Test et diagnostic de cartes et de MCMs partiellement boundary scan

M. - H. Touati

THÈSE

présentée par

Mohamed Hédi TOUATI

pour obtenir le titre de **DOCTEUR**

de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

(arrêté ministériel du 30 mars 1992)

Spécialité : **Microélectronique**

TEST ET DIAGNOSTIC DE CARTES ET DE MCMS PARTIELLEMENT BOUNDARY SCAN

Date de Soutenance : 24 Janvier 1996

Composition du jury :

Président et Rapporteur :	M. Yves Bertrand
Rapporteur :	M. Adam Osseiran
Examinateurs	M. Bernard Courtois
	Mme Meryem Marzouki

Thèse préparée au sein du Laboratoire TIMA/INPG

إِلَى أُمِّي وَ أَبِي
إِلَى إِخوَتِي

A ma femme,

A ceux et celles qui m'ont aidé

Remerciements

Si le titre de docteur semble récompenser un travail individuel, il est évident que celui-ci n'aurait jamais pu aboutir sans le soutien moral ainsi que l'aide scientifique de plusieurs personnes

Je remercie monsieur Bernard Courtois, Directeur de recherches au CNRS et Directeur du Laboratoire TIMA, pour m'avoir accueilli et accepté de co−diriger ces travaux avec madame Meryem Marzouki, Chargée de recherches au CNRS et Responsable de groupe DCS, dont les conseils et les encouragements ont été très utiles dans l'orientation et l'avancement de mes travaux.

Ensuite, je tiens à remercier monsieur Yves Bertrand , Professeur à l'Université de Montpellier II pour m'avoir fait l'honneur d'être à la fois président de mon jury et rapporteur de cette thèse, ainsi que monsieur Adam Osseiran, Professeur à l'École d'Ingénieurs de Genève, qui lui aussi m'a fait l'honneur d'être rapporteur de cette thèse.

C'est dans une ambiance très amicale, surtout au sein du bureau 101 que ce travail s'est déroulé. Je profite de cette occasion pour adresser mes remerciements à tous mes collègues de bureau : Hong Ding, pour ses précieuses remarques et son humour ; Firas Mohammed pour sa collaboration à ce travail et Maher Rahmouni, le benjamin du bureau et son grand animateur.

Je remercie aussi tous mes autres copains et notamment Mohamed Romdhani pour ses conseils, Imed Moussa, Polen Kission et Walid Maroufi pour leur gentillesse, Richard Pistorius pour sa complicité, Marcelo Lubaszewski pour sa contribution, ainsi que Lotfi Mtibaa, Mohamed Abid et tous ceux que j'ai oublié de citer et qui se reconnaîtrons dans ces remerciements.

Finalement, j'adresse mes remerciements au reste du Laboratoire TIMA, et notamment sa partie administrative (Corinne, Patou, Chantal et Isabelle 1 & 2)

Résumé

Considérant les systèmes microélectroniques actuels, circuits comprenant des millions de transistors, cartes électroniques multi−couches et les MCMs (Modules Multi-puces), les activités de test et de diagnostic, que ce soit pour la validation de prototypes ou la maintenance, prennent de plus en plus d'importance et sont de plus en plus difficiles à réaliser.

Certes, l'adoption du standard IEEE 1149.1, plus connu sous le label "Boundary Scan" (BS) a permis de résoudre une grande partie des problèmes posés par les difficultés d'accès aux nœuds à tester, en remplaçant l'accès mécanique par un accès électronique. Mais actuellement le marché est loin d'être exclusivement fourni en composants munis de ce standard. Par conséquent, on assiste à l'apparition de systèmes hétérogènes du point de vue de la testabilité, composés de parties BS et d'autres non BS, pour lesquels il faut développer des méthodes de test et de diagnostic adaptées.

Cette thèse, qui se place précisément dans ce contexte, a pour objectif de traiter les problèmes de test et de diagnostic rencontrés dans ce type de systèmes.

Nous proposons dans le cadre de ce travail une méthodologie globale ainsi que son implémentation permettant de se rapprocher de cet objectif. Elle permet la génération et l'ordonnancement de séquences de test optimales permettant la détection de fautes à la fois dans les conglomérats de circuits BS et non BS, ainsi que sur leurs interconnexions. Les modèles de collage logique, coupure et court−circuit sont pris en compte.

Au niveau du diagnostic, une première estimation des candidats à la faute est effectuée à l'aide d'une approche semi−qualitative. Le diagnostic est ensuite raffiné à l'aide d'une stratégie de recherche des meilleurs nœuds à tester, basée sur l'utilisation de la logique floue.

Cette méthodologie, qui s'applique aussi bien aux cartes qu'aux MCMs, a été implémentée sous forme d'un outil interfacé avec des ATPGs commerciaux. Les résultats expérimentaux obtenus confirment la validité de l'approche.

Mots-Clefs: ATPGs, Boundary Scan, Test, Diagnostic, Cartes, MCMs.

Abstract

Today's Integrated Circuits are made of millions of transistors, Printed Circuit Boards (PCBs) can have up to ten layers allowing integration of a large number of components and new packaging technologies have appeared, such as 3D packaging or Multi-Chip Modules (MCMs). According to this reality, Test and Diagnostic tasks have become essential for prototype validation and maintenance.

The IEEE 1149.1 Boundary Scan Standard (BS) allows to cope with state-of-the art packaging and wiring technologies. Although research evolves from the assumption of full BS availability, industry must face a market which has just started integrating this standard. So, systems will still contain BS parts and non BS clusters in the near future.

This thesis aims at finding a global approach for the test and diagnosis of such heterogeneous systems.

We propose a methodology which unifies the test and diagnosis of components and their interconnects. It is based on a scheduling approach to test simultaneously both BS components and non BS clusters, as well as interconnects. Logical stuck-at faults, as well as short and open faults are addressed.

Diagnosis is achieved thanks to a novel approach combining structural description and qualitative reasoning to generate fault candidates. A fuzzy logic-based strategy for best test point finding is used to enhance the diagnosis accuracy and efficiency.

This methodology, which can be applied to both MCMs and PCBs, has been implemented in a software, nicely interfaced to commercial ATPGs. Obtained experimental results validate this methodology.

Keywords: ATPGs, Boundary Scan, Test, Diagnostis, PCBs, MCMs.

Table des Matières

Liste des Figures

Liste des Tableaux

Introduction

Introduction

Les progrès accomplis dans le domaine de l'intégration font que l'on a affaire à des circuits de plus en plus complexes, dont la taille est de plus en plus réduite et qui vont servir à composer des cartes de plus en plus compactes. Ces circuits posent souvent des problèmes de testabilité, surtout s'ils sont hautement séquentiels. Pour améliorer leur testabilité et permettre d'effectuer le test à un coût raisonnable, le moyen le plus adapté est de poser le problème de la testabilité et de la maintenance dès la phase de conception du circuit.

Le problème se pose de manière similaire pour les cartes, étant donné qu'actuellement, d'un côté l'encapsulation des circuits tend à devenir de plus en plus compacte, et d'un autre côté, grâce notamment aux nouvelles techniques de montage des circuits sur les plaques imprimées telle que la technique CMS (Composants Montés en Surface) qui permet de monter les circuits sur les deux cotés de la carte, les carte électroniques deviennent de plus en plus miniaturisées [Cou94]. Ceci pose donc le problème d'accès aux broches des circuits. L'utilisation de lits à clous ("bed of nails") devient extrêmement difficile voire impossible à cause des risques de court-circuit au cours du contact testeur/broche, et d'une façon générale des dommages qui peuvent être occasionnés aux circuits. Ces court-circuits peuvent s'avérer aussi néfastes pour le dispositif à tester que pour le testeur lui-même s'il ne dispose pas de dispositif de protection.

L'apparition récente de la technologie MCM ("Multi-Chip Module" ou Circuit Multi-Puces) qui consiste à regrouper sur le même substrat plusieurs puces avec une forte densité d'interconnexion, a conforté davantage l'idée que la solution à ces problèmes de test passe par l'adoption de l'approche de conception en vue du test (DFT pour "Design For Testability"). Cette approche consiste à inclure la testabilité comme contrainte lors de la phase de conception.

Parmi les dispositifs de DFT mis au point à ce jour, on notera que deux techniques sont de plus en plus utilisées pour leur efficacité :

- la technique du "Scan Path" qui permet d'accéder aux nœuds internes du circuit tout en limitant le nombre d'entrées/sorties supplémentaires prévues à cet effet. Dans ce concept, le circuit est conçu de manière à présenter deux modes de fonctionnement : un mode de fonctionnement normal et un mode de test dans lequel toutes les bascules élémentaires sont interconnectées sous forme d'un ou de plusieurs registres à décalage. Pour obtenir cette configuration, on remplace les bascules utilisées usuellement par des bascules "scan" : ce sont des bascules disposant de deux entrées sélectionnables (par exemple en adoptant un multiplexeur en entrée de la bascule). Selon le mode de fonctionnement, c'est soit l'entrée test soit l'entrée de mode de fonctionnement normal qui est activée. En mode test, un vecteur est chargé en série à l'intérieur des bascules reliées entre elles pour former un registre à décalage. Après retour au mode de fonctionnement normal, le vecteur ainsi chargé est transmis à l'intérieur du circuit, et le résultat obtenu modifie ainsi le contenu des différentes bascules du circuit. Une fois le circuit stabilisé, les valeurs présentes sur les bascules sont extraites du circuit en mode série, puis comparées à la réponse correcte attendue.

- La deuxième technique utilisée est connue sous l'appellation d'auto−test intégré ou BIST ("Built-In Self-Test"). Elle consiste à intégrer dans le circuit les mécanismes nécessaires à son propre test. Un exemple est celui de l'analyse de signature qui consiste à inclure dans le circuit un générateur de stimuli pour produire des séquences de test, et un analyseur de signature pour compacter et analyser les réponses du circuit aux stimuli qui lui sont appliqués.

Malgré le surcoût engendré par la surface additionnelle, ces approches ont suscité un énorme intérêt auprès des concepteurs et des fabricants. Car en plus du fait que le "Scan Path" apporte une amélioration sensible à la testabilité des circuits séquentiels, il peut aussi présenter un intérêt certain pour le test des cartes électroniques et les MCMs. A ce niveau, une utilisation conjointe avec les techniques de BIST permet une importante réduction du coût du test et de la maintenance de ces cartes, et d'une manière générale des systèmes.

L'adoption du standard IEEE 1149.1 appelé "Boundary Scan", qu'on développera dans le prochain chapitre a permis de résoudre les problèmes inhérents aux difficultés d'accès aux nœuds des circuits. Mais la réalité actuelle du marché fait que seulement une faible (quoiqu'en augmentation régulière) proportion de circuits intègre ce standard. Par conséquent, nous allons vraisemblablement passer par une phase transitoire dans laquelle nous serons confrontés à des systèmes hybrides, constitués par des parties avec et d'autres sans Boundary Scan.

Le travail effectué au cours de cette thèse se situe précisément dans ce contexte. Nous avons mis au point une méthodologie visant l'unification du test et du diagnostic de ce type de systèmes. Le présent document détaille tous les aspects de cette méthodologie ainsi que son implémentation sous forme d'un logiciel. Il présente également les résultats que nous avons pu obtenir à l'aide de ce logiciels, sur différents benchmarks.

Le premier chapitre de ce document est consacré à la présentation du standard IEEE 1149.1 ainsi que du test hors ligne de cartes développées avec ce standard.

Dans le Chapitre 2, nous développons la méthodologie pour la génération de vecteurs de test pour les cartes hétérogènes.

Dans le chapitre 3 et après une présentation des MCMs, nous décrivons l'extension de la méthodologie développée dans le chapitre 2 au cas des MCMs.

Le problème du diagnostic est traité dans le chapitre 4. Une approche semi–qualitative pour la génération des candidats et la mise à jour des estimations y est présentée.

Dans le but d'améliorer le diagnostic, nous proposons une statégie de recherche du meilleur nœud à tester. Cette stratégie sera développée dans le chapitre 5.

Le chapitre 6 est consacré à la présentation de TANDEM, le logiciel qui a été développé au cours de cette thèse. Il est suivi du chapitre 7 dans lequel nous avons regroupé les résultats expérimentaux obtenus à l'aide de ce logiciel.

Finalement, nous concluons sur le bilan du travail effectué et les perspectives pour les futurs travaux de recherches dans ce domaine.

Chapitre 1

L'approche "Boundary Scan"

Chapitre 1

L'approche "Boundary Scan"

1.1 Introduction

Étant donnés les résultats intéressants obtenus avec les techniques de scan et l'intérêt porté par les industriels à ce type de conception en vue de la testabilité (DFT : "Design For Testability"), il était devenu impératif de songer à établir un standard de test valable autant pour les circuits que pour les cartes elles mêmes. Ce standard doit être en même temps capable de répondre aux problèmes posés par le test aujourd'hui, et assez évolutif pour s'adapter au défi de la complexité accrue que connaîtront les circuits VLSI ainsi que les cartes de demain. Des travaux ont été entrepris au sein du groupe JTAG ("Joint Test Action Group"), comprenant des compagnies et des institutions européennes et nord américaines [BeO91] pour la définition d'un standard de test répondant aux critères suivants :

– Faciliter les accès aux broches des circuits de la carte
– Permettre le test des interconnexions sur la carte
– Faciliter l'utilisation des dispositifs de test intégrés aux circuits
– Minimiser le coût en terme d'équipements de test (testeurs, sondes, etc ...)
– Diminuer le nombre de broches supplémentaires nécessaires à sa mise en œuvre.

Pour ces diverses raisons, l'adoption de la norme **IEEE 1149.1** [IEEE90] plus connue sous le label "Boundary Scan" est intervenue en 1990. Il s'agit de la définition d'une architecture qui permet le test à plusieurs niveaux hiérarchiques (circuit, carte et système) sans se soucier du problème d'accès mécanique aux nœuds internes. Cette approche permet aussi l'élaboration d'une stratégie globale de test indépendante de l'origine industrielle du produit. Elle présente deux modes d'opération majeurs qui sont:

– Le mode **"Non invasive"** dans lequel le dispositif de test est complètement transparent par rapport à l'application.

– Le mode **"Pin_Permission"** pendant lequel la logique interne de l'application est isolée du monde extérieur.

1.2 L'architecture Boundary Scan

La figure 1.1 montre l'architecture générale d'une carte munie du dispositif BS.

