Le test des PLAs optimisés topologiquement

A.-O. Fernandes

THESE

présentée par

Antonio Otavio FERNANDES

pour obtenir le titre de **DOCTEUR**

de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE
(arrêté ministériel du 5 Juillet 1984)

Spécialité: **Microélectronique**

LE TEST DES PLAs OPTIMISES TOPOLOGIQUEMENT

Date de soutenance: 9 septembre 1988

Composition du jury:

Messieurs	G.MAZARE	Président
	G.CAMBON	Rapporteur
	B.COURTOIS	
	J.FREHEL	Rapporteur

Thèse préparée au sein du laboratoire TIM3.

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Président : Georges LESPINARD **Année 1988**

Professeurs des Universités

BARIBAUD Michel	ENSERG	JOUBERT Jean-Claude	ENSPG
BARRAUD Alain	ENSIEG	JOURDAIN Geneviève	ENSIEG
BAUDELET Bernard	ENSPG	LACOUME Jean-Louis	ENSIEG
BEAUFILS Jean-Pierre	ENSEEG	LESIEUR Marcel	ENSHMG
BLIMAN Samuel	ENSERG	LESPINARD Georges	ENSHMG
BLOCH Daniel	ENSPG	LONGEQUEUE Jean-Pierre	ENSPG
BOIS Philippe	ENSHMG	LOUCHET François	ENSIEG
BONNETAIN Lucien	ENSEEG	MASSE Philippe	ENSIEG
BOUVARD Maurice	ENSHMG	MASSELOT Christian	ENSIEG
BRISSONNEAU Pierre	ENSIEG	MAZARE Guy	ENSIMAG
BRUNET Yves	IUFA	MOREAU René	ENSHMG
CAILLERIE Denis	ENSHMG	MORET Roger	ENSIEG
CAVAIGNAC Jean-François	ENSPG	MOSSIERE Jacques	ENSIMAG
CHARTIER Germain	ENSPG	OBLED Charles	ENSHMG
CHENEVIER Pierre	ENSERG	OZIL Patrick	ENSEEG
CHERADAME Hervé	UFR PGP	PARIAUD Jean-Charles	ENSEEG
CHOVET Alain	ENSERG	PERRET René	ENSIEG
COHEN Joseph	ENSERG	PERRET Robert	ENSIEG
COUMES André	ENSERG	PIAU Jean-Michel	ENSHMG
DARVE Félix	ENSHMG	POUPOT Christian	ENSERG
DELLA-DORA Jean	ENSIMAG	RAMEAU Jean-Jacques	ENSEEG
DEPORTES Jacques	ENSPG	RENAUD Maurice	UFR PGP
DOLMAZON Jean-Marc	ENSERG	ROBERT André	UFR PGP
DURAND Francis	ENSEEG	ROBERT François	ENSIMAG
DURAND Jean-Louis	ENSIEG	SABONNADIERE Jean-Claude	ENSIEG
FOGGIA Albert	ENSIEG	SAUCIER Gabrielle	ENSIMAG
FONLUPTJean	ENSIMAG	SCHLENKER Claire	ENSPG
FOULARD Claude	ENSIEG	SCHLENKER Michel	ENSPG
GANDINI Alessandro	UFR PGP	SILVY Jacques	UFR PGP
GAUBERT Claude	ENSPG	SIRIEYS Pierre	ENSHMG
GENTIL Pierre	ENSERG	SOHM Jean-Claude	ENSEEG
GREVEN Hélène	IUFA	SOLER Jean-Louis	ENSIMAG
GUERIN Bernard	ENSERG	SOUQUET Jean-Louis	ENSEEG
GUYOT Pierre	ENSEEG	TROMPETTE Philippe	ENSHMG
IVANES Marcel	ENSIEG	VEILLON Gérard	ENSIMAG
JAUSSAUD Pierre	ENSIEG	ZADWORNY François	ENSERG

Professeur Université des Sciences Sociales (Grenoble II)

BOLLIET Louis

Personnes ayant obtenu le diplôme d'HABILITATION A DIRIGER DES RECHERCHES

BECKER Monique
BINDER Zdenek
CHASSERY Jean-Marc
CHOLLET Jean-Pierre
COEY John
COLINET Catherine
COMMAULT Christian
CORNUEJOLS Gérard
COULOMB Jean- Louis
DALARD Francis
DANES Florin
DEROO Daniel
DIARD Jean-Paul
DION Jean-Michel
DUGARD Luc
DURAND Madeleine
DURAND Robert
GALERIE Alain
GAUTHIER Jean-Paul
GENTIL Sylviane
GHIBAUDO Gérard
HAMAR Sylvaine
HAMAR Roger
LADET Pierre
LATOMBE Claudine
LE GORREC Bernard
MADAR Roland
MULLER Jean
NGUYEN TRONG Bernadette
PASTUREL Alain
PLA Fernand
ROUGER Jean
TCHUENTE Maurice
VINCENT Henri

Chercheurs du C.N.R.S
Directeurs de recherche 1ère Classe

CARRE René
FRUCHART Robert
HOPFINGER Emile
JORRAND Philippe
LANDAU Ioan
VACHAUD Georges
VERJUS Jean-Pierre

Directeurs de recherche 2ème Classe

ALEMANY Antoine
ALLIBERT Colette
ALLIBERT Michel
ANSARA Ibrahim
ARMAND Michel
BERNARD Claude
BINDER Gilbert
BONNET Roland
BORNARD Guy
CAILLET Marcel
CALMET Jacques
COURTOIS Bernard
DAVID René
DRIOLE Jean
ESCUDIER Pierre
EUSTATHOPOULOS Nicolas
GUELIN Pierre
JOUD Jean-Charles
KLEITZ Michel
KOFMAN Walter
KAMARINOS Georges
LEJEUNE Gérard
LE PROVOST Christian
MADAR Roland
MERMET Jean
MICHEL Jean-Marie
MUNIER Jacques
PIAU Monique
SENATEUR Jean-Pierre
SIFAKIS Joseph
SIMON Jean-Paul
SUERY Michel
TEODOSIU Christian
VAUCLIN Michel
WACK Bernard

Personnalités agrées à titre permanent à diriger des travaux de recherche (décision du conseil scientifique)

E.N.S.E.E.G

CHATILLON Christian
HAMMOU Abdelkader
MARTIN GARIN Régina
SARRAZIN Pierre
SIMON Jean-Paul

E.N.S.E.R.G

BOREL Joseph

E.N.S.I.E.G

DESCHIZEAUX Pierre
GLANGEAUD François
PERARD Jacques
REINISCH Raymond

E.N.S.H.G

ROWE Alain

E.N.S.I.M.A.G

COURTIN Jacques

E.F.P.

CHARUEL Robert

C.E.N.G

CADET Jean
COEURE Philippe
DELHAYE Jean-Marc
DUPUY Michel
JOUVE Hubert
NICOLAU Yvan
NIFENECKER Hervé
PERROUD Paul
PEUZIN Jean-Claude
TAIB Maurice
VINCENDON Marc

Laboratoires extérieurs

C.N.E.T

DEVINE Rodericq
GERBER Roland
MERCKEL Gérard
PAULEAU Yves

UNIVERSITE Joseph FOURIER (GRENOBLE I)

Président de l'Université :
M. PAYAN Jean Jacques

Année Universitaire 1987 - 1988

MEMBRES DU CORPS ENSEIGNANT DE SCIENCES ET DE GEOGRAPHIE

PROFESSEURS DE 1ère Classe

ARNAUD Paul	Chimie Organique
ARVIEU ROBERT	Physique Nucléaire I.S.N.
AUBERT Guy	Physique C.N.R.S
AURIAULT Jean-Louis	Mécanique
AYANT Yves	Physique Approfondie
BARBIER Marie-Jeanne	Electrochimie
BARJON Robert	Physique Nucléaire ISN
BARNOUD Fernand	Biochimie Macromoléculaire Végétale
BARRA Jean-René	Statistiques-Mathématiques Appliquées
BECKER Pierre	Physique
BEGUIN Claude	Chimie Organique
BELORISKY Elie	Physique
BENZAKEN Claude	Mathématiques Pures
BERARD Pierre	Mathématiques Pures
BERNARD Alain	Mathématiques Pures
BERTRANDIAS Françoise	Mathématiques Pures
BERTRANDIAS Jean-Paul	Mathématiques Pures
BILLET Jean	Géographie
BOELHER Jean-Paul	Mécanique
BONNIER Jane Marie	Chimie Générale
BOUCHEZ Robert	Physique Nucléaire ISN
BRAVARD Yves	Géographie
CARLIER Georges	Biologie Végétale
CAUQUIS Georges	Chimie Organique
CHARDON Michel	Géographie
CHIBON Pierre	Biologie Animale
COHEN ADDAD Jean-Pierre	Physique
COLIN DE VERDIERE Yves	Mathématiques Pures
CYROT Michel	Physique du Solide
DEBELMAS Jacques	Géologie Générale
DEGRANGE Charles	Zoologie
DEMAILLY Jean-Pierre	Mathématiques Pures
DENEUVILLE Alain	Physique
DEPORTES Charles	Chimie Minérale
DOLIQUE Jean-Michel	Physique des Plasmas
DOUCE Roland	Physiologie Végétale
DUCROS Pierre	Cristallographie
FONTAINE Jean-Marc	Mathématiques Pures
GAGNAIRE Didier	Chimie Physique
GERMAIN Jean-Pierre	Mécanique,
GIRAUD Pierre	Géologie
HICTER Pierre	Chimie
IDELMAN Simon	Physiologie Animale
JANIN Bernard	Géographie
JOLY Jean-René	Mathématiques Pures
KAHANE André, détaché	Physique
KAHANE Josette	Physique
KRAKOWIAK Sacha	Mathématiques Appliquées

LAJZEROWICZ Jeanine	Physique
LAJZEROWICZ Joseph	Physique
LAURENT Pierre-Jean	Mathématiques Appliquées
LEBRETON Alain	Mathématiques Appliquées
DE LEIRIS Joël	Biologie
LHOMME Jean	Chimie
LLIBOUTRY Louis	Géophysique
LOISEAUX Jean-Marie	Sciences Nucléaires I.S.N.
LUNA Domingo	Mathématiques Pures
MACHE Régis	Physiologie Végétale
MASCLE Georges	Géologie
MAYNARD Roger	Physique du Solide
OMONT Alain	Astrophysique
OZENDA Paul	Botanique (Biologie Végétale)
PAYAN Jean-Jacques	Mathématiques Pures
PEBAY-PEYROULA Jean-Claude	Physique
PERRIER Guy	Géophysique
PIERRARD Jean-Marie	Mécanique
PIERRE Jean-Louis	Chimie Organique
RENARD Michel	Thermodynamique
RINAUDO Marguerite	Chimie CERMAV
ROSSI André	Biologie
SAXOD Raymond	Biologie Animale
SENGEL Philippe	Biologie Animale
SERGERAERT Francis	Mathématiques Pures
SOUCHIER Bernard	Biologie
SOUTIF Michel	Physique
STUTZ Pierre	Mécanique
TRILLING Laurent	Mathématiques Appliquées
VALENTIN Jacques	Physique Nucléaire I.S.N.
VAN CUTSEM Bernard	Mathématiques Appliquées
VIALON Pierre	Géologie

PROFESSEURS de 2ème Classe

ADIBA Michel	Mathématiques Pures
ANTOINE Pierre	Géologie
ARMAND Gilbert	Géographie
BARET Paul	Chimie
BLANCHI J.Pierre	STAPS
BLUM Jacques	Mathématiques Appliquées
BOITET Christian	Mathématiques Appliquées
BORNAREL Jean	Physique
BRUANDET J.François	Physique
BRUGAL Gérard	Biologie
BRUN Gilbert	Biologie
CASTAING Bernard	Physique
CERFF Rudiger	Biologie
CHIARAMELLA Yves	Mathématiques Appliquées
COURT Jean	Chimie
DUFRESNOY Alain	Mathématiques Pures
GASPARD François	Physique
GAUTRON René	Chimie
GENIES Eugène	Chimie
GIDON Maurice	Géologie
GIGNOUX Claude	Sciences Nucléaires
GILLARD Roland	Mathématiques Pures
GIORNI Alain	Sciences Nucléaires
GONZALEZ SPRINBERG Gérardo	Mathématiques Pures
GUIGO Maryse	Géographie
GUMUCHAIN Hervé	Géographie
GUITTON Jacques	Chimie

HACQUES Gérard	Mathématiques Appliquées
HERBIN Jacky	Géographie
HERAULT Jeanny	Physique
JARDON Pierre	Chimie
JOSELEAU Jean-Paul	Biochimie
KERCKHOVE Claude	Géologie
LONGEQUEUE Nicole	Sciences Nucléaires I.S.N.
LUCAS Robert	Physique
MANDARON Paul	Biologie
MARTINEZ Francis	Mathématiques Appliquées
NEMOZ Alain	Thermodymamique CNRS - CRTBT
OUDET Bruno	Mathématiques Appliquées
PECHER Arnaud	Géologie
PELMONT Jean	Biochimie
PERRIN Claude	Sciences Nucléaires I.S.N.
PFISTER Jean-Claude	Physique du Solide
PIBOULE Michel	Géologie
RAYNAUD Hervé	Mathématiques Appliquées
RICHARD Jean-Marc	Physique
RIEDTMANN Christine	Mathématiques Pures
ROBERT Gilles	Mathématiques Pures
ROBERT Jean-Bernard	Chimie Physique
SARROT-REYNAULD Jean	Géologie
SAYETAT Françoise	Physique
SERVE Denis	Chimie
STOECKEL Frédéric	Physique
SCHOLL Pierre-Claude	Mathématiques Appliquées
SUBRA Robert	Chimie
VALLADE Marcel	Physique
VIDAL Michel	Chimie Organique
VIVIAN Robert	Géographie
VOTTERO Philippe	Chimie

MEMBRES DU CORPS ENSEIGNANT DE L' IUT 1

PROFESSEURS de 1ère Classe

BUISSON Roger	Physique IUT 1
DODU Jacques	Mécanique Appliquée IUT 1
NEGRE Robert	Génie Civil IUT 1
NOUGARET Marcel	Automatique IUT 1
PERARD Jacques	EEA. IUT 1

PROFESSEURS de 2ème classe

BOUTHINON Michel	EEA. IUT 1
CHAMBON René	Génie Mécanique IUT 1
CHEHIKIAN Alain	EEA. IUT 1
CHENAVAS Jean	Physique IUT 1
CHOUTEAU Gérard	Physique IUT 1
CONTE René	Physique IUT 1
GOSSE Jean-Pierre	EEA.IUT 1
GROS Yves	Physique IUT 1
KUHN Gérard, (Détaché)	Physique IUT 1
MAZUER Jean	Physique IUT 1
MICHOULIER Jean	Physique IUT 1
MONLLOR Christian	EEA.IUT 1
PEFFEN René	Métallurgie IUT 1
PERRAUD Robert	Chimie IUT 1
PIERRE Gérard	Chimie IUT 1
TERRIEZ Jean-Michel	Génie Mécanique IUT 1
TOUZAIN Philippe	Chimie IUT 1
VINCENDON Marc	Chimie IUT 1

PROFESSEURS DE PHARMACIE

AGNIUS-DELORD Claudine	Physique	Faculté La Tronche
ALARY Josette	Chimie Analytique	Faculté La Tronche
BERIEL Hélène	Physiologie et Pharmacologie	Faculté La Tronche
CUSSAC Max	Chimie Therapeutique	Faculté La Tronche
DEMENGE Pierre	Pharmacodynamie	Faculté La Tronche
FAVIER Alain	Biochimie	C.H.R.G.
JEANNIN Charles	Pharmacie Galénique	Faculté Meylan
LATURAZE Jean	Biochimie	Faculté La Tronche
LUU DUC Cuong	Chimie Générale	Faculté La Tronche
MARIOTTE Anne-Marie	Pharmacognosie	Faculté La Tronche
MARZIN Daniel	Toxicologie	Faculté Meylan
RENAUDET Jacqueline	Bactériologie	Faculté La Tronche
ROCHAT Jacques	Hygiéne et Hydrologie	Faculté La Tronche
SEIGLE-MURANDI Françoise	Botanique et Cryptogamie	Faculté Meylan
VERAIN Alice	Pharmacie Galénique	Faculté Meylan

MEMBRES DU CORPS ENSEIGNANT DE MEDECINE

PROFESSEURS CLASSE EXEPTIONNELLE ET 1ère CLASSE

AMBLARD Pierre	Dermatologie	C.H.R.G.
AMBROISE-THOMAS Pierre	Parasitologie	C.H.R.G.
BEAUDOING André	Pédiatrie-Puericulture	C.H.R.G.
BEZEZ Henri	Orthopédie-Traumatologie	Hopital SUD
BONNET Jean-Louis	Ophtalmologie	C.H.R.G.
BOUCHET Yves	Anatomie	Faculté La Merci
	Chirurgie Générale et Digestive	C.H.R.G.
BUTEL Jean	Orthopédie-Traumatologie	C.H.R.G.
CHAMBAZ Edmond	Biochimie	C.H.R.G.
CHAMPETIER Jean	Anatomie-Topographique et Appliquée	C.H.R.G.
CHARACHON Robert	O.R.L.	C.H.R.G.
COLOMB Maurice	Immunologie	Hopital sud
COUDERC Pierre	Anatomie-Pathologique	C.H.R.G.
DELORMAS Pierre	Pneumophtisiologie	C.H.R.G.
DENIS Bernard	Cardiologie	C.H.R.G.
GAVEND Michel	Pharmacologie	Faculté La Merci
HOLLARD Daniel	Hématologie	C.H.R.G.
LATREILLE René	Chirurgie Thoracique et Cardiovasculaire	C.H.R.G.
LE NOC Pierre	Bactériologie-Virologie	C.H.R.G.
MALINAS Yves	Gynécologie et Obstétrique	C.H.R.G.
MALLION Jean-Michel	Médecine du Travail	C.H.R.G.
MICOUD Max	Clinique Médicale et Maladies Infectieuses	C.H.R.G.
MOURIQUAND Claude	Histologie	Faculté La Merci
PARAMELLE Bernard	Pneumologie	C.H.R.G.
PERRET Jean	Neurologie	C.H.R.G.
RACHAIL Michel	Hépato-Gastro-Entérologie	C.H.R.G.
DE ROUGEMONT Jacques	Neurochirurgie	C.H.R.G.
SARRAZIN Roger	Clinique Chirurgicale	C.H.R.G.
STIEGLITZ Paul	Anestésiologie	C.H.R.G.
TANCHE Maurice	Physiologie	Faculté La Merci
VIGNAIS Pierre	Biochimie	Faculté La Merci

PROFESSEURS 2ème CLASSE

BACHELOT Yvan	Endocrinologie	C.H.R.G.
BARGE Michel	Neurochirurgie	C.H.R.G.
BENABID Alim Louis	Biophysique	Faculté La Merci
BENSA Jean-Claude	Immunologie	Hopital Sud
BERNARD Pierre	Gynécologie-Obstétrique	C.H.R.G.
BESSARD Germain	Pharmacologie	ABIDJAN
BOLLA Michel	Radiothérapie	C.H.R.G.
BOST Michel	Pédiatrie	C.H.R.G.
BOUCHARLAT Jacques	Psychiatrie Adultes	Hopital Sud
BRAMBILLA Christian	Pneumologie	C.H.R.G.
CHIROSSEL Jean-Paul	Anatomie-Neurochirurgie	C.H.R.G.
COMET Michel	Biophysique	Faculté La Merci
CONTAMIN Charles	Chirurgie Thoracique et Cardiovasculaire	C.H.R.G.
CORDONNIER Daniel	Néphrologie	C.H.R.G.
COULOMB Max	Radiologie	C.H.R.G.
CROUZET Guy	Radiologie	C.H.R.G.
DEBRU Jean-Luc	Médecine Interne et Toxicologie	C.H.R.G.
DEMONGEOT Jacques	Biostatistiques et Informatique Médicale	Faculté La Merci
DUPRE Alain	Chirurgie Générale	C.H.R.G.
DYON Jean-François	Chirurgie Infantile	C.H.R.G.
ETERRADOSSI Jacqueline	Physiologie	Faculté La Merci
FAURE Claude	Anatomie et Organogénèse	C.H.R.G.
FAURE Gilbert	Urologie	C.H.R.G.
FOURNET Jacques	Hépato-Gastro-Entérologie	C.H.R.G.
FRANCO Alain	Médecine Interne	C.H.R.G.
GIRARDET Pierre	Anesthésiologie	C.H.R.G.
GUIDICELLI Henri	Chirurgie Générale et Vasculaire	C.H.R.G.
GUIGNIER Michel	Thérapeutique et Réanimation Médicale	C.H.R.G.
HADJIAN Arthur	Biochimie	Faculté La Merci
HALIMI Serge	Endocrinologie et Maladies Métaboliques	C.H.R.G.
HOSTEIN Jean	Hépato-Gastro-Entérologie	C.H.R.G.
HUGONOT Robert	Médecine Interne	C.H.R.G.
JALBERT Pierre	Histologie-Cytogénétique	C.H.R.G.
JUNIEN-LAVILLAUROY Claude	O.R.L.	C.H.R.G.
KOLODIE Lucien	Hématologie Biologique	C.H.R.G.
LETOUBLON Christian	Chirurgie Générale	C.H.R.G.
MACHECOURT Jacques	Cardiologie et Maladies Vasculaires	C.H.R.G.
MAGNIN Robert	Hygiène	C.H.R.G.
MASSOT Christian	Médecine Interne	C.H.R.G.
MOUILLON Michel	Ophtalmologie	C.H.R.G.
PELLAT Jacques	Neurologie	C.H.R.G.
PHELIP Xavier	Rhumatologie	C.H.R.G.
RACINET Claude	Gynécologie-Obstétrique	Hopital Sud
RAMBAUD Pierre	Pédiatrie	C.H.R.G.
RAPHAEL Bernard	Stomatologie	C.H.R.G.
SCHAERER René	Cancérologie	C.H.R.G.
SEIGNEURIN Jean-Marie	Bactériologie-Virologie	Faculté La Merci
SELE Bernard	Cytogénétique	Faculté La Merci
SOTTO Jean-Jacques	Hématologie	C.H.R.G.
STOEBNER Pierre	Anatomie Pathologique	C.H.R.G.
VROUSOS Constantin	Radiothérapie	C.H.R.G.

AVANT PROPOS

Tout d'abord, qu'il me soit permis d'adresser ma reconnaissance à Monsieur ***B.Courtois*** *pour la confiance qu'il m'a témoignée en m'accueillant dans son laboratoire et pour avoir dirigé avec compétence tout ce travail.*

Que monsieur ***G.Mazare*** *Professeur à l'ENSIMAG, trouve ici l'expression de ma gratitude pour l'intérêt qu'il a porté à ce travail en me faisant l'honneur de présider le jury de cette thèse.*

Monsieur ***G.Chambon,*** *Professeur à l' Université des science et technique du languedoc et Monsieur* ***J.Frehel,*** *Directeur de recherche à SGS-Thompson ont bien voulu mobiliser leur temps et leurs compétences pour juger ce travail; je tiens à les assurer de mes sincères remerciements.*

Que tous mes collègues de laboratoire trouvent ici ma gratitude pour l'ambiance sympathique et amicale de travail. Un grand merci à ***M.Nicolaidis et K.Torki*** *pour les fructueuses discussions que nous avons tenues pendant toutes ces années.*

Enfin, je ne saurais oublier les amis ***Armand, Habiba, Bassan, Denis, Dominique, Chiên, Mohamad, Meryem, Jean Pierre, Isabelle, Fernando, Jayne, Aarão, Ana, Afonso, Renata, Vladimir et Lilian.*** *Je vous suis infiniment reconnaissant.*

à ***Rosana*** *et* ***Nicolas***

RESUME.

Dans ce travail, nous avons effectué une étude complète de la testabilité des PLAs. Pour les trois classes de test (hors ligne, en ligne et unifié) nous avons proposé des schémas de test dont la compatibilité avec les PLAs optimisés a été étudiée.

En ce qui concerne le test hors ligne, le schéma de test proposé apporte une amélioration vis-a-vis des schémas publiés dans la littérature tout en gardant la compatibilité avec les PLAs optimisés.

En ce qui concerne le test en ligne, l'application de deux schémas de test aux PLAs optimisés a été analysée: le schéma utilisant des codes non ordonnés et le schéma utilisant des codes détectant des erreurs simples. Un nouvel algorithme pour la génération des codes non ordonnés a été proposé.

En ce qui concerne le test unifié, le test en ligne est assuré par la méthode utilisant des codes non ordonnés. Deux méthodes sont proposées pour la génération des vecteurs d'entrée pendant le test hors ligne: le test exhaustif et le test déterministe.

MOTS-CLES:

PLA, Circuits autotestables, Circuits autocontrôlables, Test en ligne, Test hors ligne, Test unifié, Test, Test integré.

ABSTRACT

This Thesis deals with testability of the folded PLAs. We propose a testing scheme for each class of the test (off-line, on-line and unified BIST). Their application on the topological optimization cases are discussed in detail. The constraints inherent to these applications are given.

In the off-line testing, the proposed testing scheme is an improvement over other existing schemes and it is convenient for topologically optimized PLAs.

In the on-line testing, two testing schemes are examined : the testing scheme using the unordered code and the testing scheme using the single error detecting code. A new algorithm is proposed to generate the unordered code.

In the UBIST (Unified Built-In Self-Test) the on-line testing is ensured by the method using the unordered code. The input pattern generation in the off-line phase can be ensured by two methods : determinist testing and exhaustive testing.

Keywords:

PLAs, folding, self-checking systems, UBIST.

SOMMAIRE

INTRODUCTION

1 - INTRODUCTION.

1.1 - GENERALITES.

Les circuits VLSI conçus actuellement sont marqués par une très grande complexité due au nombre de transistors qu'ils comportent ainsi qu'aux fonctions qu'ils réalisent. La conception de ces circuits doit impérativement utiliser non seulement des outils de conception assistée par ordinateur (CAO) mais aussi des structures régulières. Les RAMs (**R**andom **A**cess **M**emories), ROMs (**R**ead-**O**nly **M**emories) et PLAs (**P**rogrammable **L**ogic **A**rrays) sont les structures régulières le plus utilisées et tout particulièrement les PLAs qui sont de puissantes structures pour l'implantation de multiples fonctions combinatoires. Leur structure contient seulement deux niveaux logiques, ce qui simplifie les processus de conception et de vérification. En outre, il est relativement simple d'implanter une structure testable. Par contre, la surface qu'ils occupent par rapport à la logique anarchique limite leur utilisation. Pour résoudre ce problème, c'est-à-dire, pour augmenter l'application des PLAs, des techniques d'optimisation ont été développées. L'optimisation logique permet l'élimination des entrées, sorties et monômes redondants ainsi que la réduction du taux de remplissage i.e. du nombre de transistors dans les deux matrices du PLA. L'optimisation topologique réorganise la structure interne des PLAs pour obtenir un gain de surface soit par la technique de brisure (folding) soit par la technique de partitionnement. Ceci permet non seulement de diminuer la surface et la consommation, mais aussi d'augmenter la vitesse des PLAs tout en élargissant leur champ d'utilisation. En effet, les compromis de temps, de surface et de consommation dans les circuits intégrés sont très importants. Par conséquent les structures qui occupent une grande surface, tout en ayant une forte consommation ou dont la vitesse de commutation est faible ont une utilisation assez limitée.

Un autre facteur important dans les circuits actuels est la testabilité. Le test exhaustif (i.e. l' application de 2^n vecteurs sur un circuit à n entrées) peut être facilement appliqué pour des circuits ayant un faible nombre d'entrées, mais dès que ce nombre augmente il est impossible de tester le circuit de cette manière en un temps raisonnable. Plusieurs solutions sont données pour résoudre ce problème telles que le partitionnement du

circuit en sous circuits ou bien l'emploi d'un test déterministe (i.e. appliquer au circuit un ensemble de vecteurs capables de mettre en évidence une panne modélisée). La première solution suppose que nous ayons accès à ces sous circuits et la deuxième présente des problèmes pour la détermination des vecteurs ce qui entraîne un faible taux de couverture de pannes.

Cela fait apparaître de nouvelles classes de circuits conçus avec des facilités de test (**Design For Testability - DFT**) ou même des circuits à test intégré (**Built-In Self Test - BIST**). Ces circuits sont apparus grâce à l'augmentation de la capacité d'intégration.

Plusieurs niveaux de test peuvent être considérés dans les circuits intégrés.

Après la conception, il est nécessaire de s'assurer que les fonctions réalisées par le circuit sont bien les fonctions que nous voulons implanter et que le circuit fonctionne comme nous l'avons prévu. Il s'agit de la validation de la conception, elle est faite en testant des prototypes.

Après la fabrication, il faut éliminer les circuits défectueux et périodiquement effectuer des tests de maintenance pour les circuits assemblés sur des cartes.

Pendant le fonctionnement normal du circuit, il est important de détecter une panne avant sa propagation dans d'autres parties du système.

Ainsi les méthodes de test peut être divisées en deux catégories:

- Le test hors ligne et
- Le test en-ligne.

Le test hors ligne est réalisé hors le fonctionnement normal du circuit. Il peut être divisé en deux catégories:

- Le test paramétrique et
- Le test logique.

Dans le premier cas, les différents paramètres qui caractérisent le circuit, tels que la tension de seuil, le courant, l'impédance, etc. (paramètres statiques) et le temps de commutation, les délais, etc. (paramètres dynamiques) sont mesurés. Dans le deuxième cas, le fonctionnement logique est vérifié. Ce test permet de détecter des fautes permanentes pour

l'acceptation initiale ou pour le test de maintenance du système. Lors du test hors ligne, le fonctionnement normal du circuit est interrompu et une séquence de test est appliquée. Les sorties sont contrôlées pour détecter une possible erreur. Dans notre travail nous étudierons le test logique pour les PLAs.

Le test en ligne est réalisé pendant le fonctionnement normal du circuit. Il peut aussi être divisé en deux catégories:

- Le test en ligne discontinu et
- Le test en ligne continu.

Le test en ligne discontinu exécute des procédures de test spécifiques quand le circuit n'a aucune tâche fonctionnelle à exécuter. Le test en ligne continu assure la détection d'erreurs pendant l'exécution d'une tâche quelconque. Il est utilisé pour surveiller le système pendant son fonctionnement normal. Dans ce cas toutes les fautes, permanentes et intermittentes peuvent être détectées. Les circuits qui assurent la détection d'erreur en ligne sont appelés "self checking". Dans notre travail nous étudierons le test en ligne continu pour les PLAs.

Une nouvelle classe proposée par NICOLAIDIS [NIC 86], appelé UBIST (**U**nified **B**uilt-**I**n **S**elf **T**est), associe le test hors ligne et le test en ligne. Ainsi tous les tests nécessaire à un système VLSI sont assurés. En effet le test en ligne et le test hors ligne ont été développés indépendament l'un de l'autre. Pour pouvoir exploiter au maximum les avantages que l'un pourrait apporter à l'autre il faut développer des nouvelles méthodes. Il est important de remarquer que généralement les propriétés des "circuits self-checking" sont vérifiées pour des fautes simples mais des fautes multiples peuvent se produirent spécialement après le processus de fabrication. Il est donc nécessaire de s'assurer que les pannes multiples seront détectées avant l'utilisation du circuit, par le test hors ligne. Ainsi le schéma unifié offre une plus haute couverture de fautes et assure un test de très haute qualité.

Dans ce travail le test des PLAs est étudié. Nous avons abouti à des schémas optimisés par rapport aux schémas proposés dans la littérature et ces schémas sont compatibles avec les PLAs optimisés topologiquement.

Le chapitre 2 présente les différentes méthodes d'optimisation des PLAs.

Le test hors ligne des PLAs est exposé dans le chapitre 3. Un bref apperçu des diverses méthodes exposées dans la littérature est fait avant la proposition d'un schéma. Dans ce schéma trois caractéristiques sont prises en compte:

- La compatibilité avec les PLAs optimisés,
- La surface additionnelle et
- La couverture des fautes.

Ce schéma est une amélioration des schémas existants et il reste compatible avec la structure des PLAs optimisés. L'application de ce schéma aux PLAs optimisés est aussi détaillée.

Le test en ligne est exposé dans le chapitre 4. Une revue des schémas de test proposés dans la littérature est faite et son application dans les PLAs optimisés est démontrée. Une nouvelle méthode pour la génération du codage des sorties est proposée et des exemples d'application sont donnés.

Le test unifié est exposé dans le chapitre 5. Les modifications nécessaires à la logique additionnelle sont montrées ainsi que l'application du schéma de test aux PLAs optimisés. Le test exhaustif est utilisé pour les PLAs ayant un faible nombre d'entrées.

La couverture de faute de chaque schéma est donnée dans les annexes.

Les règles de dessin utilisées dans ce travail sont celle proposées par MEAD et CONWAY [MEA 80].

La terminologie utilisée est la suivante:

- n nombre d'entrées,
- m nombre de monômes et
- p nombre de sorties.

1.2 - LES HYPOTHESES DE PANNES.

La conception des circuits autotestables nécessite un modèle de panne qui représente les défaillances réelles. Il est nécessaire de considérer les défauts physiques qui surviennent dans une technologie donnée. Puisque ces défauts physiques sont généralement connus, il est raisonnable de déterminer les effets de ces défauts au niveau logique. Ceci correspond à la modélisation des pannes.

Le collage logique a été longtemps le modèle pris en compte pour le test

des circuits. Ce modèle couvre une partie des pannes physiques mais il n'est pas représentatif [COU 81] et [GAL 80] de toutes les pannes réelles des circuits.

Plusieurs pannes physiques ont un effet plus complexe dans les circuits. Le dosage d'implantation ionique, par exemple, peut provoquer un décalage dans les niveaux d'un transistor et peut amener sur la sortie une tension située entre les tensions assignées au "0" logique et au "1" logique. Ce comportement résulte en un "1" ou "0" faible. En effet le collage à "1" est une dégénérescence d'un "0" faible et le collage à "0" est une dégénérescence d'un "1" faible [TAM 84]. Un court-circuit impose sur les deux lignes une même valeur. Cette valeur doit être comprise entre la valeur logique "1" et la valeur logique "0" (les deux lignes sont supposées avoir des valeurs complémentaires) et donc la ligne supposée avoir la valeur "0" aura un "0" faible et l'autre un "1" faible.

Nous pouvons décrire à présent la modélisation de pannes des PLAs que nous utiliserons.

La structure régulière des PLAs permet de grouper les fautes dans un petit nombre de types. Elle permet aussi de simplifier l'analyse des effets de chaque faute modélisée ainsi que la détermination de la couverture des fautes.

Les fautes permanentes dans la structure des PLAs peuvent être modélisées par les fautes logiques suivantes [CHA 78]:

- Collage d'une entrée, monôme ou sortie à "0" ou à "1",
- Court-circuit entre deux lignes adjacentes ("bridging") et
- "Crosspoint".

Le collage logique ainsi que le court circuit correspondent à des défauts physiques dans les PLAs. Le crosspoint correspond à des fautes dont l'effet équivaut à la présence ou à l'absence anormale de transistors, dans les matrices ET et OU.

Y.TAMIR et C.H.SEQUIN [TAM 84] ont proposé un modèle de panne pour les PLAs représentatif des pannes permanentes et intermittentes. Dans ce modèle, les fautes dues à un "1" ou "0" faible et les coupures des lignes d'entrée, de monôme et de sortie sont ajoutées à l'ensemble des fautes permanentes. Ainsi l'ensemble complet des pannes permanentes et intermittentes peut être modélisé par [TAM 84]:

- 1/0 faible d'une ligne d'entrée,
- Court-circuit entre deux lignes d'entrée adjacentes,
- 1/0 faible d'un monôme,
- Court circuit entre deux monômes adjacents,
- 1/0 faible d'une sortie,
- Court circuit entre deux sorties adjacentes,
- Court circuit entre une ligne d'entrée et un monôme,
- Court circuit entre un monôme et une sortie,
- Transistor supplémentaire dans la matrice ET,
- Transistor supplémentaire dans la matrice OU,
- Coupure d'un monôme et
- Coupure d'une sortie.

Le collage à 1 est une dégénérescence d'un "0" faible tandis que le collage à "0" est une dégénérescence d'un "1" faible. L'absence d'un transistor dans la matrice ET a l'effet d'un "1" faible d'une ligne d'entrée et l'absence d'un transistor dans la matrice OU a l'effet d'un "1" faible d'un monôme. Ceci explique l'exclusion de ces pannes de l'ensemble donné.

Quand ces fautes sont propagées aux sorties des PLAs elles provoquent des erreurs unidirectionnelles. Cela a été démontré dans [MAK 82] pour les fautes simples dues à un collage, court-circuit et "crosspoint". Le "1" faible se propage de la même manière que le collage à "0". De même le "0" faible se propage de la même manière que le collage à "1". Les coupures ont l'effet soit d'un "1" faible soit d'un collage à "0". Nous pouvons donc conclure que l'ensemble des pannes modélisées provoquent des erreurs unidirectionnelles aux sorties des PLAs.

TECHNIQUES D'OPTIMISATION DES PLAs

2 - TECHNIQUES D'OPTIMISATION DES PLAs.

2.1 - GENERALITES.

Les réseaux logiques programmables (PLAs) sont des structures régulières qui permettent la réalisation de circuits logiques de type combinatoire ou séquentiel.

Les circuits combinatoires sont des circuits dont les sorties sont des fonctions booléennes, exprimées en sommes de produits des variables d'entrée. Ainsi les PLAs sont composés de deux matrices indépendantes, la matrice ET et la matrice OU. La matrice ET génère, à partir des variables d'entrée les monômes (qui sont les produits des entrées). La matrice OU génère, à partir des monômes, les sorties (qui sont les sommes des monômes).

Les circuits séquentiels sont des circuits dont les sorties dépendent non seulement des entrées au même instant (circuit combinatoire) mais aussi des entrées aux instants précédents. Des circuits séquentiels peuvent être réalisés par des PLAs à l'aide d'éléments de mémorisation placés comme des entrées additionnelles de la matrice ET et chargés par les sorties de la matrice OU (PLA statique). Cette connexion constitue le bouclage des circuits séquentiels. Dans le cas des PLAs dynamiques les éléments de mémorisation ne sont plus nécessaires, les sorties sont rebouclées directement aux entrées correspondantes.

Différentes représentations sont utilisées pour décrire les différents niveaux de conception et de vérification des PLAs. L'ensemble des descriptions utilisées, qui est montré dans la figure 2.1, peut se résumer en trois catégories, qui sont:

- Description fonctionnelle,
- Description structurelle et
- Description physique.

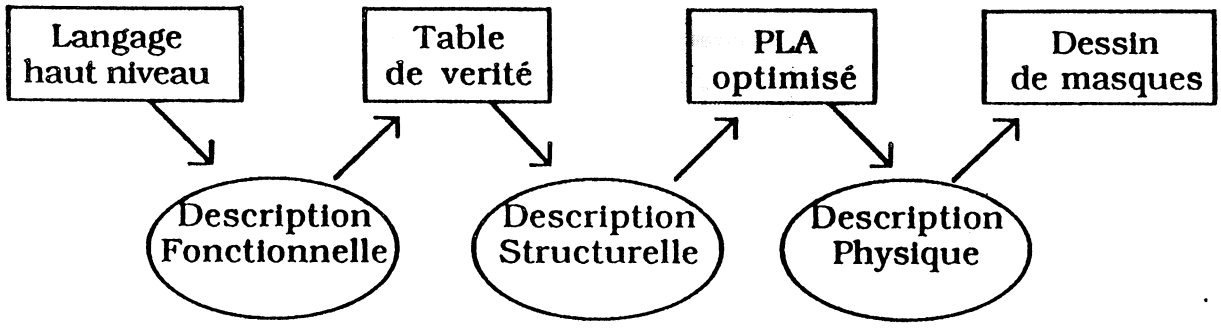


Figure 2.1: Représentation des PLAs.

Dans la description fonctionnelle, le concepteur décompose la réalisation en une série d'algorithmes structurés de complexité décroissante. L'ensemble des sorties sont décrites en fonction de l'ensemble des entrées par une langage de haut niveau ou bien par des équations booléennes. L'implantation du circuit lui même n'est pas défini dans cette phase. Ici on s'occupe principalement de la spécification des conditions qui doivent être testées, de la définition de l'architecture ainsi que des caractéristiques opérationnelles.

La description structurelle permet d'effectuer la traduction entre la description fonctionnelle et la description physique. Dans cette phase, sont effectuées les optimisations logique et topologique. Les PLAs sont décrits par des tables de vérité ou bien une représentation topologique. Dans ce travail, tous les exemples considérés sont des PLAs traités au cours de cette phase. Ceci explique sa description détaillée dans les paragraphes suivants.

Dans la description physique les caractéristiques technologiques sont prises en compte et le dessin des masques est généré.

2.1.1 - Représentation des PLAs au niveau structurel.

Les PLAs, comme tous les circuits combinatoires, peuvent être représentés par une table de vérité qui spécifie les valeurs des variables de sortie pour chaque combinaison des valeurs des variables d'entrée. Soit F une fonction de n entrées et p sorties. L'ensemble des variables d'entrée de la fonction F est:

$$X_F = (X_1, X_2, \ldots, X_n) = X_i, \; i \text{ dans } (1,\ldots,n)$$

L'ensemble des variables de sortie de la fonction F est:

$$Y_F = (Y_1, Y_2, ..., Y_p) = Y_j, j \text{ dans } (1,...,p)$$

Un cube singulier défini par $P_k = (R_k, V_k) = R_k/V_k$ est une assignation des valeurs R_{kl} et V_{kj} à chacune des variables X_l et Y_j respectivement. $R_k=(R_{k1}, R_{k2}, ..., R_{kn})$ est le cube d'entrée et $V_k = (V_{k1}, V_{k2}, ..., V_{kp})$ est le cube de sortie. Le terme singulier indique l'existence d'une relation entre le cube d'entrée et le cube de sortie.

La figure 2.2 montre un exemple d'une table de vérité compacte qui est une représentation tabulaire des cubes P_k de la fonction F.

Terme	Cube d'entrée	Cube de sortie
	$X_1 X_2 X_3 X_4$	$Y_1 Y_2 Y_3$
1	1 1 0 0	1 0 0
2	0 1 0 0	1 0 1
3	0 1 1 0	1 0 1
4	* 1 1 1	1 0 0
5	* 0 * 1	0 0 0
6	1 * 1 0	0 0 0
7	0 0 0 0	0 1 1
8	* 1 0 1	0 1 0
9	0 0 1 0	0 0 0
10	1 0 0 0	0 0 0

Figure 2.2: Table de vérité compacte.

