenir la modification voulue.



(a)



(b)

Figure 10- (a) Structure générale du ''Shéma'' 4
(b) La technique du compactage utilisée
pour la génération du ''check symbol'' Cp

L'avantage de cette structure est qu'elle permet à la fois le test du séquencement du microprogramme ainsi que le test du contenu du champ des bits d'actions. Dans le cas du MC 68000, l'utilisation de cette structure présente aussi l'avantage de tester non seulement l'adressage de la partie enchaînement des microinstructions mais aussi de la partie actions car le check symbol Cp est évalué à partir des bits de la partie enchaînement et des bits de la partie actions.

La protection fournie par cette méthode étant meilleure que dans les autres cas, malgré tout ne couvre pas 100% des erreurs provoquées par les pannes de la classe Cl. Ces erreurs sont rappelées ici, il s'agit de:
- l'adressage d'une microinstruction, partie enchaînement et/ou partie actions, qui ne correspond pas à l'adresse correcte puisqu'il existe des microinstructions avec le même check symbol Cp, ceci pouvant mener à des erreurs indétectables.
- des erreurs unidirectionnelles aux sorties de la ROM. Le codage en parité des microinstructions (enchaînement et actions) ne peut pas détecter toutes les erreurs de ce type.

II.2.3. ROM autotestable basée sur l'analyse des défaillances

Le circuit proposé dans ce paragraphe fournit une détection de 100% des erreurs provoquées par les pannes de la classe CI. Le code de BERGER est utilisé pour coder les microinstructions (17 bits de partie enchaînement et 68 bits de partie actions), ce codage nécessite d'ajouter 7 bits par microinstruction. Le code de BERGER étant un code non-ordonné, permet la détection de toutes les erreurs unidirectionnelles. Il reste donc à protéger le circuit vis-à-vis des pannes provoquant l'adressage d'une microinstruction (enchaînement et/ou action) à la place d'une autre.

Le codage des microinstructions ne recouvre pas ce type de pannes. Par exemple , si une microinstruction est adressée à la place d'une autre, alors les deux sont codées en code de BERGER et l'erreur

n'est pas détectée.

Ce type d'erreur est provoqué par les pannes affectant les entrées du décodeur lignes ou les entrées du décodeur colonnes. Ces entrées doivent alors être codées et controlées après les points critiques. Un bit P0 de parité est utilisé pour les bits A0, A1, un bit P1 de parité pour les bits A2, A3, un bit P2 de parité pour les bits A4, A5, A8, A9. Le circuit ainsi obtenu est présenté à la figure 11.

Les bits B0 à B6 composent les bits de contrôle du code de BERGER. Etant donné qu'un micromot d'actions peut être combiné avec deux ou quatre micromots d'enchaînement, il est nécessaire pour pouvoir coder les microinstructions, que les 7 bits de contrôle soient stockés dans les micromots d'enchaînement.

Un micromot d'enchaînement contient 17 bits. L'augmentation en bits de la ROM est alors 7 x Nbre micromots d'enchaînement = 7 x 544 = 3.808 bits, ce qui correspond à une augmentation en bits de 11,8%.

Les microinstructions sont controlées par le controleur de BERGER. Ce controleur sera divisé en deux parties, l'une est placée à la sortie des latches d'enchaînement pour compacter les 17 bits d'enchaînement en 6 bits. Ces 6 bits et les bits B0,...,B6 sont transférés vers les latches des parties actions où est placée l'autre partie du controleur.

Un générateur de parité est utilisé, aux sorties des latches d'enchaînement pour générer les bits de parité P0, P1, P2, P3 utiles au contrôle des entrées des circuits de décodage lignes et décodage colonnes. Ces bits sont stockés dans le registre adresse de la ROM. Dans la figure 11 on voit aussi le contrôle des entrées des décodeurs lignes et décodeurs colonnes.

Un générateur de parité est utilisé aux sorties des latches de partie actions. Les bits générés protègent les transferts des différents groupes de bits de la partie actions jusqu'aux blocs où ils sont utilisés.