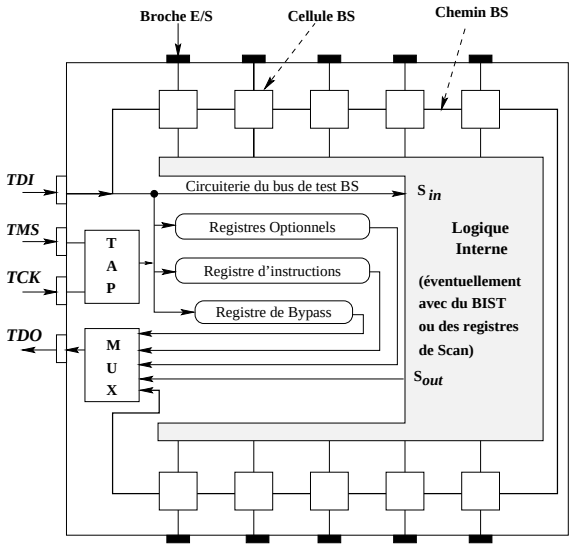

Figure 1.1: Architecture générale d'une carte BS

Les principaux éléments qui entrent dans sa composition sont les suivants :

1. La cellule "Boundary Scan" de base
2. Le bus de test
3. Le contrôleur de bus de test

4. Les registres d'instructions et de données

Nous verrons dans la suite comment tous ces éléments sont assemblés pour former l'architecture en question.

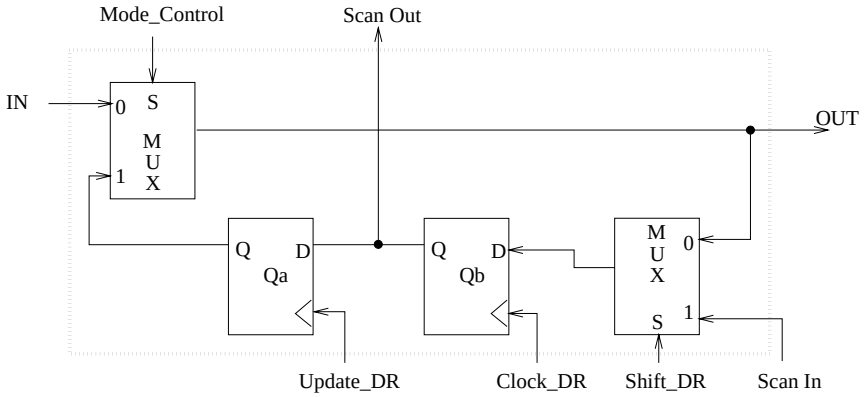

Figure 1.2: Exemple de cellule BS

1.2.1 La cellule Boundary Scan

La Figure 1.2 montre un exemple de ce type de cellule. On notera que l'architecture de cette cellule n'est pas imposée par le standard, et que par conséquent le choix est laissé au concepteur. Dans notre cas, elle est constituée de deux bascules et de deux multiplexeurs. Ces derniers assurent l'aiguillage à l'intérieur de la cellule. Le premier contrôle le signal d'entrée et le deuxième contrôle la bascule de décalage. Cette cellule présente quatre modes de fonctionnement décrits et illustrés ci-dessous :

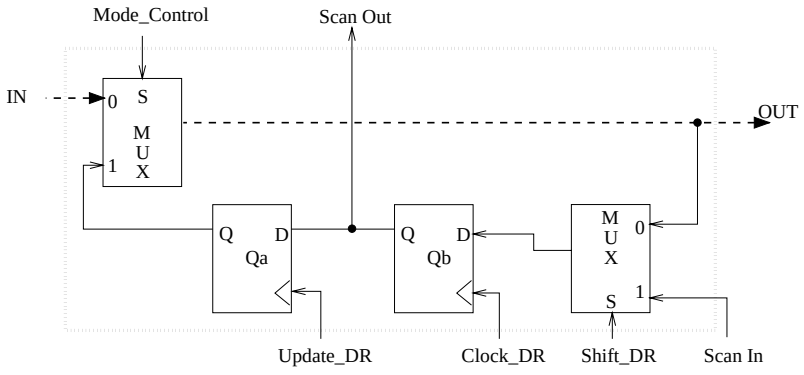

Figure 1.3: Cellule BS en mode de fonctionnement normal

– le mode normal (figure 1.3) : quand le signal **Mode_Control** a la valeur 0, l'entrée de données (IN) est simplement transférée à la sortie (OUT) de la cellule BS.

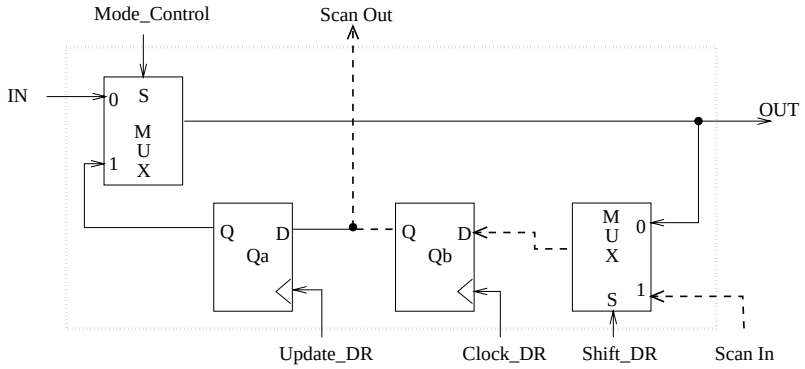

Figure 1.4: Cellule BS en mode Scan

– le mode Scan (figure 1.4) : quand le signal **Shift_DR** a la valeur 1 et le signal **TCK** est appliqué sur **Clock_DR**. L'entrée de test (Scan In) est décalée vers la sortie (Scan Out) de la cellule BS.

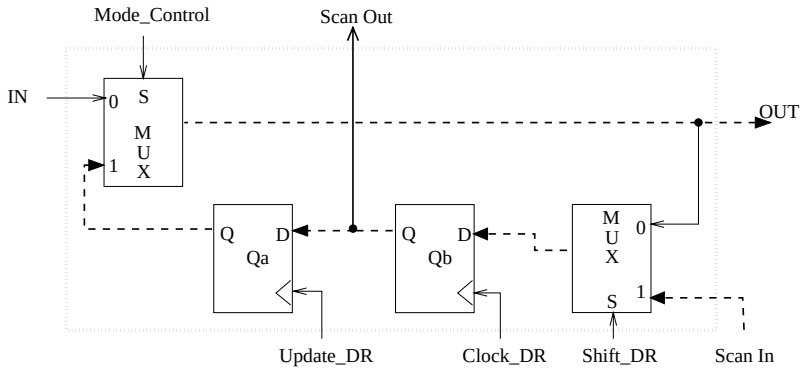

Figure 1.5: Cellule BS en mode "Control"

– le mode de contrôle (figure 1.5) quand le signal **Mode_Control** a la valeur 1 et le signal **TCK** est appliqué sur **Clock_DR** et **Update_DR**. L'entrée de test (Scan In) est transférée sur la sortie (OUT) de la cellule BS afin d'imposer sa valeur en entrée de la logique interne du circuit.

– le mode d'observation (figure 1.6) quand les signaux **Mode_Control** et **Shift_DR** ont la valeur 0 et le signal **TCK** est appliqué sur **Clock_DR**. L'entrée de donnée (IN) est transférée vers la sortie de test de la cellule BS (Scan Out), afin d'observer sa valeur.

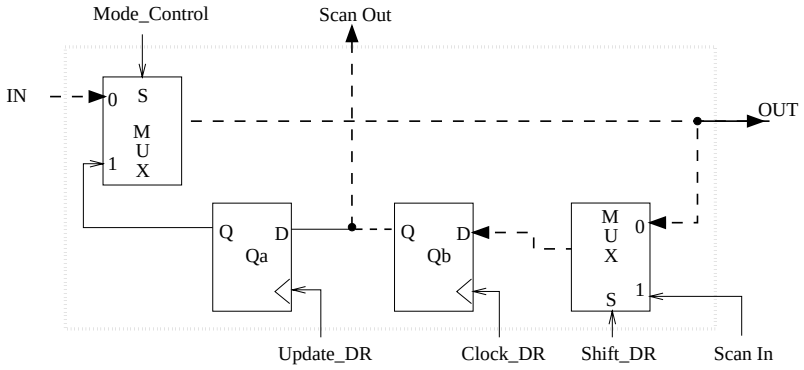

Figure 1.6: Cellule BS en mode d'observation

Une cellule BS est insérée entre chaque broche fonctionnelle et la logique interne du circuit (d'où la notion de "Boundary"). Ensuite elles sont connectées en série de façon à former le registre Boundary Scan. Il s'agit d'un registre à décalage cheminant le long du périmètre du circuit et de la carte.

1.2.2 Le bus de test

Il est composé au minimum de quatre signaux, il est directement commandé soit par un testeur, soit par un gestionnaire BS. Ces quatre signaux se classent en deux catégories :

– Les signaux de contrôle **TMS** et **TCK**, il s'agit de deux signaux généraux contrôlés par le testeur.

– Les signaux de données de test **TDI** et **TDO**, il s'agit de l'entrée et de la sortie du chemin série formé par les divers registres BS des circuits de la carte.

Ces quatre signaux sont définis de la manière suivante dans la norme BS :

– TCK (Test Clock) : C'est l'horloge du contrôleur. Elle est indépendante de celle du circuit et n'est utilisée qu'au cours du processus de test BS.

– TDI (Test Data Input) : Les données ou les instructions de test sont reçues via cette ligne et sont dirigées vers le registre approprié.

– TDO (Test Data Output) : C'est la sortie par laquelle le contenu d'un registre (instruction ou donnée) est transmis à l'extérieur en mode série.

– TMS (Test Mode Select) : Ce signal est utilisé pour contrôler la logique de test (le

contrôleur TAP) afin de lui signifier la nature des données à recevoir (c'est−à−dire des données de type instruction ou de type données de test).
Outre ces signaux obligatoires, le bus de test peut comporter un cinquième signal :
− TRST (Test Reset) : C'est un signal optionnel. Il permet d'initialiser le contrôleur TAP à l'état "Test Logic Reset". En l'absense du signal TRST, cet état est atteint par le maintien de TMS=1 pendant 5 cycles de TCK.

1.2.3 le contrôleur de bus de test (TAP)

Le TAP (pour "Test Access Port") est une machine d'états finis (16 états). Elle est synchronisée par le signal d'horloge TCK. Elle comprend une entrée unique (TMS), et ses sorties sont des signaux qui sont associés aux différents états internes. Le contrôleur ne peut changer d'état que sur une impulsion de l'horloge TCK. L'état suivant est fonction de la valeur logique portée sur la ligne de contrôle TMS. Le diagramme d'états du TAP est représenté dans la figure 1.7. On notera la présence de deux sous−ensembles, l'un relatif au contrôle du registre d'instructions et l'autre à celui de données. Dans l'état **Test_Logic_Reset**, la circuiterie de test est initialisée tandis que le fonctionnement normal de l'application reste intact. Quant à l'état **Run_Test/Idle**, il peut soit s'insérer entre les opérations de scan (état "Idle"), soit permettre l'exécution des processus de test internes tels que le test intégré ou le scan interne (état Run_Test).

1.2.4 Les Registres internes

Le dispositif comprend deux types de registres : le registre d'instruction et les registres de données.

1.2.4.1 Le registre d'instruction

Ce registre comprend lui-même deux parties, l'une sérielle et l'autre parallèle de façon à pouvoir stocker une nouvelle instruction tout en maintenant la précédente. Au début du cycle de décalage d'instruction, il sera chargé avec l'état du test précédent.

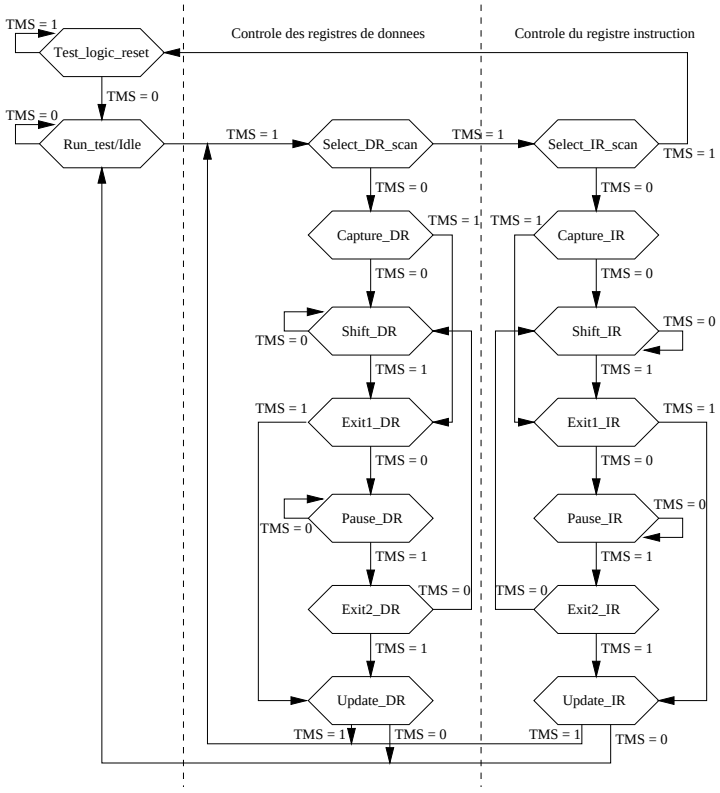

Figure 1.7: Diagramme d'état du contrôleur TAP

Ensuite, cet état sera décalé vers TDO en même temps qu'arrivera la nouvelle instruction par TDI. Chaque instruction spécifie les opérations à réaliser ainsi que le registre de données sélectionné entre les broches TDI et TDO. Trois instructions sont rendues obligatoires par le standard : *BYPASS, EXTEST et SAMPLE/PRELOAD* alors que les instructions *INTEST et RUNBIST* sont uniquement recommandées.

L'instruction *SAMPLE* permet d'échantillonner les données présentes sur les bornes d'entrées/sorties de la carte. Elle est utilisée lors d'un test fonctionnel. *PRELOAD* sert à charger des valeurs dans le registre de données. Elle est utilisée pour initialiser la carte dans un état sûr (pour éviter les dommages éventuels relatifs à des conflits de bus) avant une opération de test. Il est à noter que ces deux instructions n'affectent pas le mode de fonctionnement normal de la carte.

L'instruction *EXTEST* est utilisée pour le test de la circuiterie et des connexions entre les circuits BS.

L'instruction *INTEST* est consacrée au test interne des circuits. Elle consiste à appliquer les vecteurs de test sur les cellules BS d'entrée du circuit et à recueillir les réponses sur ses cellules BS de sortie.

1.2.4.2 Les registres de données

Ils doivent être au minimum au nombre de deux :
– le registre de *BYPASS* qui sert à court-circuiter le circuit de façon à l'isoler du chemin de balayage de la carte. Ceci permet de réduire le temps de test en permettant d'accéder directement à un circuit donné de la carte.
Ce registre est composé d'une seule bascule (figure 1.8). Elle relie, dans le même circuit, TDI à TDO lors de l'instruction *BYPASS*.