Un PLA réalise des fonctions exprimées sous la forme d'une somme des termes produits. Une fonctions Y_j est décrite par:

$$Y_j = (V_{kj} * t_k, \quad k, t_k \text{ dans } PT_F)$$

PT_F est l'ensemble des termes produits de la fonction F dont l'ensemble t_k est au moins lié à une variable de sortie Y_j ayant une valeur "1" ($V_{kj} = 1$).

Dans la figure 2.2, par exemple, $Y_1 = T_1 + T_2 + T_3 + T_4$ ou bien $Y_1=X_1X_2\overline{X}_3\overline{X}_4 + \overline{X}_1X_2\overline{X}_3\overline{X}_4 + \overline{X}_1X_2X_3\overline{X}_4 + X_2X_3X_4$. Le symbole "*", dans la table de vérité, représente une valeur indéfinie qui peut être considérée comme étant la valeur "0" ou "1".

Une autre manière de représenter une table de vérité est montrée dans la figure 2.3. Dans ce cas toutes les variables, entrée et entrée complémentaire, sont montrées. Donc une rangée représente un monôme et une colonne représente une entrée ou une entrée complémentaire ou une sortie.

	Cube d'entrée				Cube de sortie
Terme	$X_1\overline{X}_1$	$X_2\overline{X}_2$	$X_3\overline{X}_3$	$X_4\overline{X}_4$	$Y_1Y_2Y_3$
1	1 0	1 0	0 1	0 1	1 0 0
2	0 1	1 0	0 1	0 1	1 0 1
3	0 1	1 0	1 0	0 1	1 0 0
4	0 0	1 0	1 0	1 0	1 0 0
5	0 1	0 1	0 1	0 1	0 1 0
6	0 0	1 0	0 1	1 0	0 1 0

Figure 2.3: Table de vérité d'implantation.

La figure 2.4 montre la représentation topologique du PLA de la figure 2.3. L'utilisation d'une entrée (ou entrée complémentaire) dans la génération d'un monôme correspond au placement d'un transistor entre cette entrée (ou entrée complémentaire) et le monôme. De même, l'utilisation d'un monôme dans la génération d'une fonction de sortie correspond au placement d'un transistor entre ce monôme et la sortie. Ainsi la programmation des PLAs est obtenue par l'insertion des transistors dans les intersections convenables des monômes et des entrées/sorties. Généralement, dans la technologie MOS, ils sont implementés en une structure NOR-NOR. Donc, à partir de la table de vérité de la figure 2.3, les "1" correspondent au placement d'un transistor dans le croisement et le "0" signifie l'absence de transistor. Puisque la structure utilisée est NOR-NOR le cube d'entrée doit être inversé i.e. les transistors sont placés dans le croisement entre le monôme et la variable complémentaire. De même la sortie doit être inversée d'où l'inclusion des inverseurs à la sortie du PLA.

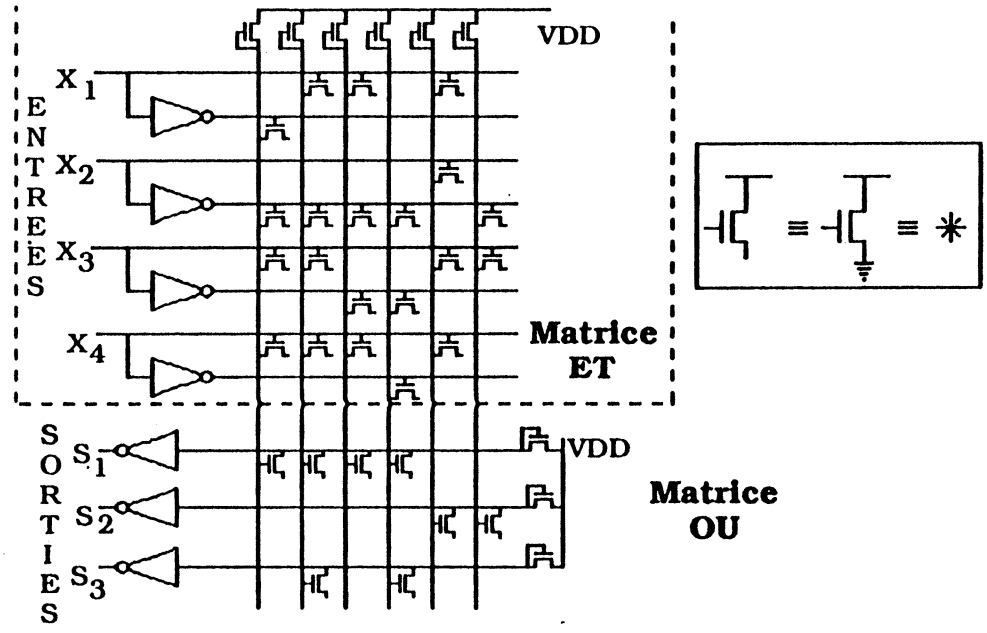


Figure 2.4: Représentation topologique d'un PLA.

2.2 - OPTIMISATION EN SURFACE DES PLAs.

La surface d'un PLA classique est proportionnelle au nombre des entrées/sorties et des monômes indépendamment des fonctions que le PLA réalise. Plus précisément, la surface d'un PLA est donnée par l'équation:

$$S_{PLA} = (2*n + p)*m + a\,n + b\,m + c\,p \qquad \text{(eq. 1)}$$

Où n, m et p sont, respectivement, le nombre des entrées, monômes et sorties et a, b et c sont des constantes associées directement à la topologie des amplificateurs-décodeurs des entrées, des sorties et des monômes. Cette surface peut être optimisée par deux techniques différentes, mais complémentaires:

- Optimisation logique et
- Optimisation topologique.

L'optimisation logique cherche la simplification des fonctions booléennes. Ainsi elle peut réduire la surface des PLAs par l'élimination des entrées, sorties et monômes redondants. Elle peut aussi réduire le nombre de transistors dans les matrices ET et OU. Cela permet de réduire le taux

de remplissage des matrices et en conséquence d'améliorer l'optimisation topologique.

L'optimisation topologique peut être obtenue par deux techniques:

- Partition ou segmentation et
- Pliage (brisure ou "folding").

La technique de partition consiste à diviser de grands PLAs en plusieurs petits PLAs [EGA 84]. La figure 2.5 montre l'exemple d'un PLA partitionné. Dans cette technique la structure du PLA est maintenue.

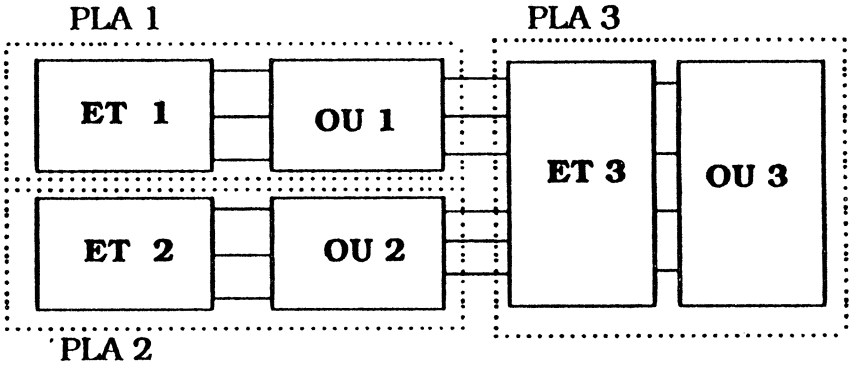


Figure 2.5: PLA partitionné.

Le pliage consiste à réorganiser les matrices ET et OU de façon à pouvoir placer deux ou plusieurs entrées et/ou sorties sur une même colonne et/ou deux ou plusieurs monômes sur une même rangée. L'optimisation par pliage peut être divisée en deux classes que nous décrirons dans les paragraphes suivants:

- Pliage simple et
- Pliage multiple.

Une ligne peut être partagée par deux monômes si et seulement si les deux monômes sont disjoints (voir définition D 2.1). Une colonne peut être partagée par deux entrées/sorties si et seulement si les deux entrées/sorties sont disjointes (voir définition D 2.2). Mais le fait que deux entrées, sorties ou monômes soient disjoints n'est pas l'unique contrainte d'application de l'optimisation topologique. Chaque paire d'entrée/sortie partageant une ligne, par exemple, introduit des contraintes sur les autres.

Ceci veut dire que deux entrées/sorties partageant une colonne doivent être l'une au dessous de l'autre et que les monômes doivent suivre cet ordonnancement. D'où l'introduction de contraintes pour les autres entrées/sorties. Ainsi un ensemble de pairs entrées/sorties disjoint peut être implanté si ses monômes peuvent être ordonnés.

DEFINITION D 2.1

Deux entrées sont dites disjointes si et seulement si elles excitent des monômes différents.

DEFINITION D 2.2

Deux sorties sont dites disjointes si et seulement si elles n'ont pas de monômes communs.

DEFINITION D 2.3

Deux monômes sont dits disjoints si et seulement si les entrées et les sorties qui leur sont rattachées sont disjointes.

En fonction de ces définitions HACHTEL, NEWTON et SANGIOVANNI-VINCENTELLI [HAC 82] ont formalisé le problème de l'optimisation topologique de la manière suivante:

"Determiner l'ensemble d'entrées, sorties et/ou monômes disjoints pour obtenir une surface minimale dont l'implantation soit possible".

Ce problème ainsi que ses variations est un problème NP-complet [LUB 82] et pour le résoudre plusieurs heuristiques ont été proposées dont la technique PAOLA [PER 80 et 85] [CHU 84] et PLEASURE [MIC 83].

2.2.1 - Le pliage simple.

Pour le pliage simple, une colonne au moins est partagée entre deux entrées et/ou sorties et/ou une ligne au moins est partagée entre deux monômes. Dans ce cas, si toutes les lignes sont partagées par deux

monômes et si toutes les colonnes sont partagés par deux entrées/sorties, le PLA sera réduit à 25% de la surface originale. Ceci est le gain maximal du pliage simple.

Le pliage simple d'entrées est décrit dans la figure 2.6. La particularité de cette technique est que les entrées sont placées sur deux côtés du PLA. La surface finale de la matrice ET, après l'optimisation topologique, est au maximum égale à 50%. Ce gain maximal est obtenu quand toutes les colonnes sont partagés par deux entrées.

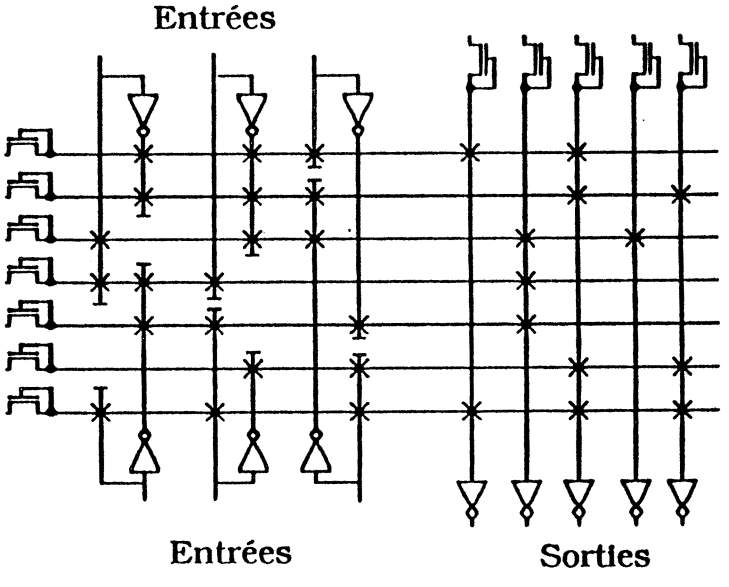


Figure 2.6: Pliage simple d'entrées.

Le pliage simple des sorties est décrit dans la figure 2.7. Dans cette technique les sorties sont placées sur deux côtés du PLA. La surface finale de la matrice OU, après l'optimisation topologique, est au maximum égale à 50%. Ce gain maximal est obtenu quand toutes les colonnes sont partagés par deux sorties. Généralement le taux de remplissage de la matrice OU est beaucoup plus faible que celui de la matrice ET. Par conséquent, l'optimisation topologique de cette matrice est normalement plus facile. Le gain de surface est aussi généralement plus important que celle de la matrice ET.

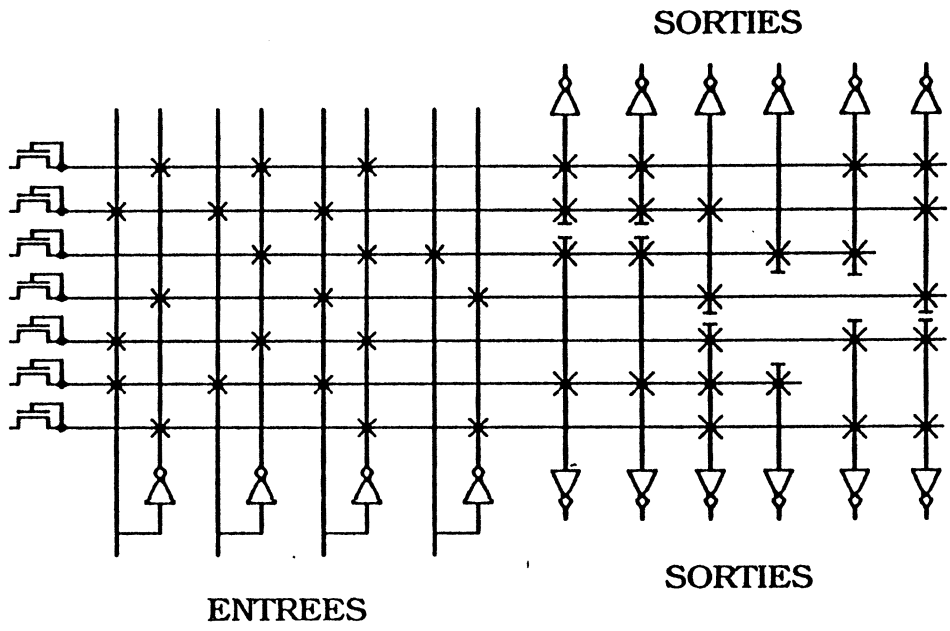


Figure 2.7: Pliage simple des sorties.

Le pliage simple des monômes présente des lignes du PLA partagées par deux monômes. Deux monômes peuvent partager une ligne si et seulement si les entrées et les sorties qui les attaquent sont différentes. Dans ce cas les sorties sont divisées en deux ensembles qui sont placés sur deux côtés distincts du PLA. Les entrées sont susceptibles d'être réordonnées. Ces contraintes limitent l'utilisation de cette technique. Par contre cette technique optimise les deux matrices. Le gain maximal est égal à 50% de la surface du PLA. La figure 2.8 décrit le pliage simple des monômes.

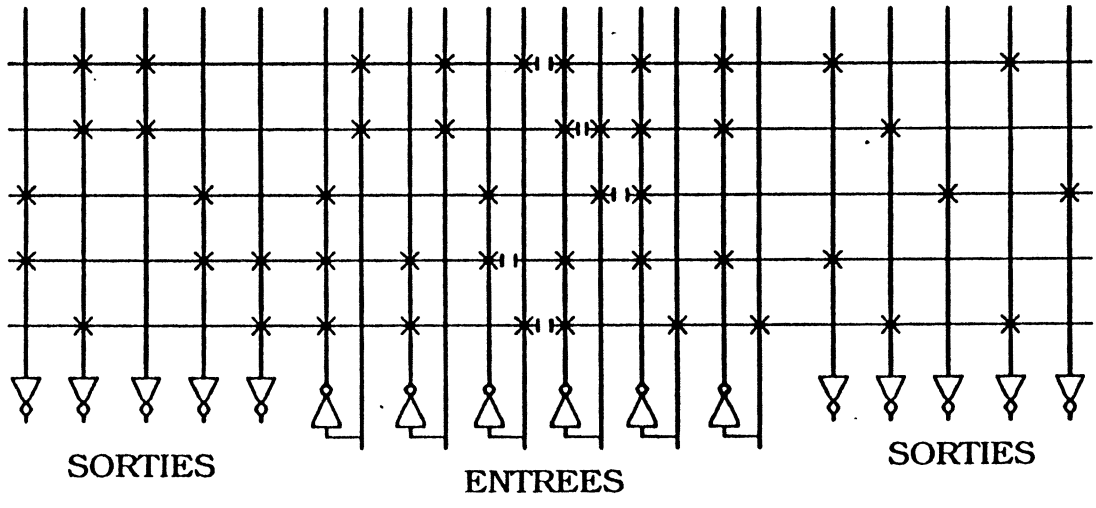


Figure 2.8: Pliage simple des monômes.

2.2.2 - Le pliage multiple.

Pour le pliage multiple, une colonne au moins est partagées entre plusieurs entrées et/ou sorties. Le pliage simple est un cas particulier du pliage multiple. Donc le gain de surface est toujours meilleur dans le cas du pliage multiple (dans le pire des cas le gain sera égal). Le gain de surface maximal est déterminé par la structure du PLA lui même. La principale difficulté pour l'application de cette technique est due au routage interne des matrices ET et OU. Chaque segment doit être relié à son connecteur placé sur le bord de la matrice. Il faut donc trouver un chemin à travers les emplacements vides à l'intérieur des matrices du PLA. Cette recherche est difficile et parfois impossible [CHU 84]. Les colonnes qui utilisent moins de monômes sont placées vers le côté externe de la matrice, où convergent toutes les connexions internes. Ceci permet de libérer la place pour le routage puisque les monômes peuvent être coupés dès que leur utilisation est finie. Dans le cas où la place libérée n'est pas suffisante une bande d'écartement doit être introduite. Cette bande rend le PLA plus haut que prévu et en conséquence l'optimisation est dégradée.

Le pliage multiple d'entrées est illustré par la figure 2.9. La particularité de cette technique est que les entrées sont généralement placées parallèlement aux monômes. Le taux de remplissage de la matrice ET doit être assez faible pour pouvoir libérer la place nécessaire aux connexions. Ceci est un facteur qui restreint l'utilisation de cette technique.

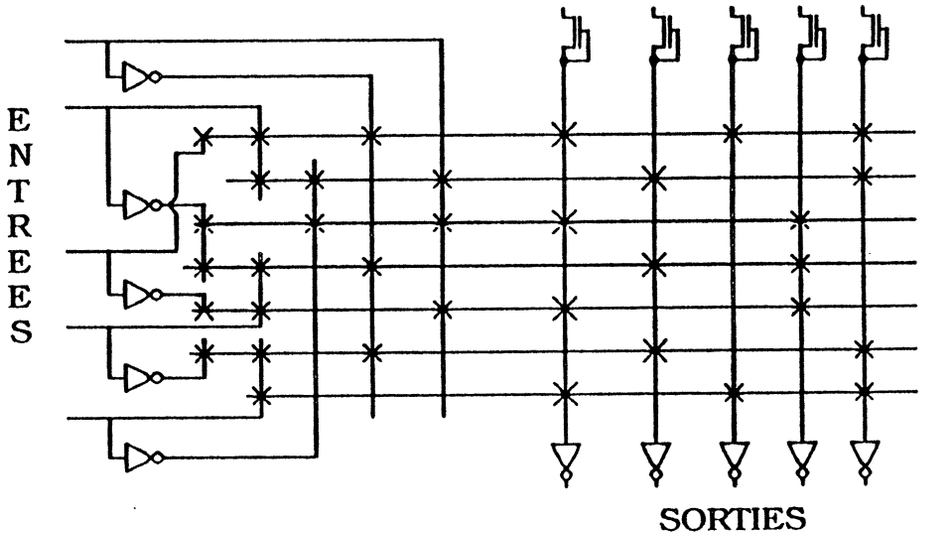


Figure 2.9: Pliage multiple des entrées.

Le pliage multiple des sorties présente les sorties parallèles aux monômes, comme le montre la figure 2.10. Cette technique s'applique aux PLAs ayant un grand nombre de monômes et de sorties où chaque sortie est générée par un petit nombre de monômes. Ceci veut dire que comme dans le cas précédent le taux de remplissage doit être faible. Généralement le taux de remplissage de la matrice OU est plus faible que celui de la matrice ET. Par conséquence l'application du pliage multiple des sorties est relativement plus facile que celle des entrées.

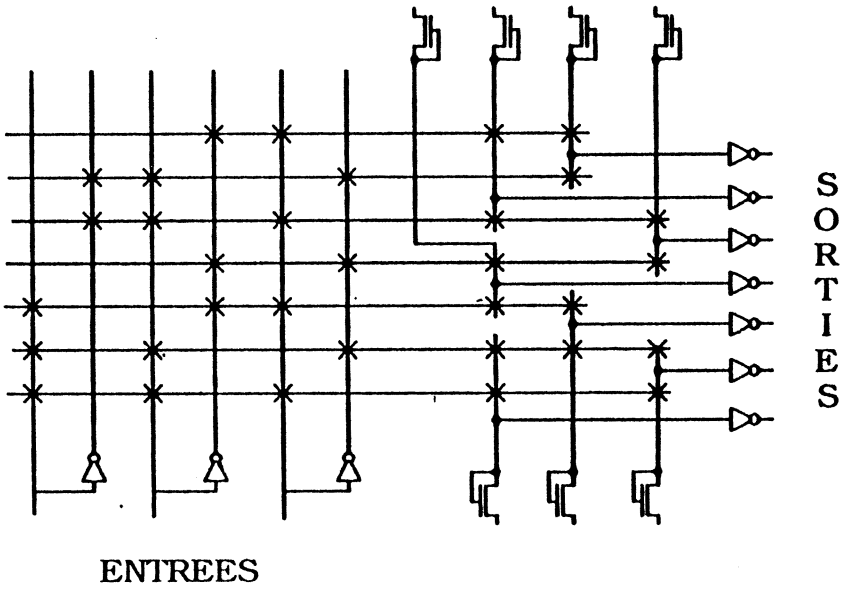


Figure 2.10: Pliage multiple des sorties.

Un autre cas du pliage multiple est le PLA monomatrice [VAR 85]. Il présente des lignes partagées par plusieurs monômes comme indiqué sur la figure 2.11. Ceci est obtenu par une duplication des entrées. Les entrées et les sorties sont placées sur les deux cotés du PLA. Cette technique donne une structure très particulière au PLA. Le gain de surface dans le PLA est faible. Par contre celui obtenu par une meilleure connexion avec les blocs voisins peut être assez important.

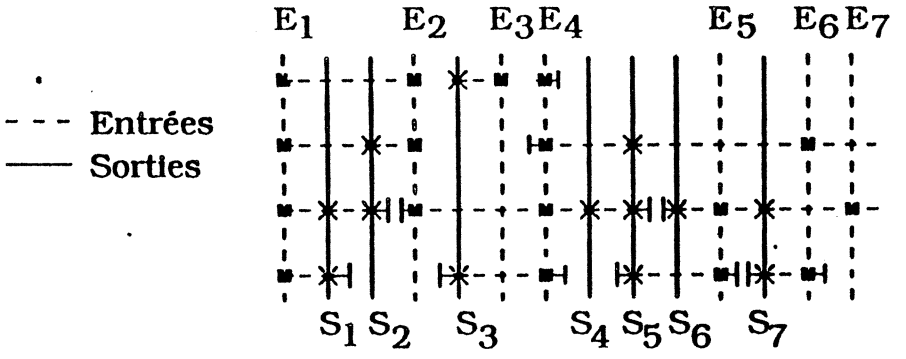


Figure 2.11: PLA monomatrice [VAR 85].

2.2.3 - La technique "PAOLA".

La technique PAOLA consiste à réduire la surface d'un PLA par un changement de sa topologie interne et par une adaptation de sa forme de manière a augmenter sa connectabilité et sa transparence [CHU 84].

La technique utilisée cherche à réorganiser la structure interne du PLA, en permutant les monômes ainsi que les entrées et les sorties. Après cette réorganisation, un compactage en rangées et en colonnes est essayé. Ainsi l'optimisation du PLA lui même est faite. D'autre part la disposition des connecteurs d'entrée et de sortie définis par le concepteur permet un câblage externe le plus simple possible.

Dans cette technique l'optimisation des PLAs se réalise en trois phases:

- Réordonnancement des monômes,
- Compactage des matrices et
- Duplication des monômes.

Les phases de réordonnancement et de compactage s'appliquent soit à la matrice ET, soit à la matrice OU. La duplication des monômes s'applique à la matrice OU mais avec des répercussions sur la matrice ET.

Le but de la première phase est de réordonner les monômes pour obtenir une distribution des transistors qui permettra un meilleur compactage. L'approche utilisée, pour réordonner les monômes, consiste a redistribuer les transistors sur une bande diagonale décrite par M et S comme le montre la figure 2.12. Ceci veut dire que les monômes et les entrées/sorties sont réordonnés de manière que leurs transistors se trouveront le plus près possible d'une bande diagonale. La diagonalisation est obtenue par la minimisation de la couverture géométrique de chaque

segment.

Un segment interne d'une entrée ou sortie est défini par le segment de colonne compris entre le premier et le dernier transistor contrôlés par cette entrée ou contrôlant cette sortie.

La couverture géométrique d'un segment interne est l'ensemble des monômes se trouvant entre le premier et le dernier transistor de l'entrée/sortie considérée, ses monômes intermédiaires étant connectés ou non à cette entrée/sortie.

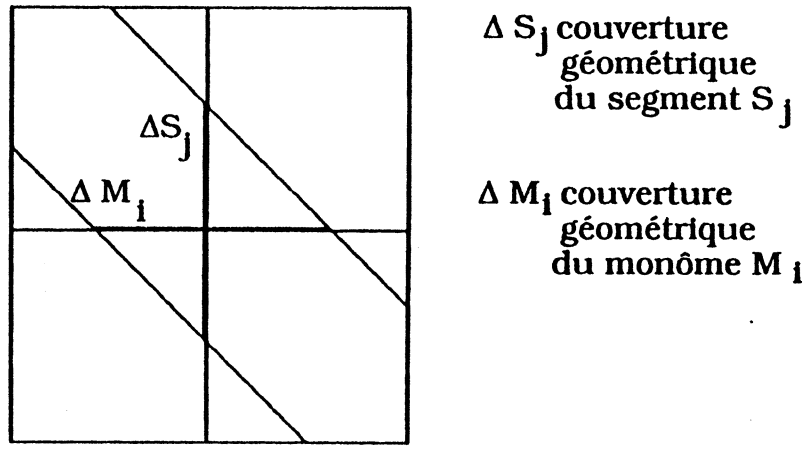


Figure 2.12: Diagonalisation de la matrice OU.

La couverture géométrique d'un monôme est l'ensemble des entrées et des sorties se trouvant entre le premier et le dernier transistor de la partie ET et OU du monôme.

On obtient la minimisation de la couverture géométrique d'un segment interne par permutation des monômes et celle d'un monôme par permutation des entrées/sorties.

Le réordonnancement des monômes peut s'appliquer soit à la matrice ET soit à la matrice OU, avec des répercussions de l'une sur l'autre bien entendu. Normalement la matrice OU est beaucoup plus creuse que la matrice ET, il est donc normal que l'ordonnancement soit fait sur cette matrice. Ceci entraîne un meilleur compactage sur la matrice OU.

C'est dans cette phase que la position des connecteurs, fixée par le concepteur, est prise en compte.

Le but du compactage des matrices est de placer deux ou plusieurs entrées/sorties sur une même colonne réalisant ainsi l'optimisation topologique des matrices.

Le regroupement des entrées/sorties sur une même colonne se fait en fonction de leur **compatibilité:** deux ou plusieurs entrées/sorties peuvent partager une même colonne si elles sont compatibles.

DEFINITION D 2.4

Deux entrées/sorties sont **compatibles** si leurs segments ne se recouvrent pas.

Il est important de remarquer que la couverture d'une entrée ou sortie est définie par l'ordre des monômes, d'où l'importance de l'ordonnancement dans l'optimisation.

Le nombre de possibilités pour regrouper les entrées/sorties en groupes de 1 à n/p éléments, sachant que le nombre de groupes peut varier entre 1 et n/p, croît de manière exponentielle avec n/p. Pour résoudre ce problème il faut utiliser des heuristiques, même pour des exemples de taille moyenne.

Une entrée/sortie peut être compatible avec une ou plusieurs entrées/sorties ou n'avoir aucune compatibilité. Le nombre de compatibilité est le nombre qui exprime la quantité d'entrées/sorties qui sont compatibles avec une entrée/sortie donnée. Dans PAOLA l'heuristique utilisée est la suivante:

Les entrées/sorties sont triées par nombre de compatibilité. A partir de l'entrée/sortie qui a le nombre le plus petit de compatiblité on choisit un ensemble de sorties compatibles deux à deux pour les regrouper dans une même colonne. Ces entrées/sorties sont éliminées du groupe de départ. Cette procédure est répétée jusqu'à ne plus avoir d'entrée/sorties.

Quand la longueur d'un PLA est plus petite que celle du bloc à commander, il est nécessaire de l'augmenter. Cette opportunité peut être utilisée pour améliorer la diagonalisation de la matrice OU par duplication des monômes comme le montre la figure 2.13. La duplication des monômes ne donne pas un gain de surface dans le PLA. Elle facilite la connexion avec les blocs voisins, ce qui represente des gains importants au niveau du routage.

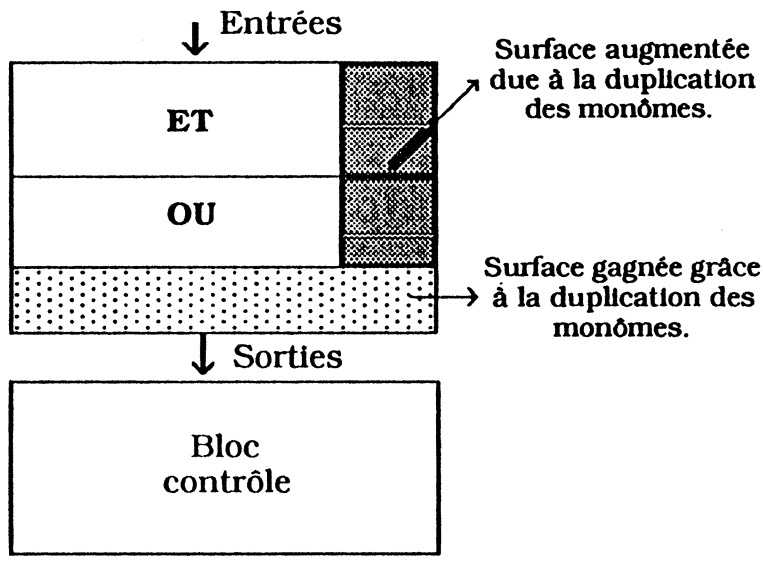


Figure 2.13: Adaptation de la structure d'un PLA par allongement.

La duplication d'un monôme, dit monôme-père, consiste à le dédoubler en deux monômes-fils distribués sur deux sorties différentes ce qui permettra un éventuel compactage. Donc les deux monômes ont les mêmes variables d'entrée dans la matrice ET. Les sorties qui était attachées à un seule monôme avant la duplication sont distribuées entre les deux monômes.

La difficulté de cette phase consiste à choisir le monôme qui, une fois dupliqué, permettra effectivement de gagner une colonne, ainsi que le point de coupure du monôme ou bien quelles sorties doivent être placées sur chaque monôme. Le choix du monôme à dupliquer est en effet une application d'une fonction heuristique qui prend en compte un des critères suivants [PER85]:

- Le nombre de sorties contrôlées,
- La couverture géométrique du monôme,
- Le facteur d'occupation,
- Le facteur d'utilisation,
- La distance barycentre moyenne,
- La taille de trou et
- Le poids des trous extrêmes de chaque sortie.

La distance barycentre moyenne est la distance entre le barycentre et le point milieu de la couverture d'un monôme. Le point milieu est défini comme le point équidistant du premier et du derniér transistor d'un monôme. Le barycentre est défini en fonction du poids donné à chaque sortie. Ce poids

est donné en fonction de l'ordre du connecteur sortie. La taille de trou d'un monôme est un ensemble de points vides contigus. Les trous extrêmes de chaque sortie prennent en compte les trous contigus au premier et au dernier transistor de la sortie.

Le point de coupure est fixé par des critères basés principalement sur le barycentre et le point milieu [PER 85]. Etant donné que les résultats sont fonction des caractéristiques des monômes et des sorties, des essais permettent de définir le meilleur critère pour chaque cas.

LE TEST HORS LIGNE DES PLAs

3 - LE TEST HORS LIGNE DES PLAs.

3.1 - GENERALITES.

La régularité des PLAs, qui favorise l'implantation d'une structure testable, présente aussi quelques particularités qui rendent difficile sa contrôlabilité. Premièrement, chaque entrée contrôle deux lignes d'entrée i.e. le changement d'une entrée implique le changement de la ligne d'entrée et de la ligne d'entrée complémentaire. Deuxièmement, chaque vecteur d'entrée sélectionne en général plusieurs monômes. Pour ces raisons la génération des vecteurs de test pour un PLA sans modifications est une tâche difficile.

Pour résoudre ces problèmes plusieurs schémas de test sont proposés dans la littérature, dont les plus importants seront décrits dans cette étude avant la presentation d'une nouvelle proposition. La clé de ces schémas est le contrôle de chaque ligne d'entrée et de chaque monôme individuellement.

Les PLAs testables peuvent être divisés en trois classes:

- Les PLAs facilement testables,
- Les PLAs fonctionnellement testables et
- Les PLAs autotestables.

La première méthode est basée sur de simples modifications dans la structure du PLA. Ces modifications permettent de contrôler soit les lignes d'entrée soit les monômes. Le contrôle des entrées ou des monômes est associé à des vecteurs d'entrée convenablement choisis pour exciter les fautes en propageant les erreurs (si possible) vers une sortie observable. Les vecteurs d'entrée ainsi que les vecteurs de sorties sont dépendants de la programmation du PLA. Les vecteurs d'entrée et les réponses de sortie sont respectivement générés et vérifiés hors circuit. La surface de la logique rajoutée est faible.

La seconde méthode est basée sur le principe qu'il existe un test universel tel que les vecteurs d'entrée et les réponses de sortie sont indépendants de la fonction que le PLA réalise. Par conséquent le coût de génération de vecteurs de test est éliminé. Les vecteurs d'entrée sont générés hors circuit et la réponse de sortie est normalement comprimée sur des bits de parité.

Dans la troisième méthode, le circuit est capable de se tester tout seul. La logique nécessaire à la génération des vecteurs d'entrée, de même que celle necessaire à la compression des vecteurs de sortie est implantée sur la même puce. Les PLAs autotestables peuvent être divisés en deux classes générales:

- Dans la première classe, les vecteurs d'entrée sont générés par des LFSRs (**L**inear **F**eedback **S**hift **R**egister) et les réponses de sortie sont comprimées par des analyseurs de signature.

- Dans la seconde classe, les entrées et les sorties sont contrôlées pour pouvoir tester à chaque cycle un seul croisement dans la matrice ET. Cette classe utilise le même principe que les PLAs fonctionnellement testables mais dans ce cas la logique nécessaire au test est implantée sur la même puce.

Par la suite, nous ferons une brève presentation des schémas de test les plus importants proposés dans la littérature. Cette description est divisée en trois classes.

3.1.1 - Les PLAs facilement testables.

KHAKBAZ [KHA 84] propose un schéma de test basé sur le contrôle individuel des monômes. Ce contrôle est obtenu par des registres à décalage. La procédure de test assigne la valeur "1" à chaque monôme sélectonné (le registre à décalage met à la masse tous les autres monômes). Le vecteur d'entrée, qui maintient ce monôme sélectionné, est appliqué. Puis à chaque cycle une entrée est changée. Le registre à décalage est maintenu dans son état tant que la dernière entrée n'est pas changée. Les monômes sont connectés à une sortie additionnelle (Z_1) qui sera à "1" si le monôme est aussi à "1" ou si le vecteur d'entrée est une "don't care condition" (i.e. il n'est pas défini pour la fonction de sortie). Sinon, la valeur de Z_1 doit changer, s'il n'y a pas de faute. La figure 3.1 décrit ce schéma.

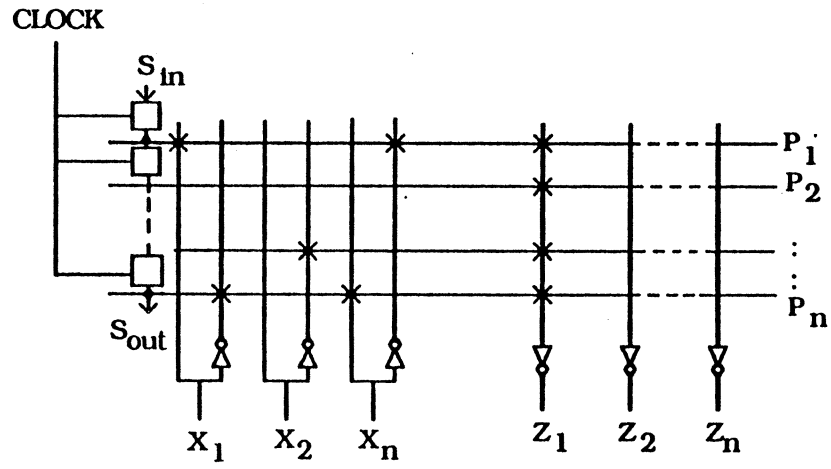


Figure 3.1: Schéma proposé par KHAKBAZ.

RAJSKI et TYSZER [RAJ84] proposent un schéma de test basé sur le contrôle de chaque ligne d'entrée. La figure 3.2 montre le schéma de test.

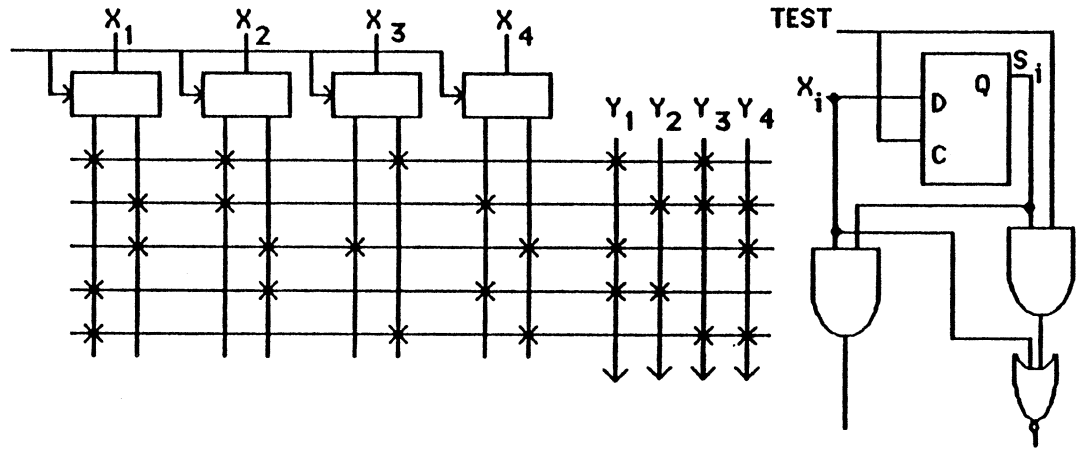


Figure 3.2: Schéma proposé par RAJSKI et TYSZER.

BOZORGUI-NESBAT et McCLUSKEY [BOZ 84] proposent un schéma de test où le contrôle des monômes est obtenu par l'addition des entrées. Ces entrées sont additionnées en vue de rendre le PLA "non concurrent" (i.e. chaque vecteur d'entrée sélectionne seulement un monôme). Dans la phase de test les entrées supplémentaires associées aux entrées normales contrôlent les monômes. Les vecteurs d'entrée sont envoyés de manière semblable à celle du schéma proposé par KHAKBAZ. Les monômes ont une distance minimale entre eux égale à deux (i.e. il existe au moins deux variables d'entrée différentes entre deux monômes). Ainsi le changement

d'une seule entrée n'est pas suffisant pour sélectionner un autre monôme. La figure 3.3 décrit ce schéma.

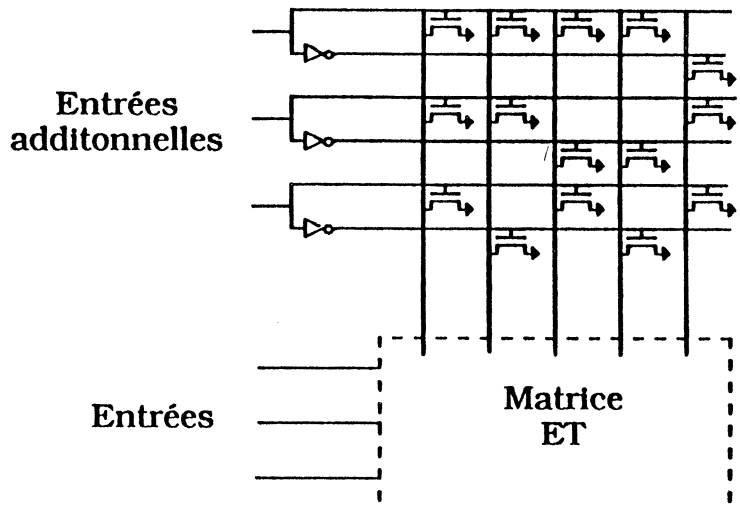


Figure 3.3: Schéma proposé par BOZORGUI-NESBAT et McCLUSKEY.

3.1.2 - Les PLAs fonctionnellement testables.

FUJIWARA et KINOSHITA [FUJ 81] proposent un schéma de test universel. Dans les schémas de test universel, les vecteurs d'entrée ainsi que les réponses de sortie sont prédéterminés indépendamment de la fonction que le PLA réalise. La logique additionnelle est composée principalement de:

- Un registre à décalage,
- Deux colonnes Y_1 et Y_2,
- Un contrôleur de parité (monômes)
- Un contrôleur de parité (sorties)
- Un monôme et
- Une sortie.

Un monôme est ajouté pour rendre impair le nombre de transistors sur chaque ligne d'entrée. La sortie additionnelle permet d'avoir un nombre impair de transistors, sur chaque monôme, dans la matrice OU. Les deux colonnes Y_1 et Y_2 permettent de sélectionner l'entrée ou l'entrée complémentaire. Ainsi, après la selection d'une ligne d'entrée ou d'une ligne d'entrée complémentaire, le registre à décalage sélectionne un monôme à chaque cycle. La parité des monômes doit être égale à la parité des sorties à

chaque cycle. La figure 3.4 décrit ce schéma.