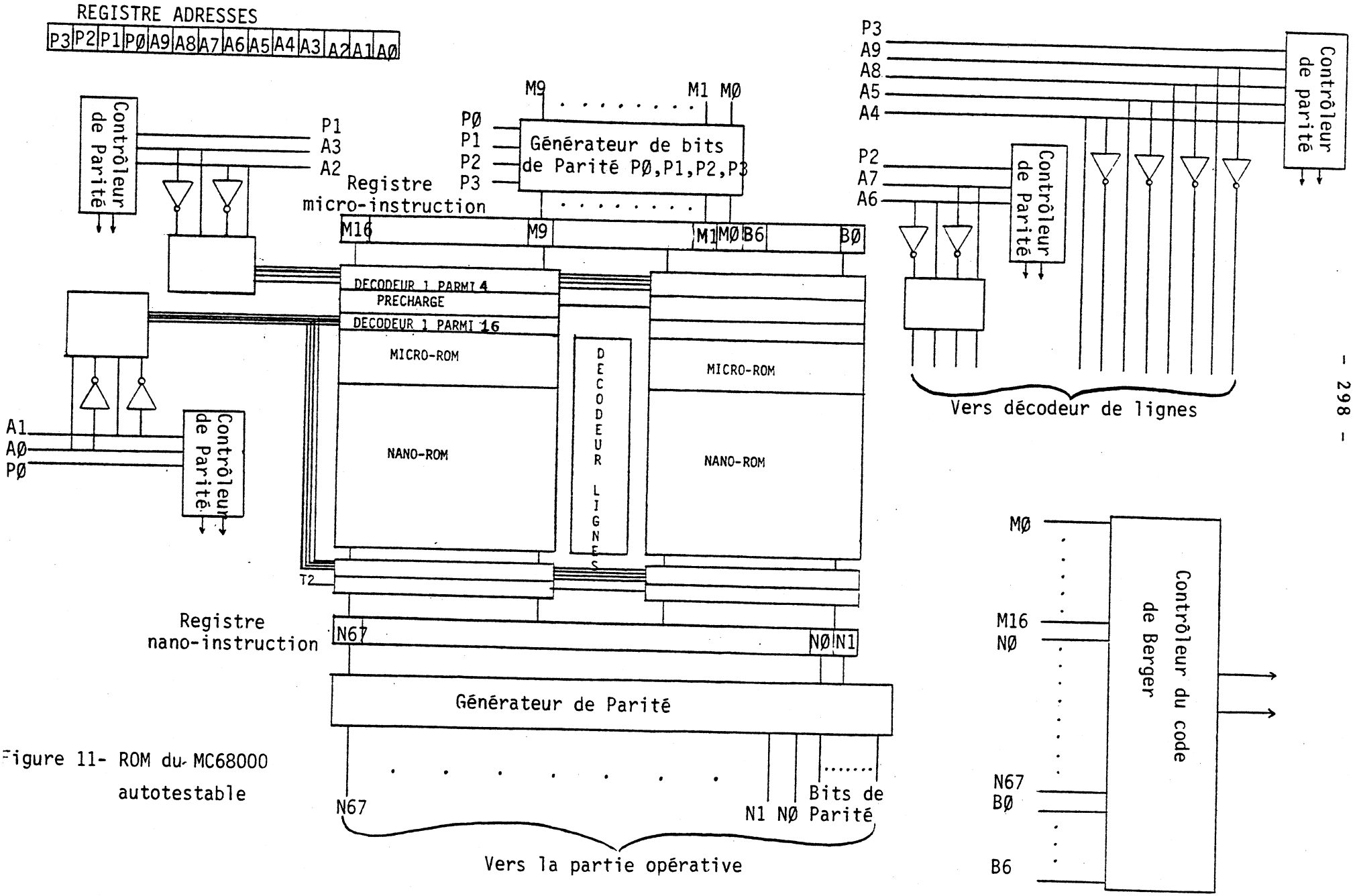


Figure 11- ROM du MC68000 autotestable

II.2.4. Augmentation en surface de la ROM

La figure 12 présente la surface occupée par les différents blocs de la ROM.

Surface totale: S = 3.820µm x 2140µm = 8,17mm2
Augmentation en surface de:
"Précharge et latches" (partie actions): S1 = 0,70mm2 x 2/68 = 0,02mm2
"1 parmi 4" (partie actions): S2 = 0,26mm2 x 2/68 = 0,0076mm2
ROM (partie actions): S3 = 0mm2
"Partie enchaînement": S4 = 1,61mm2 x 7/17 = 0,66mm2
"1 parmi 16" (partie enchaînement): S5 = 0,36mm2 x 5/17 = 0,106mm2
"Précharge" (partie enchaînement): S6 = 0,219mm2 x 5/17 = 0,064mm2
"1 parmi 4" (partie enchaînement): S7 = 0,2mm2 x 5/17 = 0,06mm2
Latches (partie enchaînement): S8 = 0,16mm2 x 5/17 = 0,045mm2
Décodeur lignes (partie enchaînement): S9 = 0mm2
Décodeur lignes (partie actions): S10 = 0mm2
"1 parmi 3" multiplexeur (mode TEST): S11 = 0mm2

Générateur de parité (partie enchaînement) nombre de transistors: NT1 = 7(2- 1) + 7(2-1) + 7(2-1) + 7(4-1) = 42
Générateur de parité (partie actions) nombre de transistors: NT2 < 7(68-1) = 469
Controleurs de parité (entrées des decodeurs lignes et colonnes) nombre de transistors: NT3 = 7(2-1) + 7(2-1) + 7(2-1) + 7(2-1) + 7(4-1) + 10(5-1) = 89
Controleur du code de BERGER (réalisation cellulaire): NT4 = 23(85-7) + 10(7-1) = 1854

La surface d'un bloc contenant NT transistors est donnée par l'équation S = k.NT.13.pp.pm donnée en [REI 83]:
k est le coefficient d'optimisation qui se situe dans l'intervalle
k = 0,9 - conception optimisée
k = 1,1 - conception latche
On choisit: k = 1

pp est le pas de poly, pp = 4µm
pm est le pas de métal, pm = 6µm
NT est le nombre de transistors
13 est le nombre de pas de poly x pas de métal par transistor, résultat de statistiques sur la logique aléatoire [REI 83].

Surface totale des générateurs de parité et des controleurs:
S12 = k(NT1 + NT2 + NT3 + NT4).13.pp.pm = 1 x 2.454 x 13 x 4m x 6m = 0,765mm2
Augmentation totale de la ROM: S' = S1 + S2 + ... + S12 = 1,72mm2
Ceci correspond à une augmentation de 20% de la surface de la ROM.

Remarque

La méthode utilisée, basée sur l'analyse des défaillances de la classe C1 a permis une couverture de 100% des pannes de la ROM avec une augmentation de 20% de la surface de départ. Ceci est un résultat très intéressant étant donné que l'augmentation de surface obtenue par [DIS 81] est de 41% pour un recouvrement des pannes bien inférieur.