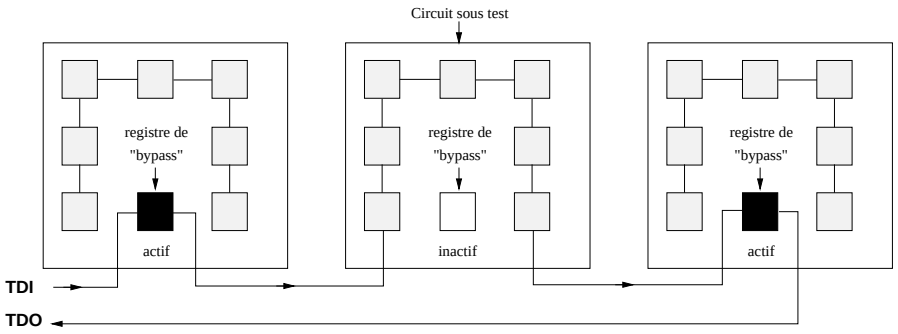

Figure 1.8: Utilisation du registre de BYPASS

– le registre de Boundary Scan, constitué par les cellules Boundary Scan interconnectées en registre à décalage (figure 1.1).

D'autres registres peuvent être connectés en option par l'utilisateur entre les ports TDI et TDO. On citera pour l'exemple le registre d'identification qui comportera entre autres, l'identité du fabricant, la version du circuit, etc. L'instruction *IDCODE* sera associée à ce registre.

La figure 1.9 présente un exemple de carte BS avec les chemins BS au niveau circuit et au niveau carte.

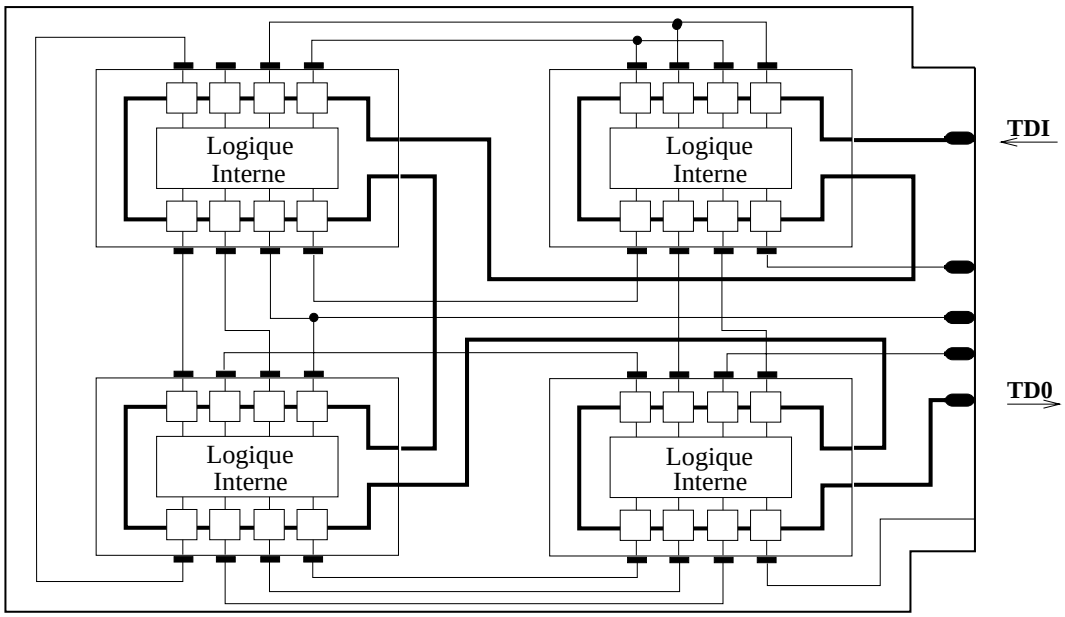

Figure 1.9: Exemple de carte BS

1.3 Les modes de test

L'architecture Boundary Scan permet trois types de test :

1.3.1 Le test externe

Dans ce mode de test (figure 1.10), on teste directement les interconnexions entre les différents circuits de la carte. Ceci permet de réaliser toutes les vérifications concernant les problèmes de court−circuit (shorts) et circuit−ouvert (opens) pouvant émaner de soudures sèches ou de pistes coupées par exemple.

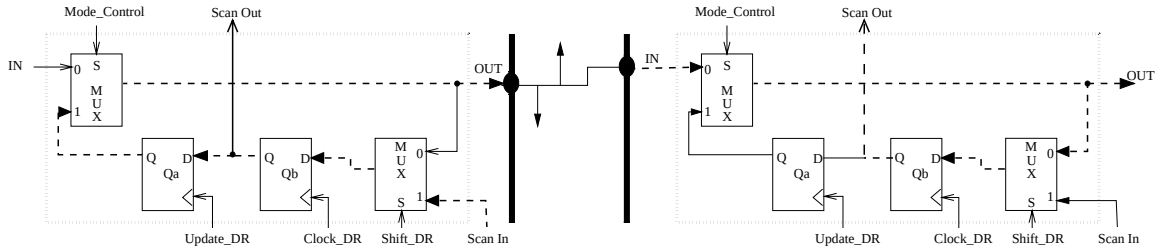

Figure 1.10: Configuration pour le test des interconnexions

Ce type de test repose sur les facilités de contrôlabilité qui permettent d'appliquer une valeur sur la sortie de la cellule BS en amont, et aux facilités d'observabilité qui

permettent d'observer la valeur présente sur l'entrée de la cellule en aval. Les valeurs qu'on applique sont appelées "Parallel Test Vector" (PTV), les valeurs recueillies sont désignées par "Parallel Response Vector" (PRV). Il arrive souvent que l'on ait besoin de plusieurs PTVs pour effectuer le test.

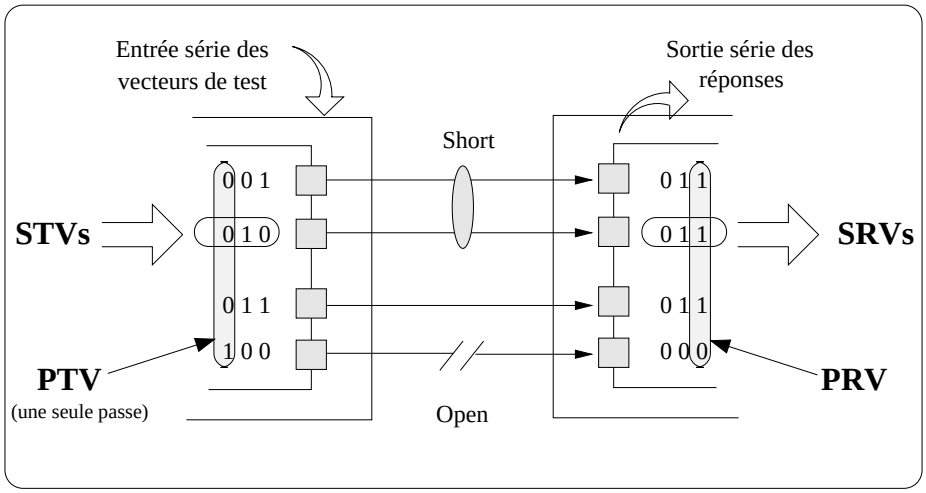

Figure 1.11: Test des connexions

Dans ce cas la suite des valeurs à appliquer à chaque nœud est désignée par "Sequential Test Vector" (STV). De la même manière, la réponse est appelée "Sequential Response Vector" (SRV) (figure 1.11). Une sélection judicieuse des STVs permet de détecter les fautes sur les connexions. Plusieurs algorithmes simples ont été définis pour ce faire, et sont décrits dans la littérature [Wag87] [HRA88] [JaY89].

1.3.2 Le test interne

Ce mode de test permet l'obtention d'une totale contrôlabilité et observabilité des diverses broches de chaque circuit. La logique interne de chaque circuit est commandée par les cellules d'entrée du Boundary Scan qui lui transmettent les stimuli. Les réponses sont observées sur les cellules de sortie (figure 1.12). Le circuit lui−même peut contenir des dispositifs internes de test du type Scan Path ou test intégré, dans ce cas les opérations de test interne sont effectuées durant ce mode de test.

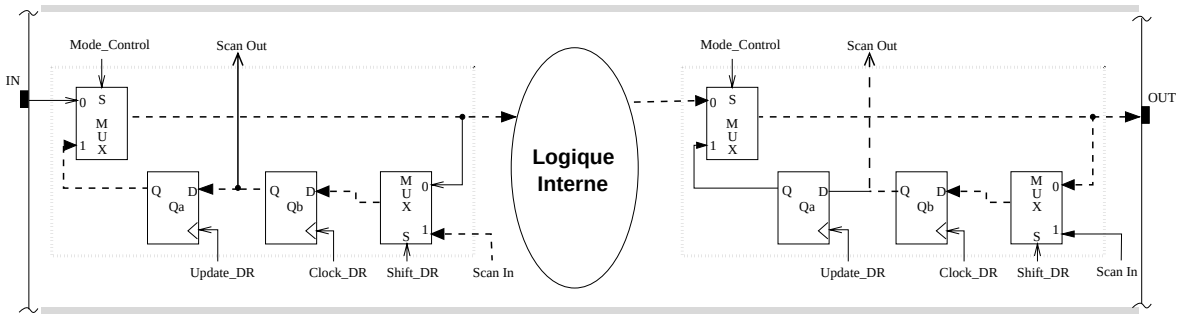

Figure 1.12: Configuration en mode test interne

1.3.3 Le test d'échantillonnage

C'est le troisième mode de test, il permet de stocker à la volée des données en entrée et en sortie des circuits supportant la norme IEEE 1149.1 (figure 1.13). Ces données peuvent être analysées pour vérifier le fonctionnement dynamique du circuit (identification de fautes de délai par exemple). Ce mode de test n'affecte pas le mode de fonctionnement normal du circuit car les données peuvent être sorties pendant que le système continue de fonctionner normalement. Ceci nécessite néanmoins que la circuiterie de scan ait été conçue de manière à ne pas interférer avec le mode de fonctionnement normal lorsqu'il y a transmission de données série.

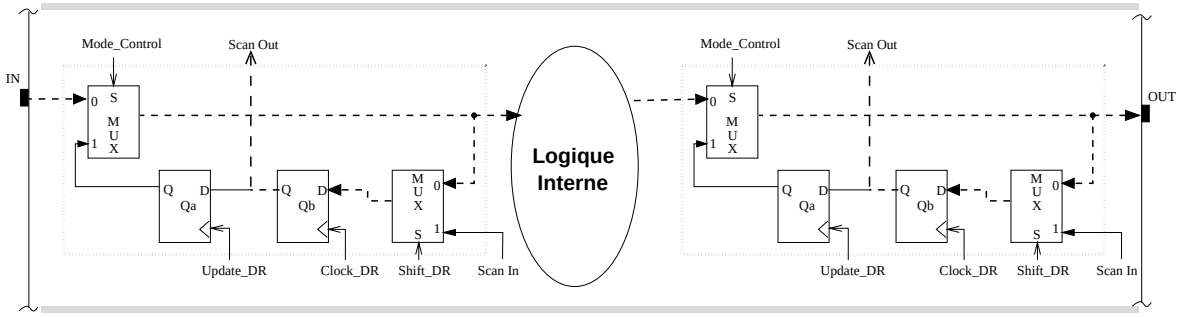

Figure 1.13: Configuration en mode échantillonnage

1.4 Stratégie de test des cartes "Boundary scan"

Du point de vue testabilité des cartes, l'idéal serait de disposer de cartes munies du dispositif Boundary Scan, et composées uniquement de circuits Boundary Scan. De cette manière tous les problèmes de contrôlabilité et d'observabilité seraient résolus.

Dans l'hypothèse d'une carte 100% "Boundary scan", une séquence globale de test est proposée dans [TuY89]. Il s'agit d'un test structurel qui se divise en trois étapes principales : une première étape d'initialisation et de vérification de la circuiterie de test, une étape de test des connexions et une dernière étape de test des circuits qui composent la carte. Le déroulement de ces étapes s'effectue de la manière suivante :

1.4.1 initialisation et vérification de la circuiterie de test

Cette étape est nécessaire dans le cas d'une chaîne afin de s'assurer que le dispositif de test est bien en place et qu'il ne présente aucune anomalie. Pour cette raison ce test est appelé test d'intégrité (figure 1.14). Il se déroule de la façon suivante :

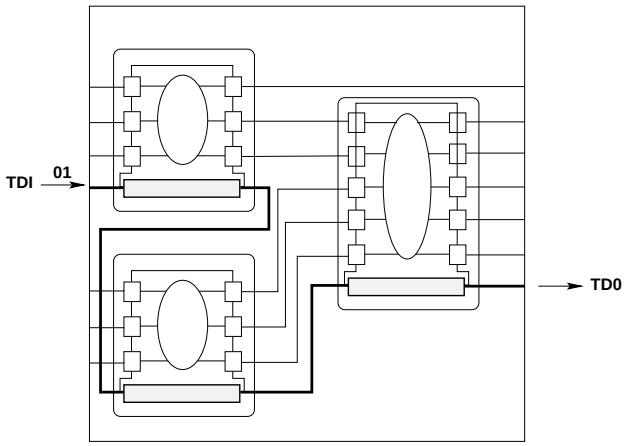

Figure 1.14: Test d'intégrité

- atteindre l'état "Test Logic Reset" dans tous les contrôleurs TAP sur la carte (5 cycles de TCK avec TMS = '1' ou encore activation de TRST)
- tester le chemin de balayage de la carte à travers les registres d'instructions. Pour cela, on écrit une séquence 'X..X01X...X01' dans tous les registres d'instructions et on vérifie le résultat sur la sortie TDO.
- précharger les registres BS de sortie avec un état sûr afin d'éviter les dommages qui pourraient être causés par des conflits de bus (on peut utiliser à cet effet l'instruction PRELOAD)
- Identifier éventuellement les circuits assemblés sur la carte par l'utilisation de l'instruction IDCODE prévue à cet effet)

- vérifier l'opération des registres de données (utilisation des instructions BYPASS, ..)

1.4.2 le test des connexions

C'est la partie la plus intéressante pour le test des cartes électroniques. Cette étape se déroule de la manière suivante :
- appliquer à toutes les connexions une séquence de test pour la détection des fautes du modèle (utilisation de EXTEST)
- en cas de faute, appliquer aux connexions suspectes une séquence de test pour le diagnostic de fautes (on utilise alors conjointement les instructions EXTEST et BYPASS).