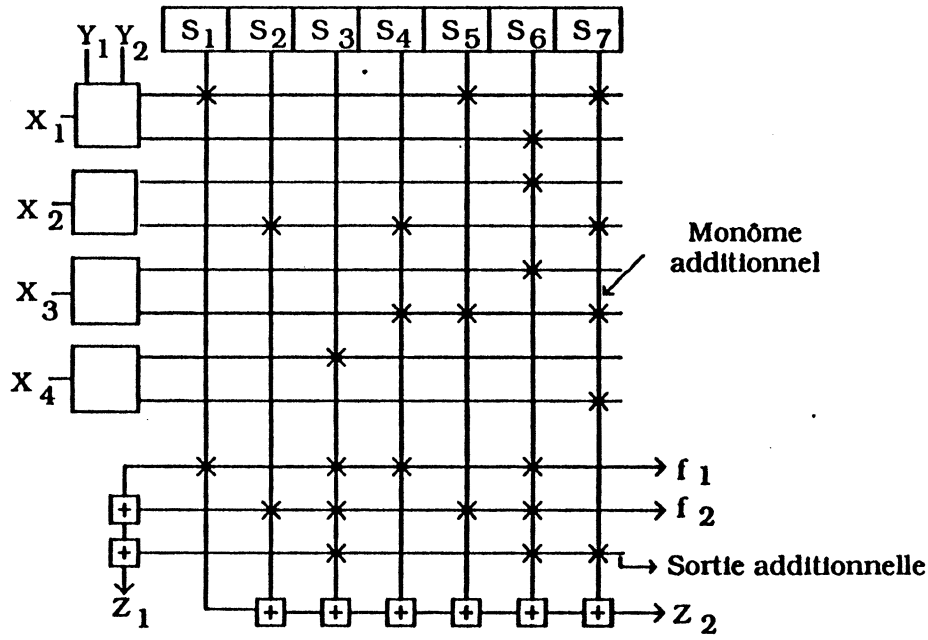


Figure 3.4: Schéma proposé par FUJIWARA et KINOSHITA.

SALUJA [SAL 83] et FUJIWARA [FUJ 84] proposent des améliorations du schéma de [FUJ 81]. Le schéma proposé par FUJIWARA est donné dans la figure 3.5.

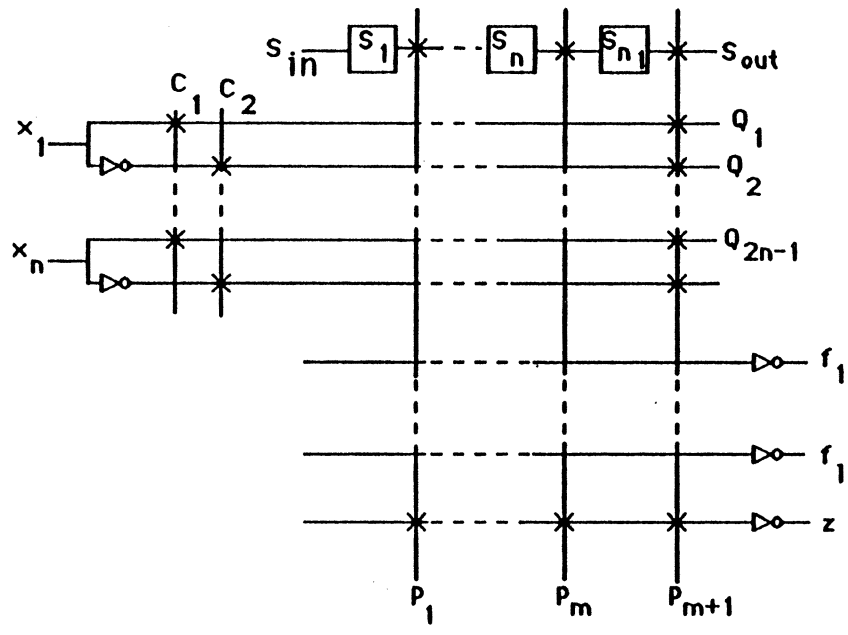


Figure 3.5: Schéma proposé par FUJIWARA.

3.1.3 - PLA autotestable (première classe).

DAEHN et MUCHA [DAE 81] proposent l'utilisation d'un registre non linéaire à rebouclage pour la génération des vecteurs d'entrée. Comme le montre la figure 3.6, les BILBOs (**Built-In Logic Block Observers**) sont capables de générer les vecteurs d'entrées et de comprimer la réponse de sortie. Chaque matrice est testée séparément et le test est divisé en trois phases. Dans la première phase le BILBO 1 génère les vecteurs d'entrée qui seront comprimés par le BILBO 2. Après la vérification de la valeur de la signature, le BILBO 2 génère les vecteurs d'entrée dont les vecteurs de sortie seront comprimés par le BILBO 3. Finalement, le BILBO 3 génère les vecteurs qui seront comprimés par le BILBO 1. La première phase permet de tester la matrice ET, la seconde, la matrice OU et la troisième les décodeurs d'entrée et de sortie. Les vecteurs d'entrée sont indépendants de la fonction que le PLA réalise par contre les vecteurs de sortie ne le sont pas. La surface de la logique ajoutée limite l'utilisation de ce schéma.

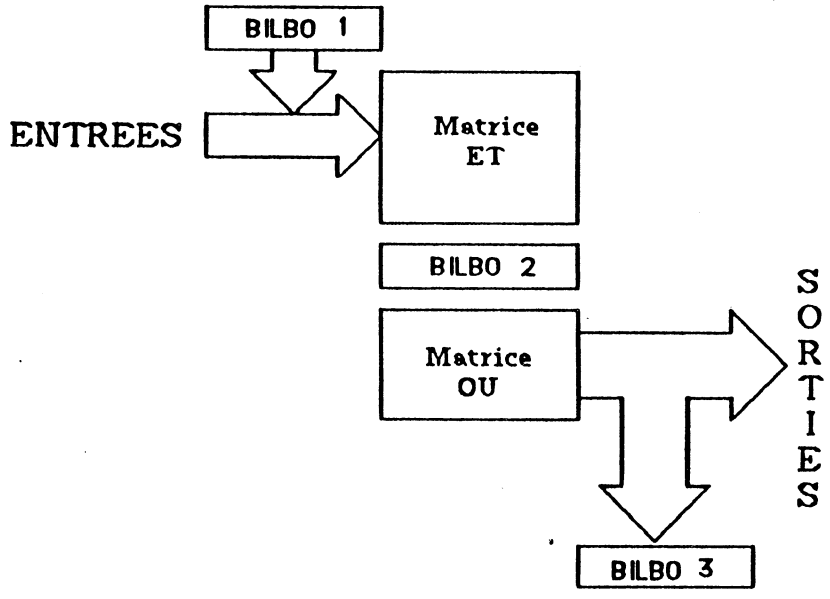


Figure 3.6: Schéma proposé par DAEHN et MUCHA.

HASSAN et McCLUSKEY [HAS 83] proposent un schéma dans lequel les LFSRs sont utilisés pour générer les vecteurs d'entrée et pour vérifier les vecteurs de sortie. Les deux LFSRs parallèles aux entrées, donnés sur la figure 3.7, compriment la valeur des entrées et des entrées complémentaires. Un autre LFSR comprime la réponse des sorties. Les trois

signatures comprimées sont comparées avec les signatures attendues pour déterminer si le PLA fonctionne correctement. Ce schéma convient aux PLAs ayant un petit nombre d'entrées puisque il est nécessaire d'appliquer 2^{n-1} vecteurs de test. Le PLA doit donc avoir un petit nombre entrées pour que le test soit appliqué dans un temps raisonnable.

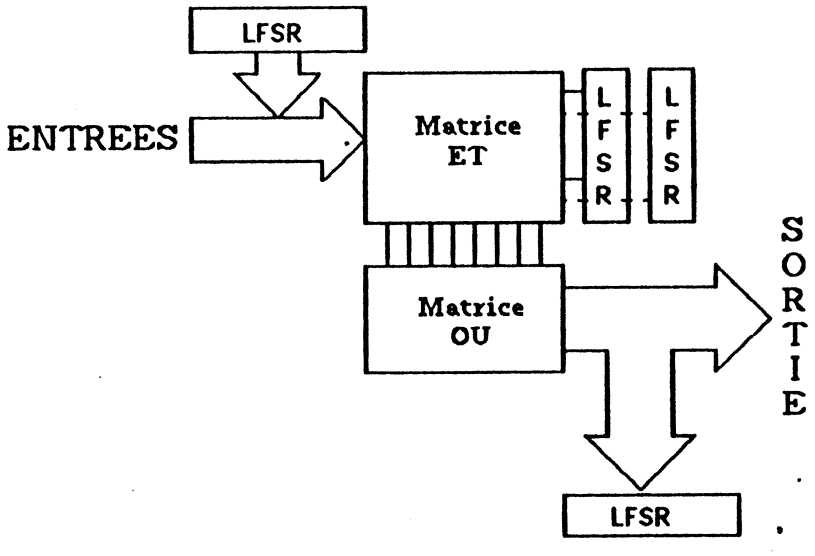


Figure 3.7: Schéma proposé par HASSAN et McCLUSKEY.

3.1.4 - Le PLA autotestable (seconde classe).

YAJIMA et ARAMAKI [YAJ 82] proposent la structure donnée dans la figure 3.8. La logique additionnelle est composée de quatre monômes, deux sorties, deux contrôleurs de parité (monômes et sorties), un décodeur d'entrée avec un registre à décalage et un contrôle des parités (feedback value generator). Les deux valeurs de parité sont utilisées par le "feedback value generator" pour la génération du vecteur d'entrée suivant. Après l'application de n + 2m + 8 vecteurs d'entrée, la réponse de sortie est comprimée dans un régistre à décalage composé par m + 4 cellules.

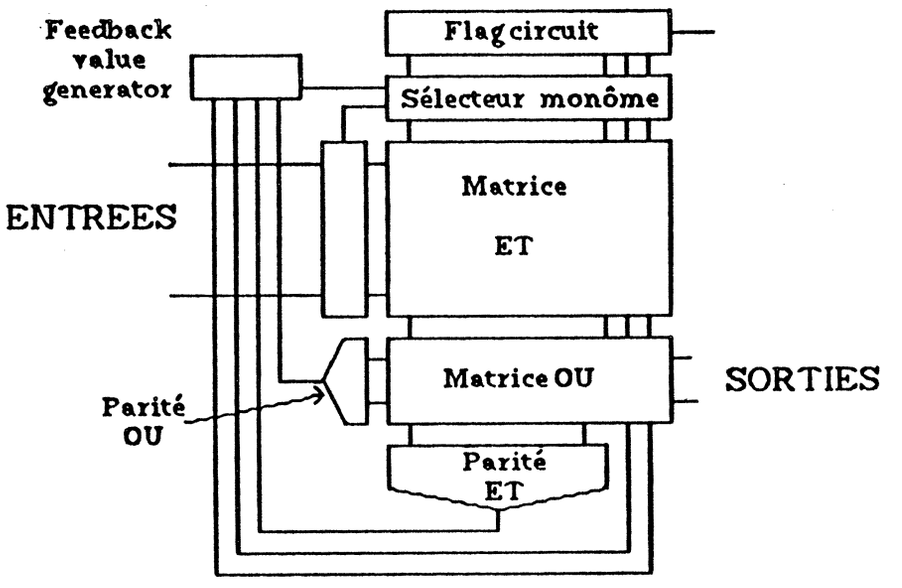


Figure 3.8: Schéma proposé par YAJIMA et ARAMAKI.

TREUER, FUJIWARA et AGARWAL [TRE 85] proposent un schéma où le contrôle des monômes est obtenu par des registres à décalage (m/2). Un autre registre à décalage est utilisé pour générer les vecteurs d'entrée. Deux lignes additionnelles contrôlent le test des lignes d'entrée ou des lignes d'entrée complémentaire. Un multiplexeur permet de sélectionner les entrées de test pendant la phase de test, ou les entrées normales au cours du fonctionnement normal du circuit. Les vecteurs de sorties sont comprimés sur des bits de parité. Cette compression est obtenue par l'addition des monômes et une sortie. Les monômes permettent d'avoir un nombre impair de croisements avec et sans transistors pour chaque ligne d'entrée dans la matrice ET. La sortie permet d'avoir un nombre impair de croisements avec transistors dans la matrice OU. Donc après l'application de 2nm + 1 vecteurs d'entrée la parité cumulative doit être impaire. Ce schéma est donné dans la figure 3.9.

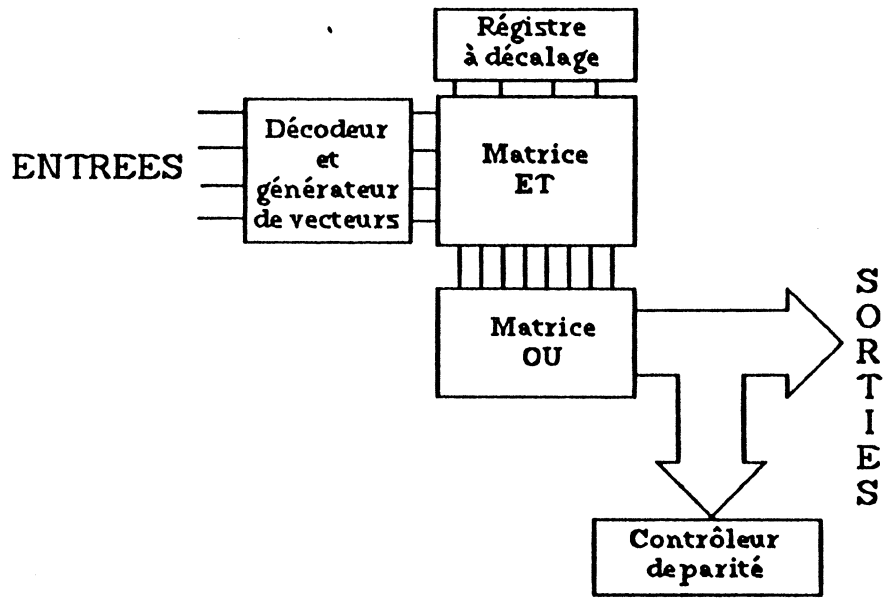


Figure 3.9: Schéma proposé par TREUER, FUJIWARA et AGARWAL.

SALICK, MERCER et UNDERWOOD [SAL 85] proposent un schéma qui utilise le même principe que celui employé par BOZORGUI-NESBAT [BOZ 85] pour le contrôle des monômes. Dans ce cas les entrées additionnelles ne sont pas minimisées, donc nous avons un décodeur composé de $\log_2 m$ entrées. Ce décodeur est associé à un LFSR qui permet de sélectionner un seul monôme à la fois. Les vecteurs d'entrée sont générés par un décodeur associé à un registre à décalage composé de $\log_2 n$ bits. Le test des entrées et entrées complémentaires est obtenu par des portes EXNOR associées à deux colonnes. Les deux colonnes permettent de sélectionner les entrées ou entrées complémentaires et les portes EXNOR permettent d'alterner la valeur appliquée aux entrées. Ainsi quand une entrée est testée, une colonne met à la masse toutes les entrées complémentaires et une sortie des portes EXNOR sera à "1". Quand une entrée complémentaire est testée, l'autre colonne met à la masse toutes les entrées et une sortie des portes EXNOR sera à "0". La figure 3.10 décrit ce schéma.

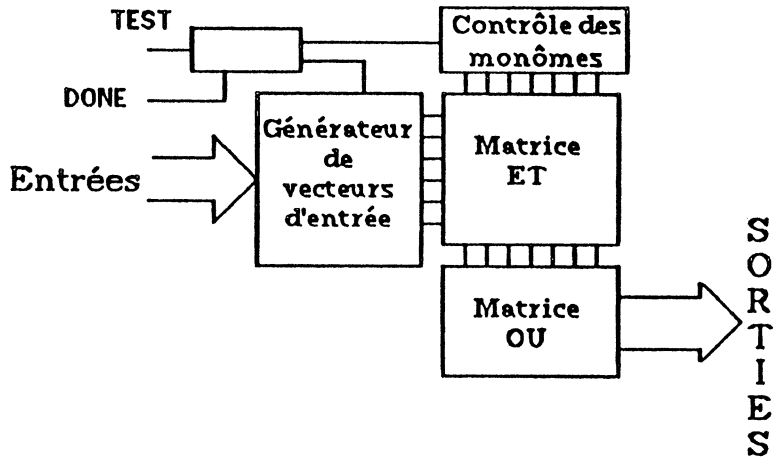


Figure 3.10: Schéma proposé par SALICK, MERCER et UNDERWOOD.

HUA, JOU et ABRAHAM [HUA 84] proposent un schéma qui utilise deux contrôleurs de parité (monômes et sorties). Les entrées et les monômes sont contrôlés de manière semblable à celle utilisée par [FUJ 81]. La figure 3.11 décrit ce schéma.

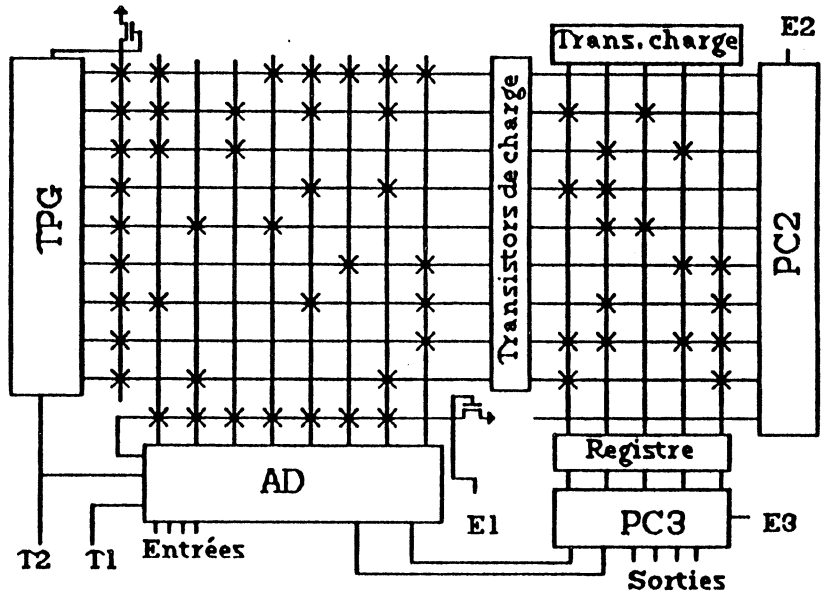


Figure 3.11: Schéma proposé par HUA, JOU et ABRAHAM.

3.2 - SCHEMA DE TEST PROPOSE.

La conception des circuits autotestables nécessite une attention particulière vis-à-vis des problèmes suivants:

- Génération des vecteurs d'entrée,
- Application des vecteurs,
- Compression des vecteurs de sorties et
- Couverture de fautes.

Les vecteurs d'entrée doivent être simples de manière à ce que de simples modifications ou augmentations du circuit original soient capables de les générer pendant la phase de test.

Le nombre de vecteurs de test qui seront appliqués au circuit doit être le plus petit possible afin que le test s'effectue en un temps raisonnable.

Les réponses de sortie doivent être comprimées dans un petit nombre de bits et de manière que la surface additionnelle soit faible.

Ces caractéristiques doivent amener à une bonne couverture de fautes.

Dans ce chapitre nous proposons un schéma qui utilise le même principe que les autres schémas décrits dans la littérature. Il est compatible avec la structure des PLAs optimisés topologiquement. En effet le schéma proposé utilise le meilleur parti des autres schémas présentés dans la littérature qui ne sont pas suffisants pour s'appliquer aux PLAs optimisés. Nous lui avons apporté des modifications pour améliorer soit la surface additionnelle soit la couverture de fautes. Ainsi nous pouvons dire que le schéma proposé est une amélioration vis-à-vis des autres schémas publiés.

Le schéma de test est décrit par la figure 3.12. La logique additionnelle est composée de:

- Un générateur de vecteurs d'entrée (GVE),
- Un contrôleur des monômes,
- Un compresseur des vecteurs de sortie et
- Une logique de contrôle.

Ces parties sont détaillées par la suite.

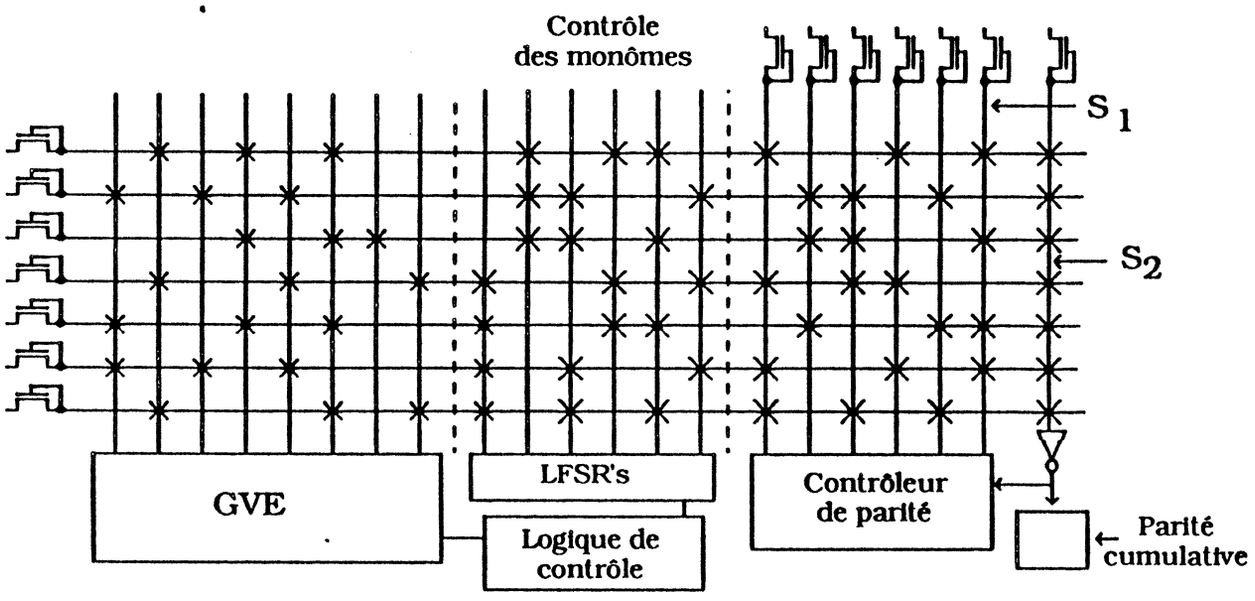


Figure 3.12: Schéma proposé.

3.2.1 - Générateur de vecteurs d'entrée.

La figure 3.13 montre la logique nécessaire pour la génération des vecteurs d'entrée. Elle est composée de deux colonnes (C et $\overline{C}$), un registre à décalage avec n cellules, n portes OU-exclusif et un multiplexeur.

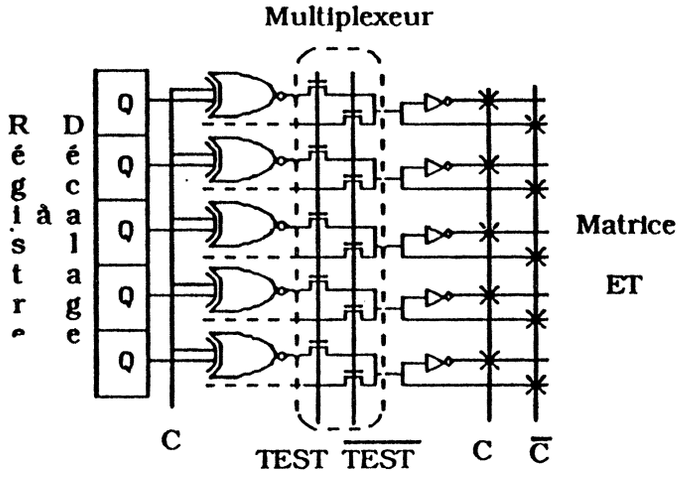


Figure 3.13: Générateur de vecteurs d'entrées.

La logique pour la génération des vecteurs d'entrée doit permettre d'avoir à chaque cycle une seule ligne d'entrée selectionnée (i.e. la ligne d'entrée selectionnée est mise à "1" et toutes les autres sont mises à "0"). Les entrées doivent donc être contrôlées après les inverseurs des entrées.

Pour cela il faut rajouter sur la logique additionnelle deux colonnes qui permettent d'amener à la masse les entrées, quand on teste une entrée complémentaire et les entrées complémentaires, quand on teste une entrée. La figure 3.14 montre le détail de cette partie.

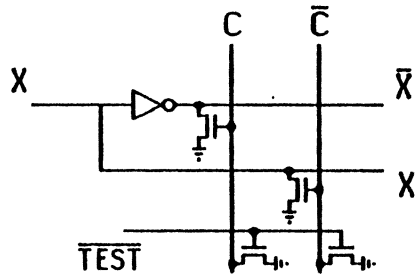


Figure 3.14: Contrôle des entrées.

La logique doit permettre de tester une entrée puis son complément avant de traiter l'entrée suivante. Ceci peut être obtenue par des portes EXNOR commandées par le même signal qui amène à la masse l'entrée ou son complément. Ainsi le registre à décalage a une entrée à "1" et toutes les autres à "0". Si C est égal à "1", les lignes d'entrée complémentaire sont mises à la masse par la colonne C et à la sortie des portes EXNOR nous avons une entrée à "1" et toutes les autres à "0". Quand C passe à "0", les lignes d'entrée sont mises à la masse et à la sortie des portes EXNOR nous avons une entrée à "0" et toutes les autres à "1". La valeur C est générée par une bascule T de la logique de contrôle.

La surface de cette logique est donnée par l'équation suivante (voir annexe 1):

$$S_{GVE} = [1456n + 720n]\ \lambda^2 \qquad \text{(eq. 3.1)}$$

Ce schéma est semblable à celui proposé par SALICK, MERCER et UNDERWOOD [SAL 85]. Le décodeur placé avant les portes EXNOR, est remplacé par le registre à décalage ce qui représente un gain de surface ($S_{GVE} = [1456n + (720 + 256n)*\lceil \log_2(n) \rceil]\lambda^2$).

3.2.2 - Contrôle des monômes.

La logique de contrôle des monômes est plus simple que celle des entrées car le problème posé pour les entrées complémentaires n'existe pas. La logique doit permettre de tester un seul monôme à la fois. Il s'agit donc de rendre le PLA non concurrent.

L'utilisation d'un décodeur comme celui proposé par SALICK, MERCER et UNDERWOOD [SAL 85] est la meilleure option pour contrôler les monômes. Il réduit la surface de la logique complémentaire et élimine la nécessité d'avoir une logique placée sur les monômes. En effet la configuration de la logique complémentaire ressemble à celle du PLA. Pour cette raison le décodeur est plus facile à placer.

La surface du décodeur décrit par la figure 3.15, est donnée par l'équation suivante où la surface des portes EXOR du LFSR n'est pas prise en compte:

$$S_{CM} = (128m + 720) \log_2 m \ \lambda^2 \quad \text{(eq. 3.2)}$$

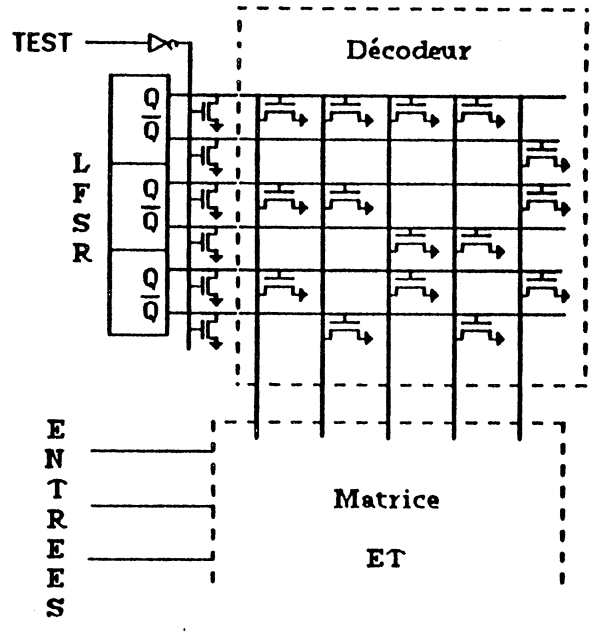


Figure 3.15: Décodeur de contrôle des monômes.

Chaque vecteur généré par le LFSR correspond à un monôme. Ainsi le LFSR doit être capable de générer m vecteurs, il doit donc être composé de $\lceil \log_2(m+1) \rceil$ bits. Après la sélection d'un monôme toutes les lignes d'entrée

sont testées. Après avoir testé la dernière entrée un autre monôme est sélectionné. Pendant le fonctionnement normal du circuit le décodeur est mis à la masse.

3.2.3 - Compression des vecteurs de sortie.

La figure 3.16 montre les colonnes et le contrôleur de sortie ajoutés au PLA en vue de la compression des vecteurs de sortie.

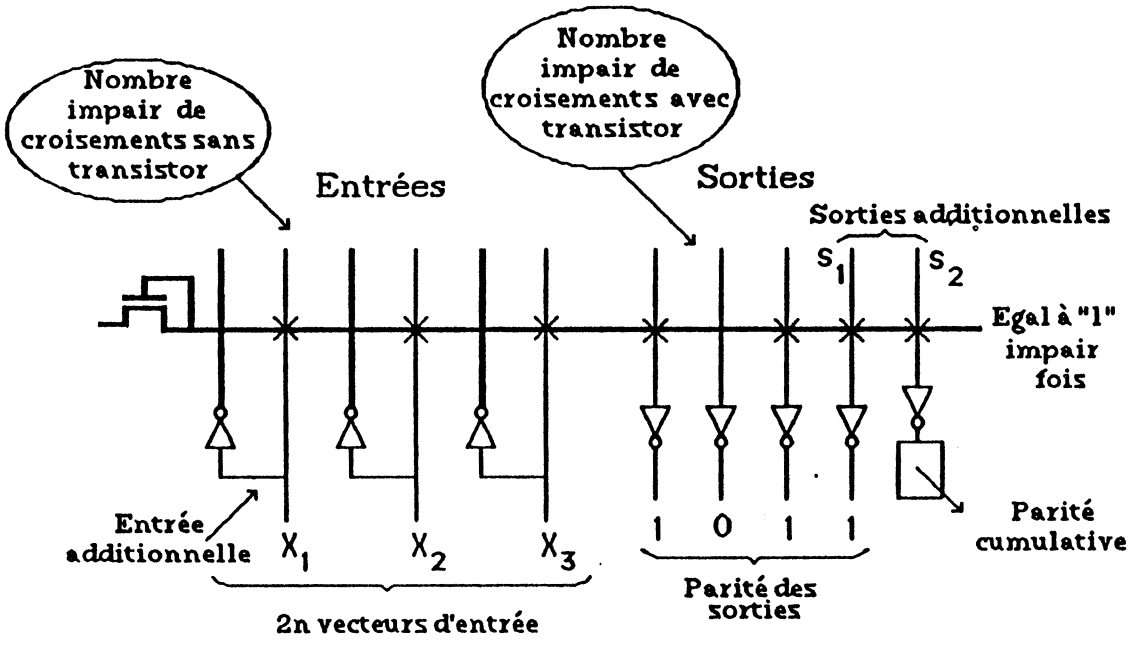


Figure 3.16: Compression des vecteurs de sortie.

Propriété 1:

Il existe un nombre impair de croisements sans transistor sur chaque monôme dans la matrice ET.

Cette propriété est assurée par une entrée additionnelle. Quand il existe un nombre pair de croisements sans transistor, sur le monôme original, un transistor sera placé à l'intersection du monôme et de l'entrée additionnelle.

Propriété 2:

Il existe un nombre impair de croisements avec transistor sur chaque monôme dans la matrice OU.

Cette propriété est assurée par une sortie additionnelle (S_1). Quand il existe

un nombre pair de croisements avec transistor, sur le monôme original, un transistor sera placé à l'intersection du monôme et de la sortie S_1.

La sortie additionnelle S_2 permet de connaître la valeur du monôme et par conséquent la valeur de la parité de sortie.

Le contrôle des entrées et des monômes dans la matrice ET permet de tester un croisement à chaque cycle. Le décodeur de contrôle des monômes sélectionne un monôme (ceci veut dire que tous les monômes sont mis à la masse excepté celui sélectionné). La valeur du monôme sélectionné va dépendre de l'existence ou non d'un transistor au croisement formé par la ligne d'entrée, qui est sélectionnée à chaque cycle, et le monôme sélectionné. Les vecteurs d'entrée montrés dans la figure 3.17, sont appliqués au PLA pour chaque monôme sélectionné.

X_1	X_2	.....	X_{n-1}	X_n	C	$\bar{C}$	
1	0	.....	0	0	1	0	
0	1	.....	1	1	0	1	
0	1	.....	0	0	1	0	Chaque
1	0	.....	1	1	0	1	
0	0	.....	1	0	1	0	monôme
1	1	.....	0	1	0	1	
0	0	.....	0	1	1	0	
1	1	.....	1	0	0	1	

Figure 3.17: Vecteurs d'entrée.

Quand il existe un transistor au croisement sélectionné, tous les monômes sont mis à la masse, donc le vecteur de sortie est composé de "0" et la parité est paire (voir la figure 3.18).

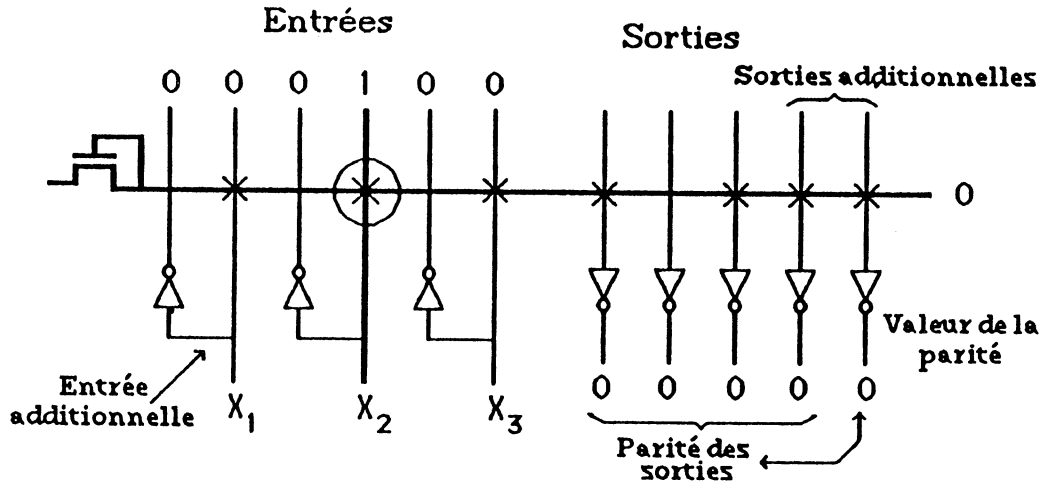


Figure 3.18: Compression du vecteur de sortie.

Quand il n'y a pas de transistor au croisement sélectionné, le monôme reste à "1", donc toutes les sorties en relation avec ce monôme sont aussi à "1". Puisque le nombre de sorties en relation avec chaque monôme est impair (propriété 2), la parité du vecteur de sortie est aussi impaire (voir figure 3.19).

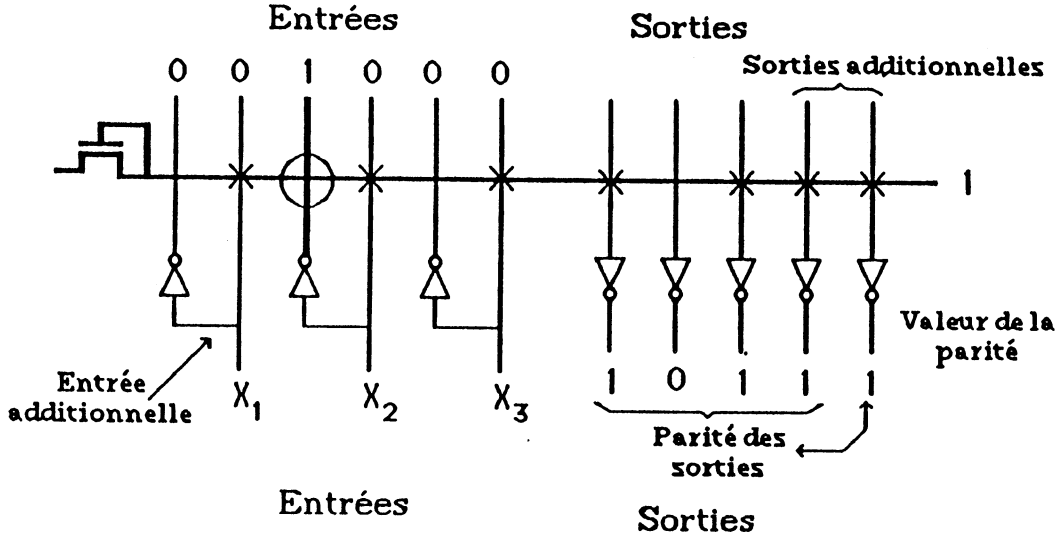


Figure 3.19: Compression du vecteur de sortie.

Chaque monôme est composé d'un nombre impair de croisements sans transistor dans la matrice ET (propriété 1). Par conséquent le monôme sera à "1" un nombre impair de fois (voir figure 3.16). Puisque quand le monôme est à "1" la parité de sortie est aussi à "1" (propriété 2) et que cela arrive un nombre impair de fois (propriété 1), la parité cumulative après 2n vecteurs d'entrée sera impaire. Cette procédure est répétée pour chaque monôme.

La figure 3.20 montre le contrôleur de parité dont la surface est donnée par l'équation suivante (voir annexe 2):

$$S_{CP} = 760p\ \lambda^2 \text{ (eq. 3.3)}$$

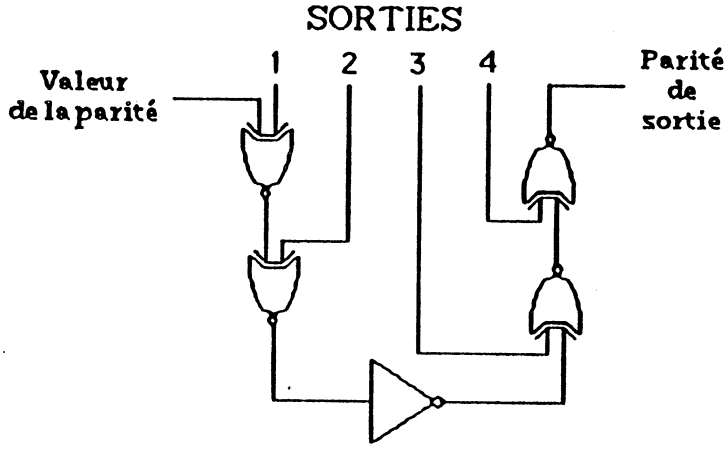


Figure 3.20: Contrôleur de parité.

3.2.4 - Logique de contrôle.

La logique de contrôle est chargée de la gestion du test. Elle doit être capable de signaler la fin de chaque cycle de monôme (i.e. quand la dernière entrée a été testée), la fin du test (i.e. quand la dernière entrée du dernier monôme a été testée) et aussi de générer le signal commandant l'alternance du test des entrées. La figure 3.21 montre la logique de contrôle dont la surface n'est pas prise en compte.

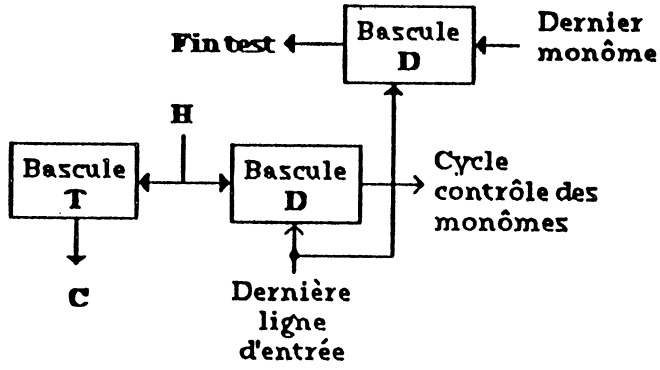


Figure 3.21: Logique de contrôle.

3.3 - COMPARAISON DES DIFFERENTES METHODES.

La comparaison des méthodes de test est basée sur quatre critères:

- Dégradation des signaux,
- Longueur du test,
- Surface additionnelle et
- Couverture des fautes.

Le premier paramètre montre la dégradation des signaux d'entrée introduite par la logique additionnelle, pendant le fonctionnement normal du circuit. Le tableau 3.1 donne une comparaison entre dégradations des circuits, exprimée en retard de portes et transistors de passage. Nous remarquons que la méthode proposée par TREUER, FUJIWARA et AGARWAL [TRE 85] présente le retard le plus grand.

METHODE	PORTES	TRANSITOR PASSAGE
SCHEMA PROPOSE	0	1
SALICK	0	1
HASSAN	0	1
DAEHN	0	1
TREUER	1	3
HUA	1	3

Tableau 3.1: Retard.

Le tableau 3.2 donne la longueur des vecteurs de test utilisés dans les différentes méthodes. La méthode proposée par HASSAN et McCLUSKEY [HAS 83] utilise 2^n vecteurs d'entrée, ce qui limite son utilisation à des PLAs ayant un petit nombre d'entrées.

METHODE	VECTEURS DE TEST
SCHEMA PROPOSE	2nm
SALICK	2nm
HASSAN	2^n
DAEHN	2n + m
TREUER	2nm +2m
HUA	2n + m

Tableau 3.2: Longueur des vecteurs de test.

Les surfaces de logique additionnelle sont comparées dans le tableau 3.3. Les valeurs sont données en λ^2 et sont basées sur les règles de dessin proposées par MEAD et CONWAY [MEA 80]. Le tableau 3.3 présente uniquement les propositions de TREUER, FUJIWARA et AGARWAL [TRE 85] et SALICK, MERCER et UNDERWOOD [SAL 85] car dans les autres méthodes la surface n'est pas détaillée.

n	m	p	TREUER	SCHEMA PROPOSE
50	190	67	366400	309120
30	120	27	228000	177840
27	181	54	304480	249856
54	134	61	298880	260480
30	153	37	272880	227712
24	44	13	111680	90336
12	58	21	104800	74976
25	42	12	111120	90976
60	200	60	401600	341120

Tableau 3.3: Surface additionnelle.

Une comparaison des couvertures de fautes est donnée dans le tableau 3.4. La méthode proposée par SALICK, MERCER et UNDERWOOD [SAL 85] n'est pas détaillée car il s'agit d'un générateur de vecteurs de test et la

couverture de faute dépend de la méthode utilisée pour la compression des vecteurs de sortie.

Le théorème et les lemmes traitant de la couverture de fautes sont donnés dans l'annexe 3.

DAEHN	Toutes les fautes simples et mutiples dues à un collage, "bridging" ou "crosspoint".
TREUER	Toutes les fautes simples et presque toutes les fautes multiples dues à un collage, "bridging" ou "crosspoint".
HASSAN	Toutes les fautes simples et mutiples dues à un collage.
HUA	Toutes les fautes simples et mutiples dues à un collage, "bridging" ou "crosspoint".
SCHEMA PROPOSE	Toutes les fautes simples et presque toutes les fautes multiples dues à un collage "bridging" ou "crosspoint".

Tableau 3.4: Couverture de fautes.