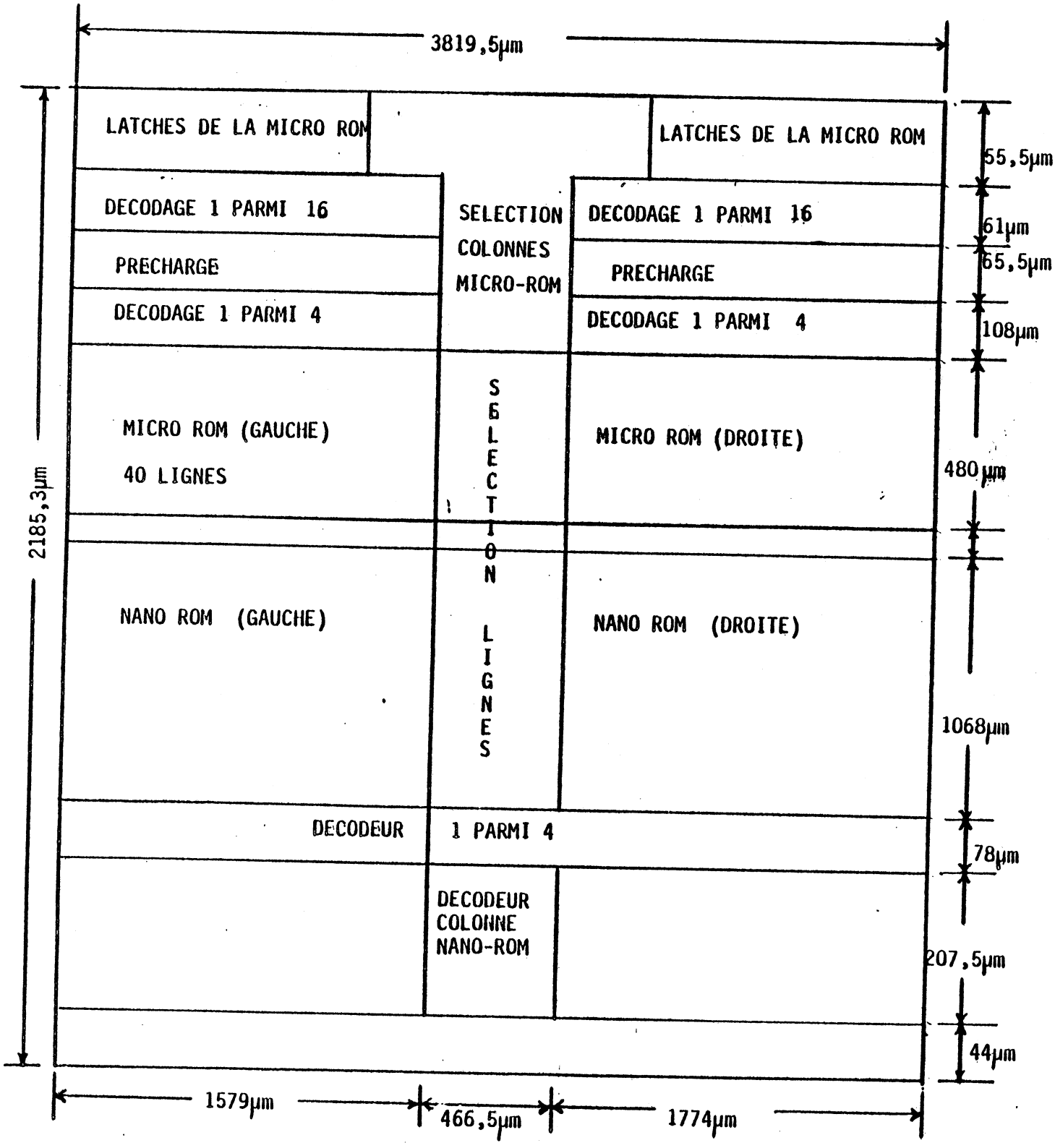

3819,5µm
LATCHES DE LA MICRO ROM
LATCHES DE LA MICRO ROM
DECODAGE 1 PARMI 16
SELECTION COLONNES MICRO-ROM
DECODAGE 1 PARMI 16
PRECHARGE
PRECHARGE
DECODAGE 1 PARMI 4
DECODAGE 1 PARMI 4
MICRO ROM (GAUCHE)
40 LIGNES
SELECTION LIGNES
MICRO ROM (DROITE)
NANO ROM (GAUCHE)
NANO ROM (DROITE)
DECODEUR 1 PARMI 4
DECODEUR COLONNE NANO-ROM
2185,3µm
55,5µm
61µm
65,5µm
108µm
480µm
1068µm
78µm
207,5µm
44µm
1579µm
466,5µm
1774µm

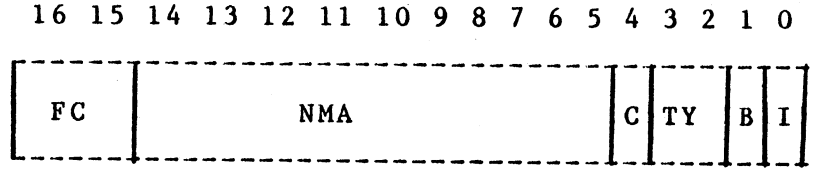
Figure 12

II.3. LES PLAs DU SEQUENCEUR

La figure 13 présente l'ensemble du séquenceur, c'est à dire la ROM (partie enchaînement), les PLAs A1, A2, A3 de décodage des codes opération, le PLA A0 d'exceptions et le PLA du branchement.

Le choix de ces PLAs est fonction de quelques bits de microinstruction (partie enchaînement). Le bit B(MR1) de la microinstruction (partie enchaînement) détermine la présence d'un branchement conditionnel.

Si B=0 alors on n'est pas en branchement
Si B=1 le PLA de branchement est sélectionné pour fournir les bits A6, A7 de l'adresse de l'instruction microprogrammée suivante.
Si B-0, le format de la microinstruction partie enchaînement est le suivant:



Le bit I(MR0) commande le chargement du registre d'instruction (IRC -> IR).

Les bits TY(MR2, MR3) définissent la source de l'adresse de l'instruction microprogrammée suivante. (TY) = (0,0) caractérise le passage en séquence, l'adresse est alors fournie par le champ NMA (10 bits). Si (TV) est différent de (0,0), alors l'adresse de l'instruction microprogrammée suivante , fournie par décodage du code opération, est sélectée aux sorties du:
PLA A1 pour TY = (0,1)
PLA A2 pour TY = (1,0)
PLA A3 pour TY = (1,1)

Le bit C(MR4) sert (avec les bits I et B) à la génération des

commandes de chargement du PLA AO par les signaux de requête d'exception.

Le champ NMA (MR5 à MR14) définit le séquencement - TY = (0,0) - il correspond à l'adresse explicite de l'instruction microprogrammée suivante.

Les FC (MR15, MR16) définissent l'état du système FC = (01): donnée, FC = (10) : programme, FC (11) : reconnaissance d'interruption.