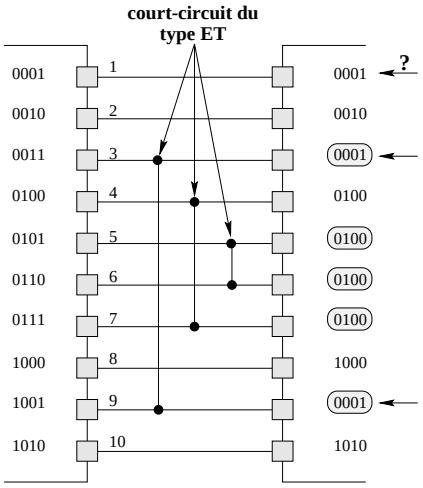

Figure 1.15: cas d'aliasing

Néanmoins, le diagnostic des court−circuits demeure une tâche extrêmement difficile. Ceci est surtout dû au fait que ce sont des fautes d'interaction entre connexions et que leur modélisation est un peu plus délicate à définir contrairement à celle des fautes de collage ou de circuits−ouverts. Actuellement deux modèles de court−circuits sont utilisés : Le court−circuit du type OU ("or-short") et celui du type ET ("and-short"). Avec ces deux modèles, on arrive à déterminer des séquences de test à l'aide d'algorithmes comme le comptage binaire [Kau74], le comptage binaire suivi de son complément [Wag87] ou le "0/1" baladeur [HRA88]. Mais le diagnostic demeure

impossible avec la première à cause du phénomène de recouvrement ("aliasing syndrome") [JaY89]. Ce phénomène consiste à avoir des signatures identiques pour les nœuds corrects et les nœuds fautifs. Dans l'exemple de la figure 1.15, on remarque que la connexion numéro 1 fait partie des nœuds suspectés alors qu'elle ne participe pas au court−circuit. Par contre, le comptage binaire suivi de son complément permet le diagnostic de "l'aliasing" mais pas celui du "Confounding Syndrome" [JaY89].

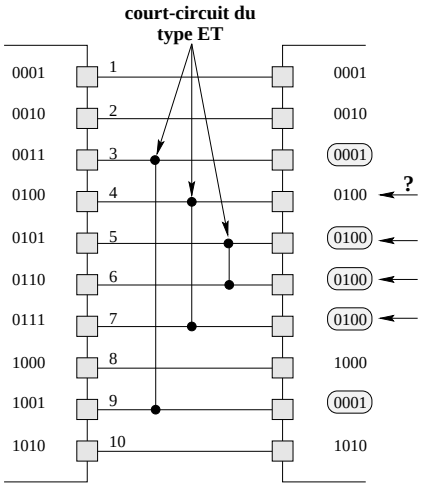

Figure 1.16: "confounding" : cas 1

Ce phénomène intervient lorsque deux ou plusieurs fautes différentes produisent la même réponse erronée. La figure 1.16 montre un exemple de ce syndrome. Dans ce cas de figure, la faute sur le nœud numéro 4 n'est pas détectée d'une manière formelle par la séquence du comptage binaire. Si on applique le complément comme dans le figure 1.17, on remarquera que bien que les deux court−circuits (4,7) et (5,6) soient indépendants, ils sont indiscernables.

1.4.3 le test des circuits

Ce mode de test utilise les facilités obtenues grâce au registre BS pour rendre les accès des circuits composant la carte totalement contrôlables et observables. Cette opération peut se faire en deux étapes :

- utilisation des instructions RUNBIST et BYPASS pour déclencher l'autotest des circuits qui intègrent cette fonctionnalité

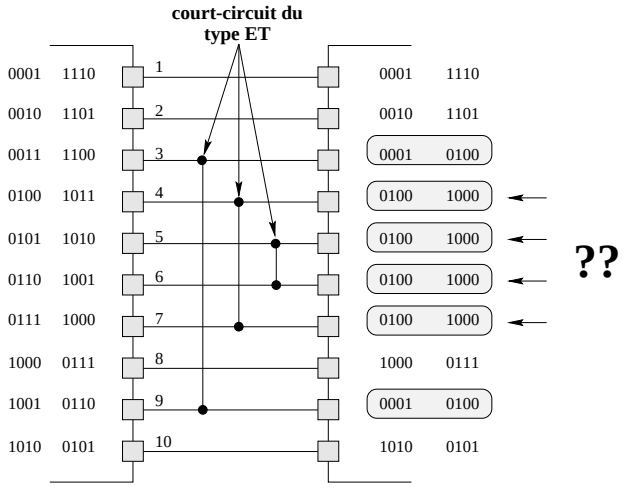

Figure 1.17: "confounding" : cas 2

- utilisation des instructions INTEST et BYPASS pour tester les autres circuits.

1.5 Conclusion

Après cette présentation des possibilités offertes par le standard IEEE 1149.1, avec surtout l'efficacité de son mode de test externe et son ouverture sur les dispositifs d'auto-test des circuits, il est maintenant clair qu'une grande partie des problèmes du test des cartes se trouve pratiquement résolue. Le seul problème qui demeure est celui du coût et par conséquent de la rentabilité de l'intégration de ce standard sur les systèmes électroniques.

Après une phase d'hésitation constatée à l'apparition de ce standard, les industriels, après des études d'évaluation [TFH92] sont de plus en plus convaincus de la parfaite rentabilité de cette intégration. Cette rentabilité sera d'autant plus grande quand tous les circuits de la cartes seront au standard BS.

Le seul inconvénient actuel est que le marché aura besoin d'une période d'adaptation, et l'on ne peut espérer atteindre "l'ère du tout Boundary Scan" avant quelques années. D'ici là, on verra apparaître des cartes mixtes, composées d'ensembles BS et d'autres non BS. Par conséquent, il faut donc réfléchir à la mise au point de méthodes efficaces permettant de tirer profit du dispositif BS pour le test des parties non BS.

Chapitre 2

Méthodologie pour le test des cartes hétérogènes

Chapitre 2

Méthodologie pour le test des cartes hétérogènes

2.1 Introduction

Actuellement, le standard IEEE 1149.1 est unanimement reconnu comme étant la meilleure solution pour la simplification du test des cartes électroniques. Grâce aux multiples possibilités que comprend ce concept, le test des cartes composées de circuits complexes devient aisé. On a vu apparaître récemment sur le marché des systèmes très compacts et peu chers (composés de micro−ordinateurs et d'interfaces d'entrées/sorties) capables de gérer le test de cartes BS très complexes. Plusieurs travaux de recherche ont été entrepris autour de ce standard, et des solutions ont été avancées aux divers cas de figure qu'on peut rencontrer. Mais les solutions proposées reposent toutes sur le fait que les éléments traités bénéficient d'une implantation totale du standard, ce qui est loin d'être le cas en réalité. En effet, le surcoût engendré par l'implantation, conjugué aux délais d'adaptation nécessaires aux fabricants, fait qu'actuellement, seulement une partie des circuits disponibles sur le marché sont munis de ce dispositif. Par conséquent, la grande majorité des cartes sont actuellement composées de circuits BS et de conglomérats non BS. La figure 2.1 montre un exemple

de ce type de cartes qu'on qualifiera de mixtes. Dans ce type de carte, des problèmes de contrôlabilité et d'observabilité se posent au niveau des nœuds qui, pour diverses raisons, sont inaccessibles aux équipements de test. Des problèmes d'initialisation concernant les interconnexions mixtes sont aussi à craindre. Ces problèmes peuvent provenir du fait que les circuits reliés aux connexions BS peuvent réagir aux changements de signaux qui se produisent sur ces connexions lors de la mise sous tension du système.

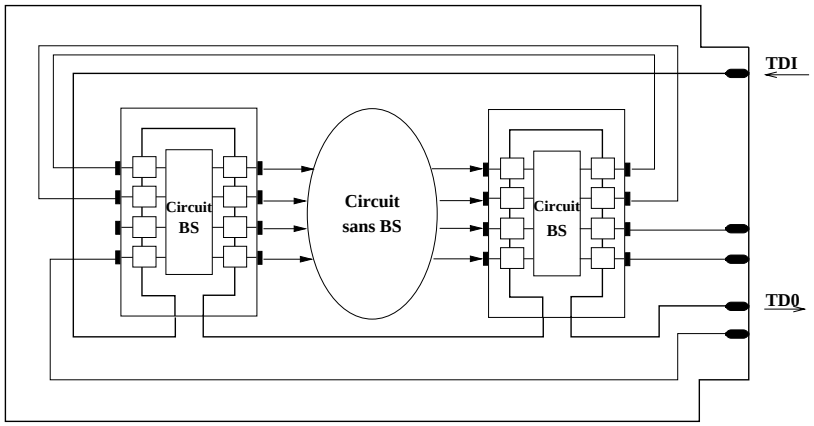

Figure 2.1: Exemple de carte partiellement BS

Deux stratégies peuvent être envisagées en vue du test de ce type de cartes :

– une première stratégie qu'on peut qualifier d'hybride, et qui consiste à tester séparément les composants BS du reste des composants. Les éléments BS ainsi que leurs interconnexions seront testés sériellement à travers le registre BS tandis que le restant des éléments est testé à l'aide de techniques de test du type "In Circuit Testing" ou de test fonctionnel.

– une seconde stratégie, plus intéressante du point de vue de l'unification du test et du diagnostic, et qui consiste à disposer autour de chaque conglomérat des dispositifs spécifiques permettant de faciliter la tâche du testeur [Han89]. En général, le type de dispositif le plus utilisé est l'insertion de points de test accessibles. De cette manière, le testeur peut entrer en contact avec ces dispositifs par l'intermédiaire de "lit-a-clous" ou de sondes individuelles. Mais les technologies de montage actuelles ou bien

la surface additionnelle nécessaire à l'implantation de connecteurs supplémentaires peuvent rendre impossible l'adoption d'une telle approche.

Une alternative à ces deux approches est proposée dans [HAR89]. Elle consiste à utiliser chaque fois que cela est possible les ressources BS des circuits voisins pour le test des conglomérats non BS. Concrètement, Ceci revient à utiliser les nœuds BS comme broches virtuelles d'un testeur. Le travail que nous présentons dans ce chapitre se place précisément dans ce contexte. Nous proposons une méthodologie pragmatique visant l'unification du test et du diagnostic dans les cartes partiellement BS.

2.2 Description de la méthodologie

L'approche que nous proposons se fixe plusieurs objectifs :

1) mettre à jour plusieurs types d'anomalie tels que :

- les circuits défectueux composant la carte ainsi que la nature de leurs fautes internes (fautes de type 1).
- les défauts d'interconnexion (short, open) dans un même conglomérat (fautes de type 2).
- les fautes d'interaction (short) entre les divers conglomérats de la carte (fautes de type 3).
- les défauts d'interconnexion entres les conglomérats et des connexions Boundary Scan de la carte (fautes de type 4).
- les fautes de court−circuits (short) entre les connexions BS elles mêmes (fautes de type 5).

2) procéder simultanément au test des nœuds internes ainsi que des interconnexions des conglomérats par le biais d'un test fonctionnel qui apparaît comme le seul

moyen de vérification lorsque des problèmes de contrôlabilité ou d'observabilité directe des nœuds internes se posent.

3) réduire au maximum le temps d'application du test en effectuant aussi souvent que possible le test en parallèle de tous les modules, ceci afin d'éviter le plus possible l'accès série à la chaîne de scan.

4) utiliser aussi souvent que possible les mêmes séquences dans les procédures de test et celles du diagnostic.

Afin d'atteindre ces objectifs, nous avons décomposé le travail en plusieurs tâches comme indiqué sur l'organigramme de la figure 2.2. Ces différentes tâches seront détaillées dans les prochains paragraphes.

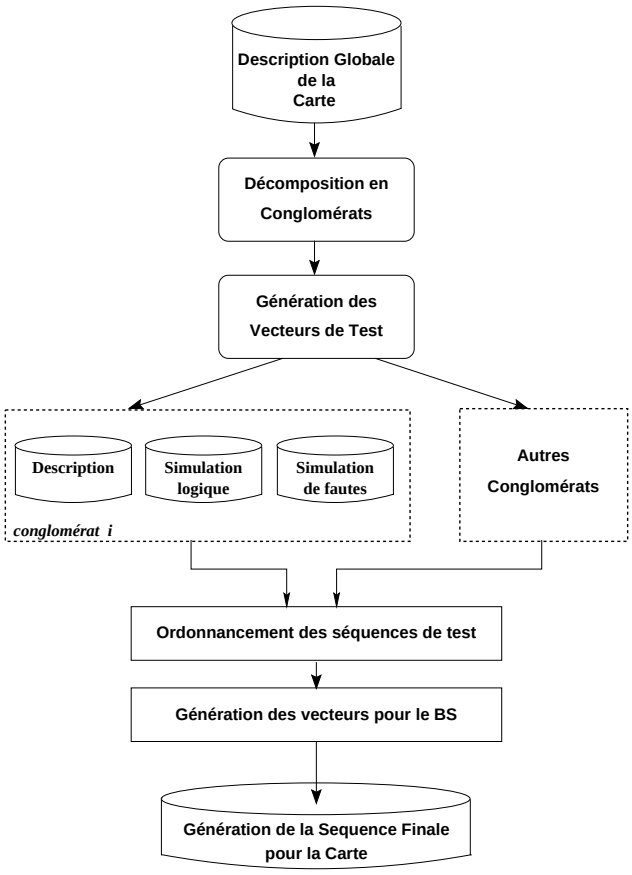

Figure 2.2: Organigramme général de la méthodologie

2.3 Génération des vecteurs de test pour un conglomérat

Une fois la décomposition de la carte en conglomérats achevée, la première tâche à effectuer consiste à générer des vecteurs de test pour chaque conglomérat. Les vecteurs de test obtenus doivent assurer une couverture aussi élevée que possible des fautes (collages, court−circuits et circuits ouverts) internes à ce conglomérat.

Étant donné qu'un conglomérat peut être composé de plusieurs modules interconnectés, deux cas de figure peuvent se présenter :

– la description au niveau logique (netlist) des modules qui le composent est disponible. Dans ce cas, une netlist globale du conglomérat est construite en combinant celles des différents modules. Ensuite, la génération des vecteurs de test pourra alors être effectuée à l'aide d'un outil de génération automatique de vecteurs de test (ATPG).

– la description interne des modules n'est pas connue (circuits confidentiels). Dans ce cas, nous proposons une méthode simple qui repose sur l'ordonnancement et la combinaison des vecteurs de chaque circuit en fonction des critères de contrôlabilité et d'observabilité des entrées/sorties primaires de chaque circuit.