Nous pouvons conclure que le schéma proposé est une amélioration vis-à-vis des schémas existants. La dégradation des signaux d'entrée pendant le fonctionnement normal du circuit est due à un seul transistor de passage. La longueur de test ne varie pas exponentiellement avec le nombre d'entrées. Il n'existe pas de contrainte pour l'application du schéma proposé dûe à la longueur de test. La surface ajoutée est la plus petite. Le schéma proposé offre une très bonne couverture de faute.

3.4 - APPLICATION AUX PLAs OPTIMISES TOPOLOGIQUEMENT.

Dans ce paragraphe nous examinons l'application du schéma de test proposé dans les PLAs optimisés topologiquement. La principale difficulté de l'application des méthodes de test dans les PLAs optimisés est due à la structure du PLA après le compactage. Par exemple, dans le cas où les entrées ou sorties sont placées sur deux côtés différents, il est necessaire de placer la logique sur les deux côtés et de ramener les signaux de contrôle. D'autres problèmes plus complexes peuvent être posés comme par exemple dans le cas du pliage multiple. Dans ce cas les entrées et les sorties sont

parallèles aux monômes et les monômes sont difficilement accessibles. L'utilisation des registres à décalage n'est pas possible dans ce cas.

La surface totale (PLA + logique additionnelle) après l'optimisation doit être inférieure à la surface avant optimisation. Cette comparaison est donnée par la suite mais elle ne prend pas en compte la surface qu'on pourrait obtenir avec une meilleure connexion dans l'optimisation topologique (cela dépend de chaque application).

3.4.1 - Le pliage simple d'entrées.

L'optimisation de la matrice ET est normalement plus difficile que celle de la matrice OU car le taux de remplissage de cette matrice est plus important que celui de la matrice OU. Cela explique le faible gain de surface dans la matrice ET. La particularité de cette technique est que les entrées sont placées sur les deux côtés du PLA. Le générateur de vecteurs d'entrée doit aussi être placé sur les deux côtés. La figure 3.22 montre un PLA avec le pliage simple des entrées et la logique complémentaire ajoutée.

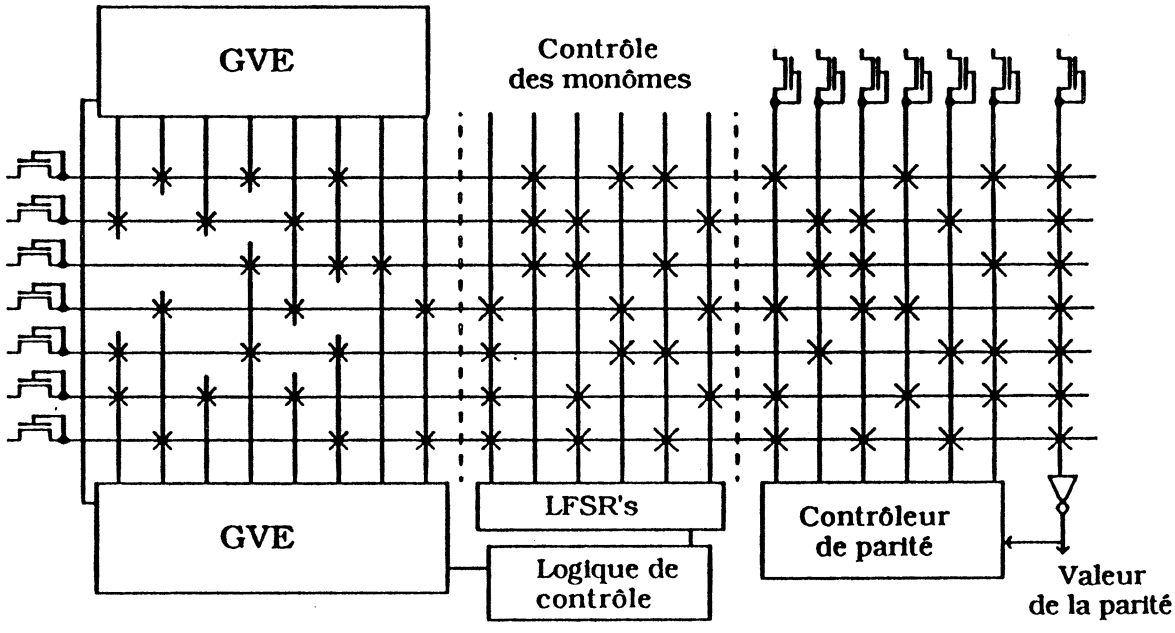


Figure 3.22: Plan d'un PLA autotestable à pliage simple des entrées.

En effet la surface de la matrice ET peut être exprimée par:

$$S_{ET} = 16n * 8m \ \lambda^2$$

La surface d'un générateur de vecteurs d'entrée est (voir equation 3.1):

$$S_{GVE} = (1456n + 720n)\ \lambda^2$$

La surface totale de la matrice ET avec le générateur de vecteurs d'entrée sans optimisation topologique est donc:

$$S_{SO} = (16n * 8m + 2176n)\ \lambda^2 \quad \text{(eq. 3.4)}$$

La surface de la matrice ET après l'optimisation topologique devient:

$$S_{AO} = [(16n * 8m)\ (1 - G)]\ \lambda^2$$

Où G = 1 - (surface après optimisation / surface initiale)

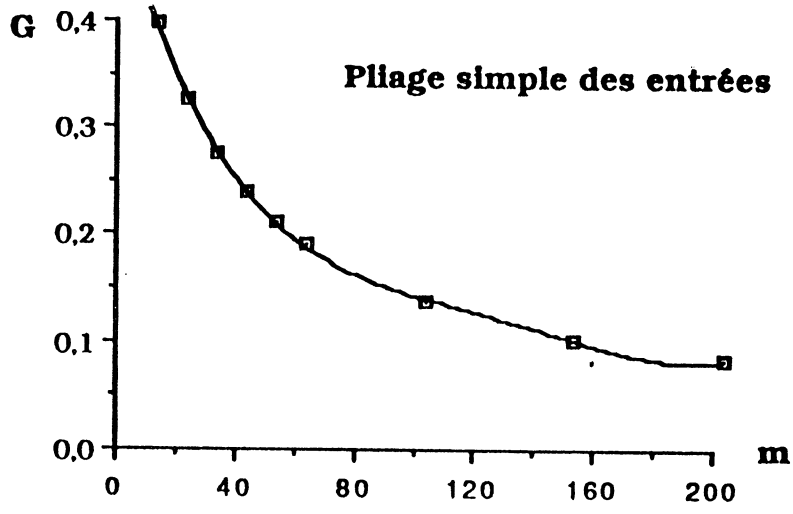
La surface des générateurs de vecteurs d'entrée sera:

$$S_{GVE} = 2 * 2176n * (1 - G)\ \lambda^2$$

La surface totale après l'optimisation sera donc:

$$S_{TAO} = [(16n * 8m)\ (1 - G) + (2 * 2176n) * (1 - G)]\ \lambda^2 \quad \text{(eq. 3.5)}$$

La surface totale (PLA + logique additionnelle) après l'optimisation topologique doit être inférieure à la surface totale avant l'optimisation topologique (i.e. $S_{SO} \geq S_{TAO}$). Ainsi, pour un gain de surface de 20% le PLA doit avoir au moins 50 monômes. Pour un gain de surface plus faible, par exemple 10%, le PLA doit comporter plus de 136 monômes. Nous pouvons donc conclure que le schéma de test est facilement applicable dans les PLAs de taille moyenne. Pour un faible nombre de monômes, il est préférable de ne pas optimiser le PLA sauf quand le gain en surface est important (voir graphique 3.1). Il faut aussi remarquer que le gain de surface maximal dans ce type d'optimisation est 50% [$G_{(max)} = 0,5$].



Graphique 3.1: Gain de surface en fonction du nombre de monômes.

Dans ce type d'optimisation les entrées ne peuvent pas être contrôlées après le croisement de la matrice ET. Les méthodes qui utilisent la première approche (HASSAN et McCLUSKEY [HAS 83] et DAEHN et MUCHA [DAE 81]) ne peuvent donc pas être employées dans ce cas.

3.4.2 - Le pliage simple des sorties.

Quand les sorties sont placées sur deux côtés du PLA, le contrôleur de parité doit être divisé en deux. La figure 3.23 montre la structure du PLA avec sa logique complémentaire dans ce cas.

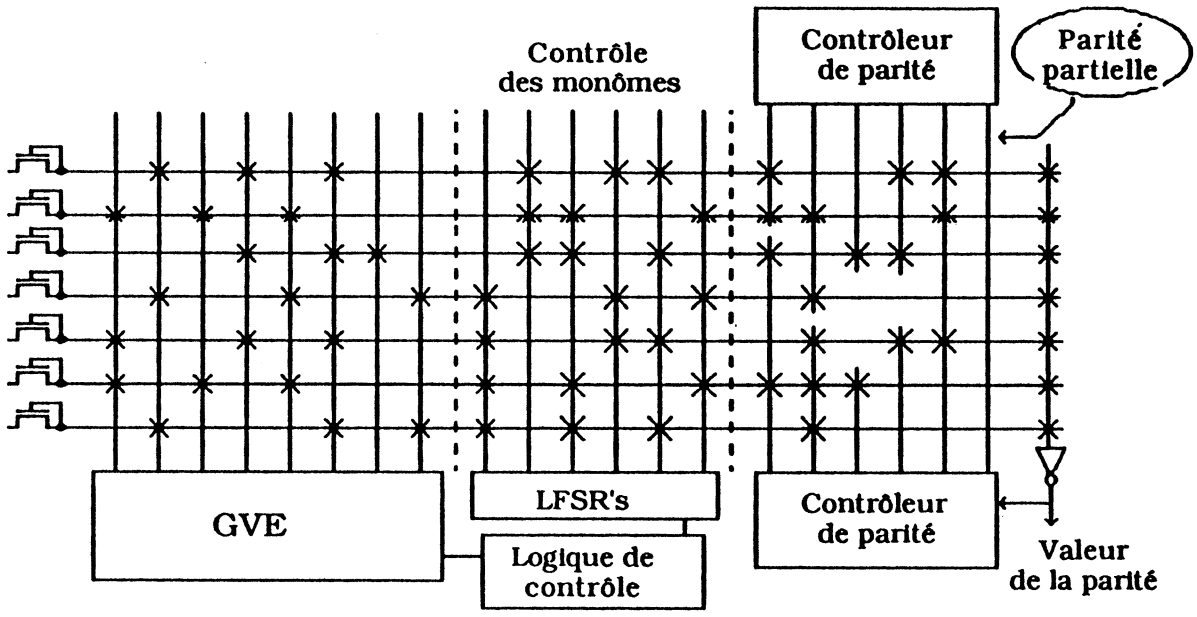


Figure 3.23: Plan d'un PLA autotestable à pliage simple des sorties.

Le gain de surface dans la matrice OU après l'optimisation topologique est donné par l'équation suivante:

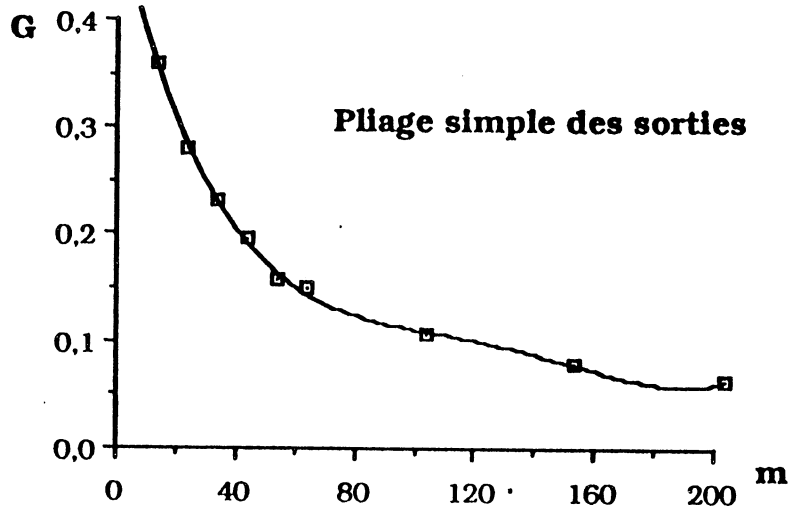
$$S_{GAO} = 8m * 8p * G \quad \lambda^2 \quad \text{(eq. 3.6)}$$

L'augmentation de la logique de test due à l'optimisation topologique est donnée par l'équation suivante:

$$S_{AT} = [2 * 95 * 8p * (1 - G) - 95 * 8p] \quad \lambda^2 \quad \text{(eq. 3.7)}$$

Généralement le taux de remplissage de la matrice OU est beaucoup plus faible que celui de la matrice ET. Par conséquent, l'optimisation topologique de cette matrice est normalement plus facile et apporte un gain de surface plus important que celle de la matrice ET. La surface additionnelle est aussi plus faible que dans la matrice ET, en effet pour un même nombre d'entrées et sorties le contrôleur de parité prend une surface plus petite. Donc l'application du schéma de test dans la matrice OU est normalement plus facile que dans la matrice ET. Néanmoins, comme dans le cas précédent, un PLA optimisé avec un gain de surface de 20% doit avoir au moins 38 monômes pour obtenir un équilibre entre la surface gagnée et la surface additionnelle ($S_{GAO} \geq S_{AT}$) dûe à l'optimisation topologique. Nous pouvons

donc conclure que le schéma de test, comme dans le cas précédent, est facilement applicable dans les PLAs de grande et moyenne taille (voir graphique 3.2).



Graphique 3.2: Gain de surface en fonction du nombre de monômes.

3.4.3 - Le pliage simple des monômes.

Les contraintes de cette technique sont plus fortes que dans les deux cas exposés précédemment. Deux monômes peuvent partager une ligne si et seulement si les entrées et les sorties qui les attaquent sont différentes. Les entrées doivent être susceptibles d'un réordonnancement et les sorties sont placées sur deux côtés différents. Ces contraintes limitent fortement le domaine d'application de ce type d'optimisation. Cependant le gain de surface obtenu par cette technique inclut le gain dans les deux matrices.

Le schéma de test est facilement applicable, mais le contrôle des monômes doit être placé sur les deux côtés et la parité de sortie doit être contrôlée sur les deux côtés indépendamment.

Le gain de surface après l'optimisation topologique est donné par l'équation suivante:

$$S_{GAO} = [\, 8m * G * (16n + 8p) \,] \quad \lambda^2 \quad \text{(eq. 3.8)}$$

L'augmentation de la logique de test due à l'optimisation topologique est

donnée par l'équation suivante (la surface du LFSR a été négligé):

$$S_{AT} = [8m * (1 - G) * 16 * \lceil \log_2 (mG + 1) \rceil] \ \lambda^2 \quad \text{(eq. 3.9)}$$

L'équilibre entre les deux surfaces ($S_{GAO} \geq S_{AT}$) dépend seulement du nombre d'entrées et de sorties. Comme le montre le graphique 3.3, le schéma de test peut donc être appliqué même dans les petits PLAs. Le problème qui se pose est l'obtention d'un gain de surface raisonnable due aux contraintes de cette technique. La figure 3.24 montre la structure du PLA avec sa logique complémentaire. Le LFSR additionnel dû à l'optimisation topologique peut être éliminé. Dans ce cas les décodeurs sont reliés (comme le montre les lignes pointillées) et les deux côtés sont testés en même temps.

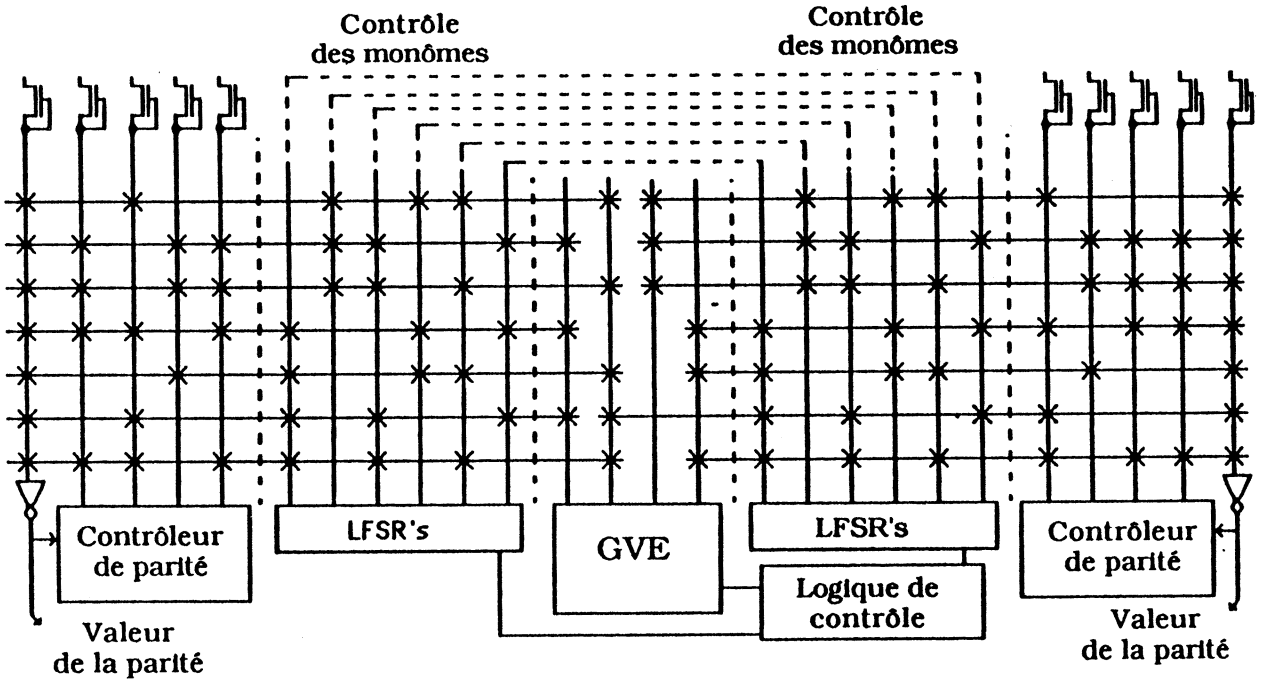
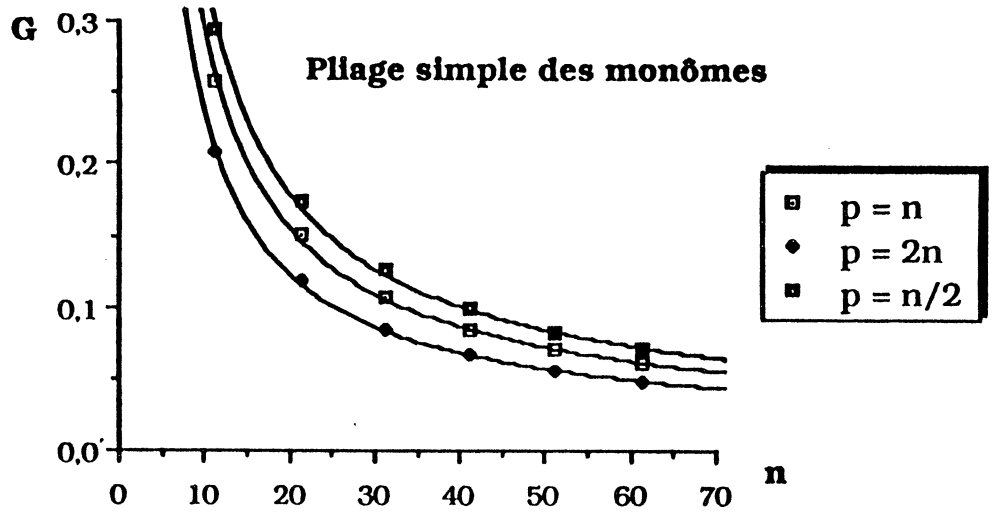


Figure 3.24: Plan d'un PLA autotestable à pliage simple des Monômes.



Graphique 3.3: Gain de surface en fonction du nombre de entrées et sorties.

3.4.4 - Le pliage multiple des entrées.

Le pliage multiple est une généralisation du pliage simple. Le gain de surface qui peut être obtenu dans ce cas est donc toujours meilleur que celui obtenu dans le cas du pliage simple (dans le pire des cas égal).

La structure des PLAs après l'optimisation topologique, representée sur la figure 3.24, présente les entrées parallèles aux monômes.

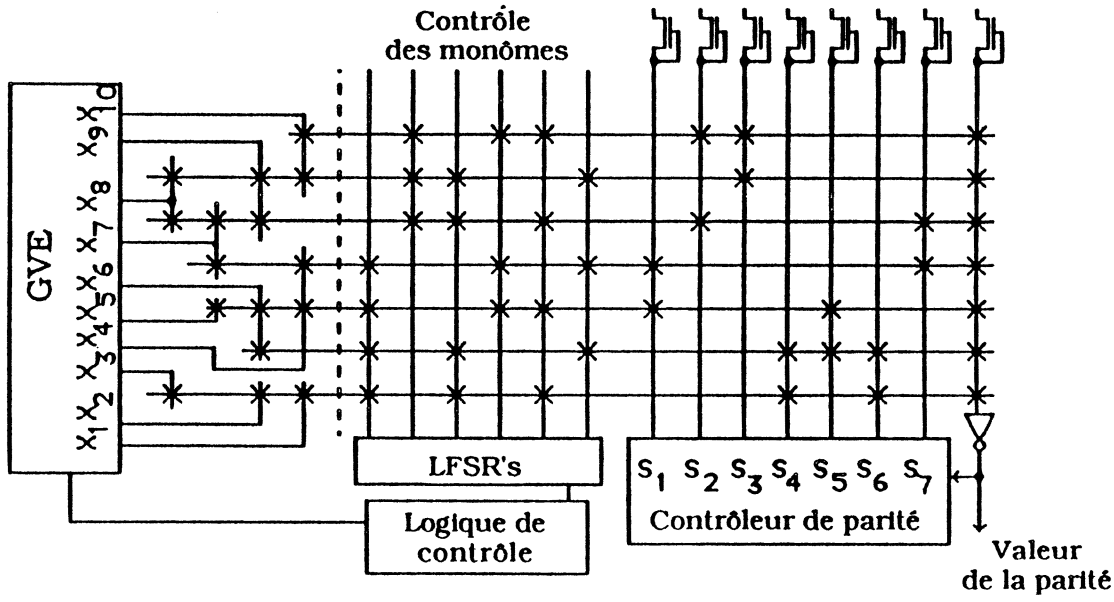


Figure 3.24: Plan d'un PLA autotestable à pliage multiple des entrées.

La surface utilisée pour placer le générateur de vecteurs d'entrée après l'optimisation topologique est:

$$S_{GVE} = 136 * 8m \quad \lambda^2$$

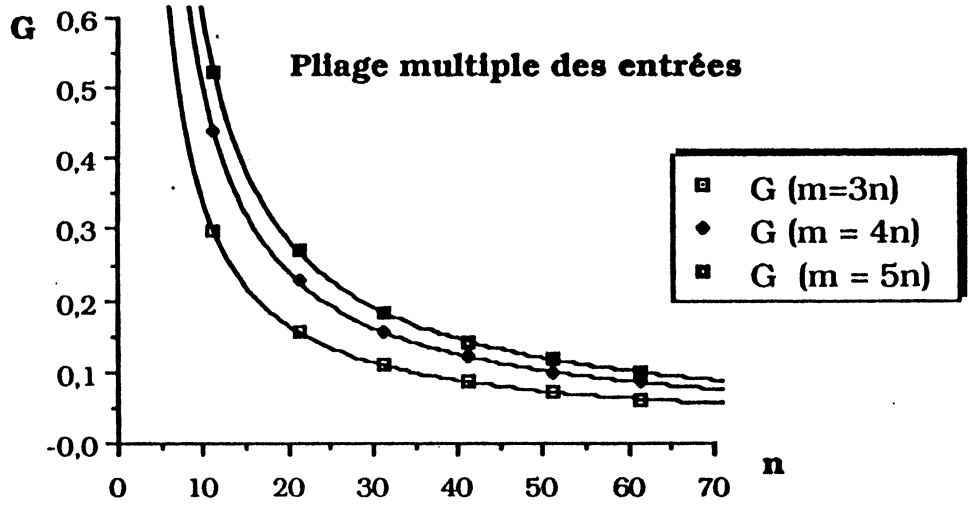
Donc la surface additionnelle due à l'optimisation topologique est:

$$S_{AT} = (136 * 8m - 136 * 16n) \quad \lambda^2 \quad \text{(eq. 3.10)}$$

Le gain de surface dans la matrice ET est donné par l'équation suivante:

$$S_{GAO} = 16n * 8m * G \quad \lambda^2 \quad \text{(eq. 3.11)}$$

L'équilibre entre le gain de surface et la surface additionnelle due à l'optimisation topologique ($S_{GAO} \geq S_{AT}$) dépend du nombre d'entrées et du nombre de monômes. L'équilibre de ces deux surfaces est donné par le graphique 3.4, paramétré par le nombre de monômes, ce nombre est fonction du nombre d'entrées.



Graphique 3.4: Gain de surface en fonction des entrées pour différents nombres de monômes.

Il montre que pour les PLAs ayant un petit nombre d'entrées (< 15) le gain de surface (G) doit être plus grand que pour les PLAs ayant un nombre élevé d'entrées. Il est clair que quand m < 2n, les entrées doivent être placées sur

les deux côtés du PLA et dans ce cas l'augmentation dans le circuit de test due à l'optimisation topologique est plus grande.

Dans ce type d'optimisation les entrées ne peuvent pas être contrôlées après le croisement de la matrice ET. Les méthodes qui utilisent la première classe, telles celles de HASSAN & McCLUSKEY [HAS 83] et de DAEHN & MUCHA [DAE 81] ne peuvent pas être employées dans ce cas. Quand le pliage multiple des entrées est associé au pliage multiple des sorties les monômes ne peuvent pas contrôlés par des registre à décalage. Ainsi les méthodes utilisant ces registres comme celle proposée par TREUER, FUJIWARA et AGARWAL [TRE 85] ne peuvent pas être employées dans ce cas.

3.4.5 - Le pliage multiple des sorties.

La structure du PLA après l'optimisation topologique, dans le cas du pliage multiple des sorties, est indiquée dans la figure 3.25 et, de manière similaire au cas précédent, les sorties sont parallèles aux monômes.

Le gain de surface dû à l'optimisation topologique dans la matrice OU est donné par l'équation suivante:

$$S_{GAO} = (8m * 8p * G) \quad \lambda^2 \quad \text{(eq. 3.12)}$$

La surface augmentée dans la circuiterie de test due à l'optimisation topologique est:

$$S_{AT} = (95 * 8m - 95 * 8p) \quad \lambda^2 \quad \text{(eq. 3.13)}$$

L'équilibre des surfaces ($S_{GAO} \geq S_{AT}$) est donné dans le graphique 3.5. Dans ce cas le gain de surface dans les PLAs ayant un petite nombre de sortie doit être beaucoup plus grand que pour un PLA ayant un nombre élevé de sorties. Si nous considérons qu'un gain de surface plus grand que 40% est difficilement réalisable, nous pouvons conclure que pour les PLAs ayant $m > 2p$ et $p < 20$ la surface après l'optimisation topologique sera plus grande qu'avant. Dans ce cas, il est préférable de ne pas optimiser le PLA.

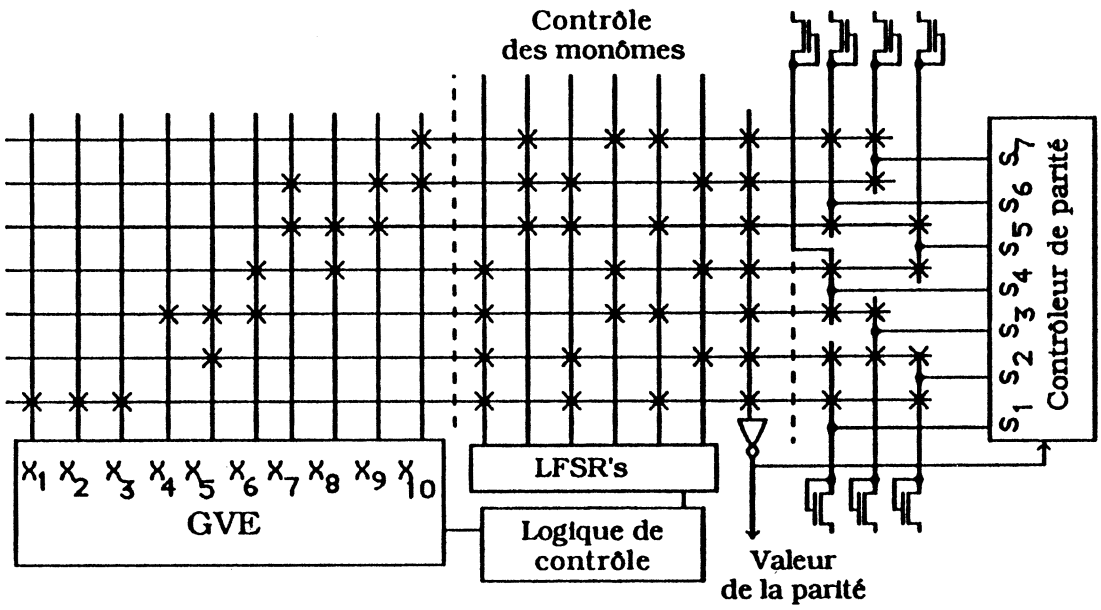
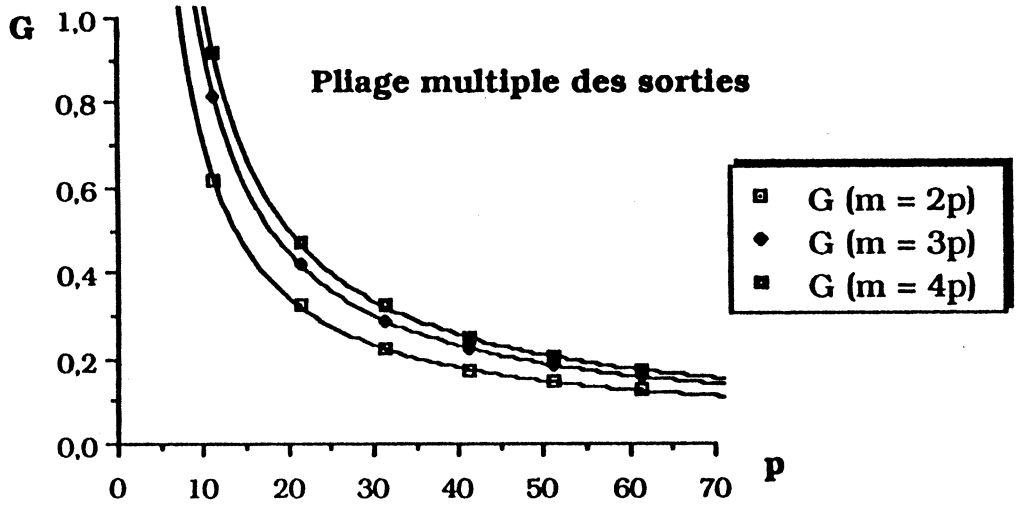


Figure 3.25: Plan d'un PLA autotestable à pliage multiple des sorties.



Graphique 3.5: Gain de surface en fonction des sorties pour différents nombres de monômes.

Dans ce type d'optimisation les monômes ne peuvent pas être contrôlés par après le croisement de la matrice OU. Donc la méthode proposée par DAEHN & MUCHA [DAE 81] ne peut pas être employée dans ce cas. Quand le pliage multiple des sorties est associé au pliage multiple des entrées les monômes ne peuvent pas contrôlés par des registre à décalage. Ainsi les méthodes utilisant ces registres comme celle proposée par TREUER, FUJIWARA et AGARWAL [TRE 85] ne peuvent pas être employées dans ce cas.

3.5 - CONCLUSION.

Le schéma de test proposé présente plusieurs avantages par rapport aux autres schémas publiés dans la littérature. Les registres à décalage associés à des portes EXNOR rendent la génération des vecteurs de test plus facile. Le contrôle des monômes par l'addition d'entrées présente la même structure que la matrice ET du PLA. Par conséquent, son implantation ne pose pas de problème. Il est important de remarquer que cette méthode élimine le registre à décalage placé sur les monômes et élimine donc les problèmes de routage. La compression des vecteurs de sortie par des bits de parité permet l'utilisation d'une méthode plus simple de compression tout en préservant une très bonne couverture de panne. Le test de la matrice OU consiste à vérifier la parité de chaque vecteur de sortie et le test de la matrice ET est effectué par la parité cumulative. Ceci améliore la couverture de panne par rapport aux schémas qui utilisent la même méthode de compression, comme par exemple celui proposé dans [TRE 85]. La surface additionnelle est plus petite que celle proposée dans d'autres schémas déjà publiés [TRE 85] et [SAL 85]. La longueur du test est proportionnelle au nombre d'entrées et au nombre de monômes (2nm). Nous pouvons donc conclure que le schéma de test proposé apporte une amélioration vis-à-vis des schémas existant dans la littérature.

Un autre facteur important est que ce schéma est compatible avec les PLAs optimisés topologiquement. Ceci a été démontré dans ce chapitre dont les principales conclusions sont les suivantes:

La structure des PLAs après l'optimisation topologique est la principale difficulté pour l'application des schémas de test. Les entrées, les sorties ou les monômes placés sur deux côtés distincts entraînent l'implantation de la circuiterie de test sur deux côtés. Cette division de la circuiterie de test augmente la surface additionnelle et cette augmentation peut être plus grande que la surface gagnée avec l'optimisation topologique. Ainsi dans le cas du pliage simple des entrées ou des sorties le schéma de test n'est convenable que pour les PLAs de grande et moyenne tailles (voir graphiques 3.1 et 3.2). Dans le cas du pliage multiple des entrées le PLA doit avoir plus de 15 entrées quand $m > 2n$ (voir graphique 3.4). Dans le cas du pliage multiple des sorties le PLA doit avoir plus de 20 sorties quand $m > 2p$ (voir graphique 3.5).

Il faut aussi remarquer que les autres schémas proposés dans la

litterature ne peuvent pas être employés dans tous les cas d'optimisation topologique. La méthode proposée par TREUER, FUJIWARA et AGARWAL [TRE 85] ne peut être employée que dans le cas du pliage simple. Dans le cas du pliage multiple cette méthode ne peut être employée que si une seule matrice est optimisée. Celle proposée par HASSAN et McCLUSKEY [HAS 83] ne peut être employée que dans le cas du pliage simple et multiple des sorties ainsi que dans le cas du pliage simple des monômes. Celle proposée par DAEHN et MUCHA [DAE 81] ne peut être employée que dans le cas du pliage simple des sorties et des monômes.

LE TEST EN LIGNE DES PLAs

4 - LE TEST EN LIGNE DES PLAs.

4.1 - GENERALITES.

Les systèmes "self-checking" sont composés d'un bloc fonctionnel et d'un bloc contrôleur, comme le montre la figure 4.1. Le bloc fonctionnel qui exécute la fonction du système a ses sorties toujours codées de manière à ce que l'existence d'une erreur puisse être détectée. Les sorties du bloc fonctionnel sont analysées par le bloc contrôleur qui, en présence d'erreurs, doit envoyer une indication d'erreur. Il doit aussi pouvoir détecter ses propres défauts.

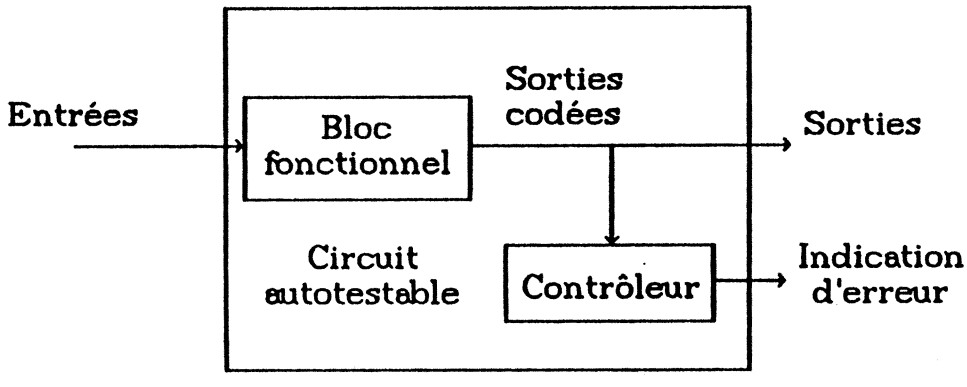


Figure 4.1: Structure générale des systèmes self-checking.

Le but des ces systèmes appelé comme "Totally self-checking goal" est d'assurer que la première sortie erronée soit détectée. En vue d'atteindre ce but, les circuits doivent être conçus de manière à ce que leur organisation interne, leur codage et les définitions de chacun des blocs les composant respectent les propriétés globales des circuits "totally self-cheking" et les hypothèses de panne du système.

Les principes des systèmes "self-checking" ont été introduits par CARTER et SCHNEIDER [CAR 68]. Ces principes ont été formellement définis comme "totally self-checking" par ANDERSON et METZE [AND 71]. Par la suite les circuits "Strongly Fault Secure" ont été définis par SMITH et METZE [SMI 78]. Ils ont aussi démontré que ces circuits sont la classe la plus large permettant d'obtenir le "totally self-checking goal" pour un même ensemble de fautes. NICOLAIDIS, COURTOIS et JANSCH [NIC 84] ont défini la classe la plus large de contrôleurs assurant le "totally self-checking goal"; ce sont les contrôleurs "Strongly Code Disjoint".

Les définitions données par la suite sont dues à ANDERSON et METZE [AND 71], SMITH et METZE [SMI 78] et NICOLAIDIS, COURTOIS et JANSCH [NIC 84]. Nous les transcrirons ici dans le but d'améliorer la clarté de ce chapitre. Ces définitions concernent les deux éléments du circuit autotestable, le bloc fonctionnel et le bloc contrôleur.

Comme dans [SMI 78], nous considérerons un bloc fonctionnel combinatoire G possédant r entrées et q sorties. Les 2^r vecteurs binaires, de longueur r, forment l'ensemble X des vecteurs d'entrée. Les 2^q vecteurs binaires, de longueur q, forment l'ensemble Y des vecteurs de sortie. Le bloc G reçoit sur ses entrées, en fonctionnement normal et avant l'occurrence d'une faute, un sous-ensemble A de vecteurs de X, appelé vecteurs du code d'entrée. Il produit un sous ensemble B de vecteurs de Y, appelé vecteurs du code de sortie. Le bloc G doit produire, en présence d'une faute, des valeurs en dehors du code de sortie B. Nous noterons G(x,f), la valeur de la sortie de G recevant x en entrée en présence d'une faute f. Avant l'occurrence d'une faute, nous noterons G(x,Ø).

DEFINITION D4.1

Un circuit G est "fault secure" (FS) pour un ensemble de fautes F si pour toutes les fautes appartenant à F et pour tous les vecteurs d'entrée, la sortie est soit correcte, soit un mot hors code, i.e.$\forall f \in F, \forall a \in A$:

- soit $G(a, f) = G(a, Ø)$
- soit $G(a, f) \notin B$

DEFINITION D4.2

Un circuit G est "self-testing" pour un ensemble de fautes F si pour chaque faute appartenant à F, il y a au moins un vecteur d'entrée que produit une sortie en dehors du code, i.e.:

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

DEFINITION D4.3

Un circuit G est "totally self-checking" (TSC) pour un ensemble de fautes F si le circuit est "fault secure" et "self-testing" pour l'ensemble F.

Une sortie erronée provoquée par une faute dans un circuit "fault secure" correspond toujours à un vecteur en dehors du code. Et un circuit "self-testing" assure que pour toutes les fautes il existe au moins un vecteur d'entrée qui produit une sortie en dehors du code. Par conséquent, les circuits qui sont à la fois "self-testing" et "fault secure", donc TSC, produiront, en présence d'une erreur, une sortie erronée détectable.

DEFINITION D4.4

Un circuit est "code disjoint" s'il produit toujours une sortie hors code pour un vecteur d'entrée hors code et une sortie appartenant au code pour un vecteur d'entrée appartenant au code, i.e.:

$$\forall a \in A, G(a, \emptyset) \in B.$$

$$\forall a \notin A, G(a, \emptyset) \notin B.$$

HYPOTHESE H1

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

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

DEFINITION D4.5

Un circuit G est redondant pour une faute f si la fonction qu'il réalise n'est pas modifiée en présence de la faute f.

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

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

Le fait qu'un circuit soit redundant pour une faute f ne l'assure pas de la propriété "self-testing", puisqu'il y a des fautes non détectées.

DEFINITION D4.6

Un circuit est "strongly redondant" pour une séquence de fautes $<f_1, f_2, \ldots f_n>$ et pour le code d'entrée A, si le circuit est redondant pour les n séquences $<f_1>$, $<f_1, f_2>$, $\ldots$ $<f_1, f_2, \ldots f_n>$ et pour le code d'entrée A.

DEFINITION D4.7

Pour une séquence de fautes $f_1, f_2, \ldots f_n$ soit k le plus petit entier pour lequel:

$$\exists\ a \in A,\ G(a \cup f_j) \neq G(a,\emptyset) \quad \text{où} \quad j = 1, \ldots, k.$$

Si un tel k n'existe pas, posons k = n. Alors G est "strongly fault secure" (SFS) pour la séquence $f_1, f_2, \ldots f_n$ si $\forall\ a \in A$:

- soit $G(a \cup f_j) = G(a,\emptyset)$
- soit $G(a \cup f_j) \notin B$, où $j = 1, \ldots, k$.

DEFINITION D4.8

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

DEFINITION D4.9

Un contrôleur est "strongly code disjoint" (SCD) pour une classe C_f d'hypothèses de pannes s'il est "code disjoint" et si, pour toutes les séquences de fautes f_i appartenant à C_f:

- soit il existe k tel que:

- le contrôleur est "strongly redundant" pour la séquence de fautes $<f_1, f_2, \ldots f_{k-1}>$, et

  $\exists\ b \in B$, tel que $G(b, U f_j) \notin C$ où $j = 1, \ldots, k$.

- soit le contrôleur est "strongly redundant" pour toutes les séquences de fautes.

HYPOTHESE H2

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

La hypothèse H2 étant assurée, un système accomplit le "totally self-checking goal" s'il est composé de [NIC 83]:

- Un circuit SFS et
- Un contrôleur SCD.

Un système est composé par plusieurs sous-systèmes qui peuvent être combinatoires ou séquentiels. Comme nous l'avons vu précédemment, deux approches peuvent être utilisées pour l'obtention du "totally self-checking goal". Dans la première approche chaque bloc fonctionnel doit être TSC et chaque contrôleur doit être code disjoint (CD) et self testing (ST). La figure 4.2(a) décrit le cas où toutes les interconnexions sont contrôlées. La figure 4.2(b) montre le cas où les blocs fonctionnels sont aussi CD. Ceci permet de contrôler seulement les sorties du système.