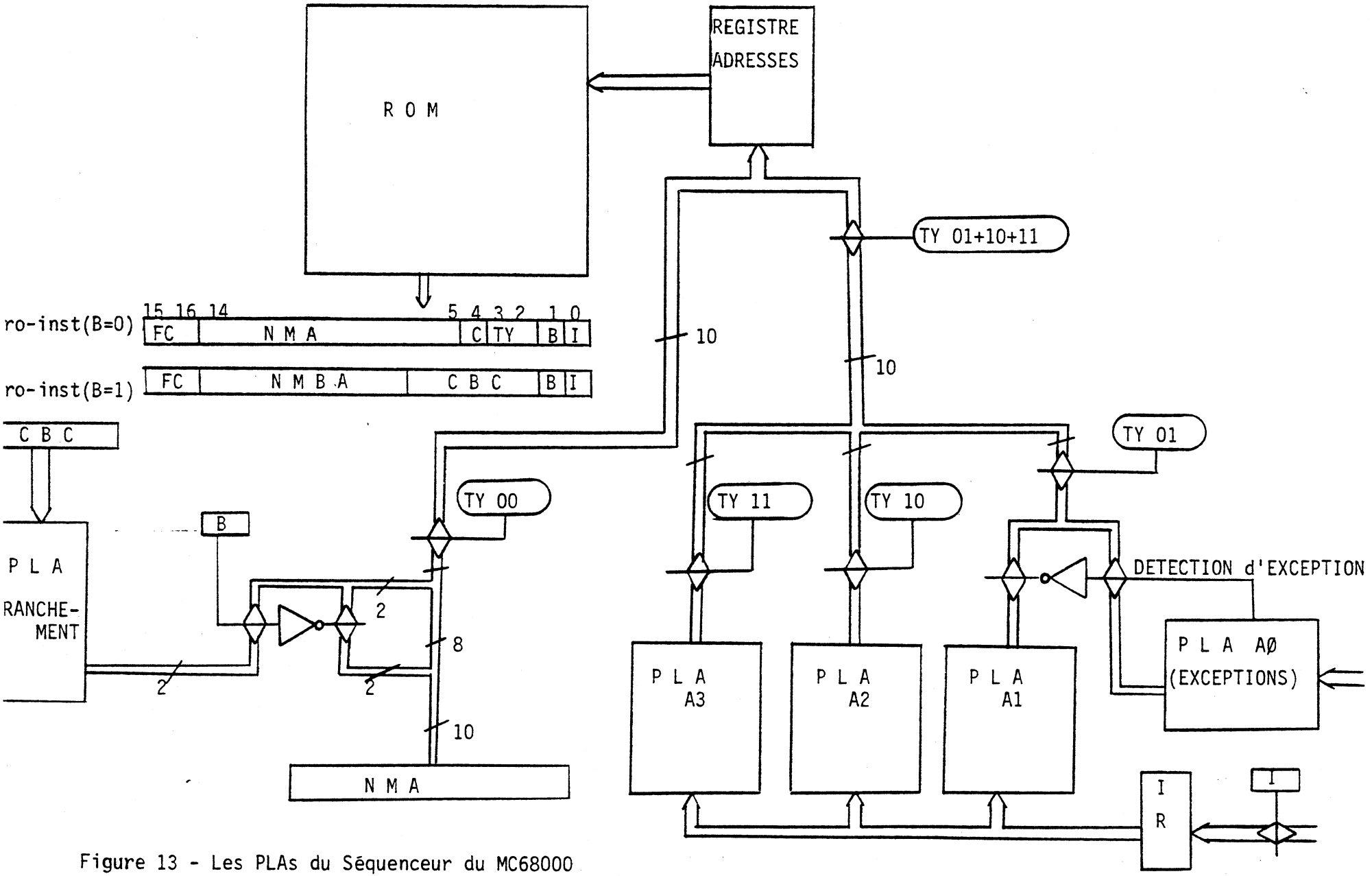
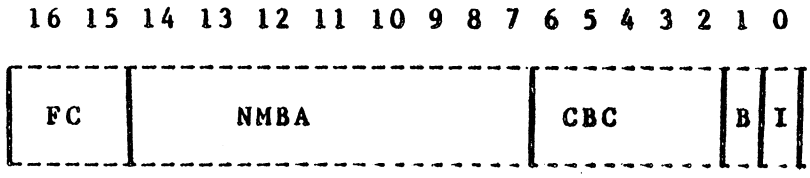


Figure 13 - Les PLAs du Séquenceur du MC68000

Si B=1, le format de la microinstruction est le suivant:



Le champ CBC (MR2 à MR6) définit la condition à tester par le PLA de branchement.

Le branchement NMBA (MR7 à MR14) donne les 8 bits d'adresse de microinstruction suivante qui est complétée par les sorties C0, C1 du PLA du branchement. Les autres bits jouent le rôle défini dans les cas de B=0.

II.3.1. Les PLAs du séquenceur autotestables

On a déjà remarqué que l'autotest des PLAs de décodage du code opération par régénération de ce code opération, proposé en [DIS 81] , n'est pas possible.

Ici on utilisera la méthode proposée dans la section V.2.2 (1ére partie) pour la conception des PLAs SFS (Strongly Fault Secure) en utilisant le code de BERGER, pour les PLAs A1, A2, A3 et A0. Le circuit augmenté est présenté figure 14.

Les sorties des PLAs A0, A1, A2 et A3 seront controlées au moyen d'un seul controleur du code BERGER. Un générateur de parité sera utilisé pour générer la parité P0 des bits A0, A1, la parité P1 des bits A2, A3, la parité P2 des bits A6, A7 et la parité P3 des bits A4, A5, A8, A9 qui sont controlés aux entrées des décodeurs lignes et des décodeurs colonnes de la ROM (voir figure 11).

Le PLA de branchement sera controlé en utilisant le code double-rail (section V.2.3, 1ère. partie). Cette technique n'est pas plus

coûteuse que le code de BERGER, étant donné que le PLA de branchement a deux sorties. Les sorties C0 C1 du PLA de branchement correspondent aux bits A6 et A7. Un générateur de parité sera alors placé aux sorties du PLA pour générer le bit P2 de parité correspondant à ces sorties.