2.3.1 Génération de vecteurs de test par ordonnancement

En général, pour les conglomérats composés de plusieurs modules, l'interconnexion entre ces modules se fait en *cascade* comme indiqué sur la figure 2.3. Pour la clarté de l'explication, nous allons nous pencher sur le cas d'un conglomérat composé uniquement de deux modules. Nous généraliserons par la suite à un nombre quelconque de modules.

Le conglomérat à étudier se compose d'un module en amont et un autre en aval. Dans cette configuration, une analyse de testabilité permet d'identifier deux types de problèmes :

- un problème de *contrôlabilité* pour le circuit aval. La contrôlabilité est une mesure destinée à quantifier la difficulté à positionner un nœud interne à un

état logique donné. Dans le cas présent, la contrôlabilité des entrées du module aval connectés aux sorties du module amont dépendra des possibilités offertes par les vecteurs disponibles pour ce dernier.

- un problème *d'observabilité* pour le circuit amont. Au même titre que la contrôlabilité, l'observabilité est une mesure destinée à caractériser la difficulté à observer sur une sortie primaire, la valeur logique présente sur un nœud interne. Pour le circuit amont, l'observabilité de ses sorties qui sont connectées aux entrées du circuit aval dépendra des vecteurs de test de ce dernier.

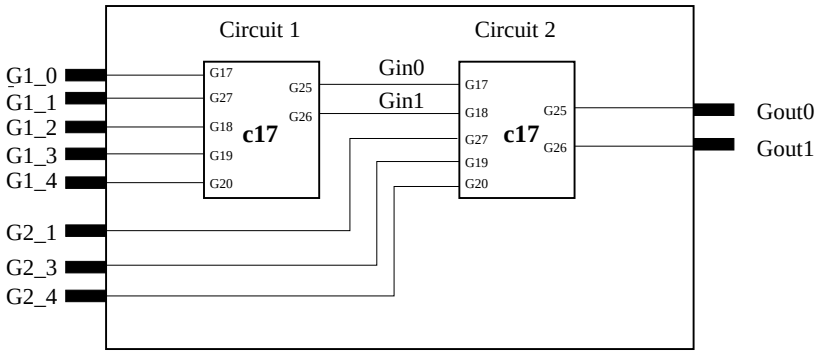

Figure 2.3: Conglomérat à tester

2.3.1.1 Contraintes sur le circuit amont

Lors du processus de génération de vecteurs de test pour le circuit aval, le circuit amont joue le rôle de générateur de stimuli. Par conséquent, il doit respecter les conditions de contrôlabilité évoquées ci-dessus. Ces conditions peuvent s'énoncer de la manière suivante :

- condition nécessaire : les vecteurs de test de ce circuit doivent assurer la détection de tous les collages logiques sur les sorties primaires connectées au circuit aval. De cette manière, on est sûr que ces nœuds subissent les commutations 0 à 1 et 1 à 0.

- condition suffisante : les vecteurs de test du circuit amont doivent générer toutes les combinaisons possibles sur les sorties connectées au circuit aval. Si N est

le nombre de ces sorties, et si l'on est en mesure d'obtenir les 2^N combinaisons possibles, on peut alors affirmer qu'on est certain de pouvoir reconstituer tous les vecteurs de test du circuit aval.

Dans la pratique, cette condition sera très difficile à réaliser. Mais on verra par la suite qu'en réalité, on aura uniquement besoin des combinaisons nécessaires à la reconstitution des vecteurs de test du circuit aval.

2.3.1.2 Contraintes sur le circuit aval

Durant la phase de test du circuit amont, le circuit aval sert de chemin de données (data−path), acheminant les informations de ses entrées vers ses sorties primaires. Pour cette raison, tous les chemins utilisés doivent être sensibilisés pour assurer une propagation correcte de ces informations. Ainsi, pour satisfaire les contraintes d'observabilité, les vecteurs de test du circuit aval doivent remplir les conditions suivantes :

- condition nécessaire : il faut que dans l'ensemble des vecteurs de test du circuit aval, il existe des vecteurs permettant de mettre en évidence toutes les fautes de collage simple sur les entrées du circuit qui sont connectées aux sorties du circuit qui le précède.
- condition suffisante : Soit N le nombre des entrées primaires du circuit aval connectées aux sorties du circuit en amont, il faut que dans l'ensemble des vecteurs de test du circuit en aval, il existe des vecteurs permettant de mettre en évidence, les 2^N combinaisons de collages simples sur ces entrées.

Cette dernière condition est particulièrement difficile à satisfaire. Mais fort heureusement, dans la pratique, on aura uniquement besoin des combinaisons de détections requises par le circuit amont.

2.3.2 Illustration de la méthode

Pour simplifier la compréhension de cette méthode, nous allons l'étudier sur le conglomérat de la figure 2.3. l'ensemble est composé de deux modules c17 (figure 2.4) de la famille des ISCAS'85 [BrF85].

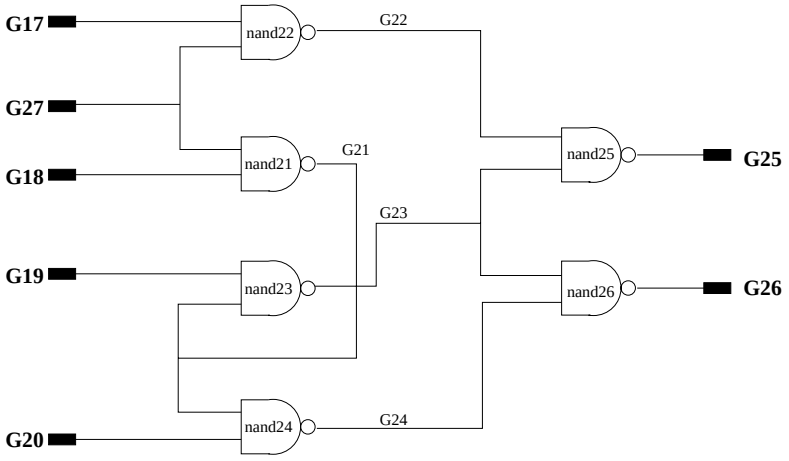

Figure 2.4: Module c17

Pour ce module, nous avons généré un ensemble de 6 vecteurs de test donnant une couverture de 100% pour les fautes de collage. Le tableau 2.1 présente les vecteurs de test obtenus ainsi que la couverture associée à chacun d'entre eux. Le terme D signifie que le vecteur détecte un collage à 0 sur le nœud considéré alors que le terme $\overline{D}$ signifie la détection d'un collage à 1.

	V1	**V2**	**V3**	**V4**	**V5**	**V6**	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
G17	1	0	1	0	0	1	D	$\overline{D}$	D	0	$\overline{D}$	1
G18	1	1	0	0	0	0	D	D	$\overline{D}$	$\overline{D}$	0	0
G19	0	0	0	1	0	0	0	0	0	D	$\overline{D}$	$\overline{D}$
G20	1	1	1	0	0	1	1	1	D	0	$\overline{D}$	D
G27	1	1	1	1	1	0	D	D	D	1	1	$\overline{D}$
G25	1	0	1	1	0	0	D	$\overline{D}$	D	D	$\overline{D}$	$\overline{D}$
G26	0	0	1	1	0	1	$\overline{D}$	$\overline{D}$	D	D	$\overline{D}$	D

Tableau 2.1: Vecteurs de test et matrice de couverture pour le c17

Comme on peut le constater, ces vecteurs permettent d'avoir les 4 combinaisons possibles {00, 01, 10, 11} sur les sorties G25 et G26. Par conséquent, ils vérifient les conditions de contrôlabilité du circuit aval. Ils permettent d'autre part les 4 combinaisons possibles {(D,D),(D,$\overline{D}$),($\overline{D}$,D),($\overline{D}$,$\overline{D}$)} (moyennant quelques manipulations explicitées plus loin) de collage simple sur les entrées G17, G18. Par conséquent, ils vérifient également les conditions d'observabilité du circuit amont.

2.3.2.1 Test du circuit en Amont

Pour pouvoir tester le maximum de fautes du circuit amont, il faut avoir les combinaisons suivantes {(D,D),(D,$\overline{D}$),($\overline{D}$,D),($\overline{D}$,$\overline{D}$)} sur les nœuds G17 et G18 (condition d'observabilité). Sur la matrice de couverture (tableau 2.1), on peut voir que : v1 donne (D,D), v2 donne ($\overline{D}$,D) et v3 donne (D,$\overline{D}$). Aucun vecteur ne donne ($\overline{D}$,$\overline{D}$). Dans ce cas, il faudrait chercher deux vecteurs *compatibles* dont l'union est susceptible de reconstituer le couple ($\overline{D}$,$\overline{D}$). Sur cette même matrice, on peut constater que les vecteurs V4 et V5 permettent de réaliser cette opération.

En conclusion on a trouvé une séquence composée de 8 vecteurs (les 6 vecteurs initiaux + 2 supplémentaires issus de la répétition de (v4,v5)), qui permet d'obtenir une couverture maximale pour le module amont.

2.3.2.2 Test du circuit en Aval

Pour obtenir 100% de couverture pour ce circuit, il faut pouvoir appliquer sur ses entrées une séquence complète des vecteurs v1 à v6. Dans ce but, on essaie de reconstituer les vecteurs de test du deuxième circuit avec les sorties du premier. Pour cela, il suffit d'identifier les vecteurs n'ayant pas été reconstitués lors de la phase du test du module amont, et de les ajouter. Dans notre exemple, parmi les vecteurs de test du deuxième module, seul le vecteur v6 manque. Il suffit donc de choisir le vecteur adéquat à appliquer sur le module 1 pour reconstituer le vecteur v6. Une solution possible est présentée en grisé dans le tableau de la figure 2.5.

Entrées du module amont					Entrées du module aval			Vecteur équivalent	Détections ciblées
G1_0	**G1_1**	**G1_2**	**G1_3**	**G1_4**	**G2_1**	**G2_3**	**G2_4**	pour le module 2	pour le module 1
S3_1 =			V1		1	0	1	**V3**	toutes
S3_2 =			**V2**		1	1	0	**V4**	sur G18 seulement
S3_3 =			**V2**		1	0	0	**V5**	sur G17 seulement
S3_4 =			V3		1	0	1	**V1**	toutes
S3_5 =			V4		1	0	1	V1	toutes
S3_6 =			**V5**		1	0	0	V5	sur G17 seulement
S3_7 =			**V5**		1	1	0	V4	sur G18 seulement
S3_8 =			V6		1	0	1	**V2**	toutes
S3_9 =			V1		0	0	1	**V6**	Aucune

Figure 2.5: Une solution possible pour le test du module 1

2.3.3 Algorithme général

Avant de détailler l'algorithme, nous allons énoncer quelques définitions.

Définition 1 : un vecteur amont a une correspondance k avec un vecteur aval si k détections des fautes de collage réalisées par le premier sur les sorties du module amont correspondent à des détections réalisées par le deuxième vecteur sur les entrées du module aval. Par exemple, le vecteur v3 du module amont a une correspondance 2 avec v1 du module aval (tableau 2.1).

Définition 2 : un vecteur amont est dit compatible avec un vecteur aval si les valeurs affectées par le premier aux sorties du module amont sont identiques à celles affectées par le deuxième aux entrées du module aval. Le tableau 2.2 montre la compatibilité et la correspondance entre les vecteurs du circuit c17.

La génération des vecteurs pour le conglomérat débute en deux étapes :

– Une phase de propagation des vecteurs du module amont (organigramme de la figure 2.6. Pour chacun de ces vecteurs, on cherche un vecteur aval ayant une

Amont / **Aval**	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**	**V1**	**V2**	**V3**	**V4**	**V5**	**V6**
V1	0	0	2	2	0	0	0	0	1	1	0	0
V2	0	0	0	0	0	2	0	0	0	0	0	1
V3	2	0	0	0	0	0	1	0	0	0	0	0
V4	0	1	0	0	1	0	0	0	0	0	1	0
V5	0	1	0	0	1	0	0	1	0	0	1	0
V6	0	0	0	0	0	0	1	0	0	0	0	0
	Correspondance						Compatibilité					

Tableau 2.2: Correspondance et compatibilité entre vecteurs de test pour le c17

correspondance totale avec ce vecteur amont. Si on ne le trouve pas, on cherche les vecteurs compatibles et on génère les combinaisons susceptibles de propager le vecteur amont. On choisit parmi ces combinaisons (si elles existent) celles qui assurent un maximum de couverture avec le moins de vecteurs. Les résultats sont ensuite stockés sous forme de liste.

– Une phase de construction des vecteurs du module aval durant laquelle on parcourt la liste des séquences générées lors de l'étape précédente en identifiant les vecteurs qui n'ont pas encore été reconstitués. Pour chacun de ces vecteurs, on parcourt l'ensemble des vecteurs amont et on lui affecte le premier vecteur qui lui est compatible.

A la fin de ces deux étapes, on se retrouve avec deux listes : celle des vecteurs à appliquer aux conglomérats et celle des vecteurs qui n'ont pas pu être propagés ou reconstruits correctement.

Dans l'exemple précédent (figure 2.3), notre raisonnement a été développé sur le cas d'un conglomérat composé de deux modules. Ce raisonnement reste valable dans le cas où le conglomérat serait composé de K modules. Il suffit simplement de répéter le même algorithme $(K-1)$ fois, en considérant à chaque fois deux blocs. Par exemple, dans le cas ou $K = 3$, on applique la première fois l'algorithme sur l'ensemble formé

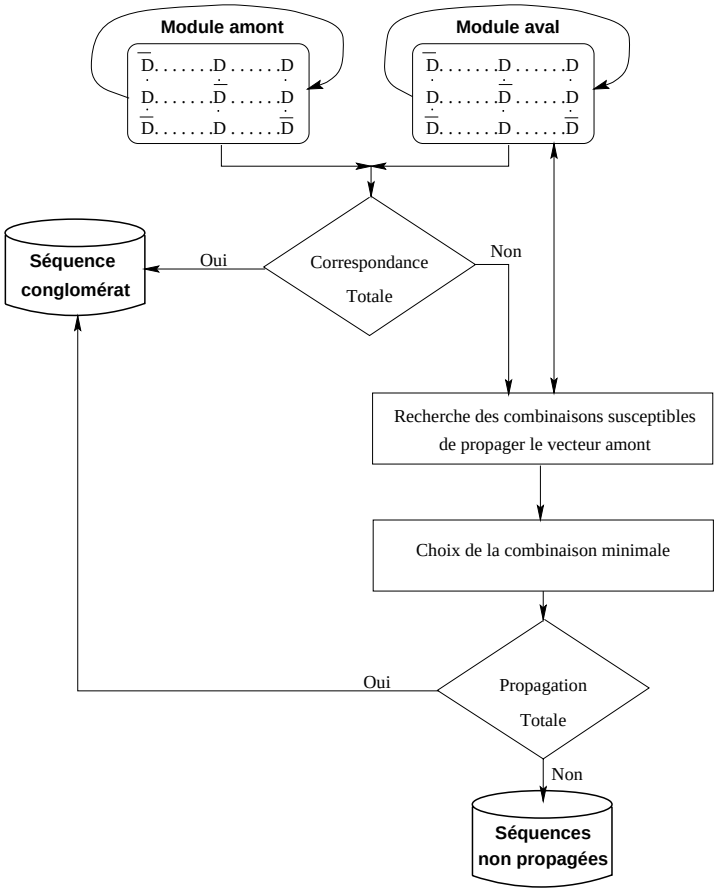

Figure 2.6: Organigramme de la phase de propagation des vecteurs amont

par le module 1 et le module 2. Ceci donne un module intermédiaire 1_2. Ensuite, on applique l'algorithme sur le nouvel ensemble réunissant le module 1_2 et le module 3.