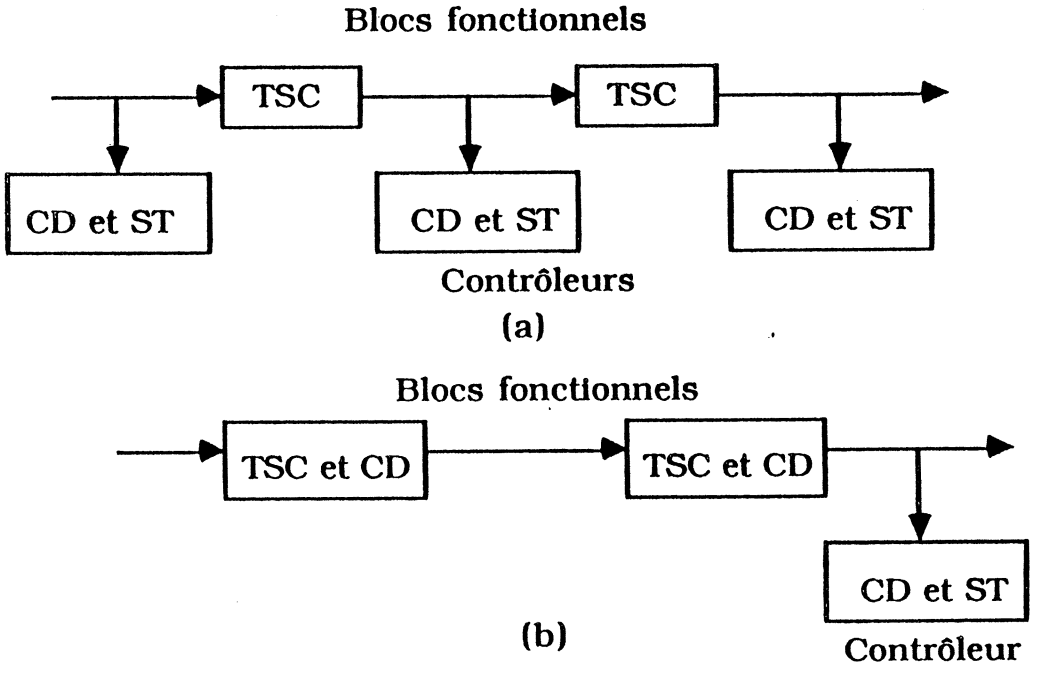


Figure 4.2: Systèmes TSC.

La deuxième approche est décrite dans la figure 4.3. Chaque bloc fonctionnel doit être SFS et chaque contrôleur SCD.

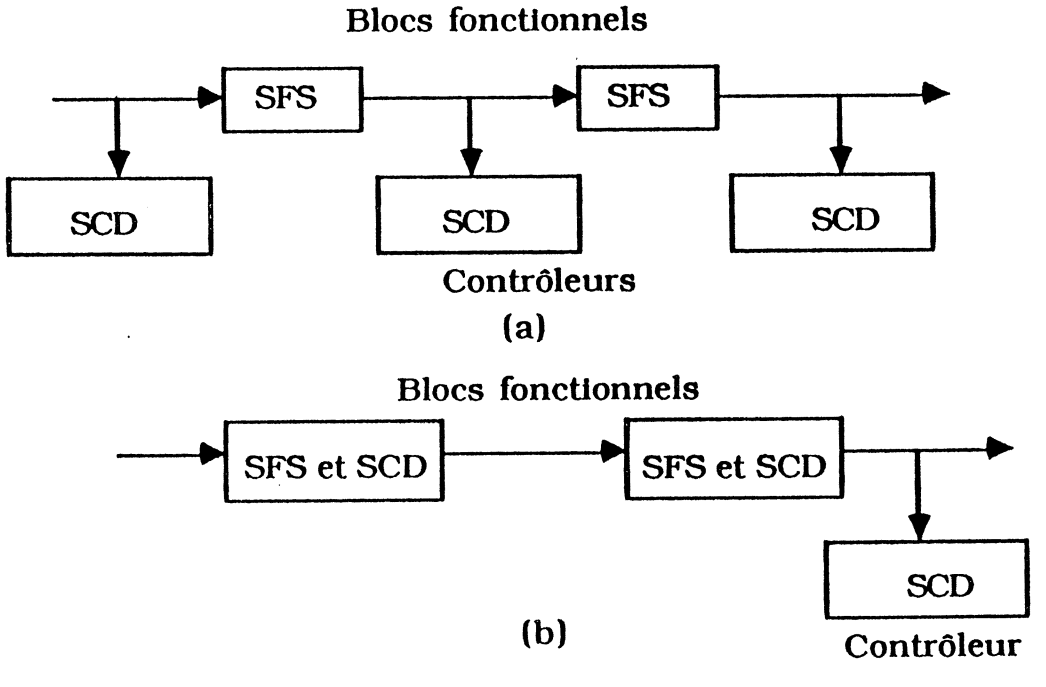


Figure 4.3: Systèmes SFS.

Comme dans le cas précédent, toutes les interconnexions sont contrôlées. Si les blocs fonctionnels sont SFS et SCD, un seul contrôleur est utilisé pour les sorties du système.

4.2 - CODES.

Les codes ont un rôle important dans les circuits self-checking. En effet, nous cherchons un codage capable de détecter toutes les erreurs possibles d'après les hypothèses de panne avec un minimum de redondance.

Les codes peuvent être divisés en deux classes, à savoir:

- Séparables et
- Non séparables.

Dans le premier cas, l'information codée est composé de deux parties, les bits d'information et les bits de contrôle (voir figure 4.4)

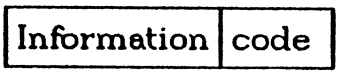


Figure 4.4: Codes séparables.

Dans le second cas, il n'existe pas de séparation entre l'information et le codage. Donc l'information contient aussi le codage.

Par la suite nous décrirons les codes utilisés dans ce travail.

4.2.1 - Code de parité.

Le code de parité est un code séparable. Il utilise un simple bit de contrôle qui est obtenu par la fonction OU-Exclusif des bits de l'information. Donc la génération des bits de parité de p variables S_i est donnée par la relation suivante:

$$S_1 \oplus S_2 \oplus S_3 \ \ldots \ S_p = P$$

Si P = 0, le code de parité est paire.
Si P = 1, le code de parité est impaire.

Ce code est capable de détecter des erreurs simples, i.e. des erreurs n'affectant qu'un seul bit dans une information, et son implantation est facile

et donc nécessite une redondance faible.

4.2.2 - Code de Berger.

Le code de Berger est un code séparable et non ordonné (voir définition D4.10) optimal.

DEFINITION D4.10

Un code C est un code non ordonné si étant donné des vecteurs distincts X (x_n, x_{n-1},... x_1) et Y (y_n, y_{n-1},... y_1) appartenant au code, il existe au moins deux positions binaires i et j telles que (sachant que les indices i et j parcourent respectivement le vecteur d'information et le code):

$$x_i \oplus x_j = 1,\ x_i \oplus y_j = 1 \text{ et } x_j \oplus y_j = 1.$$

Ou bien X ne couvre jamais Y et vice versa. X couvre Y si et seulement si X possède des bits à "1" partout où Y en a.

Le code de Berger est la représentation du nombre de zeros que contient l'information. Si l'information contient n bits, le nombre de bits du code de Berger est donné par équation suivante:

$$B = \lceil \log_2 (n + 1) \rceil.$$

La figure 4.5 montre la composition d'une information codée en Berger.

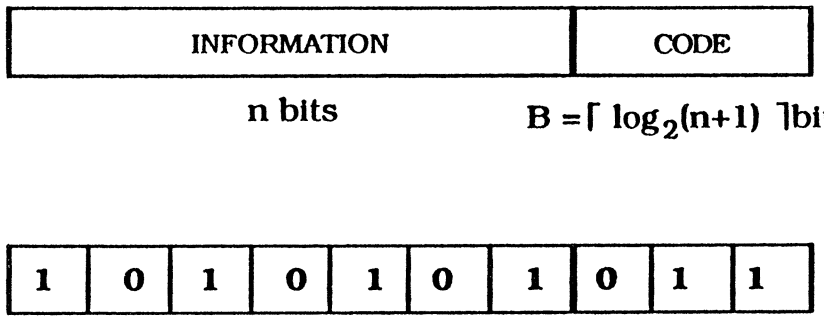


Figure 4.5: Information codée en Berger.

Le code de Berger modifié est la réprésentation de la différence entre le nombre de zeros et le nombre minimum de zeros dans les vecteurs de sortie.

Puisqu'il s'agit d'un code non ordonné, il est capable de détecter les erreurs unidirectionnelles (Voir définition D4.11).

DEFINITION D4.11

Une erreur unidirectionnelle est une erreur multiple telle que plusieurs bits sont soit à "1" à la place de "0" ou "0" à la place de "1", mais jamais les deux en même temps.

La figure 4.6 décrit le schéma de principe d'un contrôleur du code de Berger.

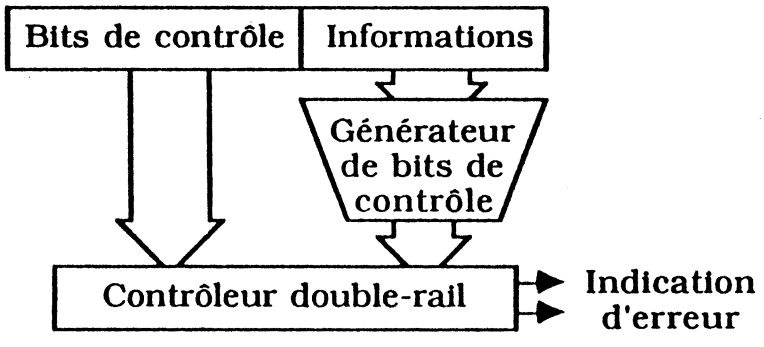


Figure 4.6: Contrôleur de Berger.

4.2.3 - Codes k parmi n.

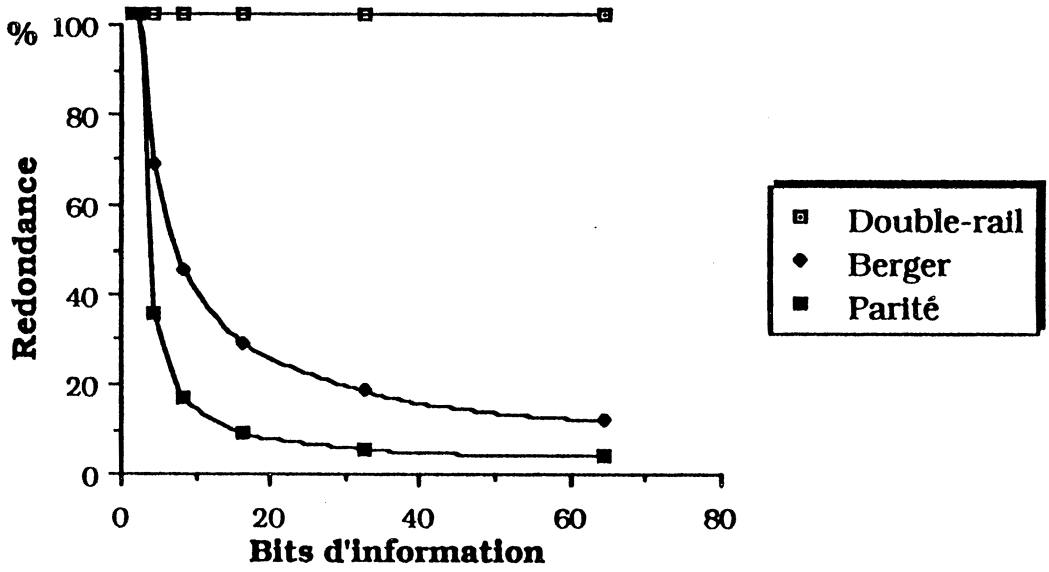
Les codes k parmi n sont des codes non séparables et non ordonnés optimaux. Ce sont des codes à pondération constante, c'est à dire que chaque mot contient exactement k bits à la valeur "1" et par conséquent n - k à la valeur "0".

4.2.4 - Codes à duplication.

Les codes à duplication sont des codes séparables dont le circuit génère soit une autre sortie identique ou bien son complément. Ces sont des codes adaptés à la détection des fautes multiples.

Le graphique 4.1 résume la redondance et le tableau 4.1 donne la

capacité de détection d'erreur pour chaque code.



Graphique 4.1: Redondance.

Erreur / Code	Mutiple	Unidirectionnelle	Simple
Double rail	Oui	Oui	Oui
Non ordonné	Non	Oui	Oui
Parité	Non	Non	Oui

Tableau 4.1: Détection d'erreur.

4.3 - PLAs SFS.

Les PLAs SFS sont conçus de telle manière qu'ils génèrent eux même leurs codes de sortie. MAK, ABRAHAM et DAVIDSON [MAK 82] ont démontré que les fautes simples dues à un collage, "bridging" et "crosspoint" dans les PLAs sont propagées comme une erreur unidirectionnelle. Les codes non ordonnés comme le code de Berger, le code de Berger modifié ou k parmi n sont suffisants pour coder les sorties en permettant la détection des erreurs unidirectionnelles. Plusieurs schémas ont été proposés en utilisant cette approche. Une autre classe de schémas utilise des codes permettant la détection des erreurs simples (code de parité). Dans ce cas le

degré de divergence (voir définition D4.12) doit être égal à 1.

DEFINITION D4.12

Le degré de divergence d'une ligne d'un circuit est le nombre de sortie qui sont connectées à cette ligne.

Puisque le degré de divergence dans les PLAs est plus grand que 1, chaque partie (i.e. lignes d'entrée, monômes et sorties du PLA) doit être testée séparément. Ainsi une faute sur une ligne d'entrée ou sur un monôme est vérifiée avant la propagation des erreurs sur plusieurs sorties. Donc le degré de divergence est maintenu à 1. Par la suite nous décrirons les schémas proposés dans la littérature pour chaque classe présentée.

4.3.1 - PLAs SFS utilisant des codes non ordonnés.

MAK; ABRAHAM et DAVIDSON [MAK 82] présentent un schéma de test utilisant des codes non ordonnés. Dans ce schéma, d'après les fonctions du PLA, le code est sélectionné parmi le code de Berger, code de Berger modifié et k parmi n. Le codage des fonctions de sortie d'un PLA dans un code non ordonné est fait grâce à l'addition de plusieurs bits de contrôle dans la matrice OU. Si le nombre de sorties avant le codage est p et le nombre maximum et minimum de "1" dans les vecteurs de sorties sont respectivement r et s, le nombre de bits de contrôle nécessaires pour le code de Berger est $\log_2(p+1)$ et r-s pour le k parmi n. Ces codes sont générés en utilisant la structure du PLA, donc généralement plusieurs monômes sont ajoutés pour la génération de ces nouvelles fonctions. Le principal avantage de ce schéma est qu'il n'est pas nécessaire de tester chaque partie du PLA séparément. Par contre les contrôleurs utilisés dans ce schéma sont plus complexes que ceux nécessaire pour des codes à parité. La figure 4.7 montre la structure du PLA et la figure 4.8 montre le tableau de Karnaugh où g_1 et g_2 sont les deux bits du code de Berger des sorties.

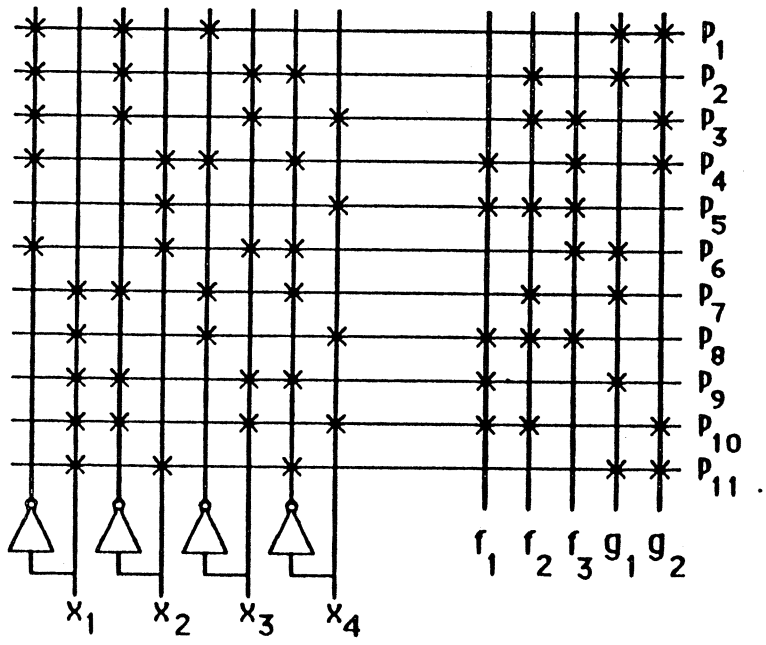


Figure 4.7: PLA SFS utilisant code de Berger.

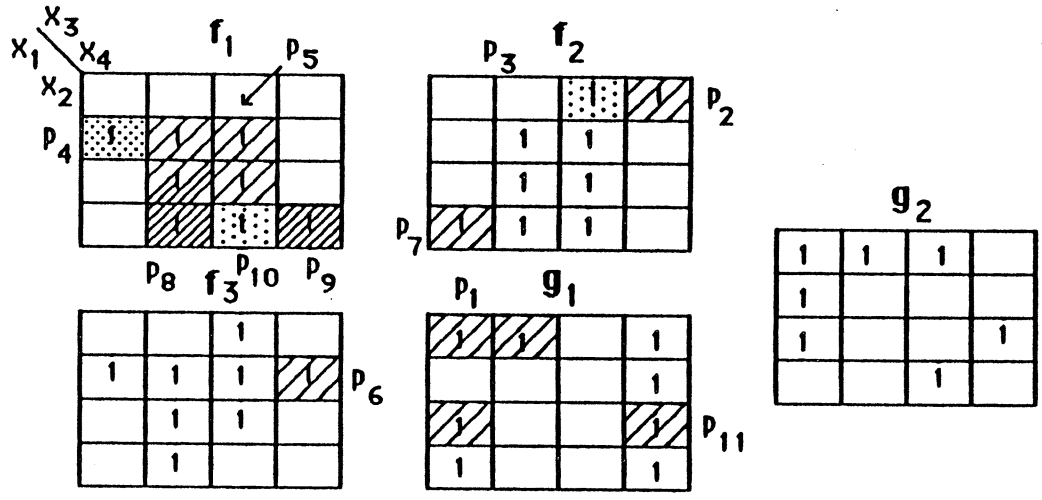


Figure 4.8: Tableau de Karnaugh.

4.3.2 - PLAs SFS utilisant des codes détectant des erreurs simples.

NICOLAIDIS [NIC 83] présente un schéma de test basé sur trois contrôles de parité. Ce schéma a été présenté aussi par CHEN, FUCHS et ABRAHAM [CHE 85]. Chaque partie du PLA est contrôlée par un code de parité. Deux sorties sont ajoutées pour générer le code de parité des sorties et des monômes. Comme dans le cas précèdent, ces sorties sont générées en utilisant la structure du PLA, donc plusieurs monômes doivent être ajoutés pour obtenir le codage.

Le principal avantage de ce schéma est qu'il utilise des contrôleurs très simples, mais chaque partie du PLA doit être testée séparément.

La figure 4.9 décrit ce schéma de test.

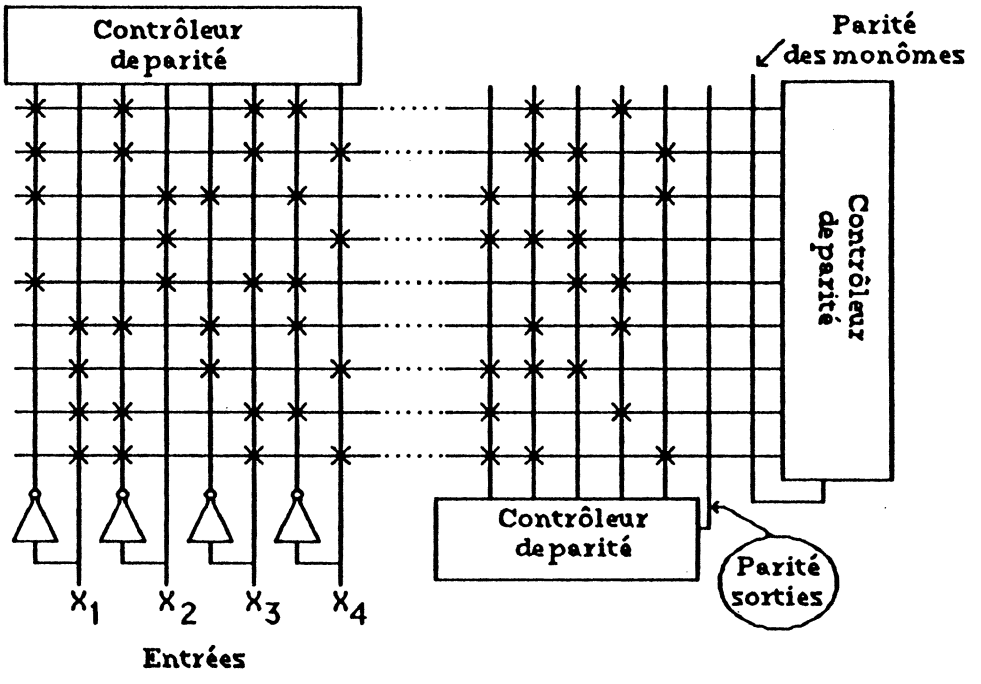


Figure 4.9: PLA SFS utilisant le code de parité.

J.KHAKBAZ et E.J.McCLUSKEY [KHA 81] proposent un schéma pour les PLAs non concurrents (un PLA est non concurrent si chaque vecteur appartenant au code d'entrée sélectionne exactement un monôme). Dans leur schéma, les entrées sont vérifiées par un contrôleur double rail (les entrées sont considérées comme n paires d'un code double rail). Les monômes non concurrents sont considérés comme un code 1 parmi m (où m est le nombre de monômes). Puisque le degré de divergence est maintenu à 1, les vecteurs de sortie peuvent être comprimés dans un simple bit de parité pour la détection des fautes simples. La parité des sorties est générée par une sortie additionnelle. La figure 4.10 montre le schéma de test.

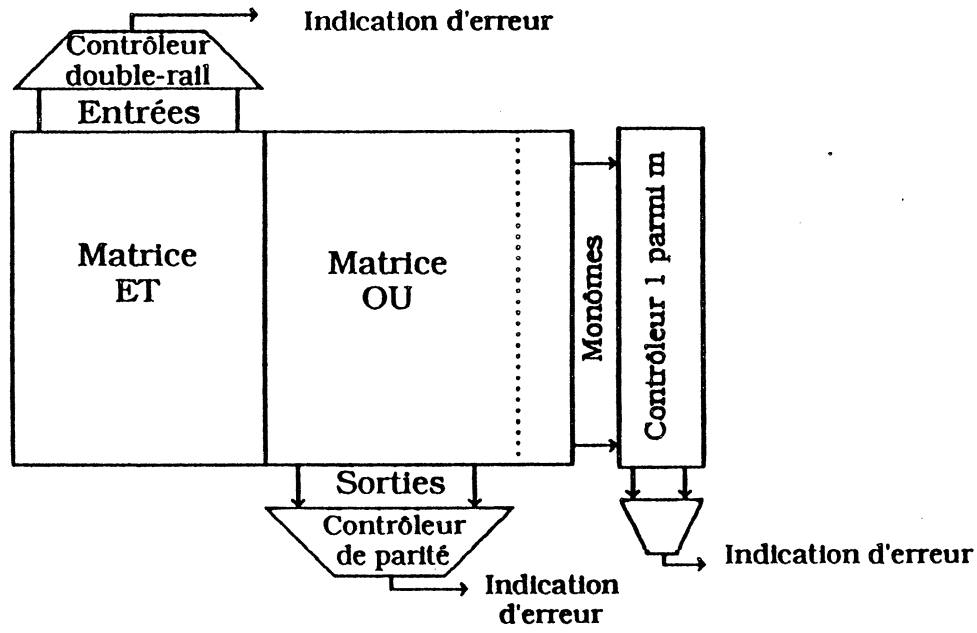


Figure 4.10: Schéma de test proposé par J.KHAKBAZ et E.J.McCLUSKEY.

4.4 - GENERATION DES CODES NON ORDONNES.

Les bits de contrôle d'un codage approprié des sorties d'un PLA sont fonction des informations des sorties elles mêmes et des informations des vecteurs d'entrée. Leur génération peut être facilement obtenue en utilisant les tableaux de Karnaugh quand le nombre des entrées est petit. Sinon une autre méthode de génération doit être utilisée. Dans [MAK 82], une méthode basée sur l'algèbre est proposée. Après avoir choisi le codage des sorties, l'algorithme proposé génére les fonctions du codage directement à partir des fonctions des sorties. Cette méthode demande un temps d'exécution très important puisqu'il s'agit d'une fonction polynomiale des entrées.

Dans ce travail nous proposons une autre méthode pour la génération des fonctions des bits de codage. La complexité de cette méthode est une fonction simple du nombre des entrées.

L'approche SFS utilisée dans ce travail est celle décrite dans la figure 4.3(a). Les vecteurs d'entrée sont donc contrôlés et seul le code d'entrée A est appliqué au PLA. Par conséquent, les fonctions des bits de contrôle doivent seulement prendre en compte le code de sortie B. Ainsi à chaque vecteur d'entrée appartenant au code d'entrée A correspond une fonction de sortie qui appartient au code de sortie B, et un code des sorties, s'il n'y a pas de faute.

Si nous considérons que chaque vecteur d'entrée sélectionne un seul

monôme i.e. si le PLA est non concurrent, les fonctions des bits de contrôle peuvent être facilement obtenues individuellement pour chaque monôme. Dans ce cas, chaque monôme correspond à un ensemble de vecteurs d'entrée. Après la génération des bits de contrôle, les fonctions de sortie (les fonctions de départ et les fonctions des bits de contrôle) peuvent être optimisées en utilisant les méthodes d'optimisation logique en vue de générer un ensemble minimal de monômes. Ainsi, la propriété de concurrence du PLA est maintenue sans restrictions. Il est important de remarquer que généralement un PLA avant optimisation logique est non concurrent. Les fonctions des bits de contrôle peuvent être générées dans cette phase (avant l'optimisation logique).

Quand le PLA est concurrent, le monôme concerné doit être éclaté pour rendre le PLA non concurrent. Ceci est obtenu par l'algorithme décrit par la suite.

La matrice ET d'un PLA est une matrice n par m éléments où chaque terme est défini de la manière suivante:

S'il existe un transistor à l'intersection du i^{eme} monôme et de la ligne d'entrée de la j^{eme} entrée (i.e. si la ligne d'entrée de la j^{eme} entrée peut mettre à la masse le i^{eme} monôme) alors A(i,j) = 1.

S'il existe un transistor à l'intersection du i^{eme} monôme et de la ligne d'entrée complémentaire de la j^{eme} entrée (i.e. si la ligne d'entrée complémentaire de la j^{eme} entrée peut mettre à la masse le i^{eme} monôme) alors A(i,j) = 0.

S'il n'y a pas de transistor aux intersections du i^{eme} monôme et de la ligne d'entrée ainsi que de la ligne d'entrée complémentaire de la j^{eme} entrée (i.e. si ni la ligne d'entrée ni la ligne d'entrée complémentaire de la j^{eme} entrée ne peuvent mettre à la masse le i^{eme} monôme) alors A(i,j) = X.

Ainsi la matrice ET du PLA décrit dans la figure 4.7 est:

0 0 0 X
0 0 1 0
0 0 1 1
0 1 0 0
X 1 X 1
0 1 1 0
1 0 0 0
1 X 0 1

1 0 1 0
1 0 1 1
1 1 X 0

Si nous considérons la représentation binaire d'un monôme, dans la matrice ET, $m_I = (a_{n-1}, a_{n-2}, \ldots a_{n0})$ et $m_J = (b_{n-1}, b_{n-2}, \ldots b_{n0})$ l'addition de ces monômes est définie comme suit:

$$m_K = (a_{n-1} \oplus b_{n-1}, a_{n-2} \oplus b_{n-2}, \ldots a_{n0} \oplus b_{n0}),$$

où $a_I \oplus b_I$ est défini par les opérations commutatives suivantes:

$$0 \oplus 0 = 0$$

$$0 \oplus 1 = 1$$

$$0 \oplus X = 0$$

$$1 \oplus 1 = 0$$

$$1 \oplus X = 0$$

$$X \oplus X = 0$$

La distance D entre deux monômes, qui sont deux points d'un n-cube, est définie par:

$$D(m_I, m_J) = \Sigma\, a_I \oplus b_I$$

Donc, deux monômes sont concurrents si et seulement si $D(m_I, m_J) = 0$. Dans ce cas deux situations sont en fait possibles:

- La redondance ou
- La concurrence.

Dans le premier cas tous les termes de m_I sont égaux aux termes de m_J.

Par conséquent le monôme m_j ou m_i peut être éliminé. Cette opération peut être accomplie par le placement de la programmation de la matrice OU dans le monôme qui restera (i.e. tous les termes égaux à 1, de la matrice OU sont maintenus).

Dans le deuxième cas, un ou plusieurs termes de a_i sont égaux à X et b_i est égal à 0 ou 1 (ou vice versa, b_i égal à X et a_i égal à 1 ou 0). Dans ce cas, le monôme qui contient le terme X est dupliqué en remplaçant le terme X par 0 et 1. Puis, un des deux nouveaux monômes sera soit concurrent soit redondant en relation avec le monôme qui contient le 0 ou 1. Donc la procédure de concurrence ou bien de redondance doit être appliquée. Cette algorithme se résume de la manière suivante:

```
DEBUT

    TANT QUE i ≠ EOF FAIRE
        i = i+1
        j = i+1

        TANT QUE j ≠ EOF FAIRE

            D = m_i ⊕ m_j;

            SI D = 0
                SI a_k = X et b_k = 1 ou 0
                    dupliquer m_j en remplaçant a_i par 1 et 0;
                SI b_k = X et a_k = 1 ou 0
                    dupliquer m_i en remplaçant a_i par 1 et 0;
                SI a_k = b_k
                    éliminer m_j en plaçant les 1 de la matrice OU
                        de m_j dans m_i;
            j = j+1;
        FIN;
    FIN;
FIN.
```

L'annexe 4 donne un exemple d'application de cette méthode de génération.

Le principal avantage de cette méthode est qu'il peut être facilement modulé. Ceci veut dire que le codage est facilement calculé par un sous-ensemble de sorties. Ainsi le PLA peut avoir plusieurs sous-ensembles de sorties codées en Berger ou en d'autres code.

4.5 - APPLICATION AUX PLAs OPTIMISES TOPOLOGIQUEMENT.

Les caractéristiques de chaque schéma de test exposées précédemment permettent de définir l'emploi de chaque méthode. Les méthodes qui utilisent le code de parité, par exemple, peuvent être employées dans seulement deux cas de PLAs optimisés. Les méthodes qui utilisent les codes non ordonnés peuvent être employées dans tous les cas de PLAs optimisés.

La surface additionnelle dans le schéma de test due à l'optimisation topologique doit être évaluée pour chaque application puisqu'elle est fortement liée au contenu du PLA. L'optimisation topologique n'entraîne pas forcement une modification du schéma de test. Par exemple l'emploi d'un schéma à code non ordonné n'impose une modification du schéma de test que si la matrice OU est modifié.

Quand le PLA a un nombre important de sorties, nous ne pouvons pas générer un seul code non ordonné pour l'ensemble des sorties. Les générateurs de codes non ordonnés ont une structure arborescente ce qui peut introduire des retards dans les circuits. Pour celà les sorties doivent être divisées en sous-ensemble(s) de manière à ce que la vitesse du circuit reste raisonnable. L'optimisation topologique de la matrice OU presente aussi les sorties divisées. Ainsi dans ce cas, le schéma de test peut être appliqué aux PLAs ayant la matrice OU optimisée sans augmentation de surface.

4.5.1 - Le pliage simples des entrées.

Les schémas de test qui utilisent des codes permettant la détection des erreurs simples ne sont pas applicables dans ce cas d'optimisation topologique. Lorsque deux lignes d'entrée partagent une colonne, il est impossible de contrôler les lignes d'entrées après le croisement de la matrice ET. Par conséquent il est impossible d'assurer un degré de divergence égal à 1 et d'utiliser ce schéma.

Par contre, les schémas qui utilisent des codes non ordonnés sont parfaitement applicables dans ce cas d'optimisation. L'optimisation topologique change la structure de la matrice ET et les contrôleurs sont placés sur les sorties du PLA. L'application du schéma de test ne nécessite donc aucune altération ni dans le PLA ni dans le contrôleur. La figure 4.11 montre un PLA avec les sorties additionnelles pour la génération du code non ordonné.

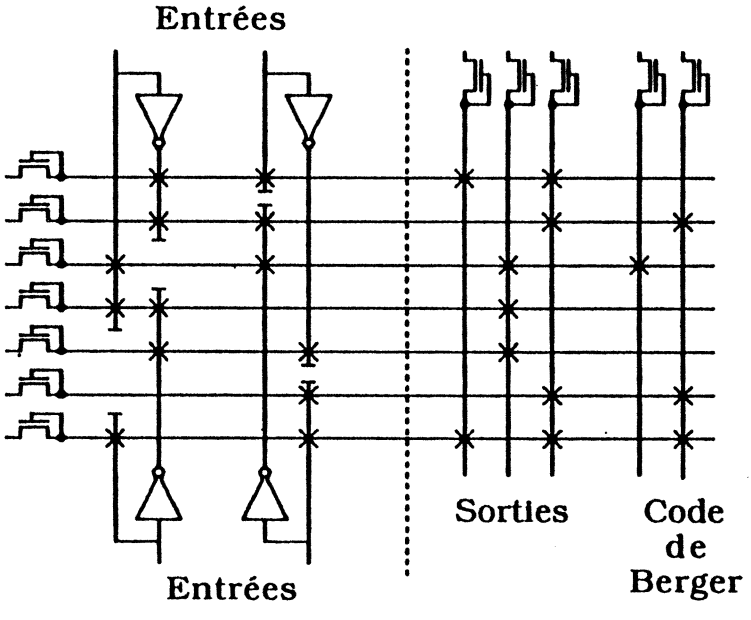


Figure 4.11: Plan d'un PLA SFS à pliage simple des entrées.

4.5.2 - Le pliage simples des monômes.

Dans ce cas les deux schémas de test peuvent être employés. L'utilisation de codes non ordonnés nécessite une génération du code sur chacun des côtés séparément. Le nombre de bits de contrôle ainsi que les monômes nécessaires à sa génération sont augmentés. Si le PLA a un nombre important de sorties, le nombre de sorties codées doit être limité (compromis entre la vitesse et le nombre de bits de contrôle) et ces caractéristiques peuvent donc être complémentaires. Dans ce cas, la surface de la circuiterie de test n'est pas augmentée avec l'optimisation topologique.

Pour le schéma qui utilise le code de parité, la parité peut être calculée soit séparément pour chaque côté, soit globalement pour tous les monômes et aussi pour les sorties. La première solution implique une augmentation du nombre de monômes nécessaires pour la génération de la parité des monômes et des sorties. Pour la deuxième solution, la valeur de la parité

partielle d'un côté est reliée à celle de l'autre côté pour calculer la parité totale des monômes et des sorties.

La figure 4.12 montre la structure du PLA et les sorties additionnelles nécessaires lors de la génération du code de Berger. La figure 4.13 montre la structure du PLA, les contrôleurs de parité et les sorties additionnelles pour la génération de la parité.

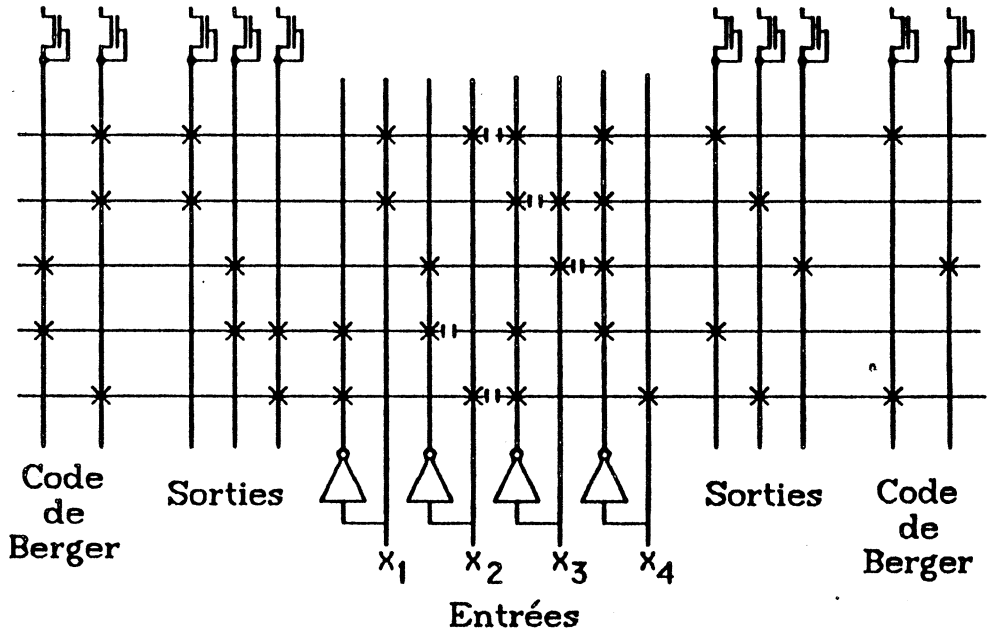


Figure 4.12: Plan d'un PLA SFS à pliage simple des monômes.

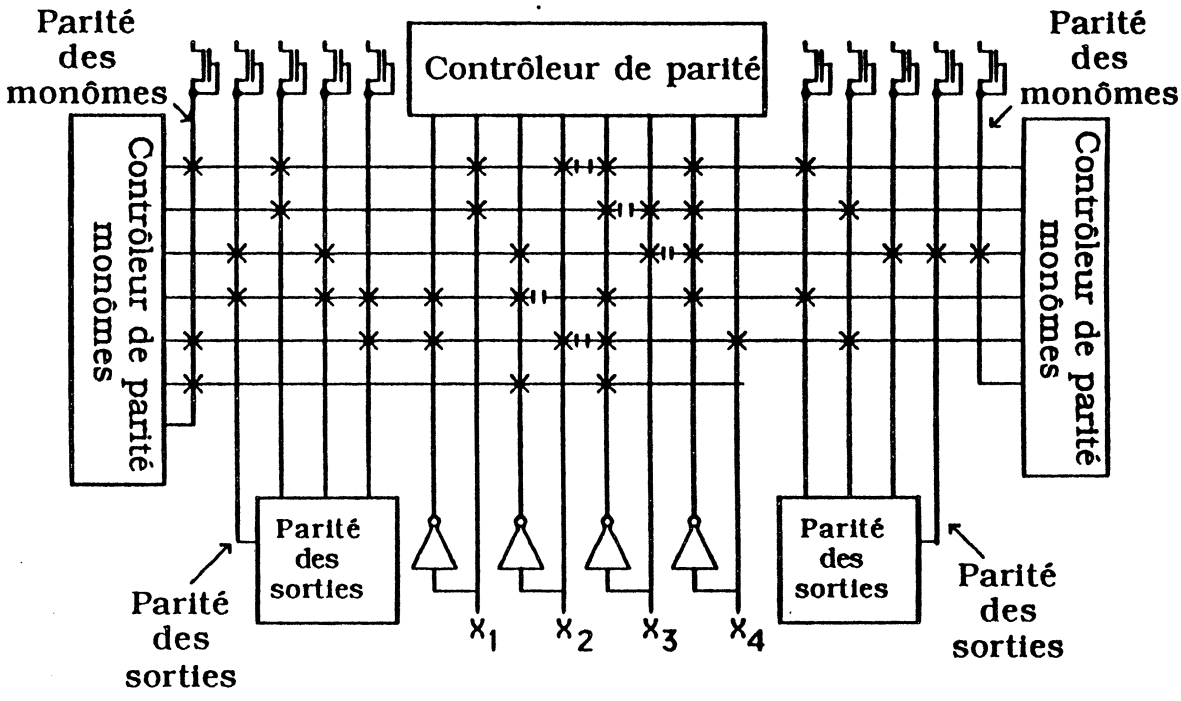


Figure 4.13: Plan d'un PLA SFS à pliage simple des monômes.

4.5.3 - Le pliage simple des sorties.

Dans ce cas d'optimisation topologique la structure du PLA permet l'utilisation des deux schémas de test. Dans les deux cas, les contrôleurs des sorties sont placés des deux côtés de la matrice OU. Les codes peuvent être générés soit pour tout l'ensemble des sorties soit pour chaque côté indépendamment. Pour la première solution, la valeur partielle doit être ramenée sur l'autre côté du PLA. La deuxième solution implique une augmentation des monômes pour la génération des nouveaux codes.

Dans le cas de la génération séparée des codes non ordonnés pour chaque côté, si nous supposons que les sorties sont brisées nous pouvons considérer que l'augmentation de surface est due uniquement aux monômes ajoutés. Le gain de surface dans la matrice OU est donné par l'équation suivante:

$$S_A = 8m * 8p * G\,\lambda^2 \quad \text{(eq. 4.1)}$$

L'augmentation de la surface due à l'ajout des monômes pour la génération du codage après l'optimisation topologique est:

$$S_{PT} = [8m_A * (16n + 8p)]\ \lambda^2 \quad \text{(eq. 4.2)}$$

Où m_A est le nombre de monômes ajoutés. Puisque le gain de surface dû à l'optimisation topologique doit être plus grand que l'augmentation de surface dans la circuitérie de test due à l'ajout des monômes nous avons donc:

$$(8m * 8p * G) \geq [8m_A * (16n + 8p)]$$

Si m_A est donné par un facteur de m (m_A = Am) après les simplifications nous avons:

$$p\,(G\text{-}A) \geq 2\,An \quad \text{(eq. 4.3)}$$

La génération des codes non ordonnés sur chaque côté ne peut donc être

employée que si le gain de surface G est plus grand que le taux de monômes ajoutés A. Sinon la deuxième solution doit être employée i.e. un seul code est généré et la valeur partielle est ramenée sur l'autre côté pour le calcul total. Dans ce cas, la surface nécessaire pour réaliser le routage est donnée par l'équation suivante:

$$S_O = 8m * 8 * \lceil \log_2 (pG + 1) \rceil \lambda^2 \quad \text{(eq. 4.4)}$$

De même, le gain en surface dû à l'optimisation topologique doit être plus grand que la surface nécessaire au routage c'est à dire:

$$(8m * 8p * G) \geq 8m * 8 * \lceil \log_2 (pG + 1) \rceil \quad \text{(eq. 4.5)}$$

Puisque cette équation est toujours vraie cette méthode peut être employée dans tous les cas. La génération séparée sur chaque côté doit être utilisée quand les sorties sont divisées en plusieurs groupes à cause du compromis entre la vitesse du circuit et le nombre de sorties codées. Les équations (eq. 4.2 et eq. 4.4) permettent aussi de comparer la surface additionnelle qui peut être un critère de choix dans le cas où les deux solutions peuvent être employées. Comme dans le cas précédent, si le PLA a un nombre important de sorties, le nombre de sorties codées doit être limité (compromis entre la vitesse et le nombre de bits de contrôle) et ces caractéristiques peuvent donc être complémentaires. Dans ce cas, la surface de la circuiterie de test n'est pas augmentée avec l'optimisation topologique.