La solution donnée ci-dessus doit être complétée avec quelques circuits qui généreront les signaux complémentaires tels que: les commandes de sélection des PLAs. Ces commandes seront controlées après avoir commandé les MOS de sélection des PLAs.

Notons aussi que chaque octet du code opération entrant dans la puce est codé en parité et qu'il est testé en entrée par le contrôleur des entrées/sorties des données (voir chapître I, 2ème. partie figure 11 et figure 40). Une porte XOR est utilisée pour générer un bit unique de parité du code opération, ce bit sera stocké dans les registres IRC et IR. Un controleur de parité sera utilisé pour contrôler les entrées des PLAs A1, A2, A3 comme il est proposé pour la conception des PLAs SFS en chapître V, 1ère. partie.

Ces circuits ne sont pas présentés à la figure 14.

L'évaluation de l'augmentation de la surface du circuit ainsi obtenu ne peut pas être estimée sans un redessin complet des PLAs afin d'obtenir leurs sorties avec le code de BERGER.

D'autre part, il est évident que cette méthode est beaucoup plus économique que la duplication utilisée génèralement pour concevoir des PLAs autotestables. Le réencodage proposé en [DIS 81] nécessite aussi la duplication de la surface.

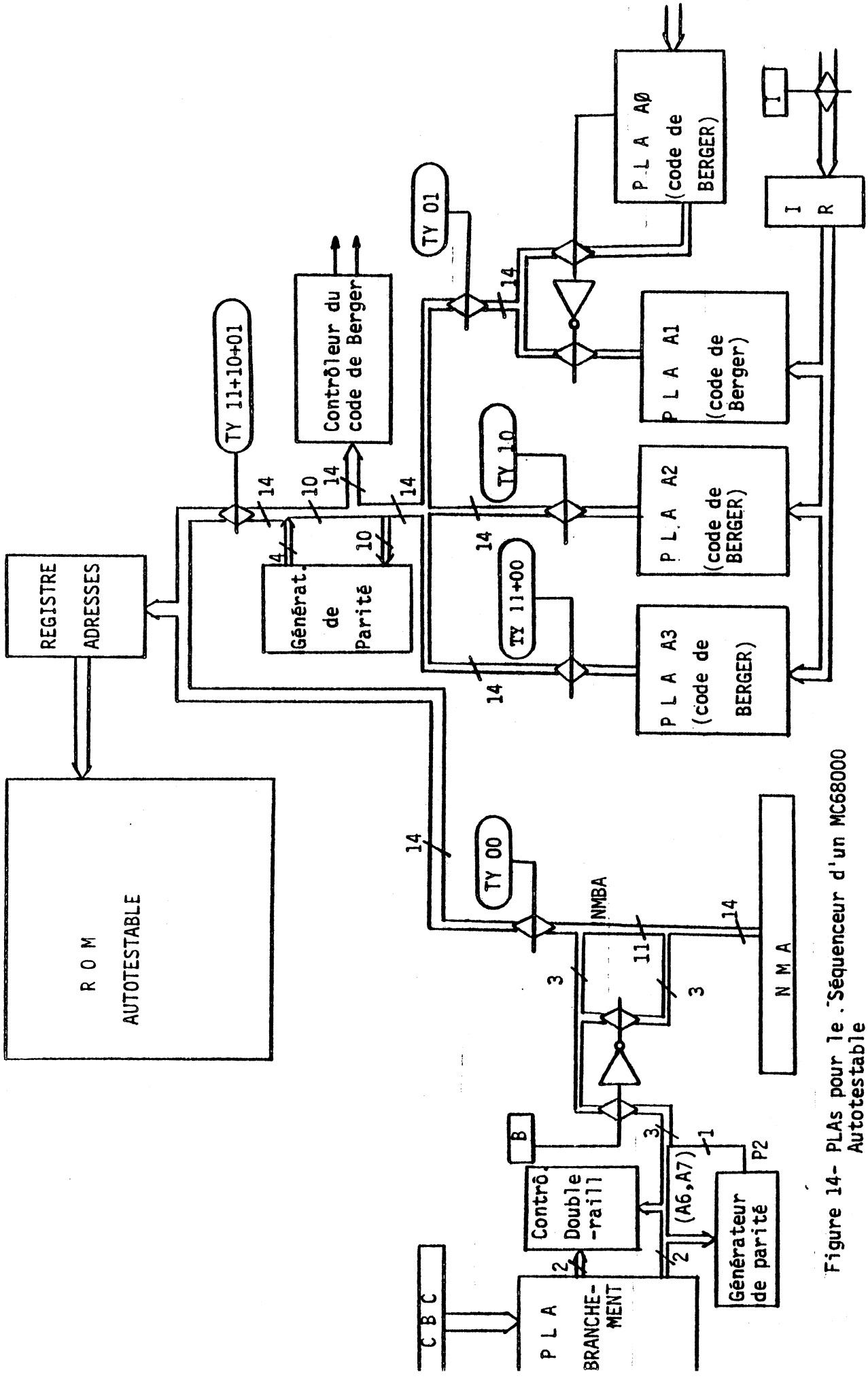


Figure 14- PLAs pour le Séquenceur d'un MC68000 Autotestable

III - LE PLA IRD - LES CIRCUITS DE CONTROLE DE LA PO

Le dessin des circuits augmenté n'est pas détaillé ici. Les modifications suivantes sont utilisées pour assurer la sécurité de ces blocs.

PLA IRD

Les différentes parties du PLA IRD sont testées par codage de leurs sorties en code de BERGER généralisé. La majorité de ces parties est composée d'une seule matrice NOR, l'analyse présentée dans la section V.2.2.3 de la lère. partie sera utilisée . La partie du PLA IRD qui sert à l'extraction des paramètres affectant le contrôle de l'ALU est composée de trois matrices NOR. Cette partie sera testée par un code de BERGER (voir section V.2.2.2, lère. partie).

Le bit de parité du code opération sera stocké aussi dans le registre IRD. Ce bit sera utilisé pour controler les entrées du PLA IRD.