Cette méthode ne se limite pas aux modules combinatoires. Elle peut s'appliquer aussi aux conglomérats composés de modules combinatoires reliés à des modules séquentiels, car conceptuellement rien ne l'empêche. Néanmoins, on s'attend à ce que les résultats obtenus soient moins bons que dans le cas de conglomérats composés de modules combinatoires uniquement. Quant à ceux composés de modules séquentiels uniquement, si en théorie la méthode reste applicable, sa mise en œuvre en pratique devient trop fastidieuse. En effet, dans ce cas, nous n'avons plus affaire à des vecteurs de test indépendants, mais à des séquences de vecteurs indissociables et par conséquent très difficiles à reconstruire (il suffit qu'un seul vecteur manque pour que toute la séquence soit abandonnée).

2.4 Ordonnancement pour un ensemble de conglomérats

Cette étape a pour but de trouver une séquence minimale permettant de réaliser le test en parallèle de tous les conglomérats composant la carte pour les court−circuits d'interaction. Si les ATPGs actuels permettent souvent d'obtenir des taux de couverture appréciables pour les fautes de collage, il n'en n'est pas de même pour les court−circuits ou les circuits−ouverts. Dans ce cas, on est souvent amené à recourir à des simulations de fautes pour atteindre de bons taux de couverture. Néanmoins cette solution reste envisageable seulement pour des circuits de faible complexité. Dans notre cas, la mise au point de ces vecteurs de test est obtenue de la manière suivante : les différentes séquences de test obtenues pour chaque conglomérat lors de la première étape sont regroupées et ordonnées de manière à mettre en évidence ces fautes. Cet ordonnancement est basé sur l'ordre d'association des différentes séquences entre elles, ainsi que sur la possibilité de répéter au besoin certaines séquences à l'intérieur de la séquence globale. Néanmoins, cet ordonnancement doit être réalisé avec le souci de produire une séquence finale de longueur égale à celle de la plus longue des séquences de l'ensemble des conglomérats. Nous avons toutefois remarqué au cours de nos expérimentations, que cette contrainte (non restrictive, rappelons le) est souvent satisfaite dans le cas de conglomérats combinatoires. Elle l'est moins dans le cas où ils sont séquentiels. Ceci peut s'expliquer par le fait que souvent, le taux de couverture de fautes par vecteur de test pour les circuits combinatoires est supérieur à celui des circuits séquentiels. Ceci entraîne donc la nécessité de dupliquer davantage de séquences dans le cas séquentiel.

D'une manière générale pour détecter une faute dans un circuit, il faut pouvoir activer cette faute (condition de contrôlabilité) et la propager au moins vers une des sorties primaires du circuit (condition d'observabilité). Dans l'exemple de la figure 2.7, le vecteur de test doit être choisi de manière à ce que les stimuli appliqués aux lignes I1, I2, I3 et I4 donnent des valeurs opposées sur les nœuds n1 et n2, tandis que la valeur appliquée à la ligne I5 doit permettre la propagation sur la sortie S.

Pour cet exemple, le vecteur {00000} permet de détecter un "and-short" entre n1

et n2, alors qu'un "or-short" sera indétectable dans ce cas de figure.

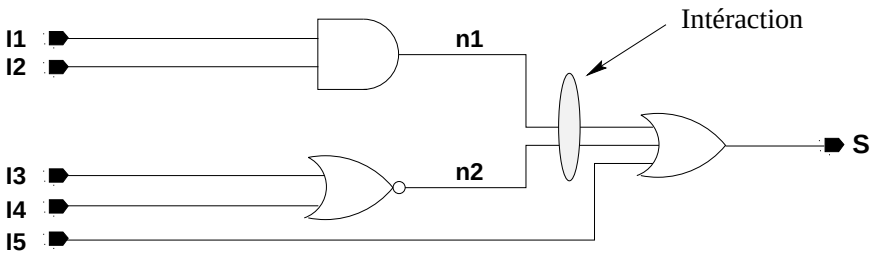

Figure 2.7: Détection d'une faute d'interaction

Par conséquent, pour détecter une faute d'interaction entre deux nœuds appartenant chacun à un conglomérat, il faut trouver au moins un vecteur appartenant à chaque ensemble et qui vérifie en premier lieu les conditions de contrôlabilité et d'observabilité, mais qui doit aussi engendrer des niveaux logiques opposés sur les nœuds suspectés afin de mettre en évidence cette possible interaction. De ce fait, l'algorithme que nous proposons (algorithme 2.1) permet d'ordonner les vecteurs en vue de la détection de n'importe quel type de court−circuit entre conglomérats. Pour implémenter cet algorithme, nous avons adopté une représentation matricielle qu'on décrit dans le paragraphe 2.5.1.

---

pour tous les nœuds du conglomérat i
{
    pour tous les nœuds du conglomérat j
    {
        associer les vecteurs $V_k(i), V_l(j)$ *qui produisent des détections opposées*
    }
}

Algorithme 2.1 :Détection des court-circuit

2.5 Test des conglomérats et des connexions BS

Une fois que les phases 1 (génération des vecteurs de test pour les conglomérats) et 2 (ordonnancement des vecteurs pour maximiser la détection des fautes d'interaction) sont achevées, la dernière phase consiste à générer les vecteurs pour les connexions BS. Étant donné que les conglomérats partagent l'accès électronique des mêmes registres BS, si on applique en parallèle le test de tous les conglomérats, ceci permettra de réduire le temps de test. En plus, pour assurer la détection des court−circuits d'interaction, les sorties de tout conglomérat et de toute connexion BS doivent être observées de manière continue durant le test [RoD90]. Par conséquent, le test des connexions BS et celui des conglomérats doivent être réalisés simultanément chaque fois que cela est possible [Han89]. Les connexions BS n'ayant pas pu être testées à ce moment-là, seront testées dans la phase suivante en utilisant les algorithmes mis au point à cet effet [Wag87] [HRA88] [JaY89]. Les circuits BS seront alors vérifiés en phase finale, étant donné que les mêmes ressources sont nécessaires au test des conglomérats et des connexions BS.

2.5.1 La détection des fautes

Pour avoir ce test simultané, l'idée principale ici est que les vecteurs doivent garantir la propagation aux sorties primaires des conglomérats, des fautes d'interaction entre les nœuds de ces conglomérats et les connexions BS, et que le test des court−circuits dans les parties BS commence par la génération d'une première partition de l'ensemble total de leurs connexions. Pour cela, nous avons besoin :

– d'identifier en premier lieu les chemins de propagation à partir d'un nœud donné aux sorties primaires des conglomérats.

– de fixer les conditions qui permettent la détection de n'importe quel court−circuit entre un nœud interne d'un conglomérat et une connexion BS.

Pour réaliser ces tâches, nous avons adopté une représentation matricielle des données. Cette représentation consiste à générer une matrice qui représente la couverture des

fautes de collage pour les conglomérats. Elle est obtenue par superposition des résultats obtenus par deux simulations successives :

– une simulation logique qui va affecter à chaque nœud la valeur logique que lui attribue le vecteur de test.

– une simulation de fautes qui va donner la couverture de chaque vecteur de test sur chaque nœud des conglomérats. Cette matrice se présente sous l'aspect suivant :

– Chaque colonne correspond à un PTV de la séquence de test (cf. paragraphe 1.3.1).

– Chaque ligne correspond à un STV pour le nœud considéré.

– Chaque élément $M(i,j)$ de cette matrice donne la couverture du vecteur j pour les fautes de collage sur le nœud i de la manière suivante :

- M(i,j) = D => le vecteur j détecte un collage à 0 sur le nœud i.
- M(i,j) = $\overline{D}$ => le vecteur j détecte un collage à 1 sur le nœud i.
- M(i,j) = 0/1 => le vecteur j ne détecte aucun collage sur le nœud i.

Avec ce type de représentation, on obtient facilement les informations concernant le conglomérat, et les conditions énumérées précédemment peuvent être formalisées et les algorithmes pourront être implantés. Étant donné que, maintenant il suffit de parcourir les colonnes de la matrice pour identifier les chemins de propagation aux sorties des conglomérats, la condition pour la détection d'un court−circuit quelconque entre deux nœuds i et j appartenant à deux conglomérats différents dont les matrices sont notées respectivement M1 et M2 devient :

Pour chaque couple (M1(i),M2(i)) de STVs, il doit exister au minimum un PTV k pour lequel on a:

$$\{M1(i,k) = D \wedge M2(j,k) = \overline{D}\} \vee \{M1(i,k) = \overline{D} \wedge M2(j,k) = D\}$$

Dans le cas où l'on a affaire à un court−circuit du type "and−short" (comportement modélisé par une porte "AND"), l'état logique dominant est le 0 logique, et la condition devient:

$$\{M1(i,k) = 0 \wedge M2(j,k) = D\} \vee \{M1(i,k) = D \ \wedge \ M2(j,k) = 0\}$$

Dans le cas où le court−circuit est du type "or−short" (comportement modélisé par une porte "OR"), pour lequel le 1 logique est l'état dominant, la condition devient:

$$M1(i,k) = 1 \wedge M2(j,k) = \overline{D}\} \vee \{M1(i,k) = \overline{D} \ \wedge \ M2(j,k) = 1\}$$

Concernant la détection des court−circuits quelconques ("and−short" et "or−short") entre un nœud interne i d'un conglomérat et une connexion BS, la condition à satisfaire est la suivante : pour chaque nœud i, il faut trouver des vecteurs j, k, avec $j \neq k$ tels que :

$$\{M(i,j) = D\,,\ toutes\ (BS = 0)\} \wedge \{M(i,k) = \overline{D}\,,\ toutes\ (BS = 1)\}$$

Dans ce cas, la stratégie pour la détection des court−circuits avec les connexions BS revient à déterminer deux ensembles minimaux de vecteurs de tests (PTVs) qu'on désignera par *D-set* et $\overline{D}$*-set* tels que :

− l'application simultanée des vecteurs appartenant au *D-set* et des 0 (logiques) sur toutes les connexions BS permet d'observer les court−circuits d'interaction du type "and−short" sur les sorties des conglomérats

− l'application simultanée des vecteurs appartenant au $\overline{D}$*-set* et des 1 (logiques) sur toutes les connexions BS permet d'observer les court−circuits d'interaction du type "or−short" sur les sorties des conglomérats.

Pour déterminer ces deux ensembles, on procède de la manière suivante :

− dans un premier temps, on génère les *D-set* et $\overline{D}$*-set initiaux.* Il s'agit de deux ensembles composés de PTVs pour lesquels, seulement un seul D (pour le *D-set*) et un seul $\overline{D}$(pour le $\overline{D}$*-set*) sont trouvés dans un STV. Ces ensembles représentent les vecteurs devant *obligatoirement* faire partie de la séquence finale de vecteurs de test. En effet, la faute (D ou $\overline{D}$) est seulement détectée par ce vecteur. Par conséquent,

s'il venait à manquer dans la séquence finale, la détection de la faute ne se ferait pas. L'algorithme 2.2 permet le calcul des deux ensembles *D-set* et $\overline{D}$*-set* initiaux.

---

procédure **créer_set (X)**
{
    pour toutes les lignes de la matrice de couverture faire
    {
        si (un seul X)
        alors (X_set := X_set $\bigcup$ *la colonne_correspondante)*
    }
}

Algorithme 2.2 : Calcul des *D-set* et $\overline{D}$*-set initiaux*

Une fois que les *D-set* et $\overline{D}$*-set* initiaux sont générés, la phase suivante consiste à déterminer la matrice des *D-candidats* et celle des $\overline{D}$*-candidats*. Ces deux matrices vont servir à déterminer l'ensemble des vecteurs susceptibles de faire partie de la séquence finale. Les vecteurs sélectionnés seront rajoutés aux *D-set* et $\overline{D}$*-set* initiaux afin de construire cette séquence finale de test (cf. paragraphe 2.6).

Ces deux matrices (*D-candidats* et $\overline{D}$*-candidats*) sont obtenues de la façon suivante:
– Dans un premier temps on élimine de la matrice de couverture, tous les nœuds auxquels est associé un D (resp. un $\overline{D}$) par un PTV qui fait déjà partie du *D-set* (resp. du $\overline{D}$*-set*). Dans un deuxième temps, on retire les STVs et les PTVs pour lesquels il n'y a aucun D (resp. $\overline{D}$) qui reste, et en dernier on élimine de ces matrices les STVs dont la couverture en D (resp. en $\overline{D}$) englobe celle d'un autre STV. L'algorithme 2.3 permet de déterminer ces deux matrices.