Bien que chaque solution dépende du contenu du PLA, quand les sorties sont codées en parité la génération d'une seule valeur est généralement préférable (il existe un seul bit à relier).

La figure 4.14 présente la structure du PLA et les contrôleurs de parité, pour un schéma qui utilise des codes de parité. La figure 4.15 décrit le schéma de test qui utilise un code non ordonné.

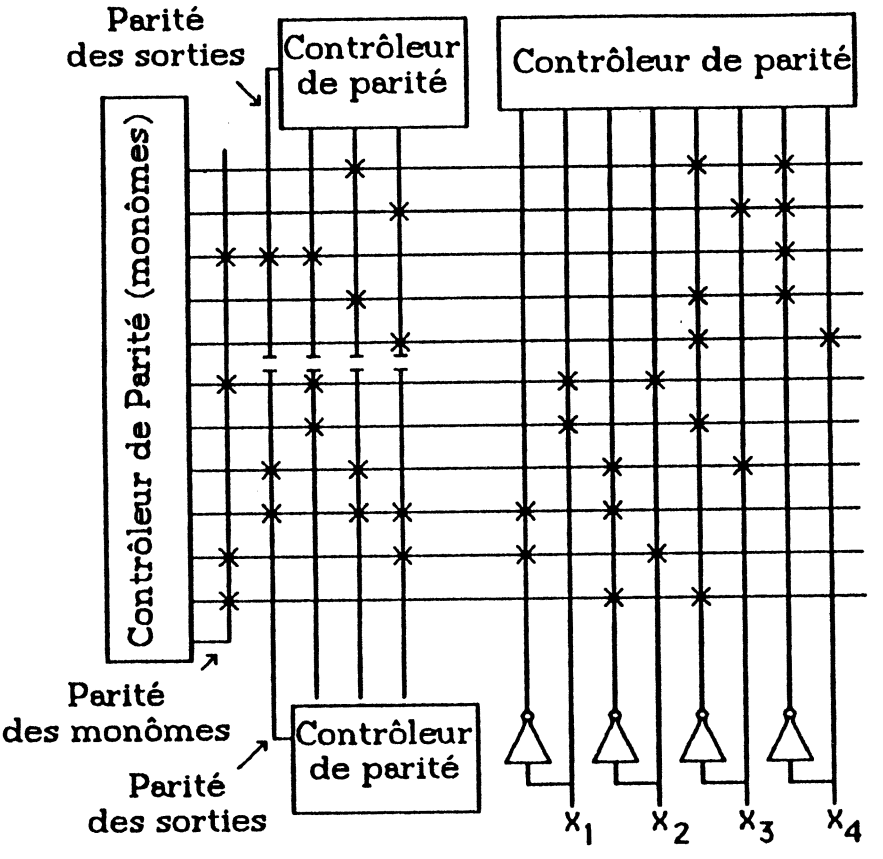


Figure 4.14 : Plan d'un PLA SFS à pliage simple des sorties.

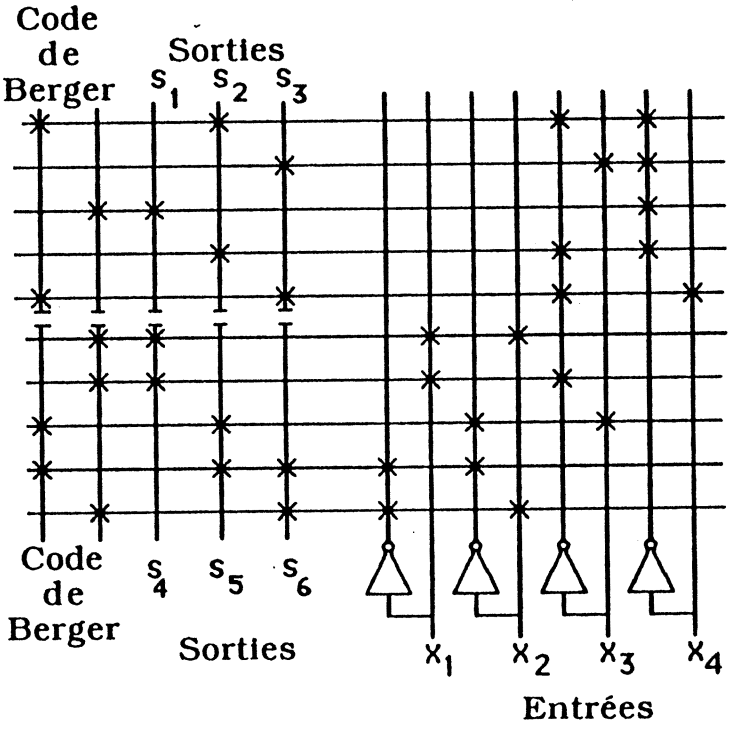


Figure 4.15: Plan d'un PLA SFS à pliage simple des sorties.

4.5.4 - Le pliage multiple des entrées.

Dans ce cas d'optimisation, plusieurs entrées partagent une même colonne. Ainsi les schémas qui utilisent des codes permettant la détection des erreurs simple ne peuvent pas être employés, comme dans le cas du pliage simple d'entrée. Par conséquent, seuls les schémas qui utilisent des codes non ordonnés sont applicables dans ce type d'optimisation. Aucune modification n'est nécessaire dans le schéma de test.

La figure 4.16 montre la structure du PLA avec les sorties additionnelles pour la génération du code.

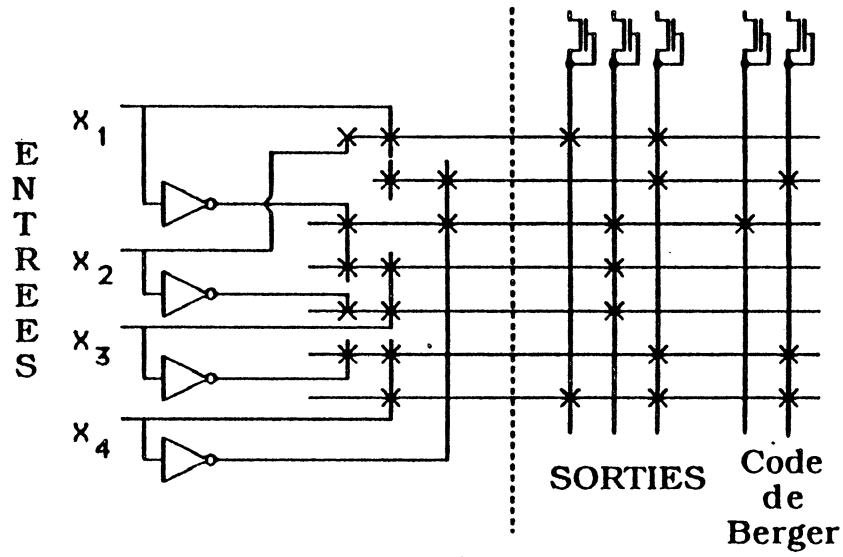


Figure 4.16: Plan d'un PLA SFS à pliage multiple des entrées.

4.5.5 - Le pliage multiple des monômes.

Dans ce type d'optimisation seuls les schémas qui utilisent les codes non ordonnés peuvent être employés. Les lignes sont partagées par plusieurs monômes, ce qui rend impossible le contrôle des monômes après la matrice OU. Le schéma qui utilise des codes non ordonnés est maintenu sans altération après l'optimisation topologique. La figure 4.17 donne la structure du PLA et les sorties nécessaires à la génération du code.

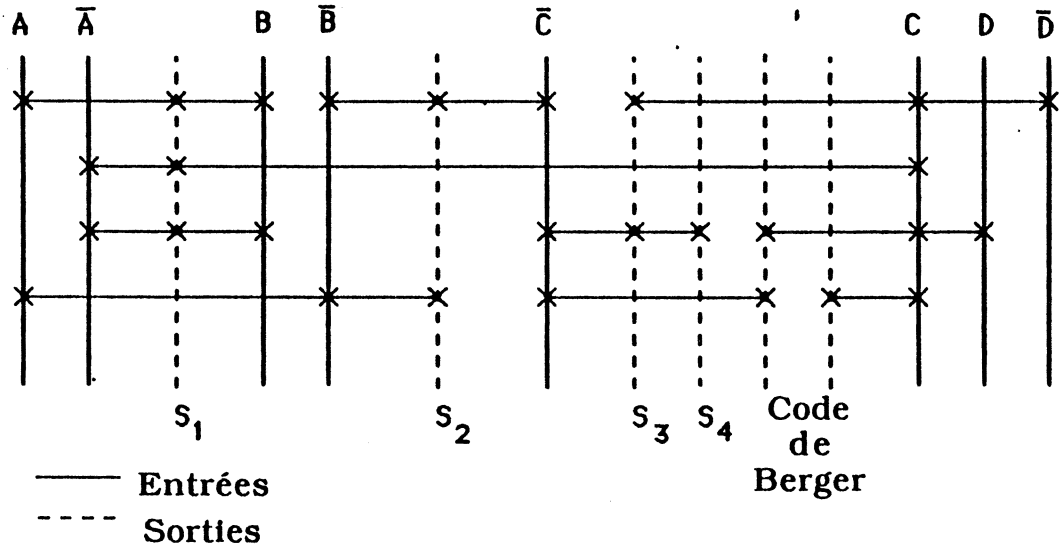


Figure 4.17: Plan d'un PLA SFS à pliage multiple des monômes.

4.5.6 - Le pliage multiple des sorties.

Dans ce type d'optimisation, les monômes ne traversent pas la matrice OU. Ceci est nécessaire pour pouvoir effectuer le routage des sorties. Ainsi, comme dans le cas précédent, il est impossible de tester les monômes après la matrice OU et par conséquent les schémas qui utilisent les codes détectant des erreurs simples ne peuvent pas être employés. Puisque, généralement, dans cette technique d'optimisation les sorties sont placées sur un seul côté du PLA, l'utilisation des codes non ordonnés ne pose pas de problème comme le montre la figure 4.18. Le schéma de test est donc maintenu sans altération après l'optimisation topologique.

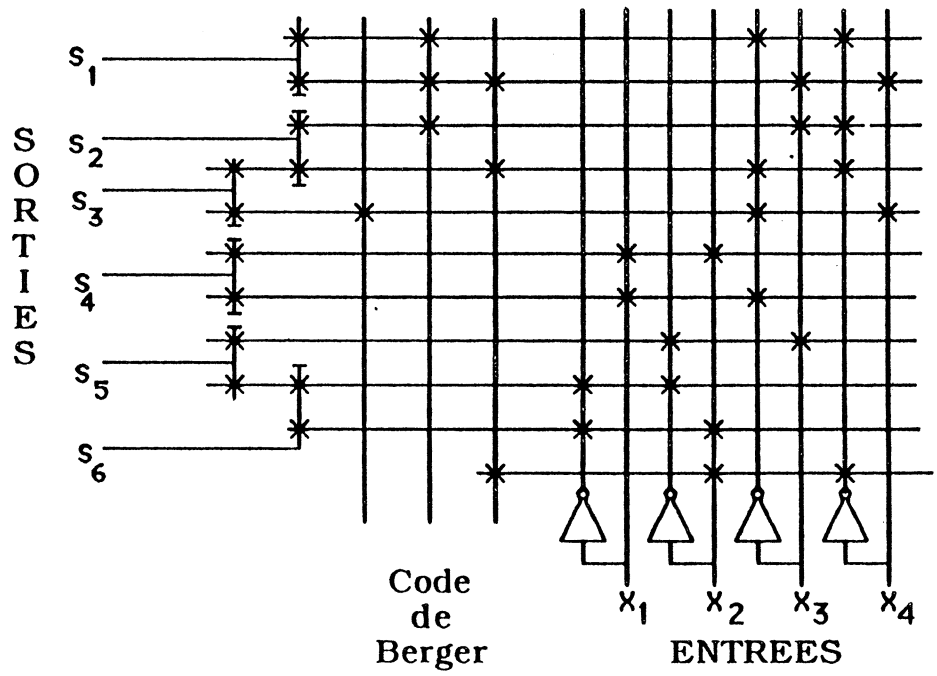


Figure 4.18: Plan d'un PLA SFS à pliage multiple des sorties.

4.6 - CONCLUSION.

L'utilisation des codes non ordonnés pour la détection concurrente des erreurs est parfaitement compatible avec la structure des PLA optimisés topologiquement. Le contrôleur est placé sur la matrice OU ce qui signifie que le schéma de test ne doit être adapté que dans le cas du pliage simple des sorties (les sorties sont placées sur deux côtés du PLA).

Les schémas de test qui utilisent des codes détectant des erreurs simples ne peuvent être employés que dans deux cas de PLAs optimisés: pliage simple des monômes et pliage simple des sorties.

L'application de ces schémas est fortement liée à la programmation du PLA. Nous devons alors vérifier la surface additionnelle pour chaque application. La surface additionnelle peut être un critère de choix du schéma de test quand la structure du PLA permet l'utilisation des deux schémas présentés.

Nous avons aussi proposé un algorithme pour la génération des codes non ordonnés. Cet algorithme offre plusieurs avantages par rapport à celui proposé dans [MAK 82]. Il permet la génération des codes pour des sous-ensembles des sorties et son temps d'exécution est fonction du nombre d'entrées, des sorties et des monômes.

LE TEST UNIFIE DES PLAs

5 - LE TEST UNIFIE DES PLAs.

5.1 - GENERALITES.

Le test unifié, comme nous avons vu précédemment, permet de réaliser tous les tests nécessaires aux circuits. L'intégration des tests hors ligne et en ligne permet de diminuer les redondances tout en disposant d'une couverture de pannes plus importante. Elle permet aussi d'obtenir la vérification des propriétés SFS pour une classe plus large de pannes.

Nous considérons ici que le test en ligne est assuré par le schéma qui utilise les codes non ordonnés. Notre choix est basé sur le fait que le schéma qui utilise des codes de parité ne peut être employé que dans deux cas de PLAs optimisés topologiquement. Dans les exemples qui vont suivre, nous utiliserons le code de Berger (non ordonné). Mais les codes m parmi n et Berger modifié peuvent aussi être appliquées.

Deux méthodes sont proposées pour l'obtention du test hors ligne:

- Le test déterministe et
- Le test exhaustif.

Dans le premier cas, un schéma similaire à celui qui nous avons proposé dans le chapitre 3 est employé (voir figure 3.12). Dans le deuxième cas, les 2^n-1 vecteurs d'entrée (tous les cas possibles excepté 000...0) sont appliqués au PLA. Dans les deux cas, des codes non ordonnés sont utilisés pour la compression des vecteurs de sortie. Par la suite nous détaillerons ces deux schémas.

5.2 - SCHEMA A TEST DETERMINISTE.

Le schéma proposé pour le test hors ligne est décrit par la figure 5.1. La logique rajoutée est composée d'un générateur de vecteurs d'entrée (GVE), d'un décodeur de contrôle des monômes, et de $\lceil \log_2 p+1 \rceil$ sorties (code de Berger des sorties pour le test hors ligne).

Pendant le test hors ligne, le décodeur de contrôle des monômes sélectionne un seul monôme à la fois. Dans cette phase le PLA fonctionne de manière non concurrente. Il est donc nécessaire de générer un code

spécifique pour cette phase puisque généralement les PLAs sont concurrents pendant leur fonctionnement normal. Le PLA doit donc générer deux codes distincts, un pour le test en ligne et l'autre pour le test hors ligne. La génération d'un code non ordonné pour le test hors ligne utilise les monômes déjà existants dans le PLA. Ainsi, nous n'avons qu'à ajouter $\lceil \log_2 p+1 \rceil$ sorties au PLA original codé pour le test en ligne. Si le PLA est non concurrent pendant le fonctionnement normal, le code non ordonné utilisé pour le test en ligne et le test hors ligne est le même, il n'y a donc pas de sortie additionnelle.

La logique additionnelle du générateur de vecteurs d'entrée ainsi que celle du contrôle des monômes est la même que celle employée dans le schéma décrit au chapitre 3. Par contre, la compression des vecteurs de sortie est modifiée. Dans ce qui suit, nous ne détaillerons que la compression des vecteurs de sortie puisque les autres parties ont été déjà détaillées dans le chapitre 3.

La couverture de pannes prend en compte le modèle proposé précédemment et elle est présentée dans l'annexe 5.

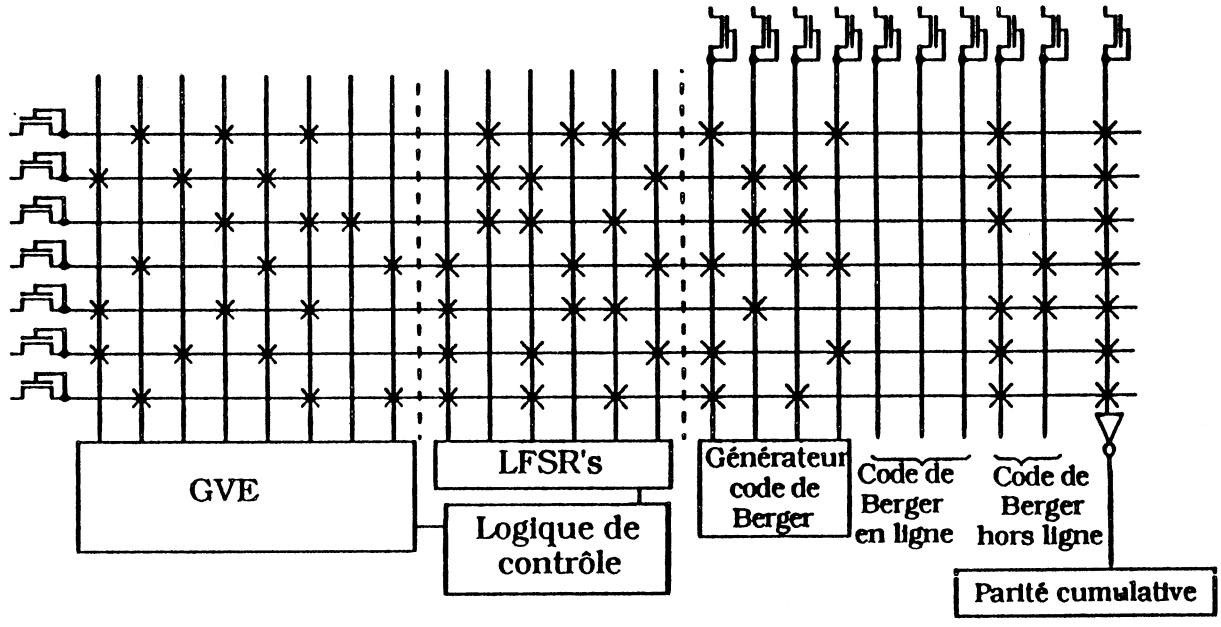


Figure 5.1: Schéma proposé.

La figure 5.2 montre les colonnes (entrées et sorties) ajoutées au PLA pour la compression des vecteurs de sortie.

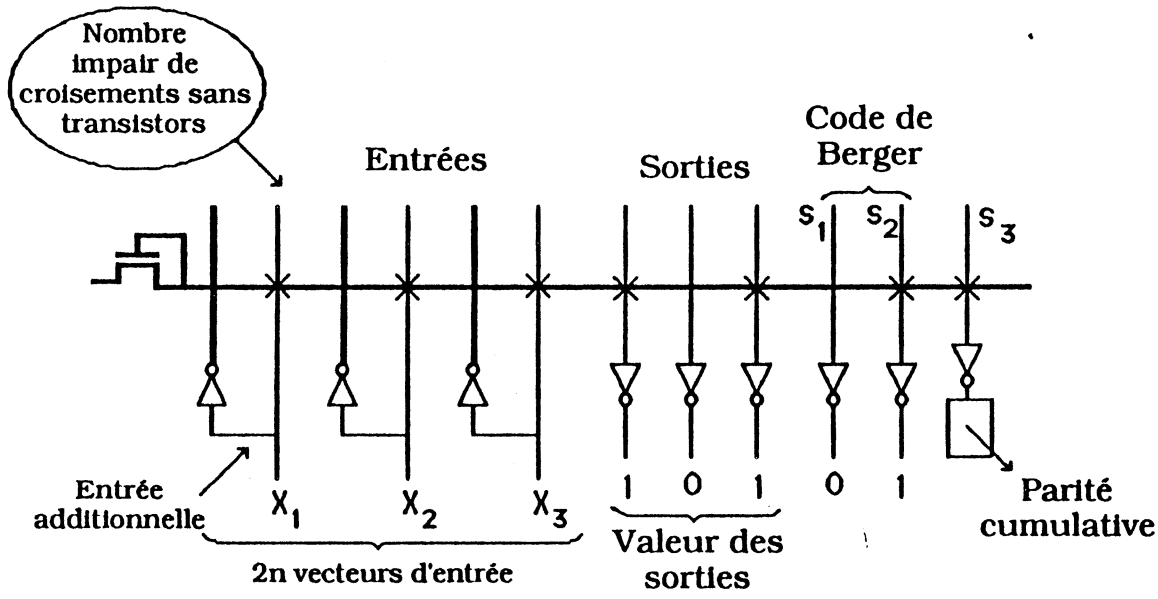


Figure 5.2: Compression des vecteurs de sortie.

Propriété 5.1:

Il existe un nombre impair de croisements sans transistor sur chaque monôme, dans la matrice ET.

Cette propriété est assurée par une entrée additionnelle. Quand il existe sur le monôme original un nombre pair de croisements sans transistor, nous plaçons un transistor à l'intersection du monôme et de l'entrée additionnelle.

Les deux sorties additionnelles (S_1 et S_2) dans la matrice OU génèrent le code de Berger pour le test hors ligne.

La sortie additionnelle S_3 permet de connaître la valeur du monôme.

Le contrôle des entrées et des monômes dans la matrice ET permet de tester un croisement à chaque cycle. Le décodeur de contrôle de monôme sélectionne un monôme (ceci veut dire que tous les monômes sont mis à la masse excepté celui sélectionné). La valeur du monôme sélectionné va dépendre de l'existence ou non d'un transistor sur le croisement formé par la ligne d'entrée, qui est sélectionnée à chaque cycle, et le monôme sélectionné. Les vecteurs d'entrée décrits dans la figure 5.3, sont appliqués au PLA pour chaque monôme sélectionné.

X_1	X_2	.....	X_{n-1}	X_n	C	$\bar{C}$
1	0	.....	0	0	1	0
0	1	.....	1	1	0	1
0	1	.....	0	0	1	0
1	0	.....	1	1	0	1
0	0	.....	1	0	1	0
1	1	.....	0	1	0	1
0	0	.....	0	1	1	0
1	1	.....	1	0	0	1

Chaque monôme

Figure 5.3: Vecteurs d'entrée.

Quand il existe un transistor au croisement sélectionné, tous les monômes sont mis à la masse, donc le code de Berger ne peut pas être généré par le PLA. La sortie additionnelle S_3 permet de détecter cette condition (dans ce cas la sortie S_3 est aussi à "0"). Ainsi le code de sortie peut être généré par la sortie S_3 comme le montre la figure 5.4. Les transistors de passage, contrôlés par la valeur complémentaire du monôme, mettent à la masse les sorties convenables du code de Berger et génèrent de cette manière le bon code.

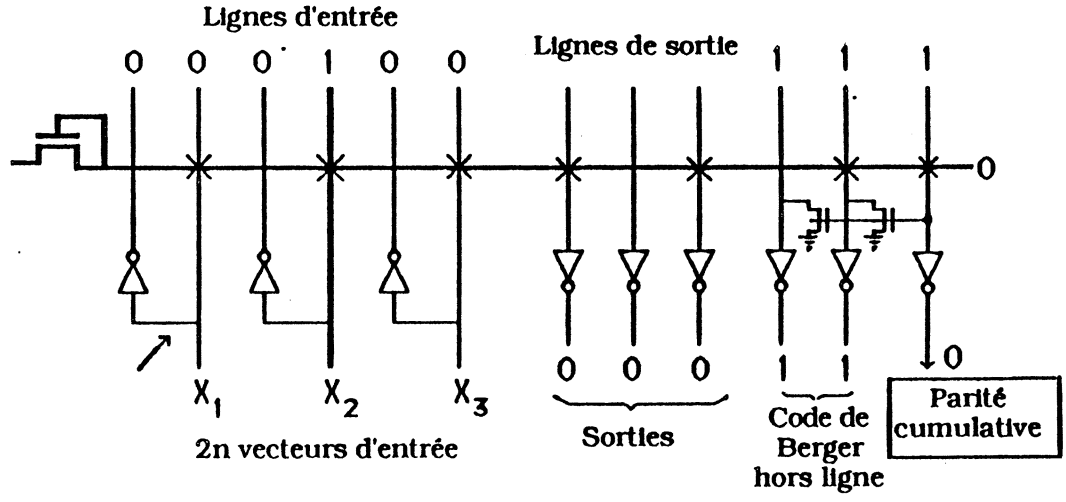


Figure 5.4: Compression des vecteurs de sortie.

Quand il n'y a pas de transistor au croisement sélectionné, le monôme reste à "1", donc toutes les sorties en relation avec ce monôme sont aussi à "1". Dans ce cas le code de Berger des sorties peut être généré par le PLA permettant ainsi le contrôle de la matrice OU, comme le montre la figure 5.5.

Chaque monôme est composé d'un nombre impair de croisements sans transistor dans la matrice ET (propriété 5.1). Par conséquent le monôme sera à 1 un nombre impair de fois (voir figure 5.5), donc la parité cumulative doit être impaire après l'application des 2n vecteurs d'entrée. Ceci permet de tester la matrice ET.

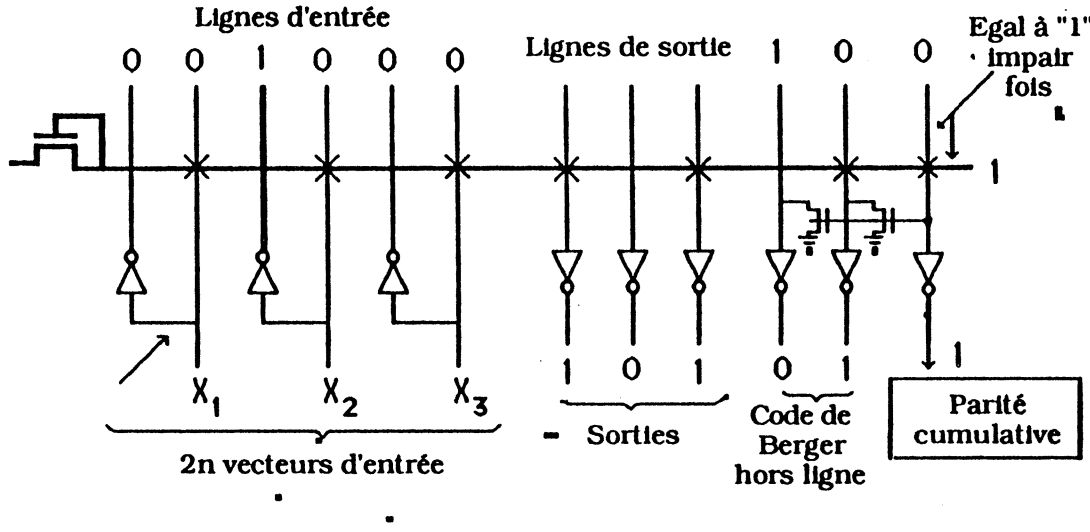


Figure 5.5: Compression des vecteurs de sortie.

5.3 - SCHEMA A TEST EXHAUSTIF.

Le test exhaustif consiste à générer les 2^n (ou les 2^n - 1) vecteurs possibles pour les n variables d'entrée. Dans le schéma que nous proposons, cette génération est effectuée par des LFSRs utilisant un polynome caractérisque primitif. Les vecteurs de sortie sont contrôlés par un code non ordonné. En effet, le test en ligne et le test hors ligne utilisent le même principe. La phase hors ligne consiste à générer les vecteurs d'entrée qui seront comprimés par le même contrôleur que celui utilisé dans la phase en ligne.

Un PLA qui contient n entrées n'utilise pas forcement toutes les 2^n combinaisons possibles pendant son fonctionnement normal. Ceci veut dire qu'il peut exister des vecteurs d'entrée dont les fonctions de sortie ne sont pas définies (faux mintermes - voir définition D 5.1). Dans ce cas aucun monôme n'est sélectionné. Il faut assurer que le PLA soit capable de générer les codes correspondant à chaque vecteur d'entrée, puisque les 2^n - 1 (tous les cas possibles excepté 000...0) sont appliqués au PLA. Il faut donc ajouter des monômes pour la génération du codage des faux mintermes. Cet ajout de monômes permet d'obtenir une couverture complète du PLA i.e. chacun des

2^n - 1 mintermes d'un PLA sélectionne au moins un monôme. Il existe un code de sortie pour chacun des 2^n - 1 vecteurs d'entrée.

DEFINITION D 5.1

Un minterme n_i pour une fonction simple de sortie est appelé minterme vrai si f = 1 et faux minterme si f = 0 quand n_i est appliqué.

DEFINITION D 5.2

Un minterme n_i pour une fonction multiple de sortie est appelé minterme vrai si au moins une fonction de sortie est à "1" quand n_i est appliqué.

DEFINITION D 5.3

Un minterme n_i pour une fonction multiple de sortie est appelé minterme faux si toutes les fonctions de sortie sont à "0" quand n_i est appliqué.

La figure 5.6 donne un exemple d'application de ce schéma de test où p_1 et p_{11} sont des monômes ajoutés pour générer le code de Berger des faux mintermes. Le tableau de Karnaugh est décrit figure 5.7. Le nombre de monômes qui doivent être ajoutés dépend de la programmation du PLA.

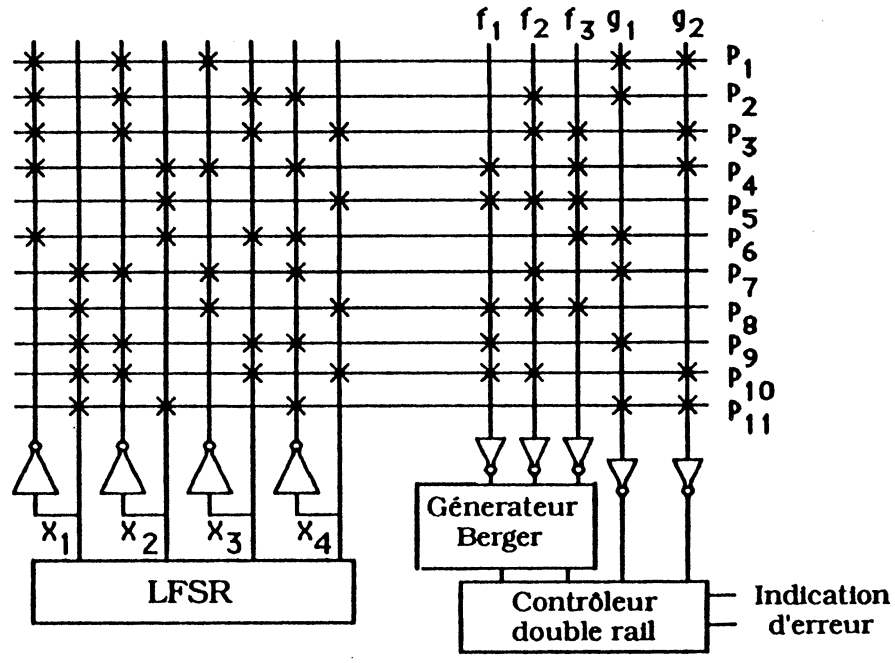


Figure 5.6: Schéma à test exhaustif.

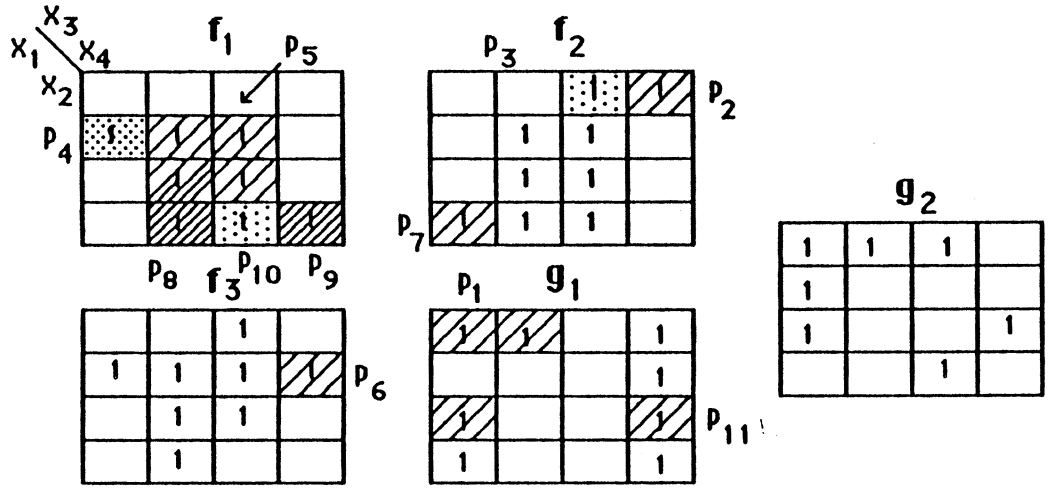


Figure 5.7: Tableau de Karnaugh.

Le LFSR est composé de n bascules D, qui sont les unités de délai, et de portes OU-exclusifs. La figure 5.8 donne la réalisation d'un LFSR. Son polynome caractéristique f (x) peut être défini par:

$$f(x) = 1 + h_1x + h_2x^2 + \ldots + h_{n-1}x^{n-1} + h_nx^n.$$

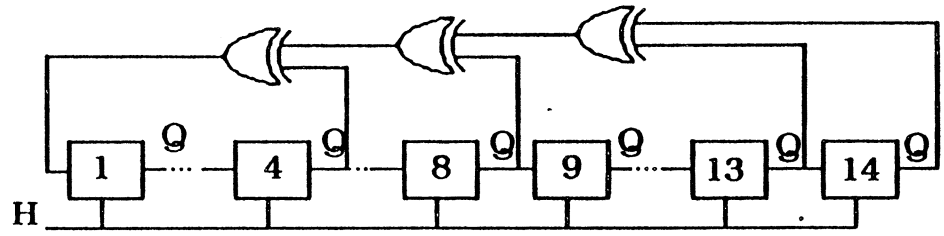


Figure 5.8: LFSR 14 bits.

Où h_i indique la présence possible d'une connexion de réaction qui vient de la sortie de chaque bascule. Si $h_i = 1$, il existe une réaction venant de la bascule i. Si $h_i = 0$ la bascule i n'est pas connecté. Le tableau 5.1 donne les connexions nécessaire pour la génération d'un polynome caractéristique primitif (i.e. un polynome qui génère $2^n - 1$ vecteurs).

Portes	Vecteurs	Réalimentation
2	3	1 X 2
3	7	2 X 3
4	15	3 X 4
5	31	3 X 5
6	63	5 X 6
7	127	4 X 7
8	255	4 X 5 X 6 X 8
9	511	5 X 9
10	1023	7 X 10
11	2047	9 X 11
12	4095	6 X 8 X 11 X 12
13	8191	9 X 10 X 12 X 13
14	16383	4 X 8 X 13 X 14
15	32767	14 X 15

Tableau 5.1: Réactions pour la génération d'un polynome caractéristique primitif.

La surface d'un LFSR peut être exprimée par (en négligeant le(s) porte(s) EXOR):

$$S_{LFSR} = 720\ n \quad \lambda^2 \qquad \text{(eq. 5.1)}$$

Il est clair que si n (nombre des entrées) est grand le test exhaustif devient impraticable dans un temps raisonnable. Ce schéma de test ne peut être employé que dans des PLAs qui contiennent un nombre relativement petit d'entrées. Si le PLA contient un nombre important d'entrées, l'approche déterministe doit être employée.

5.4 - COMPARAISON DES DEUX SCHEMAS.

La comparaison entre les deux schémas est basée sur trois critères: couverture des fautes, surface de la logique additionnelle et longueur du test.

La longueur du test est le facteur restrictif principal du test exhaustif, puisqu'il ne peut être appliqué que dans les PLAs ayant un nombre relativement petit d'entrées.

Le test déterministe génère les sorties codées, pour le test hors ligne, sans addition de monômes. Par contre quand le PLA est concurrent il faut ajouter les sorties nécessaires à la génération du codage ($\lceil \log_2 p+1 \rceil$ sorties). La couverture complète nécessaire à l'application du test exhaustif peut entraîner l'addition de plusieurs monômes. Le nombre de monômes ajouté est dépendant de la programmation du PLA. Ainsi la surface ajoutée doit être analysée pour chaque application. La logique nécessaire pour la génération des vecteurs d'entrée est plus petite dans le cas du test exhaustif. Ainsi, si le PLA original a déjà une couverture complète des mintermes, la logique additionnelle pour le test exhaustif est plus petite que celle du test déterministe.

Le test exhaustif assure la détection des fautes simples et multiples qui sont propagées comme des erreurs unidirectionnelles. En effet le PLA est testé de la même manière dans les deux phases de test (en ligne et hors ligne). Le test déterministe détecte toutes les fautes simples et presque toutes les fautes multiples. Les fautes multiples qu'il ne détecte pas sont celles de la matrice ET où le nombre de croisements est maintenu impair. Comme cette faute se propage comme une erreur unidirectionnelle, elle sera détectée pendant le test en ligne. Par conséquent, l'association des deux techniques de test, en ligne et hors ligne, donne une couverture de toutes les fautes simples et multiples. La couverture des fautes du test déterministe est

plus grande que celle du test exhaustif.

Le tableau 5.2 résume les caractéristiques de chaque schéma de test. Nous pouvons conclure que le test déterministe doit être employé pour les PLAs ayant un nombre important d'entrées. Par contre pour les PLAs ayant un petit nombre d'entrées les deux méthodes peuvent être utilisées. Les critères de choix sont:

- La couverture des fautes qui est meilleure pour le test déterministe et
- La surface additionnelle qui peut être moins importante pour le test exhaustif.

Schéma / Critère	Deterministe	Exhaustive
Couverture des fautes	Toutes les fautes simples et presque toutes les fautes multiples dues à un collage, court-circuit et crosspoint	Toutes les fautes simples et multiples qui sont propagées par des erreurs unidirectionnelles
Surface additionnelle	$2176n + 128m \log_2 m$	720m
Vecteurs de test	2nm	2^n

Tableau 5.2: Comparaison entre les schémas proposés.

Les deux schémas peuvent être employés dans tous les cas de PLAs optimisés topologiquement. Les détails de cette application sont donnés dans la suite.

5.5 - APPLICATION AUX PLAs OPTIMISES TOPOLOGIQUEMENT.

Etant donné que les schémas utilisent le même principe que les méthodes décrites précédemment, leur application dans les PLAs optimisés topologiquement présente les mêmes contraintes.

Le test en ligne est associé à l'observation des vecteurs de sortie, comme nous l'avons vu dans le chapitre précédent. Le contrôleur est placé sur la matrice OU. Ainsi les changements de structure de cette matrice entraînent aussi des changements sur le schéma de test en ligne. Ces contraintes ont été discutées dans le chapitre précédent par conséquent nous ne détaillerons pas

l'application de ce schéma.

De même, le test hors ligne agit sur la matrice ET puisque il est associé à la génération des vecteurs d'entrée. Les changements dans cette matrice entraînent des changements dans le schéma de test hors ligne. Le test déterministe a déjà été discuté dans le chapitre 3. Les détails de l'application du test exhaustif, dans les matrices ET optimisées, ainsi qu'une comparaison entre les deux schémas seront donnés par la suite.

5.5.1 - Le pliage simple des entrées.

Cette technique d'optimisation topologique agit sur la matrice ET ce qui entraîne que seule la logique nécessaire au test hors ligne doit être adaptée à la nouvelle structure du PLA.

La figure 5.9 donne la structure du PLA après l'optimisation topologique dans le cas de l'utilisation du test déterministe. Les détails de cette application sont donnés dans le chapitre 3 et peuvent être résumés par le graphique 5.1.

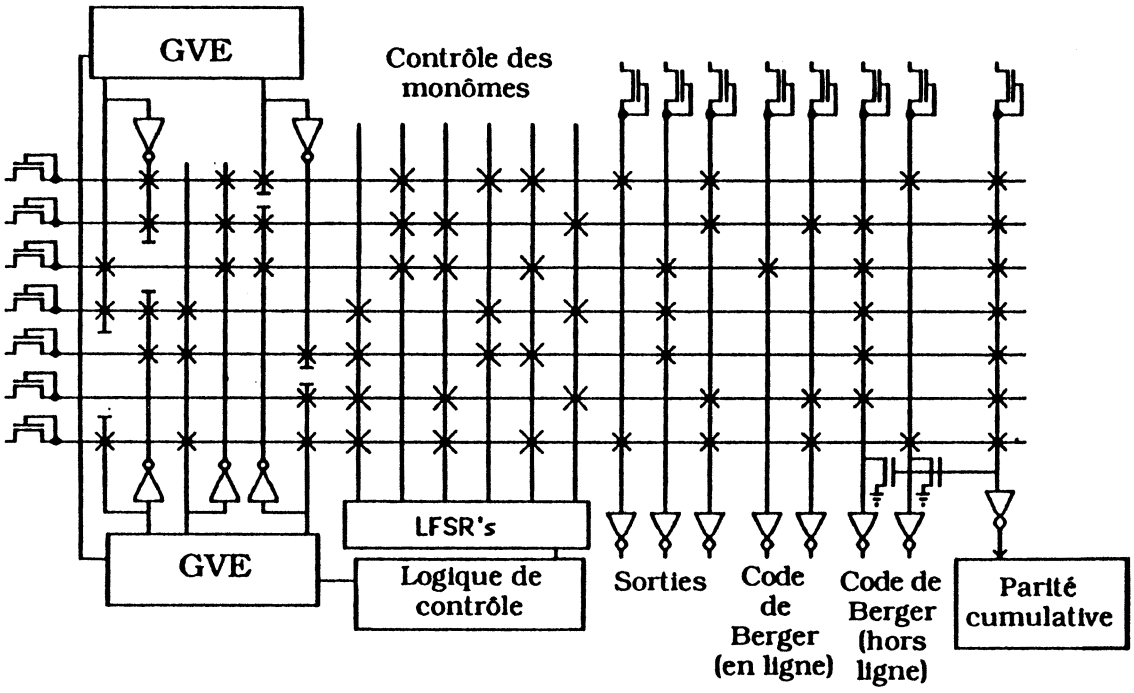
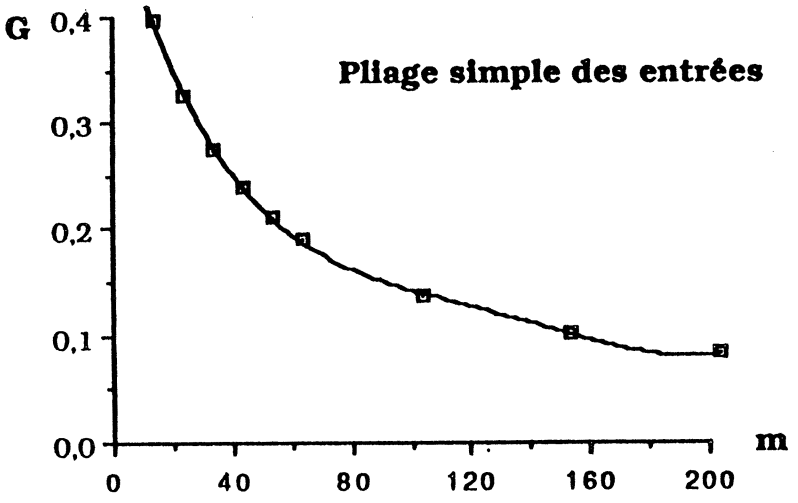


Figure 5.9: Plan d'un PLA autotestable à pliage simple des entrées.