PLA TRAP:

Le PLA TRAP est testé en utilisant le code de BERGER

CIRCUITS DE CONTROLE DE LA PO:

Les circuits de contrôle des différents blocs de la PO sont réalisés en logique aléatoire, l'utilisation d'une méthode autre que la duplication nécessitera de redessiner ces blocs afin qu'ils respectent les règles nécessaires. Etant donné la complexité de ces blocs, leur redessin n'entre pas dans le cadre de ce travail. En conséquence, la méthode proposée ici est la duplication, mais dans le cas d'une étude destinée à la réalisation d'un MC 68000, il serait très intéressant du point de vue gain de surface, d'appliquer les méthodes proposées dans la lère. partie.

Dans ce rapport, on se limite alors à dupliquer les blocs suivants:
- contrôle de l'ALU et contrôle du code condition: duplication,
- contrôle des registres poids faibles: duplication,
- contrôle des AU, MUxI/O, FOOL IT: duplication,
- contrôle des registres pois forts: duplication,
- registre d'état: les différents champs du registre d'état seront protégés en utilisant un bit de parité par champ. En conséquence, quelques corrections seront nécessaires pour permettre de modifier les bits de parité lors des modifications des bits du registre d'état.

IV - DETECTEUR DES CODES INVALIDES

Le MC 68000 possède 19.721 codes invalides qui représentent environ 30% des codes possibles [MAR 83].

Le MC 68000 possède un détecteur de codes invalides. Il s'agit d'un PLA dont la sortie unique est activée lorsqu'un code invalide est présenté dans le registre IR.

Ce détecteur est très important dans le cas des méthodes de test n'utilisant pas de redondance massive car certains types de pannes ne sont détectées que par ce circuit. Dans un tel cas, il est primordial de posséder un détecteur de codes invalides autotestables ; dans le cas contraire, les défauts du détecteur mettront en cause l'efficacité de la méthode de test.

Dans un circuit totalement autotestable, le rôle du détecteur de codes invalides au niveau du test est moins important.

IV.1. Le circuit augmenté

Les méthodes utilisées sur l'ensemble de tous les autres blocs du

MC 68000, pour obtenir des blocs fonctionnels SFS, ne peuvent être employées dans le cas présent.

Le détecteur de codes invalides ne doit pas être considéré comme un bloc fonctionnel, mais comme un contrôleur. En effet, il existe un ensemble de vecteurs d'entrée qui est appliqué pendant le fonctionnement correct du système et un autre ensemble des vecteurs d'entrée qui est appliqué au circuit en cas d'anomalies.

Un contrôleur doit respecter la propriété SCD (Strongly Code Disjoint) et non la propriété SFS. Le dessin des circuits SCD est en général difficile. Des règles générales n'existent pas et on doit examiner le circuit pour toutes les pannes individuelles possibles ; d'autre part, il est possible qu'il n'existe pas de dessin SCD pour un circuit considéré.

A cause de ces difficultés, la solution donnée dans la section VI.4, lère. partie, pour les contrôleurs spécifiques est adoptée ici.

Notons que la sortie unique $\overline{ILLI}$ du détecteur de codes invalides sera remplacée par deux sorties codées en double-rail, car il est clair qu'un controleur possédant une seule sortie ne peut pas être SCD.

La figure 15 présente la solution choisie.

Le compteur de 15 bits doit compter de 0 à 19.720. Le PLA autotestable générera les 19.721 codes invalides qui sont les vecteurs de test à appliquer au décodeur des codes invalides pendant la phase de test (signal TEST égal à '1'). Pendant cette phase, l'une des sorties du décodeur est inversée de façon à ce que si le code invalide est reconnu (sorties du décodeur à 00 ou à 11), la logique des erreurs reçoit un signal de bon fonctionnement (10 à 01) et si le code invalide n'est pas reconnu, la logique des erreurs reçoit un signal de fonctionnement incorrect (00 ou 11).

A la figure 15 le signal TEST est remplacé par T1 et l'instant T2 est utilisé comme horloge de compteur. Ainsi à chaque cycle d'horloge un vecteur de test est appliqué.

Etant donné que les commandes AOC et EXC de chargement du PLA d'exceptions AO sont activées pendant T3, la détection pendant la phase de test d'une panne du décodeur n'est pas prise en compte par la logique des exceptions ; donc le fonctionnement de celle-ci n'est pas perturbé par la modification du détecteur de codes invalides. Finalement le signal $\overline{\text{ILLI}}$ (sortie du décodeur dans le MC 68000) est obtenu en utilisant une porte XOR.