Une fois qu'on détermine les deux matrices de candidats, on procède à l'ultime étape de calcul qui consiste à déterminer la séquence minimale de test. Pour cela, il suffit de trouver la combinaison minimale de PTVs telle qu'il existe au moins un D (resp. $\overline{D}$) dans la matrice des *D-candidats* (resp. $\overline{D}$*-candidats*). Pour déterminer cette combinaison minimale, on adopte la démarche suivante : Soit nD (resp. $n\overline{D}$) le cardinal du

```
procédure créer matrice_candidat (X)
{
    éliminer de la matrice de couverture toutes les lignes où
    un X est associé a un X-set élément ;
    Tant que la réduction est possible faire
    {
        éliminer de la matrice_candidat (X) toutes les lignes et colonnes dans
        lesquels il n'y a plus de X;
        éliminer de la matrice_candidat (X) les lignes englobant une autre;
    }
}
```

Algorithme 2.3 : Création des matrices des candidats

D-set (resp. $\overline{D}$-*set*) et p un entier par lequel on désigne un index. En partant de $p = 1$, on va chercher les combinaisons de p parmi nD (resp. $n\overline{D}$) PTVs tel qu'au moins un D (resp. $\overline{D}$) existe dans chaque STV appartenant à la matrice des *D-candidats* (resp. $\overline{D}$-*candidats*). A chaque étape de calcul, un nouvel ensemble de combinaisons est obtenu. Son intersection avec le *D-set* (resp. $\overline{D}$-*set*) et avec l'ensemble de combinaisons de l'étape précédente est calculée. Si cette intersection est vide, cela implique qu'on a trouvé une séquence minimale et par conséquent on arrête le processus de calcul. On ajoute les vecteurs retenus au *D-set* (respectivement $\overline{D}$-*set*) initial. Mais dans le cas où une intersection vide n'est pas trouvée, on calcule alors toutes les combinaisons possibles, et on ne retient que celle qui engendre une intersection minimale. Les vecteurs ainsi retenus sont ensuite ajoutés au *D-set* (respectivement $\overline{D}$-*set*) initial. La procédure de calcul des combinaisons a été implantée selon l'algorithme 2.4.

```
Procédure générer_combinaison(X,Y)
{   index=1;min_intersec=max(X_candidats,Y_candidats);
    tant que ((index <= max(X_candidats,Y_candidats) et (min_intersec > 0)) faire
        {
        si (X−candidats) { X_comb=next_combinaison(matrice_candidats(X),index);}
        si (Y−candidats) { Y_comb=next_combinaison(matrice_candidats(Y),index);}
        si ((X_candidats) et pas (Y_candidats) et (X_comb))
        {
            intersec=card(X_comb U Y-set);
            si (intersec < min_intersec){ min_intersec=intersec;}
        }
        si ((Y_candidats) et pas (X_candidats) et (Y_comb))
        {
            intersec=card(Y_comb U X-Set);
            si (intersec < min_intersec){ min_intersec=intersec;}
        }
    si ((X_candidats) et (Y_candidats)
    {
        pour tous les (X_comb)
        pour tous les (Y_comb)
        {
            intersec=card((X_comb ⋂ Y-set) U (Y_comb ⋂ X-set) U (X_comb ⋂ Y_comb);
            si (intersec < min_intersec) { min_intersec=intersec;}
        }
    }
    si (X_candidats) { X-set=X-set U (X_comb retenues)}
    si (Y_candidats) { Y-set=Y-set U (Y_comb retenues)}
    }
}
```

Algorithme 2.4 : Calcul des combinaisons

2.6 Exemple d'application

Nous allons maintenant illustrer la méthodologie en l'essayant sur un exemple. Le circuit de la figure 2.8 représente le s27, un benchmark des ISCAS'89 [BBK89] dont les flip-flops ont été remplacées par un TI BS SCOPE OCTAL [TI89]. De cette manière, on obtient une carte avec une partie BS et une partie conglomérats.

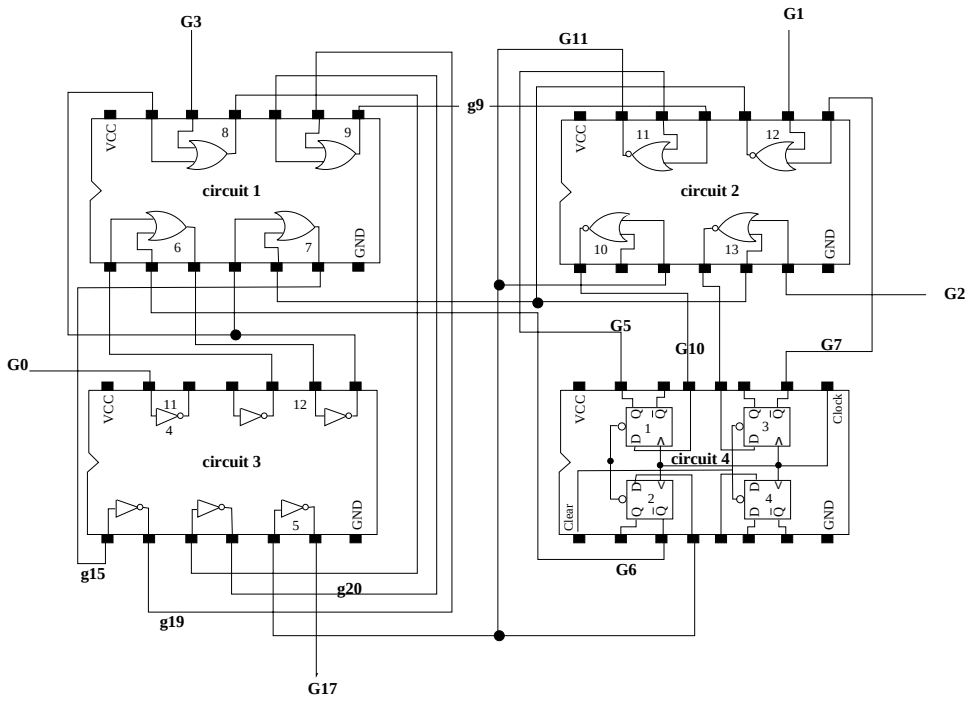

Figure 2.8: carte implantée à partir du s27

Nous avons généré une séquence de huit vecteurs de test à l'aide de Hitest [Ben84], l'ATPG de SYSTEM HILO [Ved94]. Ces vecteurs couvrent la totalité des fautes de collage, de circuits−ouverts et de court−circuits internes.

2.6.1 Génération de la matrice de couverture

Après une simulation logique et une simulation de fautes à l'aide du simulateur Hifault de SYSTEM HILO, nous avons construit la matrice de couverture. Le tableau suivant représente le résultat obtenu.

	V1	**V2**	**V3**	**V4**	**V5**	**V6**	**V7**	**V8**
G0	D	0	$\overline{D}$	D	d	$\overline{D}$	$\overline{D}$	1
G1	D	D	0	D	D	D	D	D
G2	$\overline{D}$	0	$\overline{D}$	$\overline{D}$	0	0	D	0
G3	1	D	1	1	$\overline{D}$	1	0	D
g4	1	0	0	$\overline{D}$	D	0	$\overline{D}$	1
G5	1	$\overline{D}$	0	$\overline{D}$	0	D	$\overline{D}$	$\overline{D}$
G6	0	1	D	$\overline{D}$	0	1	$\overline{D}$	0
G7	0	$\overline{D}$	D	0	$\overline{D}$	$\overline{D}$	1	$\overline{D}$
G8	0	0	$\overline{D}$	D	$\overline{D}$	0	D	0
G9	1	$\overline{D}$	D	$\overline{D}$	D	0	$\overline{D}$	$\overline{D}$
G10	D	$\overline{D}$	$\overline{D}$	$\overline{D}$	D	$\overline{D}$	$\overline{D}$	D
G11	$\overline{D}$	D	$\overline{D}$	D	$\overline{D}$	$\overline{D}$	D	D
G12	$\overline{D}$	D	$\overline{D}$	$\overline{D}$	D	D	0	D
G13	D	$\overline{D}$	D	D	$\overline{D}$	$\overline{D}$	$\overline{D}$	$\overline{D}$
G14	$\overline{D}$	1	D	D	$\overline{D}$	D	D	0
G15	0	D	$\overline{D}$	D	1	1	D	D
g16	1	D	1	D	$\overline{D}$	1	D	D
G17	D	$\overline{D}$	D	$\overline{D}$	D	D	$\overline{D}$	$\overline{D}$
g18	1	1	D	$\overline{D}$	D	1	$\overline{D}$	1
g19	1	$\overline{D}$	D	$\overline{D}$	0	0	$\overline{D}$	$\overline{D}$
G20	0	$\overline{D}$	0	$\overline{D}$	D	0	$\overline{D}$	$\overline{D}$

Tableau 2.3: Matrice de couverture de la carte

2.6.2 Calcul des *D-set* et $\overline{D}$*-set* initiaux

Pour cela, nous appliquons l'algorithme 2.2. Le déroulement de cet algorithme donne les résultats suivants :

- *D-set* = {V3,V5,V6,V7}
- $\overline{D}$*-set* = {V3,V5}

2.6.3 Calcul des matrices des candidats

Le calcul des matrices des *D-candidats* et $\overline{D}$*-candidats* suivant l'algorithme du tableau 2.3 donne :

	V1	**V2**	**V4**	**V8**
G1	D	–	D	–
G3	–	D	–	D

matrice de D-candidats

	V4	**V7**
g18	$\overline{D}$	$\overline{D}$

matrice de $\overline{D}$-candidat

2.6.4 calcul des combinaisons

L'algorithme de calcul des combinaisons donne pour cet exemple les résultats suivants:

– Pour les *D-candidats* une seule combinaison de PTV avec une intersection vide et qui est {V1,V2}.

– Pour les $\overline{D}$*-candidats* une seule combinaison de PTV avec une intersection vide et qui est {V4}. Au vu de ces résultats, le *D-set* et $\overline{D}$*-set* finaux seront :

– D−set ={ V1,V2,V3,V5,V6,V7 }

– $\overline{D}$−set ={ V3,V4,V5 }

On remarquera que le vecteur V8 est absent des deux ensembles. Néanmoins, il devra faire partie de la séquence finale de test afin de retrouver un taux de couverture de 100% pour les court−circuits internes. On remarquera aussi que les vecteurs V3 et V5 font partie des deux ensembles. Pour cela ils doivent apparaître par deux fois dans la séquence finale. Tout ceci nous donne donc la séquence définitive suivante : {V1,V2,V3,V3,V4,V5,V5,V6,V7,V8} avec {000110100X} comme Boundary Scan STV, puisque l'on associe 0 aux éléments du *D-set* et 1 à ceux du $\overline{D}$*-set*. La valeur 'X' présente dans cette séquence peut être exploitée par exemple pour le test des interconnexions BS.

2.7 Conclusion

Nous venons de présenter une méthode permettant d'établir rapidement et à faible coût des séquences de test pour des cartes Boundary Scan Partiel. Cette méthode

permet de diminuer le temps d'application du test en générant des séquences minimales pour le test simultané des conglomérats et des interconnexions. Les essais qui ont été effectués sur les circuits benchmark et dont les résultats sont donnés dans le chapitre 7 sont très encourageants [MLT93]. Son automatisation ainsi que la possibilité de l'étendre au test des MCMs comme nous allons le montrer dans le prochain chapitre font qu'elle pourrait servir de complément à des produits commerciaux tels que le PM 3790 BSD de Fluke&Philips [Phi93]. La possibilité de l'interfacer avec un outil de diagnostic (chapitres 4 et 5) permet d'obtenir un environnement unique dédié au test et diagnostic unifié de ce type de carte.

Chapitre 3

Adaptation au cas des MCMs

Chapitre 3

Adaptation au cas des MCMs

3.1 Introduction

A la fin des années 80, L'apparition sur le marché d'outils de conception de plus en plus performants et fiables conjuguée avec l'évolution extrêmement rapide de la mise au point des technologies sub-microniques ont permis aux fabricants de semi-conducteurs de produire des circuits à hautes performances qui fonctionnent à des fréquences très élevées. Cependant, les techniques de "packaging" n'ont pas connu la même évolution et commencèrent à constituer un frein au développement de systèmes rapides et compacts, surtout au niveau des temps de propagations (delay on chip-to-chip signal path), temps de montée, etc. De ce fait, une solution s'imposait et les recherches ont été entreprises dans le but de trouver un moyen de diminuer l'espacement moyen entres les différents circuits dans les systèmes électroniques.

3.1.1 Technologie WSI

L'approche utilisée dans la technologie WSI (Wafer Scale Integration) consiste à regrouper sur la même tranche de silicium tous les composants requis par l'application envisagée et les interconnecter directement par des diffusions. Mais ce type de réalisation s'est heurté à plusieurs obstacles :

- le coût de développement et de production qui s'est avéré important à cause des pertes importantes engendrées par les tranches défectueuses (si un des composants de la puce est défectueux, alors toute la puce devient inutilisable).
- l'obligation d'unicité du fabricant pour les différents composants.
- L'impossibilité d'implanter sur la même tranche des technologies différentes.

Ces nombreuses limitations ont fait en sorte que cette technologie coûteuse n'a pas connu un grand succès auprès des fabricants.

3.1.2 Technologie MCM

L'approche MCM ("Multi-Chip Module" ou Modules multi-puces) est inspirée de la technologie WSI sauf qu'au lieu de fabriquer les composants sur la même tranche de silicium, elle se limite à les placer côte à côte sur le même substrat. Ce dernier va servir de support mécanique pour les puces et assurera en même temps l'interconnexion de ces puces entres elles à travers des lignes conductrices. Cette configuration permet aux puces d'être interconnectées avec des liaisons très courtes et ainsi de garder pratiquement intactes leurs performances tout en réduisant la taille de l'ensemble après encapsulation. Elle a aussi l'avantage d'être indépendante de la technologie utilisée ce qui enlève toute contrainte sur l'origine des puces à implanter sur le substrat. Le coût de développement est lui aussi diminué vu qu'il est en partie supporté par les fabricants des puces. Au vu des performances qu'on peut atteindre [HaW92] [Gul92], cette technologie se présente désormais comme la plus viable actuellement pour les besoins des systèmes à hautes performances actuels [Cou94].

3.2 Architecture des MCMs

Comme cité précédemment, un MCM est composé de 3 parties :

- un substrat

- les puces à connecter

- le boîtier

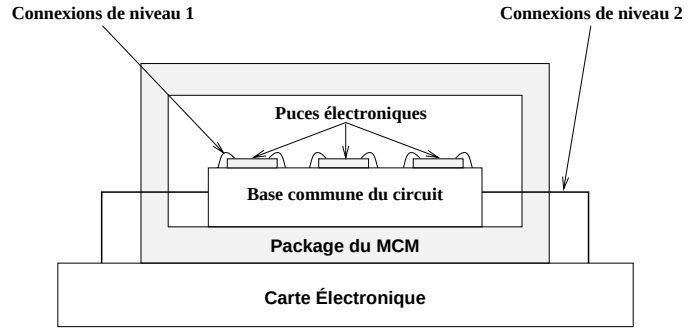

Figure 3.1: Architecture générale d'un MCM

3.2.1 Description du substrat

Le substrat est la base du MCM. C'est l'élément qui sert de support mécanique aux puces tout en assurant leurs interconnexions. Il s'agit d'un bloc multi-couches composé de matériaux conducteurs séparés par un matériau isolant à faible coefficient diélectrique (de l'ordre de 4) afin de permettre des fréquences de travail très élevées (fréquences de coupure élevées). Les couches conductrices sont reliées entre-elles par des "vias". Les puces sont reliées au substrat par l'intermédiaire des connexions de premier niveau (first level connection) qui assurent en même temps l'alimentation en courant (Vcc, Gnd). Ce type de montage permet d'obtenir des densités d'interconnexion de l'ordre de 90% sachant que celles obtenues dans les cartes classiques est de l'ordre de 10%. A ce jour, trois types de technologie de réalisation de substrat ont été standardisés (Standard IPC-MC-790) :

– MCM-L ("Laminated") : C'est une forme avancée de PCB avec des conducteurs en cuivre sur des plaques de diélectrique laminées.