Graphique 5.1: Gain de surface en fonction du nombre de monômes.

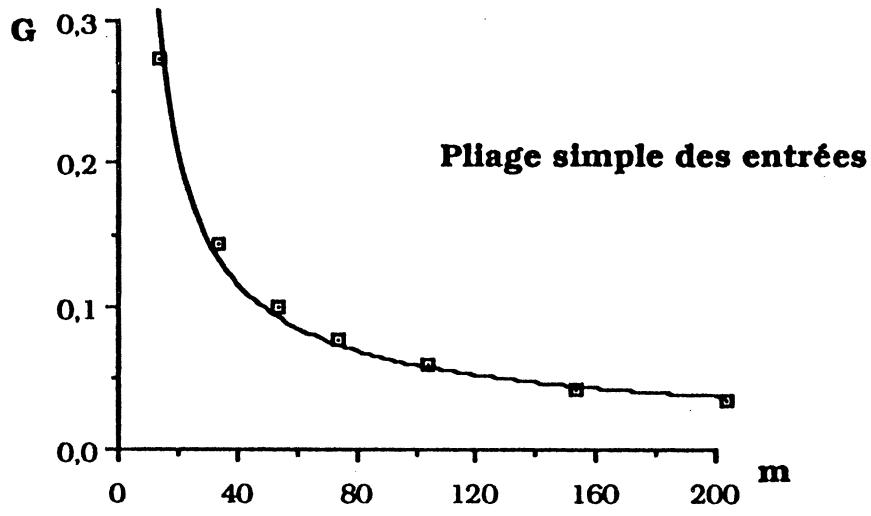
Pour un schéma utilisant le test exhaustif, la surface du PLA avec sa logique additionnelle avant l'optimisation topologique est donnée par l'équation suivante:

$$S_{AO} = 16n * 8m + 720n\ \lambda^2 \quad \text{(eq. 5.2)}$$

La surface du PLA avec la logique additionnelle après l'optimisation topologique, pour un schéma utilisant le test exhaustif, est donnée par l'équation suivante:

$$S_{SO} = [(16n * 8m) * (1 - G) + 2(1 - G) * (720n)]\ \lambda^2 \quad \text{(eq. 5.3)}$$

L'équilibre entre les deux surfaces est donné par le graphique 5.2. Il montre que le schéma est applicable même dans les petites PLAs. Sachant que le test exhaustif peut nécessiter l'ajout de plusieurs monômes, la surface nécessaire au test déterministe peut être plus petite.



Graphique 5.2: Gain de surface en fonction du nombre de monômes.

5.5.2 - Le pliage simple des sorties.

Cette optimisation topologique n'agit que sur la matrice OU donc seule l'observation des sorties doit être modifiée. La génération des vecteurs d'entrée pour le test hors ligne est maintenue sans aucune modification. Par contre la génération des codes de sortie, pour le test en ligne, est appliquée indépendamment pour chaque côté ou pour l'ensemble des sorties, comme nous l'avons vu dans le chapitre 4. La compression des vecteurs de sortie, pendant le test hors ligne, doit utiliser le même principe que celui du test en ligne. Quand le code est généré indépendamment pour chaque côté pour le test en ligne, il doit aussi être généré pour chaque côté lors du test hors ligne. Par conséquent le nombre de bits de contrôle nécessaires au test hors ligne est doublé. La figure 5.10 donne la structure du PLA après l'optimisation topologique.

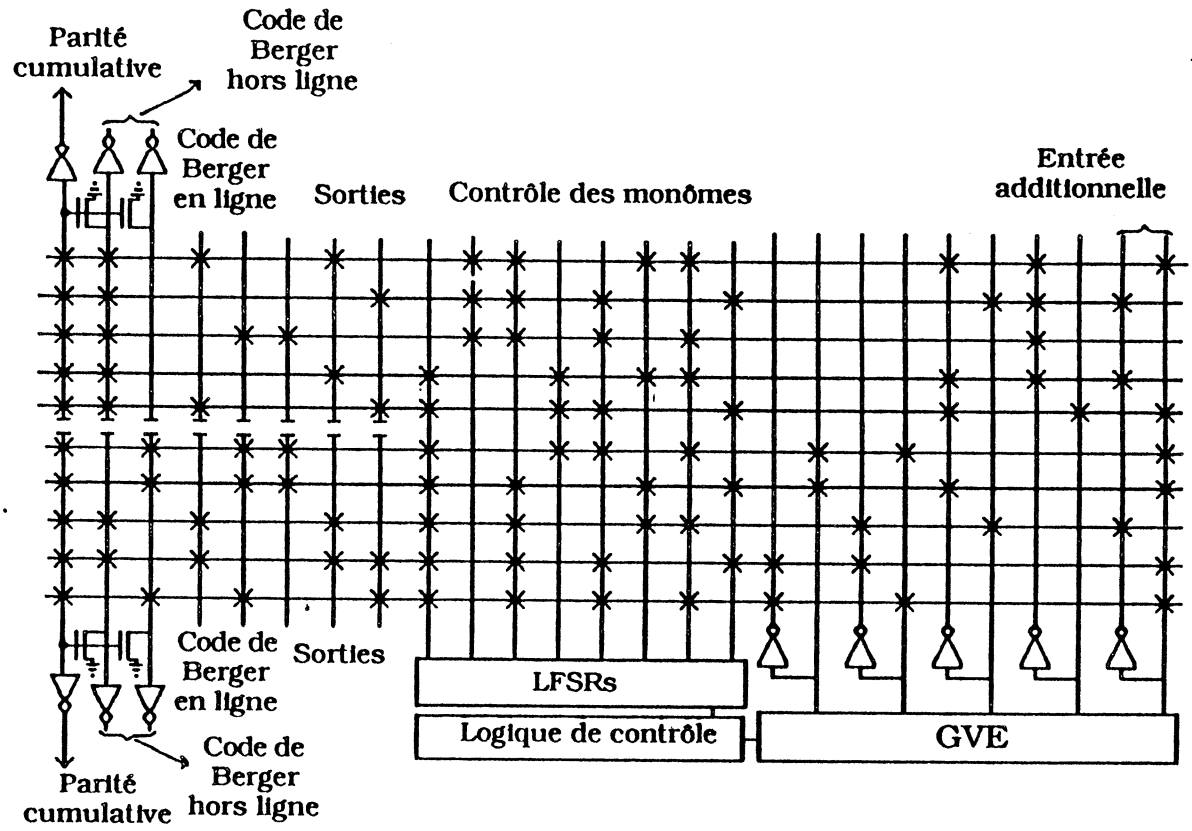


Figure 5.10: Plan d'un PLA autotestable à pliage simple des sorties.

5.5.3 - Le pliage simple des monômes.

Dans cette technique d'optimisation les sorties sont placées sur deux côtés du PLA. Les codes de Berger peuvent être générés soit séparément pour chaque côté soit globalement pour toutes les sorties. Dans le premier cas, le nombre de monômes nécessaires à la génération du code de Berger ainsi que le nombre de bits de contrôle sont augmentés. Cette méthode peut être employée quand le nombre de monômes supplémentaires pour la génération du code de Berger (due à l'optimisation topologique) est plus petit que le nombre de monômes brisés. Dans le deuxième cas, la valeur partielle du code de Berger d'un côté doit être ramenée sur l'autre. Cette méthode peut être employée quand le nombre de monômes brisés est plus grand que le nombre de bits de contrôle reliés $\lceil \log_2 p'+1 \rceil$ (Où p' est le plus petit nombre de sorties parmi les deux côtés).

Le test hors ligne n'est pas modifié lorsque le test exhaustif est employé. Quand le test déterministe est utilisé, les bits de contrôle doivent aussi être reliés. Ceci entraîne une augmentation de $\lceil \log_2 p'+1 \rceil$ bits de

contrôle à relier. Par conséquent le nombre de monômes brisés doit être plus grand que $(2 * \lceil \log_2 p'+1 \rceil)$.

La figure 5.11 montre la structure d'un PLA avec sa logique de test après optimisation topologique.

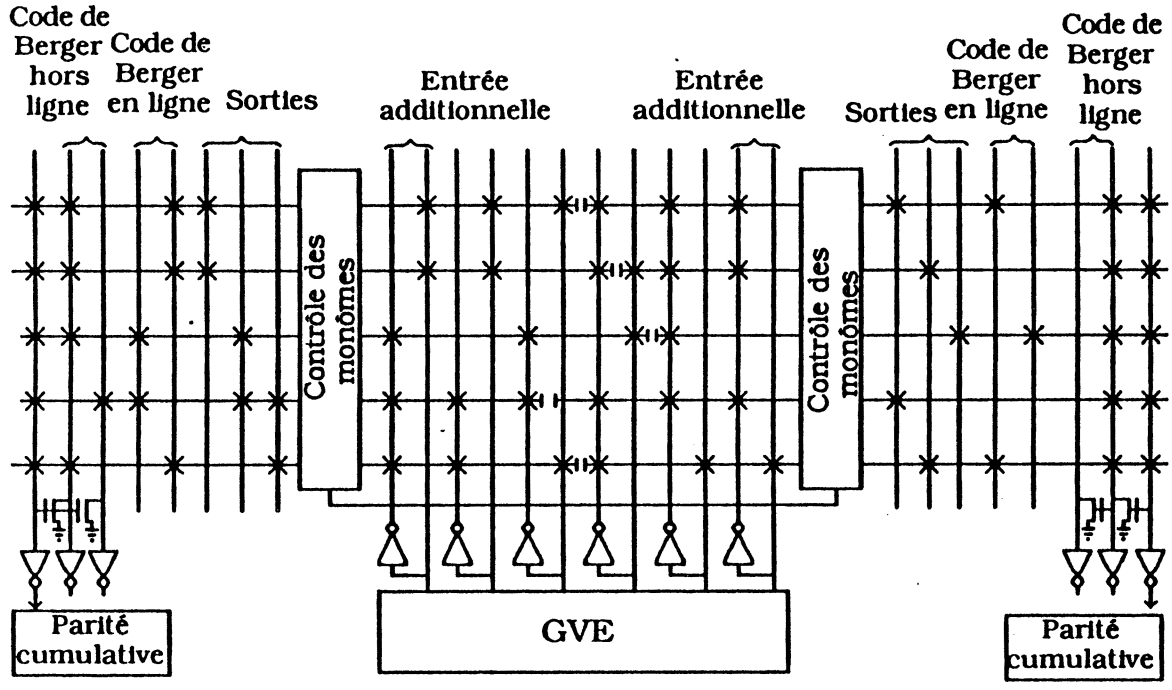
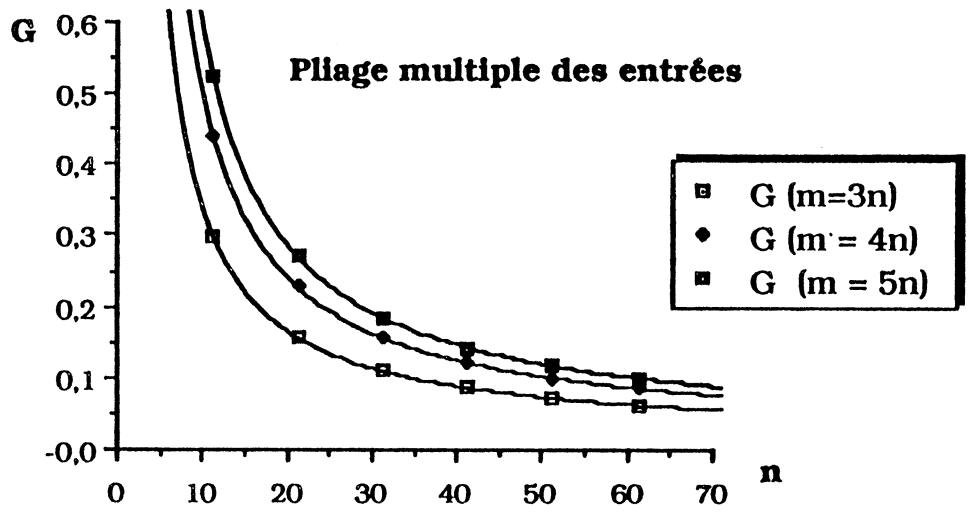


Figure 5.11: Plan d'un PLA autotestable à pliage simple des monômes.

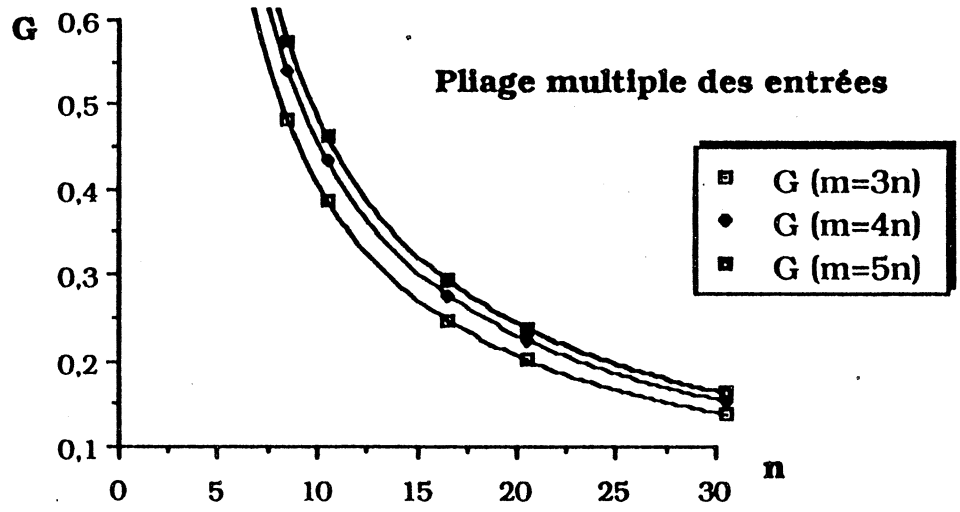
5.5.4 - Le pliage multiple des entrées.

La structure du PLA après le pliage multiple des entrées, décrit par la figure 5.12, présente des entrées parallèles aux monômes. Les contraintes de surface pour l'application du test déterministe sont les mêmes que celles décrites dans le chapitre 3. Les graphiques 5.3 et 5.4 résument respectivement l'application du schéma de test déterministe et de test exhaustif.

Le test en ligne est maintenu sans altération puisque le schéma de test n'agit que sur la matrice ET.



Graphique 5.3: Gain de surface en fonction des entrées pour différents nombres de monômes (test déterministe).



Graphique 5.4: Gain de surface en fonction des entrées pour différents nombres de monômes (test exhaustif).

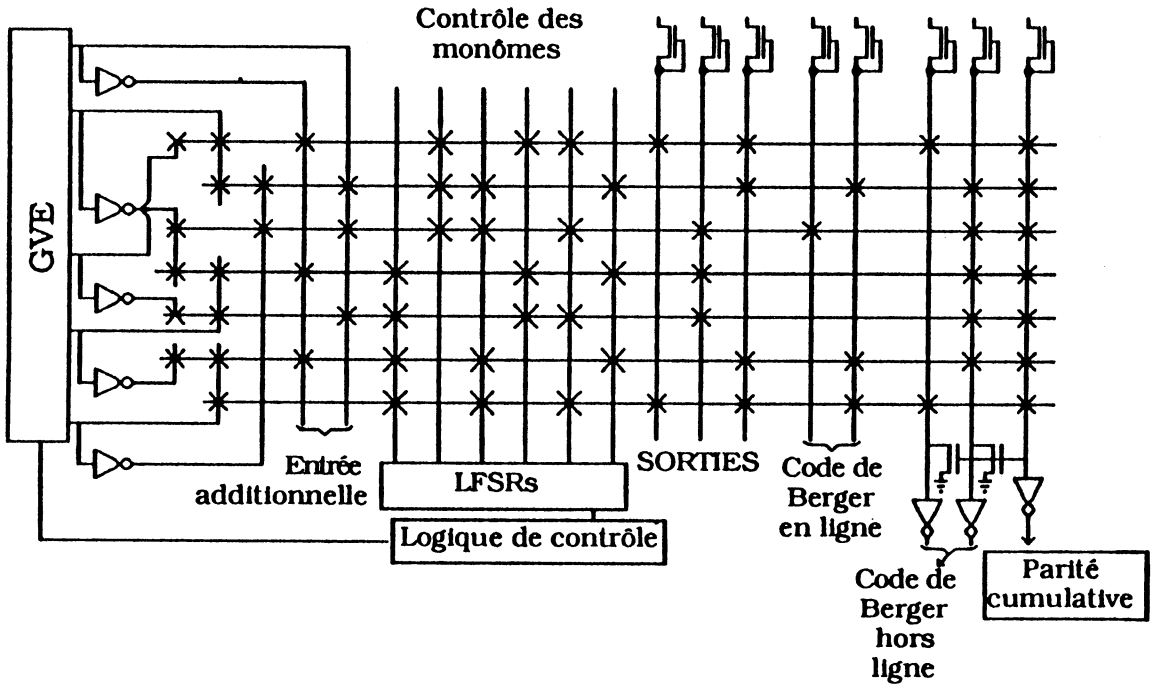


Figure 5.12: Plan d'un PLA autotestable à pliage multiple des entrées.

5.5.5 - Le pliage multiple des sorties.

Dans cette technique d'optimisation, la structure de la matrice OU est modifiée mais généralement, les sorties sont placées sur un seul côté du PLA. Ainsi le test en ligne est maintenu sans altération après l'optimisation topologique. Le test hors ligne est aussi maintenu sans altération après cette optimisation. La figure 5.13 décrit le schéma de test.

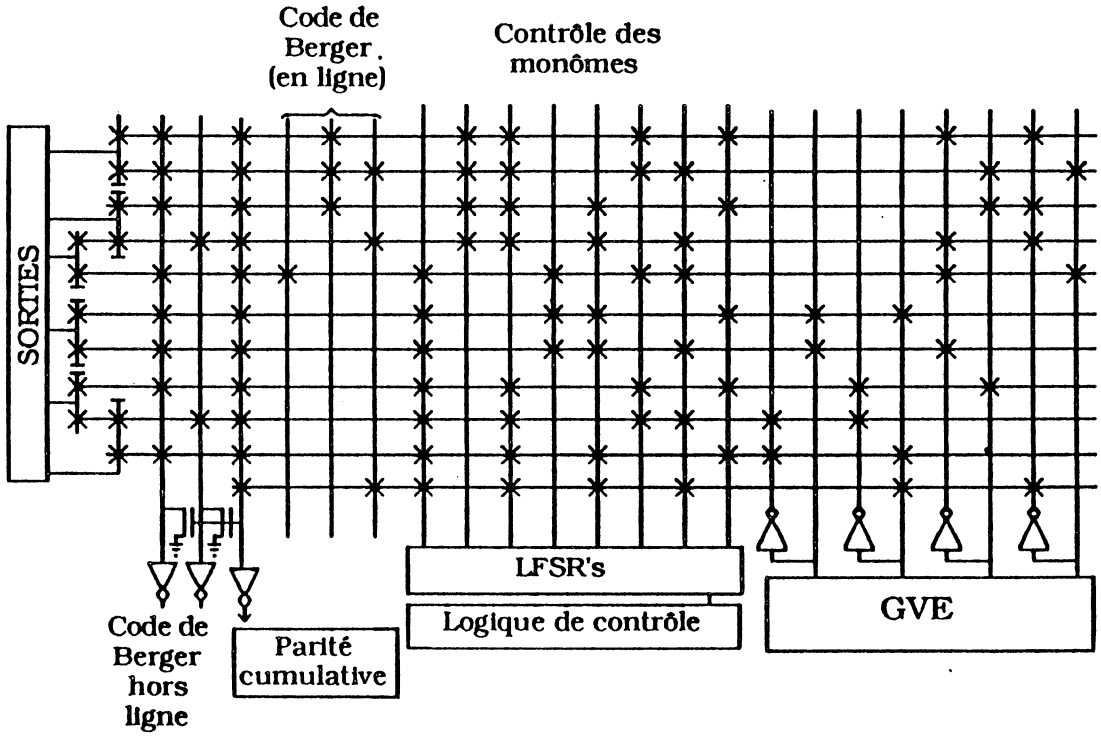


Figure 5.13: Plan d'un PLA autotestable à pliage multiple des sorties.

5.6 - CONCLUSION.

Le test en ligne est assuré par la méthode utilisant des codes non ordonnés et le test hors ligne utilise la même méthode pour la compression des vecteurs de sorties. Deux méthodes sont proposées pour la génération des vecteurs d'entrée pendant le test hors ligne: Le test exhaustif et le test déterministe. Le test exhaustif ne peut être appliquée qu'aux PLAs ayant un petit nombre d'entrées et le PLA doit être capable de générer le codage des 2^n-1 vecteurs. Ceci peut augmenter le nombre de monômes nécessaires à la génération du codage des sorties. Le test déterministe utilise le même principe que celui présenté par le schéma du chapitre 3. Dans ce cas il n'y a pas de facteur restrictif mais son utilisation nécessite l'addition de $\lceil \log_2 p+1 \rceil$ sorties pour la génération du code non ordonné pendant le test hors ligne.

Puisque les schémas de test utilisent le même principe qui celui des méthodes décrites précédemment leur application aux PLAs optimisés topologiquement est soumise aux mêmes contraintes.

CONCLUSION

6 - CONCLUSION.

L'utilisation des méthodes d'optimisation topologique rend la structure des PLAs plus souple. Ceci favorise une utilisation plus universelle des PLAs dans la conception à très haute intégration. Dans ce travail, nous avons effectué une étude complète de la testabilité des PLAs. Pour les trois classes de test (hors ligne, en ligne et unifié) nous avons proposé des schémas de test dont la compatibilité avec les PLAs optimisés a été étudiée. Par la suite les principales conclusions de chaque méthode de test sont données.

En ce qui concerne le test hors ligne, le schéma de test proposé présente plusieurs avantages par rapport aux autres schémas publiés dans la littérature. La génération des vecteurs d'entrée est facilement obtenue par des registres à décalage associés à des portes EXNOR. Le contrôle des monômes par l'ajout d'entrées présente la même structure que la matrice ET et son implantation ne pose donc pas de problème. La méthode de compression des vecteurs de sortie utilise le code de parité. Le test de la matrice OU consiste à vérifier la parité de chaque vecteur de sortie et le test de la matrice ET est effectué par la parité cumulative. Ainsi, en utilisant la méthode la plus simple de compression, nous obtenons une très bonne couverture de pannes. Le schéma présente néanmoins une meilleure couverture de panne par rapport aux schémas utilisant la même méthode de compression, comme par exemple celui proposé dans [TRE 85]. La surface additionnelle est plus petite que celles proposées dans les autres schémas déjà publiés [TRE 85] et [SAL 85]. La longueur du test est proportionnelle au nombre d'entrées et au nombre de monômes (2nm). Nous pouvons donc conclure que le schéma de test proposé apporte une amélioration vis-à-vis des schémas existants dans la littérature.

La compatibilité de ce schéma avec les PLAs optimisés a été étudiée dans le chapitre 3. La structure des PLAs après l'optimisation topologique est la principale difficulté pour l'application des schémas de test. Les entrées, les sorties ou les monômes placés sur deux côtés distincts entraînent l'implantation de la circuiterie de test sur deux côtés. Cette division de la circuiterie de test augmente la surface additionnelle et cette augmentation peut être plus grande que la surface gagnée avec l'optimisation topologique. Ces contraintes ont été analysées et sont résumées dans le tableau 6.1. Chaque cas d'optimisation topologique est décrit séparément mais il reste

valable pour des combinaisons compatibles, comme par exemple pliage simple d'entrées et pliage multiplé des sorties.

Pliage / Elément	Simple	Multiple
Entrées	Applicable aux PLAs de grande et moyenne taille	Applicable aux PLAs ayant plus de 15 entrées (m > 2n)
Sorties	Applicable aux PLAs de grande et moyenne taille	Applicable aux PLAs ayant plus de 20 sorties (m > 2p)
Monômes	Applicable sans restriction	Non applicable

Tableau 6.1: Application du schéma de test hors ligne.

En ce qui concerne le test en ligne, l'application de deux schémas de test aux PLAs optimisés topologiquement a été analysée. Le schéma utilisant des codes non ordonnés pour la détection concurrente des erreurs est parfaitement compatible avec la structure des PLA optimisés. Le contrôleur est placé sur la matrice OU ce qui signifie que le schéma de test ne doit être adapté que dans le cas du pliage simple des sorties (les sorties sont placées sur deux côtés du PLA). Par contre les schémas de test qui utilisent des codes détectant des erreurs simples ne peuvent être employés que dans deux cas de PLAs optimisés: pliage simple des monômes et pliage simple des sorties.

Nous avons proposé un nouvel algorithme pour la génération des codes non ordonnés. Cet algorithme offre plusieurs avantages par rapport à celui proposé dans [MAK 82]. Il permet la génération des codes pour des sous ensembles des sorties et son temps d'exécution est raisonnable et est fonction du nombre d'entrées, de sorties et de monômes. Plusieurs exemples ont été traités.

En ce qui concerne le test unifié, le test en ligne est assuré par la méthode utilisant des codes non ordonnés et le test hors ligne utilise la même méthode pour la compression des vecteurs de sortie. Deux méthodes sont proposées pour la génération des vecteurs d'entrée pendant le test hors

ligne: le test exhaustif et le test déterministe. Le test exhaustif ne peut être appliqué qu'aux PLAs ayant un petit nombre d'entrées et le PLA doit être capable de générer le codage de tous les vecteurs possibles (i.e. 2^n - 1). Ceci peut augmenter le nombre de monômes nécessaires à la génération du codage des sorties. Le test déterministe utilise le même principe que celui présenté par le schéma du chapitre 3. Dans ce cas il n'y a pas de facteur restrictif mais son utilisation nécessite l'addition de $\lceil \log_2 p+1 \rceil$ sorties pour la génération du code non ordonné pendant le test hors ligne. La couverture des pannes du test déterministe est améliorée.

Puisque les schémas de test utilisent le même principe que celui des méthodes décrites précédemment, leur application aux PLAs optimisés topologiquement est soumise aux mêmes contraintes. Le tableau 6.2 résume l'application des deux schémas de test hors ligne aux PLAs optimisés (test unifié).

Pliage \ Test		**Déterministe**	**Exhaustif**
Simple	Entrées	Applicable aux PLAs de grande et moyenne taille	Applicable sans restrictions
	Sorties	Applicable sans restrictions	Applicable sans restrictions
	Monômes	Applicable sans restrictions	Dépend de la programation du PLA
Multiple	Entrées	Dépend du gain de surface obtenu	Dépend du gain de surface obtenu
	Sorties	Applicable sans restrictions	Applicable sans restrictions
	Monômes	Non applicable	Non applicable

Tableau 6.2: Application du schéma de test hors ligne (test unifié).

BIBLIOGRAPHIE

BIBLIOGRAPHIE.

[AND 71] ANDERSON D.A
"Design of self-checking digital networks using coding techniques". Urbana, CSL University of Illinois, Sep. 1971 (report 527).

[BOZ 84] BOZORGUI-NESBAT S. et McCLUSKEY E.J.,
"Lower overhead for testability of programmable logic arrays." Proc. IEEE International test conference -1984 - pp. 856-865.

[CHA 78] C.W.CHA
"Testing strategies for PLAs."
Proc. 15th Design Automation Conference - Las Vegas, Nevada- June 1978

[CAR 68] CARTER W.C., SCHNEIDER P.R.
"Design of dynamically checked computers". IFIP Congress, Edinburgh 1968, Inf. Processing 68, Amsterdam, North Holland, 1969.

[CHE 85] CHEN C.Y., FUCHS W.K., ABRAHAM J.A.
"Efficient concurrent error detection in PLAs and ROMs". ICCD, Port Chester, N.Y., Oct. 1985.

[CHU 84] CHUQUILANQUI BERNAOLA S.H.
"Une nouvelle aproche pour l'optimisation topologique et l'automatisation du dessin des masques des PLA's complexes."
Thesis - INPG - Grenoble France - october 1984.

[CHU 83a] CHUQUILANQUI BERNAOLA S.H. et PEREZ SEGOVIA T.
"A VLSI topological optimization strategy applied to PLA design". IEEE Int. Conf. on Cad., Santa Clara,CA - September 1983

[CHU 83b] CHUQUILANQUI BERNAOLA S.H.
"Internal connection problem in large optimized PLA."
20 DAC, Miami, Florida - June 1983.

[CHU 82] CHUQUILANQUI BERNAOLA S.H. et PEREZ SEGOVIA T.
"PAOLA: A tool for topological optimization on large PLA's."
19 DAC, Las Vegas, Nevada - June 1982.

[COU 81] COURTOIS B.
"Failure mechanisme, fault hypotheses, and analytical testing of LSI-NMOS (HMOS) circuits". VLSI 81, Univ. of Edinburgh, August 18/21 1981, UK, Academic Press.

[CRO 78] CROUZET Y.
"Conception de circuits à large échelle d'intégration totalement autotestable". Toulouse, Université Paul Sabatier de Toulouse, 1978. (Thése n. 35).

[DAE81] DAEHN W. et MUCHA J.
"A hardware approach to self-testing of large programmable logic arrays."
IEEE Trans on circuit and syst - vol cas 28 - November 1981.

[EGA 84] EGAN J.R. et LIU C.L.
"Bipartite folding and partitioning of a PLA."
IEEE Trans on computers - vol cad 3 - July 1984.

[FER 86] FERREIRA A.
"An optimal $O(n^2)$ algorithm to fold special PLAs."
Proc. optimization Day - Montreal, Canada - April 1986.

[FUJ 85] SALUJA K., FUJIWARA H. et KINOSHITA K.
"A testable design of programmable logic arrays with universal control and minimal overhead."
Proc.IEEE International test conference - 1985.

[FUJ 84a] FUJIWARA H., TREUER R. et AGARWAL V.K.
"A low overhead, high coverage, built-in, self-test PLA design."
Report nº 84.13, VLSI Design Laboratory, McGill university, Quebec, Canada - 1984.

[FUJ 84b] FUJIWARA H.
"A new PLA design for universal testability."
IEEE Trans on computers - vol c-33 nº 8 - August 1984.

[FUJ 81] FUJIWARA H. et KINOSHITA K.
"A design of progrmmable logic arrays with universal tests."
IEEE Trans on computers - vol c-30 nº 11 - November 1981.

[GAL 80] GALIAY J., CROUZET Y. et VERGNIAULT M.
"Physical versus logical fault models MOS LSI circuits". IEEE Transactions on computers. Juin 80.

[GRE 76] D.GREER
"An associative logic matrix."
IEEE Journal of solid-state circuits - vol sc-11, nº 5 october 1976.

[HAC 82] HACHTEL G.D., NEWTON A.R et SANGIOVANNI-VINCENTELLI A.L.
"An algorithm for optimal PLA folding."
IEEE Trans on computers - vol cad-1 nº 2 - April 1982.

[HAS 83] HASSAN S.Z. et McCLUSKEY E.J.
"Testing PLAs using multiple signature analyzers."
CRC technical report - nº 83 13, Center for Reliable Computing, Computer systems Laboratory, Stanford University - California.

[HUA 84] HUA K.A., JOU J.Y. et ABRAHAM J.A.
"Built-in tests for VLSI finite-state machines."
Proc. 14th international symposium on fault tolerant computing June 1984.

[MEA 80] MEAD C. et CONWAY L.
"Introduction to VLSI systems."
Addison-Wesley publishing company - 1980

[OST 79] OSTAPKO D.L. et HONG S.J.
"Fault analysis and test generation for programmable logic arrays."
IEEE Trans on computers - vol c 28 - September 1979.

[LEW 84] LEWANDOWSKI J.L. et LIU C.L.
"A branch and bound algorithm for optimal PLA folding."
21st Design Automation Conference - 1984

[MAK 82] MAK G.P. ABRAHAM J.A., DAVIDSON E.S.
"The design of PLAs with concurrent error detection". Proc.12th Fault Tolerant Computing Symposium, Santa Monica - June 1982.

[MIC 84] De MICHELI G. et SANGIOVANNI-VINCENTELLI A.L.
"PLEASURE: A computer program for simple/multiple contrained/unconstrained folding of programmable logic arrays."
Computer-aided design, vol 16 n°2 - January 1984.

[MIC 83] De MICHELI G. et SANGIOVANNI-VINCENTELLI A.L.
"Multiple constrained folding of programmable logic array: theory and applications. IEEE Trans on computers - vol cad 2 n° 3 - July 1983.

[NIC 83] NICOLAIDIS M. et COURTOIS B.
"Design of self-checking system based on analytical fault hypotheses" IMAG, research report RR 353, March 1983.

[NIC 84a] NICOLAIDIS M.
"conception de circuits intégrés autotestables pour des hypothèses de pannes analytiques". Grenoble, INPG, Janvier 1984.

[NIC 84b] NICOLAIDIS M., JANSCH I. et COURTOIS B.
"Strongly code disjoint checkers". 14th Fault Tolerant Computing Symposium, Kissemmee, USA, 1984.

[NIC 86a] NICOLAIDIS M.
"An unified BIST approach using specific Strongly Code Disjoint Checkers Design" IMAG report No RR 599, Mars 1986.

[NIC 86b] NICOLAIDIS M. et COURTOIS B.
"Design of NMOS strongly fault secure circuits using unidirectional errors detecting codes" 16th fault tolerant computing symposium, Vienna July 1986, Austria.

[NIC 86c] NICOLAIDIS M. et COURTOIS B.
"Self-checking logic arrays"
Rapport de recherche, IMAG, December 1986.

[NIC 88] NICOLAIDIS M.
"A Unified Built-In-Self-Test scheme: UBIST"
18th Fault Tolerant Computing Symposium, Tokio, June 1988.

[PAI 81] PAILLOTIN J.F.
"Optimization of the PLA area."
Proc. 18th Design Automation Conference - Nashville - June 1981.

[PER 85] PEREZ SEGOVIA T.
"PAOLA: Un systeme d'optimisation topologique de PLA."
Thesis - INPG - Grenoble France - october 1985.

[PER 80] PEREZ SEGOVIA T.
"Optimisation en surface des PLAs."
Rapport de recherche nº 216, IMAG, October 1980.

[SAL 83] SALUJA K.K., KINOSHITA K. et FUJIWARA H.
"An easily testable design of programmable logic arrays for multiples faults."
IEEE, Trans on computers - vol c - 32 nº11 - november 1983.

[SAL 85] SALICK J., MERCER M.R. et UNDERWOOD B.
"Built-in self test input generator for programmable logic arrays."
Proc.IEEE International test conference - 1985.

[SMI 78] SMITH J.E. et METZE G.
"Strongly fault secure logic networks". IEEE Trans. on Comp., vol C-27, n°6, June 1978.

[TAK 88] TAKASHI N. et TOSHIAKI K.
"Error secure/propagating concept and its application to design of strongly fault-secure processors". IEEE Trans. on Comp., vol 37, n°1, January 1988.

[TAM 84] TAMIR Y. et SEQUIN C.H.
"Design and application of self-testing comparators implemented with MOS PLAs.
IEEE Trans on computers - vol c 33 - June 1984.

[TOR 86] TORKI K.
"Etude de l'architecture de parties contrôle autotestables pour le compilateur de silicium SYCO".
Rapport de DEA, Grenoble, INPG IMAG, June 1986.

[TRE 85] TREUER R., FUJIWARA H. et AGARWAL V.K.
"Implementing a built-in self-test PLA design."
IEEE, Design & Test - April 1985.

[VAR 85] VARINOT P. et CHUQUILANQUI BERNAOLA S.H.
"Method of PLA implementation: The monoplane PLA."
ESSIRC, Toulouse - September 1985.

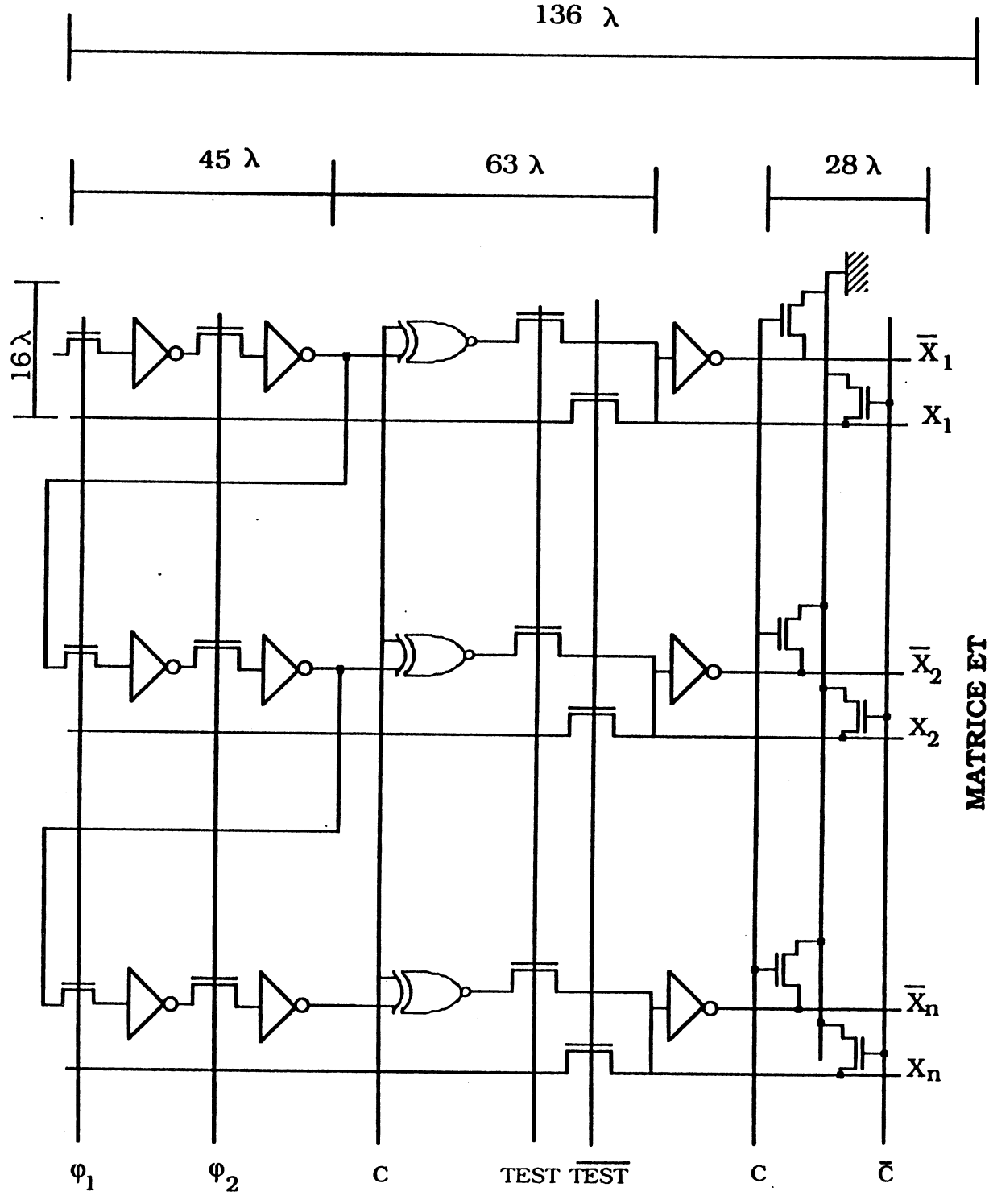
[WAD 78] WADSACK R.L.
"Faults modeling and logic simulation of CMOS and NMOS integrated circuits". The bell systems technical journal, vol 57, n° 5, May 1978.

[WOO 79] WOOD R.
"A high density programmable logic array chip".
IEEE Trans on computers - vol c 28 - September 1979.