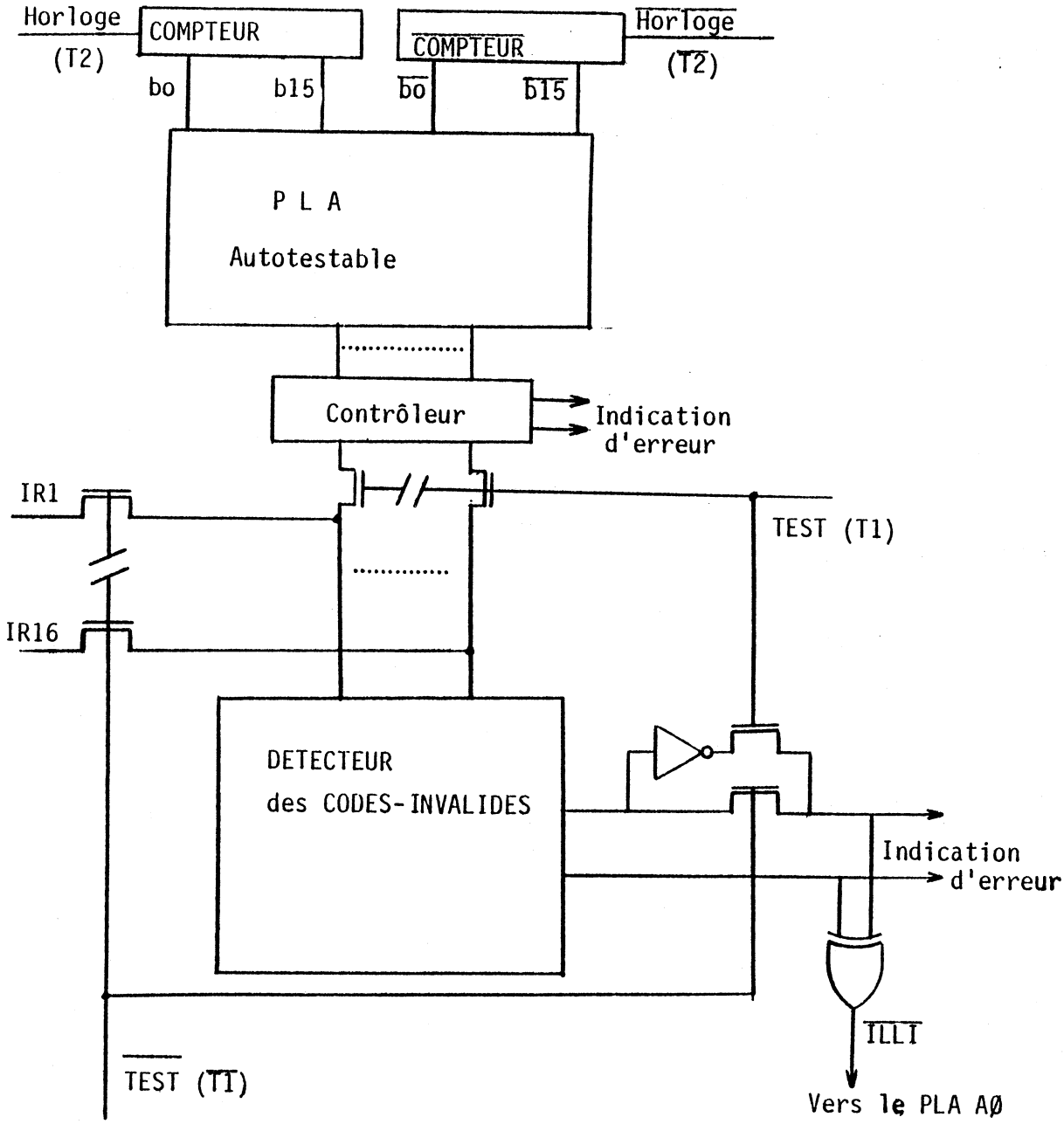


Figure 15- Principe de conception d' un détecteur de codes-invalides autotestable.

V - INTERRUPTIONS - HORLOGES - CONTROLE DU BUS

Pour les mêmes raisons que celles données pour les circuits de contrôle de la partie opérative, la duplication est utilisée dans les blocs suivants:

- logique d'interruption: duplication
- génération des horloges: duplication
- logique de contrôle du bus composé par les blocs:

* automate d'arbitrage d'attribution de bus: duplication
* circuit de commande de l'automate de contrôle du bus asynchrone: duplication
* automate de contrôle du bus asynchrone: duplication.

VI - BROCHES

Les broches du microprocesseur sont codées en parité ou en code double-rail. La correspondance entre les broches du circuit de départ et du circuit autotestable est donnée au Tableau I.

TABLEAU I

broches initiales	broches de codage	broches de circuit autotestable
Données (16 broches)	parité(2 broches)	(18 broches)
Adresses (23 broches)	parité (1 broche)	(24 broches)
Interruptions: IPL0,IPL1,IPL2 (3 broches)	parité (1 broche)	(4 broches)
Code de fonction: IFC0,IFC1 (2 broches)	parité (1 broche)	(3 broches)
IFC2 (1 broche)	double rail (1 broche)	(2 broches)
Autres broches de contrôle (14 broches)	double rail(14 broches)	(28 broches)
Horloge clock (1 broche)	duplication (1 broche)	(2 broches)
Alimentation:		
Vdd (2 broches)	" (0 broche)	(2 broches)
Vss (2 broches)	" (0 broche)	(2 broches)
0 broche de détection d'erreurs	double rail (2 broches	(2 broches)
64 broches	23 broches	87 broches

VII - EVALUATION DE LA SURFACE DU MC 68000 AUTOTESTABLE

L'estimation exacte de la surface du MC 68000 autotestable d'après l'analyse présentée dans cette étude nécessite le calcul du nombre des monômes des PLAs controlés par le code de BERGER. Etant donné que ce calcul n'est pas fait on utilisera une estimation de 30% à 40% en moyenne d'augmentation de surface de ces PLAs, y compris la surface du controleur de BERGER.

L'augmentation en surface des circuits doublés est considéré de 107%, y compris la surface des controleurs double rail.

Un controleur double rail (logique des erreurs) est aussi considéré pour compacter une trentaine de paires de sorties double rail des différents controleurs en une seule paire double rail. Le nombre des transistors d'un controleur est 10(30-1) = 290 et sa surface est 1 x 290 x 13 x 4µm x 6µm = 0,09mm2.

Le tableau II donne l'augmentation en surface du MC 68000 autotestable, l'augmentation de 40% est considérée pour les PLAs controlés par le code de BERGER. L'augmentation en surface de chaque bloc comprend aussi la surface des controleurs qui y sont utilisés.

La surface initiale des différents blocs a été mesurée sur la puce du MC 68000 par le microscope optique.

TABLEAU II

AUGMENTATION DE SURFACE DU MC 68000 AUTOTESTABLE

bloc	surface initiale	augmentation en surface
Partie opérative	8,06mm2	5,54mm2
Plot des données	2,13mm2	0,40mm2
Plot des adresses	2,84mm2	0,34mm2
Zone de routage (PO)	2,45mm2	0,74mm2
ROM (avec les circuits annexes)	8,17mm2	1,64mm2
PLAs A1 A2 A3 (décodage CO)	2,64mm2	1,05mm2
PLA de branchement	0,31mm2	0,12mm2
PLA A0 (exception)	0,23mm2	0,09mm2
Détection des codes invalides	0,55mm2	0,81mm2
PLA IRD	1,90mm2	0,76mm2
Circuits de contrôle de la PO	1,73mm2	1,85mm2
Buffers des commandes de la PO	0,98mm2	0,48mm2
Logique de contrôle du bus externe	1,1mm2	1,17mm2
Logique d'interruptions	0,29mm2	0,31mm2
Logique des horloges	1,89mm2	1,81mm2
Plots (autres que les plots d'adresses et de données)	3,00mm2	2,76mm2
Routage (PC)	6,34mm2	1,60mm2
Logique des erreurs	-	0,09mm2
	44,57mm2	21,56mm2

Par le tableau II on obtient une augmentation totale de surface de 21,41mm2, soit 48% de la surface de départ.