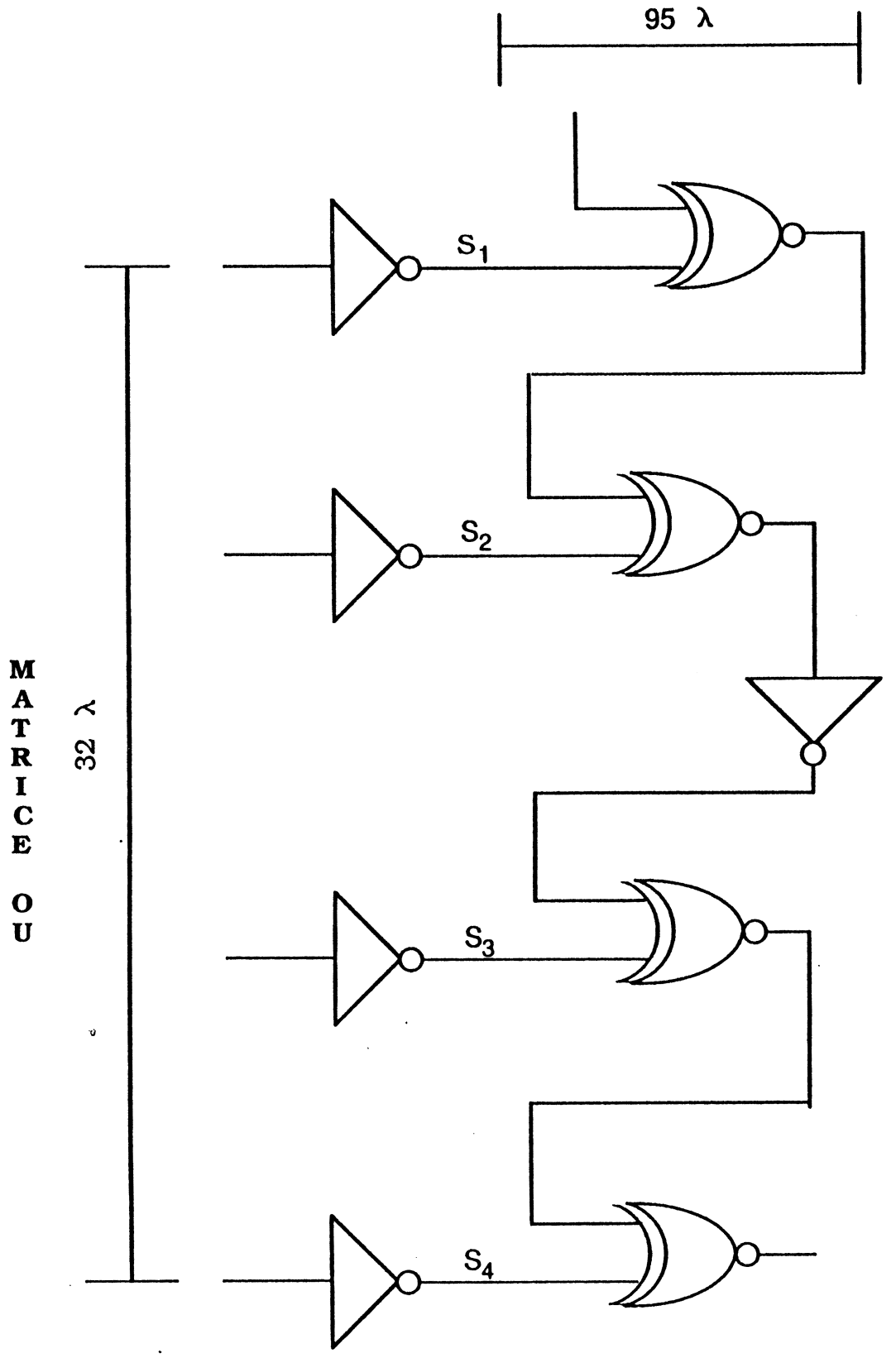
[YAJ 82] YAJIMA S., ARAMAKI T. et YASURA H.
"The design of autonomously testable PLA."
5th annual IEEE workshop on design for testability - April 1982.

ANNEXES

ANNEXE 1 : SURFACE DU GENERATEUR DES VECTEURS D'ENTREE.



ANNEXE 2: SURFACE DU CONTROLEUR DE PARITE.



ANNEXE 3

Couverture des pannes du test hors ligne.

THEOREME 1

Toutes les fautes simples et presque toutes les fautes multiples dues à un collage, un court-circuit et un "crosspoint" sont détectées par le schéma proposé.

Lemme 1

Les fautes simples dues à un "crosspoint", dans la matrice ET produit une erreur dans la parité cumulative.

Preuve

S'il existe une faute simple due à un "crosspoint" dans la matrice ET, la valeur du monôme prend l'inverse de la valeur attendue. La parité cumulative devient donc paire.

Lemme 2

Presque toutes les fautes multiples dues à un "crosspoint" dans la matrice ET, produit une erreur dans la parité cumulative.

Preuve

Le PLA est conçu de manière à avoir un nombre impair de croisements sans transistors dans la matrice ET. Ainsi le schéma de test détecte toutes les fautes multiples qui changent la valeur de la parité cumulative (i.e. un nombre impair de fautes multiples).

Lemme 3

Les fautes simples dues à un "crosspoint" dans la matrice OU, produit une erreur dans la parité du vecteur de sortie.

Preuve

S'il existe une faute simple due à un "crosspoint" dans la matrice OU, la valeur de la sortie correspondante est inversée. En conséquence la valeur de parité des sorties est aussi inversée.

Lemme 4

Presque toutes les fautes multiples dues à un "crosspoint" produit une erreur dans code de parité de sortie.

Preuve

S'il existe un nombre impair de fautes "crosspoint" du même type (transistor manquant ou en plus) dans la matrice OU, le nombre de croisements avec transistor devient pair. La parité du vecteur de sortie est donc inversée.

Lemme 5

Tous les collages simples et presque tous les collages multiples à "0" des lignes de sortie sont détectés par le schéma de test proposé.

Preuve

Quand le vecteur d'entrée sélectionne un "crosspoint" avec transistor, dans la matrice ET, toutes les lignes de sortie restent à "1". Par conséquence un nombre impair de lignes de sortie collé à "0" modifie la parité du vecteur de sortie. Cette condition doit être vérifié par tous les monômes. Un nombre pair de fautes peut être détecté quand au moins un monôme a un nombre impair de sorties erronées.

Lemme 6

Tous les collages simples et presque tous les collages multiples à "1" des lignes de sortie sont détectés par le schéma de test proposé.

Preuve

Quand le vecteur d'entrée sélectionne un crosspoint avec transistor, dans la matrice ET, toutes les lignes de sortie restent à "1". Par conséquence un nombre impair de lignes de sortie collé à "1" modifie la parité du vecteur de sortie. Cette condition doit être vérifié par tous les monômes. Un nombre pair de fautes peut être détecté quand au moins un monôme a un nombre impair de sorties eronées.

Lemme 7

Tous les collages simples et multiples à "0" des lignes de monôme sont détectés par le schéma de test proposé.

Preuve

Si le monôme est collé à "0" pendant l'application du test, la parité cumulative après 2n vecteurs sera inversée.

Lemme 8

Tous les collages simples et multiples à "1" des lignes de monôme sont détectés par le schéma de test proposé.

Preuve

Le monôme reste à "1", mais un nombre pair de vecteurs sont appliqués à ce monôme. Ainsi la parité cumulative après 2n vecteurs sera paire ce qui est l'inverse de la valeur attendue.

Lemme 9

Tous les collages simples à "0" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Les monômes sont composés par un nombre impair de croisement sans transistors. Ceci permet d'avoir une parité cumulative impaire après 2n vecteurs. Si une ligne d'entrée est collée à "0" la parité cumulative, du monôme concerné, devient paire.

Lemme 10

Presque tous les collages multiples à "0" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Cette faute équivaut à de fautes "crosspoint" multiples dans la matrice ET. Généralement deux entrées ne sont pas toujours connectées aux même monômes. Par conséquence le schéma proposé peut détecter les collages à "0" en deux entrées. Pour un nombre plus grand que deux, la faute peut être

détectée quand au moins sur un monôme la parité cumulative est modifiée. C'est à dire que le schéma ne peut pas détecter un collage multiple des lignes d'entrées quand la combinaison des lignes ayant une faute équivaut à un nombre pair de fautes "crosspoint", pour tous les monômes.

Lemme 11

Tous les collages simples et multiples à "1" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Les monômes connectés à une ligne d'entrée collé à "1" sont mis à la masse. Cette faute équivaut donc à un collage multiple à "0" des monômes.

Lemme 12

Tous les court-circuits simples et multiples des monômes sont détectés par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage à "0" d'un monôme.

Lemme 13

Toutes les fautes simples et presque toutes les fautes multiples des lignes d'entrée dues à un court-circuit sont détectées par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage multiple à "0" des lignes d'entrée.

Lemme 14

Toutes les fautes simples et presque toutes les fautes multiples des lignes de sortie dues à un court-circuit sont détectées par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage multiple à "0" des lignes de sortie.

ANNEXE 4

PLA ORIGINAL (n= 8; m= 201 et p= 38)

**** TABLE D'IMPLANTATION AVANT OPTIMISATION ****

```
00000000011111111 | 00000000011111111112222222223333333333
12345678901234567 | 12345678901234567890123456789012345678
=================================================================
01010101010101010 | 10011000001000010100100010000000000111
01010101010101011 | 10011000111000010100100010000000000111
01010101010110010 | 10011000110000010100100010000000000111
01010101010110110 | 00010001100000001010010001000000000001101
01010101011001010 | 00010000011000110100100010000000000001101
01010101011001011 | 00010000011000100000100000000000000001000
01010101011010010 | 00010000011000110100100100110000000000101
01010101011010110 | 00010000000000010000011000001000000001000
01010101100101010 | 00001000101000100000000000011000000000
01010101100101011 | 00001000110001010100100010010000000101
01010101100110010 | 00010000011011010100100010000000001101
01010101100110110 | 00010000011011010000000000100000001000
01010101101001010 | 00001000101110101010010001001000000101
01010101101001011 | 00010000000000010000011000001000000001000
01010101101010010 | 00001000101000100000000000011000000000
01010101101010110 | 00001001001010111010010001001000000101
01010110010101010 | 00010001100000011010010001000000001101
01010110010101011 | 10011000110001010100100010000000000111
01010110010110010 | 10011001001010111010010001000000000111
01010110010110110 | 10011001001100110100100010000000000111
01010110011001010 | 11011000001000010100100010000000000011
01010110011001011 | 10011000001000010100000000010110000011
01010110011010010 | 11011000111000010100100010000000000011
01010110011010110 | 10011000111000010100000000010110000011
01010110100101010 | 11011000110000010100100010000000000011
01010110100101011 | 10011000110000010100000000010110000011
01010110100110010 | 00010000011000010100000000000000001001
01010110100110110 | 10011001001010111010000000000000000011
01010110101001010 | 11011000110001010100100010000000000011
01010110101001011 | 10011000110001010100000000010110000011
01010110101010010 | 11011000110000010100100010000000000011
01010110101010110 | 10011001001010111010000000010110000011
01011001010101010 | 10011001001010111010000000010110000011
01011001010101011 | 10011001001100110100000000010110000011
01011001010110010 | 01010000011000010100100010000000001001
01011001010110110 | 00010000011000010100000000010110000001001
01011001011001010 | 00000000000000000001000000000000000001001
01011001011001011 | 10111000001000010100100010000000000101
01011001100101010 | 10011000001000010100100010000000010101
01011001100101011 | 10011000001000010100100010000000000101
01011001100110010 | 10111000111000010100100010000000000101
01011001100110110 | 10011000111000010100100010000000010101
01011001101001010 | 10111000111000010100100010000000000101
01011001101001011 | 10011000111000010100100010000000010101
01011001101010010 | 10111000111000010100100010000000000101
01011001101010110 | 10011000111000010100100010000000010101
01011010010101010 | 10111001001010111010010001000000000101
01011010010101011 | 10011001001010111010010001000000010101
01011010011001010 | 10010000011000010100100010000000000101
01011010011001011 | 00101000101000010100100010000000000101
01011010011100101 | 00001000101000010100100010000000010101
01011010011100110 | 00000000000000000001000000000000000101
01011010011101001 | 00001000101000010100100010010000000101
01011010011101010 | 00010000011000010100100100110000000000101
01011010100101010 | 00000000000000010100100010000000000101
01011010100101011 | 00000000000000110100100010000000000101
01100101010101010 | 11111000001000010100100010000000000001
01100101010101011 | 11011000001000010100100010000000010001
01100101010110010 | 11011000001000010100000000010110000001
01100101010110110 | 10011000001000010100000000010110001001
01100101011100101 | 11111000111000010100100010000000000001
01100101011100110 | 11011000111000010100100010000000010001
```

```
0110010101101001 | 1011110001110000101000000000101100000001
0110010101101010 | 1001100011100000101000000000101100010001
0110010110010101 | 1111100011000000101001000100000000000001
0110010110010110 | 1101100011000000101001000100000000010001
0110010110011001 | 1011100011000000101000000000101100000001
0110010110011010 | 1001100011000000101000000000101100010001
0110010110100101 | 1101000110000000101001000100000000000001
0110010110100110 | 0000100101000000101000000000101100010001
0110010110101001 | 1001000001100000101000000000000000000001
0110010110101010 | 0000100010100000101000000000000000010001
0110011001010101 | 0000100010100000111000000000101100110001
0110011001010110 | 0000100010100001111000000000101101010001
0110011001011001 | 1101000001100011010010001000000000000001
0110011001011010 | 0000100010100011010010001000000000010001
0110011001100101 | 1001000001100010000001000000000000000000
0110011001100110 | 0101000001100011010010011000000000000001
0110011001101001 | 0000100010100010000001000000000000010000
0110011001101010 | 0001000001100011010000010010110000000001
0110011010010101 | 1001000000000001000001100001000000000000
0110011010010110 | 0000100010100010000000000011000000000000
0110011010011001 | 0100100011000101010010001001000000000001
0110011010011010 | 0000000000000001000001100001000000000000
0110011010100101 | 0000100010100010000001000000010000000000
0110011010100110 | 0001100011000101010000010010110000010001
0110011010101001 | 1101000001101101010010001000000000000001
0110011010101010 | 0000100010110101010100000000100101010001
0110100101010101 | 1001000001101100000000000000100000000000
0110100101010110 | 0100100010110101010010001001000000000001
0110100101011001 | 0000100010110100000010000000000000010000
0110100101011010 | 0001000001101101010000010010110000000001
0110100101100101 | 1001000000000001000001100001000000000000
0110100101100110 | 0000100010100010000000000011000000000000
0110100101101001 | 0100100100101011010010001001000000000001
0110100101101010 | 0000000000000001000001100001000000000000
0110100110010101 | 0000100010100010000001000000010000000000
0110100110010110 | 0001100100110011010000010010110000010001
0110100110011001 | 1101000110000011010010001000000000000001
0110100110011010 | 0000100101010000110100000000101100010001
0110100110100101 | 0000100010100001010000000010111001000001
0110100110100110 | 0000100010100001010000000010111100010001
0110100110101001 | 1011100100101011010000000000000000000001
0110100110101010 | 1001100100101011010000000000000000010001
0110101001010101 | 1111100011000101010010001000000000000001
0110101001010110 | 1101100011000101010010001000000000010001
0110101001011001 | 1011100011000101010000000010111000000001
0110101001011010 | 1001100011000101010000000010111000010001
0110101001100101 | 1111100100101011010010001000000000000001
0110101001100110 | 1101100100101011010010001000000000010001
0110101001101001 | 1011100100101011010000000010111000000001
0110101001101010 | 1001100100101011010000000010111000010001
0110101010010101 | 1001000001100000000000000001000000000000
0110101010010110 | 0010100000000000000001000000000000000000
0110101010011001 | 0001010000000000000001011000000000000000
0110101010011010 | 0001010000000000000001011000000000000000
0110101010101001 | 0001010000000000000001011000000000000000
0110101010101010 | 0001010000000000000001011000000000000000
0110101010100101 | 0001010000000000000001011000000000000000
0110101010100110 | 0001010000000000000001011000000000000000
0110101010101001 | 0001010000000000000001011000000000000000
0110101010101010 | 0001010000000000000001011000000000000000
1001010101010101 | 0001010000000000000001011000000000000000
1001010101010110 | 0001010000000000000001011000000000000000
1001010101100101 | 0001010000000000000001011000000000000000
1001010101100110 | 0001010000000000000001011000000000000000
1001010101011001 | 0001010000000000000001101000000000000000
1001010101101001 | 0001010000000000000001101000000000000000
1001010101011010 | 0001010000000000000001011000000000000000
1001010101101010 | 0001010000000000000001011000000000000000
1001010110010101 | 0001010000000000000001011000000000000000
1001010110010110 | 0001010000000000000001011000000000000000
1001010110011001 | 0001010000000000000001011000000000000000
```

```
1001010110011010 | 0001010000000000000001011000000000000000
1001010110100101 | 0001010000000000000001011000000000000000
1001010110100110 | 0001010000000000000001011000000000000000
1001010110101001 | 0001010000000000000001011000000000000000
1001010110101010 | 0001010000000000000001011000000000000000
1001011001010101 | 0001010000000000000001011000000000000000
1001011001010110 | 0001010000000000000001011000000000000000
1001011001011001 | 0001010000000000000001011000000000000000
1001011001011010 | 0001010000000000000001011000000000000000
1001011001100101 | 0001010000000000000001011000000000000000
1001011001100110 | 0001010000000000000001011000000000000000
1001011001101001 | 0001010000000000000001011000000000000000
1001011001101010 | 0001010000000000000001011000000000000000
1001011001010101 | 0001010000000000000001011000000000000000
1001011001010110 | 0001010000000000000001011000000000000000
1001011010010101 | 0001010000000000000001011000000000000000
1001011010010110 | 0001010000000000000001011000000000000000
1001011010011001 | 0001000001100001010010010000000000000001
1001011010011010 | 1001000001100000000000000001000000000000
1001011010100101 | 0010100000000000000001000000000000000000
1001011010100110 | 0001000100101010000011010000000000000000
1001011010101001 | 0001001000000000000001101000000000000000
1001011010101010 | 0001001000000000000001101000000000000000
1001100101010101 | 0001001000000000000001101000000000000000
1001100101010110 | 0001001000000000000001101000000000000000
1001100101011001 | 0001001000000000000001101000000000000000
1001100101011010 | 0001001000000000000001101000000000000000
1001100101100101 | 0001001000000000000001101000000000000000
1001100101100110 | 0001001000000000000001101000000000000000
1001100101101001 | 0001001000000000000001101000000000000000
1001100101101010 | 0001001000000000000001101000000000000000
1001100110010101 | 0001001000000000000001101000000000000000
1001100110010110 | 0001001000000000000001101000000000000000
1001100110011001 | 0001001000000000000001101000000000000000
1001100110011010 | 0001001000000000000001101000000000000000
1001100110100101 | 0001001000000000000001101000000000000000
1001100110100110 | 0001001000000000000001101000000000000000
1001100110101001 | 0001001000000000000001101000000000000000
1001100110101010 | 0001001000000000000001101000000000000000
1001101001010101 | 0001001000000000000001101000000000000000
1001101001010110 | 0001001000000000000001101000000000000000
1001101001011001 | 0001001000000000000001101000000000000000
1001101001011010 | 0001001000000000000001101000000000000000
1001101001100101 | 0001001000000000000001101000000000000000
1001101001100110 | 0001001000000000000001101000000000000000
1001101001101001 | 0001001000000000000001101000000000000000
1001101001101010 | 0001001000000000000001101000000000000000
1001101010010101 | 0001001000000000000001101000000000000000
1001101010010110 | 0001001000000000000001101000000000000000
1001101010011001 | 0001001000000000000001101000000000000000
1001101010011010 | 0001001000000000000001101000000000000000
1001101010010101 | 0001001000000000000001101000000000000000
1001101010100101 | 0001001000000000000001101000000000000000
1001101010100110 | 0001001000000000000001001000000000000000
1001101010101001 | 0001000000000000000000101000000000000000
1001101010101010 | 0000000001100001010010000000000000000001
1010010101010101 | 0110100010100001010010001000000000000001
1010010101010110 | 0100100010100001010010001000000000010001
1010010101011001 | 0100000000000000000010000000000000000001
1010010101011010 | 0100100010100001010010001001000000000001
1010010101100101 | 0101000001100001010010011000000000000001
1010010101100110 | 1001000001100001010000000010110000000001
1010010101101001 | 1000000000000000000010000000000000000001
1010010101101010 | 0100000000000000101001000100000000000001
1010010110010101 | 0000000000000000101000000001011000000001
1010010110010110 | 0100000000000001101001000100000000000001
1010010110011001 | 0000000000000001101000000001011000000001
1010010110011010 | 0000000000000000000010001000000000000001
1010010110100101 | 0000000000000000000000000000000000000001
```

**** FIN****

PLA SANS LE CODE DE SORTIE (n= 8; m= 139 et p= 38)

**** TABLE D'IMPLANTATION APRES OPTIMISATION LOGIQUE ****

```
00000000011111111 | 00000000011111111112222222223333333333
12345678901234567 | 12345678901234567890123456789012345678
==================|=======================================
01100110010101101 | 00000000000000000100000000000000001000000
10100101100110101 | 00000000000000000000001000100000000000001
10100101010110011 | 01000000000000000000100000000000000000001
10100101100110011 | 00000000000000010000000000000000000000001
01011010011001101 | 00000000000000000000100000000000000000101
01011001101010001 | 00000000000000100000000000000000000000000
10010110101001011 | 00101000000000000000010000000000000000000
10100101011010011 | 10000000000000000000100000000000000000001
01011001011001011 | 00000000000000000000100000000000000001001
01010110101001001 | 00000000000000100000000000000000000000000
01011010000101101 | 00000000000000010000000000000000000000000
01101010100101101 | 00101000000000000000010000000000000000000
10100101010110101 | 00001000101000000000000000010000000000000
01101001011010101 | 00000000000000010000011000001000000000000
01010111001101000 | 00000000110000000000000000000000000000000
01100110100110101 | 00000000000000010000011000001000000000000
10010110101010001 | 00010010000000000000011010000000000000000
01100101011010011 | 00101000000000000000000000010110000000000
01010101100110011 | 00000000000001100000010001000000000000100
10010110100110101 | 10010000011000000000000000010000000000000
01011001100010001 | 00000000110000000000000000000000000000000
10010101010010011 | 00010100000000000000011010000000000000000
01010100000101101 | 00000000110000000000000000000000000000000
01100101011001000 | 00000000110000000000000000000000000000000
01101010100101011 | 10010000011000000000000000010000000000000
10010110101001101 | 00010001001010100000011010000000000000000
10100101100100011 | 00000000000000001010000000010110000000000
01101001011001101 | 00001000101000100000000000011000000000000
01010101101001101 | 00010000000000010000011000001000000001000
00100101000000101 | 00000000000000000000000000000000000000001
01010101011010101 | 00010000000000010000011000001000000001000
01100110100101101 | 00001000101000100000000000011000000000000
01010101101010011 | 00001000101000100000000000011000000000000
01101001011001011 | 10010000000000010000011000001000000000000
01010101100101011 | 00001000101000100000000000011000000000000
01101001100101011 | 00001000101000100000010000001000000000000
10011010101010101 | 00000000011000010100100000000000000000001
01100110100101011 | 10010000000000010000011000001000000000000
01100110101001011 | 00001000101000100000010000001000000000000
01101001011011001 | 00001000101101010000001000000000000010000
01101010101001000 | 00010100000000000000010110000000000000000
10011000000000001 | 00010000000000000000010100000000000000000
01101010100010000 | 00010100000000000000010110000000000000000
01011010010101101 | 10011001001010000000000000010000000000000
01101001010101011 | 10010000011011000000000000010000000000000
01100110011101001 | 00001000101000100000010000000000000010000
10010100000101001 | 00010100000000000000010110000000000000000
01010101011101001 | 00010000011000100000100100000000000000000
10010110010000000 | 00010100000000000000010110000000000000000
01011010010101011 | 10111001001010100000000000000000000000000
10010101100000000 | 00010100000000000000010110000000000000000
10010101000000010 | 00010100000000000000010110000000000000000
10010101000000100 | 00010100000000000000010110000000000000000
01100110011001011 | 10010000011000100000100000000000000000000
10100101100101101 | 01000000000000011010010001000000000000001
01010101100110101 | 00010000011011010000000000100000001000
10011000000000100 | 00010010000000000000010010000000000000000
01010101010110011 | 10011000110000000000010000000000000000010
10011000000010000 | 00010010000000000000011010000000000000000
10100101010010101 | 01000000000000010100100010000000000000001
10011000010000000 | 00010010000000000000011010000000000000000
```

```
1001100100000000 | 0001001000000000000001101000000000000000
0101010101100100 | 0001000001100010000010000000000000001000
1001011010011001 | 0001000001100001010010010000000000000001
1010010101010101 | 0110100010100001010010001000000000000000
1010010101100101 | 0101000001100001010010011000000000000000
0101010101000001 | 0000000000000001010000001000000000000101
0101010110010110 | 0000100000000101010010001001000000000101
0110010110100110 | 0000100101000001010000000010110001000 1
0101101000010100 | 0000000000000001010010001000000000000101
1010010101100110 | 1001000001100001010000000010110000000001
0110010110100101 | 1101000110000001010010001000000000000000
0110100110100110 | 0000100010100001010000000010111001000 1
0101010010011001 | 0001000001100001010000000000000000001001
0110100110011010 | 0000100101000011010000000010110001000 1
0110100110100101 | 0000100010100001010000000010111011110001
0101100101011010 | 0001000001100001010000000010110000010001
0110010100101010 | 0000100010100001010000000000000000010001
0110011010101010 | 0000100010111010101000000000100101010001
1010010101010110 | 0100100010100001010010001000000000010001
0101101001101001 | 0000100010100001010010001001000000000101
0110011001010101 | 0000100010100001110000000010110011010001
0101101001011010 | 0010100010100001010010001000000000000101
0101101001100101 | 0000100010100001010010001000000000010101
0101010101011010 | 0001000110000001010010001000000000001101
0101101001101010 | 0001000001100001010010011000000000000101
0110010100101001 | 1001000001100001010000000000000000000001
0110011010011001 | 0100100011000101010010001001000000000001
0110100101011010 | 0001000001101101010000010010110000000001
0101100101011001 | 0101000001100001010010001000000000001001
0110011001101010 | 0001000001100011010000010010110000000001
0110100110101001 | 1011100100101011010000000000000000000001
0101011001010101 | 0001000110000011010010001000000000001101
0110100110101010 | 1001100100101011010000000000000000010001
0110100101010110 | 0100100010110101010010001001000000000001
0101101001011001 | 1001000001100001010010001000000000000101
0101011010011010 | 1001100100101011010000000000000000000011
0101010110100101 | 0000100010110101010010001001000000000101
0110100101101001 | 0100100100101011010010001001000000000001
0110100110011001 | 1101000110000011010010001000000000000001
0101011001010110 | 1001100000000010101001000100000000000111
0110011010100110 | 0001100011000101010000010010110001000 1
0101010110101010 | 0000100100101011010010001001000000000101
0110010110010101 | 1111100011000001010010001000000000000000
0110101001011001 | 1011100011000101010000000010110000000001
0110101001011010 | 1001100011000101010000000010110001000 1
0110011001100110 | 0101000001100011010010011000000000000001
0110010110011001 | 1011100011000001010000000010110000000001
0110010101011001 | 1101100000010000101001000100000000000001
0110010110011010 | 1001100011000001010000000010110001000 1
0110100110010110 | 0001100100111001101000001001011000100 1
0110011010101001 | 1101000001101101010010001000000000000001
0110011001010010 | 0000100010100011010000000010110001000 1
0110101001101001 | 1011100100101011010000000010110000000001
0101100101010110 | 1001100100111001101000000010110000000011
0110101001101010 | 1001100100101011010000000010110001000 1
0110011001011001 | 1101000001100011010010001000000000000001
0101011001100010 | 1001100000010000101001000100000000000011
0110101001010101 | 1111100011000001010010001000000000000001
0110010101000101 | 1111100000010000101001000100000000000000
0101100101010101 | 1001100100101011010000000010110000000011
0110101001010110 | 1101100011000001010010001000000000000001
0101011010000110 | 1001100011000001010000000010110000000011
0110010110010110 | 1101100011000001010010001000000000010001
0110010101001010 | 1001100000010000101000000010110001000 1
0101100110010001 | 1011100000010000101001000100000000000101
0110101001100101 | 1111100100101011010010001000000000000001
0101011010101010 | 1101100100111001101001000100000000000011
0101100110100001 | 1011100011000001010010001000000000000101
0101100110010010 | 1001100000010000101001000100000000010101
0110101001100110 | 1101100100101011010010001000000000010001
0101100110100010 | 1001100011000001010010001000000000010101
```

```
0101011001100001 | 1101100000100001010010001000000000011
0101011010101001 | 1101100100101011010010001000000000011
0101011001011010 | 1001100100110011010010001000000000111
0101011010000101 | 1101100011000001010010001000000000011
0101011001011001 | 1001100100101011010010001000000000111
0101010101010100 | 1001100000100001010010001000000000111
0110010101000110 | 1101100000100001010010001000000010001
```

**** FIN ****

PLA CODE (n= 8; m= 144 et p= 44)

**** TABLE D'IMPLANTATION APRES OPTIMISATION LOGIQUE****

	CODE DE BERGER
0000000001111111	0000000001111111111222222222233333333334444
1234567890123456	1234567890123456789012345678901234567890123
0100100110101001	00100000000000000000000000000000000000000000
0101100110000001	00100000000000000000000000000000000000000000
0101011010101001	00000001001010100000000000000000000000000000
1010010110100101	00000000000000000000000000000000000001101001
1001101010101001	00010000000000000000010100000000000000110001
1001101010100110	00010010000000000000100100000000000000010001
1010010110011010	00000000000000000000010001000000000001110001
0100100110000010	00000000000000000000000000000000000010000000
1010010101011001	01000000000000000010000000000000000001110001
1010010101101001	10000000000000000010000000000000000001110001
1001011010100101	00101000000000000000100000000000000000110001
0101011010100100	00000000110001000000000000000000000000100000
0101100101100101	00000000000000000010000000000000001001110001
0101101001100110	00000000000000000010000000000000010101110001
0110100101101010	00000000000000010000011000001000000000000010001
0101011001100100	00000000001000000000000000000000000000000110000
0110011010011010	00000000000000010000011000001000000000000010001
0110101001010110	00101000000000000000100000000000000000110001
1001011010101000	00010010000000000000110100000000000000100001
0101011001101000	00000000111000000000000000000000000000100000
0110100101100110	00001000101000100000000000110000000000000001
0101010110100110	00010000000000100000110000100000001000000001
0101010101101010	00010000000000100000110000100000001000000001
1001010101001001	00010100000000000000110100000000000000100001
1001011010011010	10010000011000000000000000100000000000100001
0110011010010110	00001000101000100000000000110000000000000001
0101010110101001	00001000101000100000000000110000000000000001
0110100101100101	10010000000000100000110000100000000000000001
0101010110010101	00001000101000100000000000110000000000000001
1010010110010101	00000000000000010100000000101100000001000001
0110100110010101	00001000101000100000100000010000000000000001
0110011010010101	10010000000000100000110000100000000000000001
0110011010100101	00001000101000100000100000010000000000000001
0110011001101001	00001000101000100000100000000000010000000001
0110101010010101	10010000011000000000000000100000000000100001
1001101010101010	00000000011000010100100000000000000001000001
0101010101100110	00010000011000100000100000000000000001000000001
1010010101101010	01000000000000010100100010000000000000001000001
0110101010100000	00010100000000000000010110000000000000000100001
0110101010001000	00010100000000000000010110000000000000000100001
0110011001100101	10010000011000100000100000000000000000000000001
1001100000000101	00010010000000000000011010000000000000000100001
0101101010010101	00000000000000010100100010000000000001010000001
1001011001000000	00010100000000000000010110000000000000000100001
1001010100000010	00010100000000000000010110000000000000000100001
1001010000010100	00010100000000000000010110000000000000000100001
1001010110000000	00010100000000000000010110000000000000000100001
1001010100000100	00010100000000000000010110000000000000000100001
1001100000010000	00010010000000000000011010000000000000000100001
1001100001000000	00010010000000000000011010000000000000000100001
1001100100000000	00010010000000000000011010000000000000000100001
1001011010100110	00010001001010100000011010000000000000000011110
0110100101011001	00001000101101010000001000000000000010000111110
0101010110011010	00010000011011010000000000010000000100001111110
0110100101010101	10010000011011000000000000010000000000001111110
1010010110011001	00000000000000011010000000010110000000011111110
1010010110010110	01000000000000011010010001000000000000011111110
0110010110100110	00001001010000010100000000010110001000100111110

```
1001011010011001 | 0001000001100001010010010000000000000010111110
0101101010010110 | 0000000000000011010010001000000000010111111110
1010010101100110 | 1001000001100001010000000010110000000100111110
0101100101011010 | 0001000001100001010000000010110000010010011110
1010010101011010 | 0100100010100001010010001001000000000010011110
1010010101010101 | 0110100010100001010010001000000000000010011110
1010010101010110 | 0100100010100001010010001000000000100010011110
1010010101100101 | 0101000001100001010010011000000000000010011110
0101101001101001 | 0000100010100001010010001001000000010100111110
0110100110100101 | 0000100010100001010000000010110111000101010110
0101101001011010 | 0010100010100001010010001000000000010100111110
0101011010011001 | 0001000001100001010000000000000000010011111110
0110011010101010 | 0000100010111010101000000001001010100010101110
0101010101011010 | 0001000110000001010010001000000000011010011110
0110010110101010 | 0000100010100001010000000000000000100011111110
0101101001100101 | 0000100010100001010010001000000000101010011110
0101101001101010 | 0001000001100001010010011000000000010100111110
0110011001010101 | 0000100010100001111000000000101110011000101010110
0110100110011010 | 0000100101000011010000000001011000000001110110
0101011000100110 | 1001100000000000101000000000101100000011000110
0101001010111001 | 0101000001100001010010001000000000010010011110
0101011001100010 | 1001100000000000101000000000101100000011000110
0110100110100110 | 0000100010100001010000000001011100000011101110
0110010110100101 | 1101000110000001010010001000000000000010011110
0110100101011010 | 0001000001101101010000010010110000000010101110
0110100110010110 | 0001100100110011010000010010110000000010001110
0110010110101001 | 1001000001100001010000000000000000000011111110
0101101001011001 | 1001000001100001010010001000000000010100111110
0110011001010110 | 0000100010100011111000000000101101010001100110
0110100101010110 | 0100100010111010101001000100100000000010101110
0110100101101001 | 0100100100101011010010001001000000000010101110
0110101001101001 | 1011100100101011010000000010110000000010001110
0110101001101010 | 1001100100101011010000000010110001000100011110
0101010110100101 | 0000100010111010101001000100100000000010101110
0101100101010110 | 1001100100110011010000000010110000100011001110
0101010110101010 | 0000100100101011010010001001000000010101010110
0101100101010101 | 1001100100101011010000000010110000000110001110
0101011010010110 | 1001100001100000101000000000101100000011010110
0110011010100110 | 0001100011000101010000010010110001000100110110
0110101001011001 | 1011100011000101010000000010110000000011001110
0101010110011001 | 0001000001101101010010001000000000011010101110
0110011010011001 | 0100100011000101010010001001000000000011101110
0110101001011010 | 1001100001100001010000000010110001000100110110
0110011001011010 | 0000100010100011010000000010110001000111010110
0110010110011001 | 1011100011000001010000000010110000000010101110
0101010110010110 | 0000100011000101010010001000100000000010111010110
0110011001101010 | 0001000001100011010000010010110000000011101110110
0110010110011010 | 1001100011000001010000000010110001000100101010110
0101011010011010 | 1001100100101011010000000000000000000011111010110
0101011001010101 | 0001000110000011010010001000000000001101111010110
0110011010101001 | 1101000001101101010010001000000000000010101010110
0110101001100101 | 1111100100101011010010001000000000000001000110
0101101001010101 | 1011100100101011010010001000000000000010100011010
0110100110011001 | 1101000011000001101001000100000000000001110110
0101011010101010 | 1101100100110011010010001000000000000011000110
0110101001100110 | 1101100100101011010010001000000000100010001110
0101101001010110 | 1001100100101011010010001000000000101010001110
0101011000100001 | 1101100000000000101001000100000000000011000110
0101011001011010 | 1001100100110011010010001000000000000111000110
0101011010010101 | 1101100001100000101001000100000000000110101110
0110100110101000 | 1001100100101011010000000000000000000001110110
0101011001011001 | 1001100100101011010010001000000000000111000110
0110101001010101 | 1111100011000101010010001000000000000011001110
0110010101011010 | 1001100000010000101000000000101100010001110110
0110101001010110 | 1101100011000101010010001000000000010001100110
0101010101011001 | 1001100011000000101001000100000000000111010110
0110010101011001 | 1101100000010000101000000001011100000011101110
0110010110010101 | 1111100011000001010010001000000000000010101110
0101010101100101 | 0001000001100011010010001000000000001101111010110
0110011001100110 | 0101000001100011010010011000000000000001110110
0110010110010110 | 1101100011000001010010001000000000010001010110
```

```
0101010101101001 | 0001000001100011010010011000000000010111011O
0101011001010110 | 1001100011000101010010001000000000011110011O
0110010101101001 | 1011100011100001010000000010110000000011001O
0101100110100100 | 1001100011000001010010001000000000010101011O
0110010101101010 | 1001100011100001010000000010110001000110011O
0110011001011001 | 1101000001100011010010001000000000000111011O
0101100110101000 | 1001100011000101010010001000000000010110011O
0101010101010101 | 1001100000100001010010001000000000001111101O
0110010101010101 | 1111100000100001010010001000000000000111011O
0110010101010110 | 1101100000100001010010001000000001000111011O
0101100110010100 | 1001100000100001010010001000000000010111011O
0101010101010110 | 1001100011100001010010001000000000001111001O
0110010101100101 | 1111100011100001010010001000000000000110011O
0110010101100110 | 1101100011100001010010001000000001000110011O
0101100110011000 | 1001100011100001010010001000000000010110011O
```

**** FIN ****

ANNEXE 5

Couverture des pannes du test déterministe (hors ligne).

THEOREME 1

Toutes les fautes simples et presque toutes les fautes multiples dues à un collage, un court-circuit et un "crosspoint" sont détectées par le schéma proposé.

Lemme 1

Les fautes simples dues à un "crosspoint", dans la matrice ET produit une erreur dans la parité cumulative.

Preuve

S'il existe une faute simple due à un "crosspoint" dans la matrice ET, la valeur du monôme prend l'inverse de la valeur attendue. La parité cumulative devient donc paire.

Lemme 2

Presque toutes les fautes multiples dues à un "crosspoint" dans la matrice ET, produit une erreur dans la parité cumulative.

Preuve

Le PLA est conçu de manière à avoir un nombre impair de croisement sans transistors dans la matrice ET. Ainsi le schéma de test détecte toutes les fautes multiples qui changent la valeur de la parité cumulative (i.e. un nombre impair de fautes multiples).

Lemme 3

Les fautes simples dues à un "crosspoint", dans la matrice OU produit une erreur dans le code non ordonné.

Preuve

S'il existe une faute simples due à un "crosspoint" dans la matrice OU, la valeur de la sortie concerné est inversé. Le code de sortie est donc modifié.

Lemme 4

Toutes les fautes multiples dues à un "crosspoint" du même type (i.e. transistor manquant ou transistor en plus), dans la matrice OU, produit une erreur dans le code non ordonné de la sortie.

Preuve

Les fautes multiples dues à un même type de faute crosspoint si propage comme une erreur unidirectionnelle. Le code non ordonné est donc capable de la détecter.

Lemme 5

Tous les collages simples et multiples à "0" des lignes de sortie sont détectés par le schéma de test proposé.

Preuve

Quand le vecteur d'entrée sélectionne un crosspoint avec transistor, dans la matrice ET, toutes les lignes de sortie restent à "1". Par conséquence une ou plusieurs lignes de sortie collé à "0" modifie de manière unidirectionnelle la valeur d'une ou de plusieurs sorties.

Lemme 6

Tous les collages simples et multiples à "1" des lignes de sortie sont détectés par le schéma de test proposé.

Preuve

Cette faute inverse de manière unidirectionnelle la valeur d'une ou de plusieurs sorties.

Lemme 7

Tous les collages simples et multiples à "0" des lignes de monôme sont détectés par le schéma de test proposé.

Preuve

Si un monôme est collé à "0" pendant l'application du test, la parité cumulative après 2n vecteurs sera inversé.

Lemme 8

Tous les collages simples et multiples à "1" des lignes de monôme sont détectés par le schéma de test proposé.

Preuve

Le monôme reste à "1", mais un nombre pair de vecteurs sont appliqués à ce monôme. Ainsi la parité cumulative après 2n vecteurs sera paire qui est l'inverse de la valeur attendue.

Lemme 9

Tous les collages simples à "0" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Les monômes sont composés par un nombre impair de croisement sans transistors. Ce qui permet d'avoir une parité cumulative impaire après 2n vecteurs. Si une ligne d'entrée est collé à "0" la parité cumulative, du monôme concerné, devient paire.

Lemme 10

Presque tous les collages multiples à "0" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Cette faute équivaut à de multiple fautes "crosspoint" dans la matrice ET. Généralement deux entrées ne sont pas toujours connectées aux même monômes. Par conséquence le schéma proposé peut détecter les collages à "0" en deux entrées. Pour un nombre plus grande que deux, la faute peut être détectée quand au moins sur un monôme la parité cumulative est modifiée. C'est à dire que le schéma ne peut pas détecté un collage multiple des lignes d'entrées quand la combinaison des lignes ayant une faute équivaut à un nombre pair de fautes "crosspoint", pour tous les monômes.

Lemme 11

Tous les collages simples et multiples à "1" des lignes d'entrée sont détectés par le schéma de test proposé.

Preuve

Les monômes connectés à une ligne d'entrée collé à "1" sont mis à la masse. Cette faute équivaut donc à un collage multiples à "0" des monômes.

Lemme 12

Tous les court-circuits simples et multiples des monômes sont détectés par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage à "0" des monôme.

Lemme 13

Toutes les fautes simples et presque toutes les fautes multiples des lignes d'entrée dues à un court-circuit sont détectées par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage multiple à "0" des lignes d'entrée.

Lemme 14

Toutes les fautes simples et toutes les fautes multiples des lignes de sortie dues à un court-circuit sont détectées par le schéma proposé.

Preuve

Si nous supposons qu'un court-circuit produit sur les deux lignes la valeur logique d'une fonction ET, alors cette faute équivaut à un collage multiple à "0" des lignes de sortie.

INPG département des études doctorales

AUTORISATION de SOUTENANCE

VU les dispositions de l'article 15 Titre III de l'arrêté du 5 juillet 1984 relatif aux études doctorales

VU les rapports de présentation de

Messieurs G. CAMBON
J. FREHEL

Monsieur FERNANDES Antonio

est autorisé(e) à présenter une thèse en soutenance en vue de l'obtention du diplôme de DOCTEUR de L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, spécialité MICROELECTRONIQUE.

Fait à Grenoble, le 24 Août 1988

Georges LESPINARD
Président
de l'Institut National Polytechnique
de Grenoble
P.O. le Vice-Président,

RESUME.

Dans ce travail, nous avons effectué une étude complète de la testabilité des PLAs. Pour les trois classes de test (hors ligne, en ligne et unifié) nous avons proposé des schémas de test dont la compatibilité avec les PLAs optimisés a été étudiée.

En ce qui concerne le test hors ligne, le schéma de test proposé apporte une amélioration vis-a-vis des schémas publiés dans la littérature tout en gardant la compatibilité avec les PLAs optimisés.

En ce qui concerne le test en ligne, l'application de deux schémas de test aux PLAs optimisés a été analysée: le schéma utilisant des codes non ordonnés et le schéma utilisant des codes détectant des erreurs simples. Un nouvel algorithme pour la génération des codes non ordonnés a été proposé.

En ce qui concerne le test unifié, le test en ligne est assuré par la méthode utilisant des codes non ordonnés. Deux méthodes sont proposées pour la génération des vecteurs d'entrée pendant le test hors ligne: le test exhaustif et le test déterministe.

MOTS-CLES:

PLA, Circuits autotestables, Circuits autocontrôlables, Test en ligne, Test hors ligne, Test unifié, Test, Test integré.