En comparant avec l'étude donnée en [DIS 81] qui demande 58% d'augmentation en surface , on obtient ainsi une économie de 10%.

On doit remarquer aussi que dans le tableau I donné en [DIS 81] la surface des controleurs n'est pas prise en compte pour la majorité des blocs. D'autre part, dans le même tableau, le code de parité est proposé pour le test du circuit de génération des commandes de la PO ; ce circuit est composé par le PLA IRD d'extraction des paramètres et par les circuits de contrôle de la PO. L'utilisation d'un code de parité pour l'autotest d'un tel circuit est très difficile, voire impossible, une telle solution ne peut pas être proposée sans une analyse détaillée du circuit concerné.

VIII - CONCLUSION

Dans les chapîtres I et II de la 2ème. partie, la méthode de dessin des circuits SFS (1ère. partie) est utilisée pour étudier un MC 68000 autotestable.

Cette méthode nous a permis de redessiner facilement les différents blocs du MC 68000 en utilisant des règles générales, sans avoir besoin d'examiner les grandes listes des défauts introduits par la classe CI.

D'autre part, avec cette méthode nous avons obtenu un recouvrement de 100% de la classe CI avec une démonstration rigoureuse de cette protection. Les remarques suivantes sont retenues pour l'application au MC68000.

L'étude de la partie opérative est plus ou moins simple étant donné sa régularité.

La majorité de ces circuits, conçue en structure bit slice, est testée par code de parité.

La majorité des opérateurs est doublée et quelques opérateurs sont testés avec le code de parité.

L'étude des blocs réguliers de la partie contrôle est relativement simple et on peut obtenir des solutions peu coûteuses en surface.

L'étude des blocs conçus en logique anarchique est assez compliquée, pour cela la duplication de ces blocs est retenue.

Notons que la totalité de blocs non doublés est testée en utilisant la méthode du contrôle des entrées primaires (section IV.1.4 et IV.2.4, 1ére partie).On peut citer ici:

* Les registres de la partie opérative, le méchanisme des entrées-sorties, l'opérateur FOOL-IT e.t.c. dont les lignes de commande sont testées aprés avoir commander tous les bits du circuit respective.
* L'opérateur DCR (contrôle de commandes et contrôle des entrées de la partie decodeur).
* Les PLA A0,A1,A2,A3,le PLA de branchement et le PLA IRD (contrôle des entrées primaires).
* La ROM (contrôle de bits d'adressage A0, A1, ..., A9).

L'augmentation en surface, de l'ordre de 48%, est satisfaisante étant donné que le circuit est protégé à 100% pour la classe I. Cette augmentation peut être minimisée en faisant une étude plus approfondie des blocs conçus en logique anarchique. Ceci permettrait de tester ces blocs en utilisant un code moins coûteux que la duplication. La modification de ces blocs et l'utilisation de structures régulières permettront plus facilement de résoudre ce problème. De toute façon, il ést clair que les structues règulières sont très avantageuses pour le dessin des circuits autotestables.

BIBLIOGRAPHIE

[DIS 81] DISPARTE CH.
A design approach for an electronic engine controler self-checking microprocessor
EUROMICRO Symposium, Paris, September 1981.

[DUR 74] DURANTE C., BERTRAND J.C., MARTIN J.P.
Sur la maintenance automatique des systèmes informatiques
Journées d'études sur les calculateurs numériques embarqués et leurs applications, Toulouse, juin 1972.

[MAR 83] MARCHAL P.
Test en-ligne du microprocesseur MC 68000 - Modélisation et programmes de test
Thèse de docteur de 3eme cycle, INPG, juillet 1983.

[NAM 82] NAMSOO M.
Design of concurrently testable microprogrammed control units
Technical Report No. 82-6, Juin 1982
Center for Reliable Computing, Stanford University.

[OBR 83] OBREBSKA M.
Etude comparative de différentes méthodes de conception des parties contrôle des microprocesseurs
Thèse de docteur-ingénieur, INPG, juin 1982.

[REI 83] DA LUZ REIS R.A.
Evaluateur topologique prédictif pour la génération automatique des plans de masse de circuits VLSI
Thèse de docteur-ingénieur, INPG, janvier 1983.

[TOY 78] TOY W.N.
Fault tolerant design of local ESS processors
Proc. IEEE, vol. 66, pp. 1126/1145, October 1978.

AUTORISATION DE SOUTENANCE

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

VU les rapports de présentation de Messieurs

. L. BOLLIET, Professeur
. B. COURTOIS, Chargé de recherche

Monsieur NICOLAIDES Michel

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

Fait à Grenoble, le 21 décembre 1983

Le Président de l'INP-G

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

P.O. le Vice-Président,

RESUME

Des études récentes montrent que le modèle de collage logique ne convient pas pour représenter les défauts réels qui peuvent survenir dans les circuits intégrés. Ceci a amené certains auteurs à chercher des méthodes de test basées sur des hypothèses de pannes analytiques. Dans cette étude, le problème de conception des circuits autotestables vis à vis d'hypothèses de pannes analytiques, est abordé. Des méthodes et des règles générales de conception de circuits fonctionnels N-MOS "Stongly Fault Secure" sont proposées. De nouveaux codes sont introduits. La définition de la plus grande classe de contrôleurs, "Strongly Code Disjoint", qui peuvent exister est donnée. Des examples validant les méthodes sont donnés. Enfin, les méthodes sont utilisées pour l'étude d'un microprocesseur MC 68000 autotestable.

MOTS-CLES : Circuits autotestables - Circuits "Stongly Fault Secure" - Hypothèses de pannes - Codes - Contrôleurs - Conception VLSI - Microprocesseurs.