– MCM-C (Cofired) : C'est une technologie hybride qui consiste en un empilement de couches d'isolant et de couches conductrices sur des substrats en céramique. Le couches sont obtenues par déposition, et le tout est chauffé en même temps afin d'obtenir un bloc compact.

– MCM-D (Deposited) : les connexions sont réalisées par déposition de films métalliques très minces sur un diélectrique qui peut être un polymère photo-sensible

d'origine organique. Les interconnexions sont ensuite réalisées par photo-sélection (photo-etching). Cette méthode donne de meilleurs résultats au niveau de la stabilité et des distorsions.

3.2.2 Fixation des puces sur le substrat

Trois principales technologies de fixations des puces sur le substrat sont utilisées :

– Wire Bond : C'est la technique classique qui consiste à souder un fil métallique entre le substrat et la puce (figure 3.2 (a)).

– TAB (Tape Automated Bonding) : dans cette technique, les plots sont préalablement fixés sur un film plastique. A l'aide d'une matrice, on fixe simultanément les plots sur la puce et le substrat (figure 3.2 (b)).

– Flip-Chip : la puce est fixée au substrat par l'intermédiaire de billes de soudure (figure 3.2 (c)). Ceci est obtenu par pression de la puce contre le lit de billes à la température de fusion du matériau qui les compose.

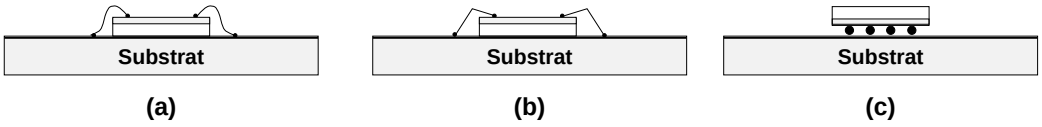

Figure 3.2: Différents modes de fixation

Selon le type de fixation puce-substrat choisi, on peut atteindre des densités de connexion allant de $400/cm^2$ (pour le Wire Bond ou le TAB) à $1600/cm^2$ (pour le Flip-Chip). Cela ne va donc pas manquer de soulever de graves problèmes de testabilité.

3.3 Test des MCMs

Le test des MCMs est une tâche extrêmement difficile à réaliser. Ceci est dû à plusieurs facteurs parmi lesquels :

- la multiplicité des circuits complexes qui les composent.

- une forte densité d'interconnexion.

- l'impossibilité d'accès aux nœuds internes après l'encapsulation du module.

Pour ces diverses raisons, l'adoption d'une méthodologie de test et de diagnostic devient impérative. Elle doit être basée sur l'utilisation des techniques de DFT afin de permettre un test fiable à un coût raisonnable, et permettre un diagnostic précis et rapide pour faciliter la maintenance.

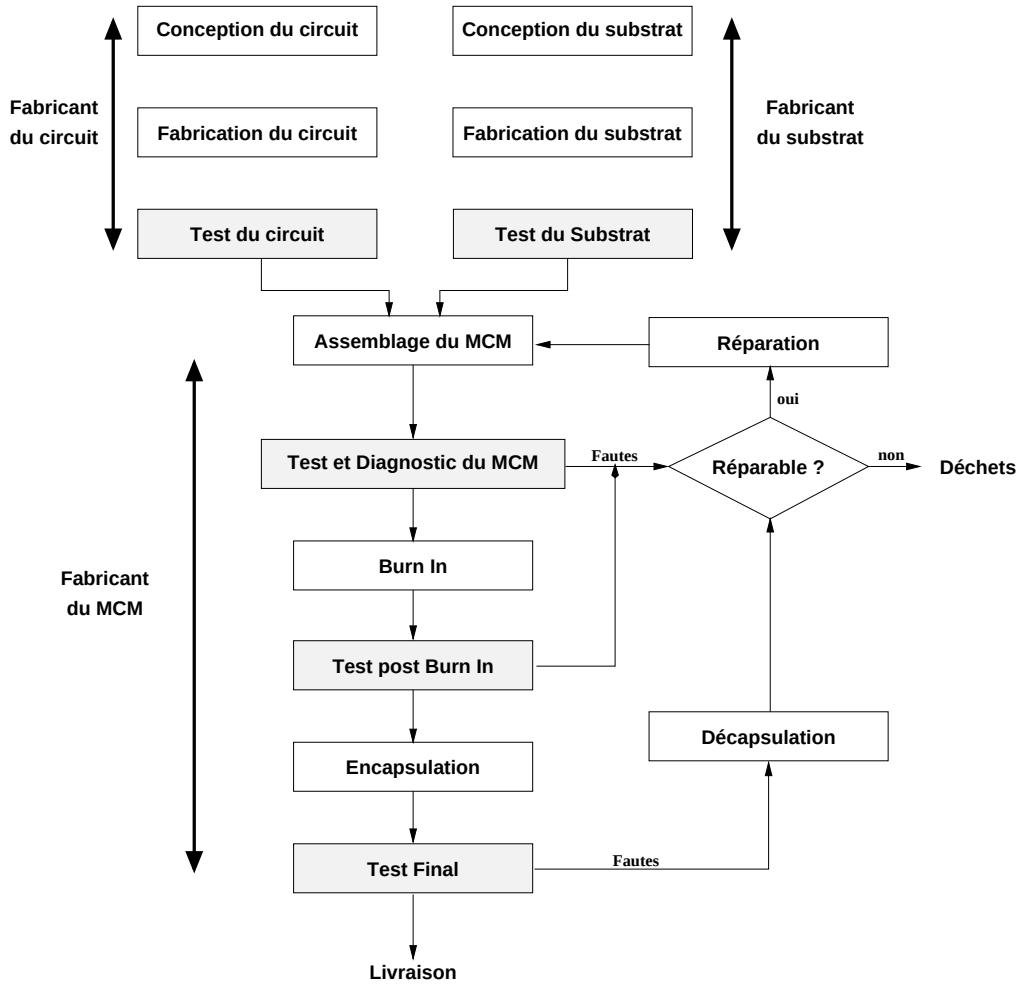

Figure 3.3: Les différentes phases dans la production des MCMs

La figure 3.3 [Zor95] relate les différentes étapes dans la chaîne de fabrication d'un MCM. On remarque que le test y figure à plusieurs reprises. Dans [Gil95], on estime à 25% les problèmes posés par le test par rapport aux problèmes majeurs (figure 3.4) rencontrés lors de la réalisation du MCM, ce qui représente une part non négligeable.

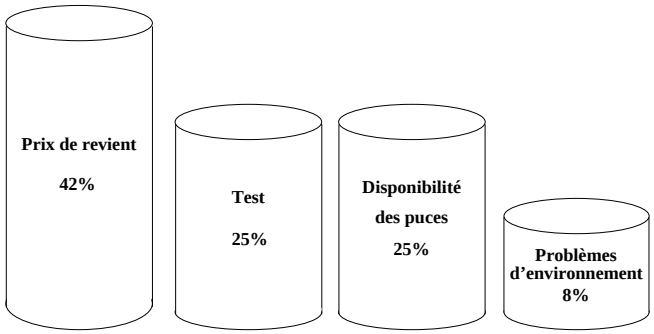

Figure 3.4: Les problèmes qui affectent les MCMs

3.3.1 Test du substrat

Le test du substrat avant la fixation des puces a pour but de détecter les anomalies sur les lignes de connexions. Avec la multitude de couches et la complexité d'interconnexion (figure 3.5), deux principaux types d'anomalie sont fréquents : les court-circuits (shorts) résultant d'un contact entre les lignes, et les circuit-ouverts (opens) découlant d'une éventuelle rupture sur ces lignes. La détection de ces pannes avant l'assemblage est impérative, cela permet ainsi d'éviter un surcoût prohibitif engendré par une intervention sur le MCM. En effet, le coût du substrat est relativement peu élevé comparé à celui du MCM complet. En plus, ce coût est normalement supporté par le fabricant de substrat qui doit par conséquent garantir la qualité des éléments qu'il produit.

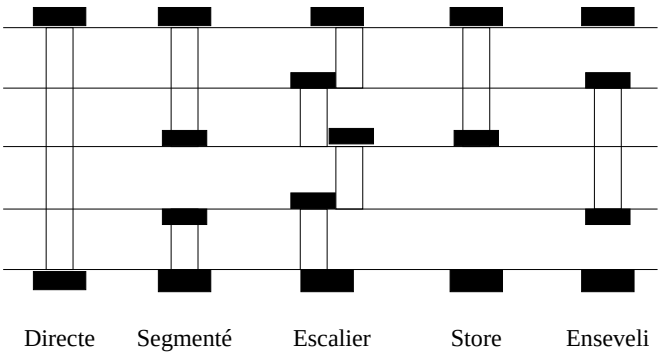

Figure 3.5: Les différentes structures de "vias"

Il existe principalement quatre techniques qui permettent d'effectuer ce genre de

test : les machines à pointes, les mesureurs de capacitances, le "flying probe" et finalement le test sans contact à l'aide d'un microscope électronique à balayage [Woo91].

Ne faisant pas partie des objectifs de ce travail, le test du substrat ne sera pas davantage développé. Néanmoins, nous indiquons à l'intention des lecteurs intéressés, que les méthodes citées précédemment ont été largement développées et discutées dans [JTB91] [MCMC92] [MCMC93].

3.3.2 Test des puces électroniques

Étant donné que les puces électroniques qui participent à la composition du MCM peuvent être d'origines diverses, leur test avant assemblage du MCM incombe en premier lieu à leurs fabricants et non à celui des MCMs. De ce fait, le fabricant de MCM ne peut garantir la qualité du produit final que relativement à celle des puces assemblées. Il est donc obligé de se fier aux indications du fournisseur de la puce concernant son état. Le rendement final de la chaîne de fabrication du MCM (nombre de circuits réussissant le test par rapport au nombre de circuits testés) dépendra directement de l'indication "Known-Good Die" (KGD) donnée par les fabricants de puces. Ce paramètre représente la probabilité que la puce ne comprend aucun défaut. Le tableau de la figure 3.1 [Gil95] montre la forte dépendance qui existe entre le rendement final de la chaîne et le KGD.

Probabilité (%) de KGD	**Rendement (%) MCM à 10 puces**	**Rendement (%) MCM à 20 puces**	**Rendement (%) MCM à 40 puces**
99.9	99	98	96
99	90	82	67
95	60	36	13
90	35	12	1
80	11	17	0
50	0	0	0

Tableau 3.1: Rendement en fonction du KGD

En général, les puces subissent des tests fonctionnels et des tests paramétriques (courant, tensions, fréquences, etc) par le fabricant. Ensuite un tri est exécuté pour

isoler les puces opérationnelles afin de les envoyer à l'encapsulation. Pour des puces destinées à être montées individuellement, ce processus peut s'avérer suffisant car d'autres test comme le vieillissement ("Burn-in") peuvent être effectués au niveau circuit. Pour celles destinées aux MCMs, ceci n'est pas acceptable car on court le risque de déceler une panne une fois l'assemblage terminé, et ceci entraînera un surcoût important. Par conséquent, pour ces puces, les tests complémentaires doivent se faire à ce niveau (puce nue). Or, les tests paramétriques au niveau des puces nues sont difficiles et très coûteux. Pour ces raisons, les fabricants se contentent souvent de réaliser des classifications de lots [Lan91]. Ceci a l'avantage de coûter moins cher, mais engendre en contrepartie une diminution du degré de certitude.

En conclusion, on voit bien qu'on ne peut atteindre un très bon taux de certification des puces avec un coût raisonnable. Cela induit donc que des étapes de test vont être déférées aux fabricants de MCMs. Pour cette raison, ces fabricants recommandent fortement l'implantation de techniques de conception adaptées au test (DFT) sur les puces.

3.3.3 Test du circuit après encapsulation

Après encapsulation, les circuits obtenus présentent en général une très grande complexité, ce qui rend leur test particulièrement difficile. Les fabricants de MCM définissent le paramètre DL (Defect level) qui désigne le pourcentage de MCMs fournis, susceptible de contenir des défauts par :

$$DL = 1 - {Y_{MCM}}^{(1-FC)} \times 100$$

où Y_{MCM} est le rendement et FC le taux de couverture de fautes.
Par conséquent, le test final du MCM doit couvrir 100% des fautes pour pouvoir affirmer que le MCM commercialisé est exempt de toute défaillance. Dans ce contexte, on comprend mieux la nécessité d'inclure des techniques de DFT au niveau des puces nues.

Parmi ces techniques, on notera le BIST et surtout le Boundary Scan qui semble le plus approprié pour faciliter ces opérations de test. Motorola a mis sur le marché un outil basé sur l'utilisation du standard 1149.1 pour le test des MCMs à l'aide d'un testeur. Le logiciel comprend un générateur de vecteurs de test ainsi qu'un générateur automatique de dictionnaire de fautes pour le test et diagnostic des MCMs. Une autre solution utilisant les facilités offertes par le standard BS a été proposée dans [Zor92]. Dans ce papier, une solution au test des MCMs est décrite. Mais cette solution repose sur la condition sine qua non que tous les modules incluent du BIST et que le BS complet est implanté sur le MCM. Or ce type de solution n'est malheureusement pas toujours applicable actuellement. Ceci est dû au fait que le pourcentage de circuits intégrés incluant du BIST/Boundary Scan reste assez faible (inférieur à 10% en 1992 [HaW92]), et que malgré l'évolution rapide du marché, on ne peut raisonnablement tabler sur la disponibilité de tels circuits dans un avenir proche.

3.4 Adaptation de la méthodologie au cas des MCMs

Pour les raisons invoquées précédemment, on peut logiquement penser que le marché va passer par une étape intermédiaire où les MCMs seraient composés par deux types de puces :

- (1) des puces munies du Boundary Scan (avec ou sans BIST)
- (2) des puces sans aucune méthodologie de DFT

Dans ce cas, comme le montre la figure 3.6, on peut identifier à l'intérieur du MCMs les composants munis du Boundary Scan, et les conglomérats ("clusters") réunissant le reste des composants (sans BS). De cette manière, le MCM pourra être traité comme une carte mixte au sens de la terminologie utilisée dans le chapitre 2.

Dans ce contexte, nous proposons une solution permettant de tirer profit des dispositifs de test présents sur certaines puces pour le test du MCM complet. Il est bien entendu que l'efficacité de cette solution sera proportionnelle au nombre de puces munies de ces facilités de test.

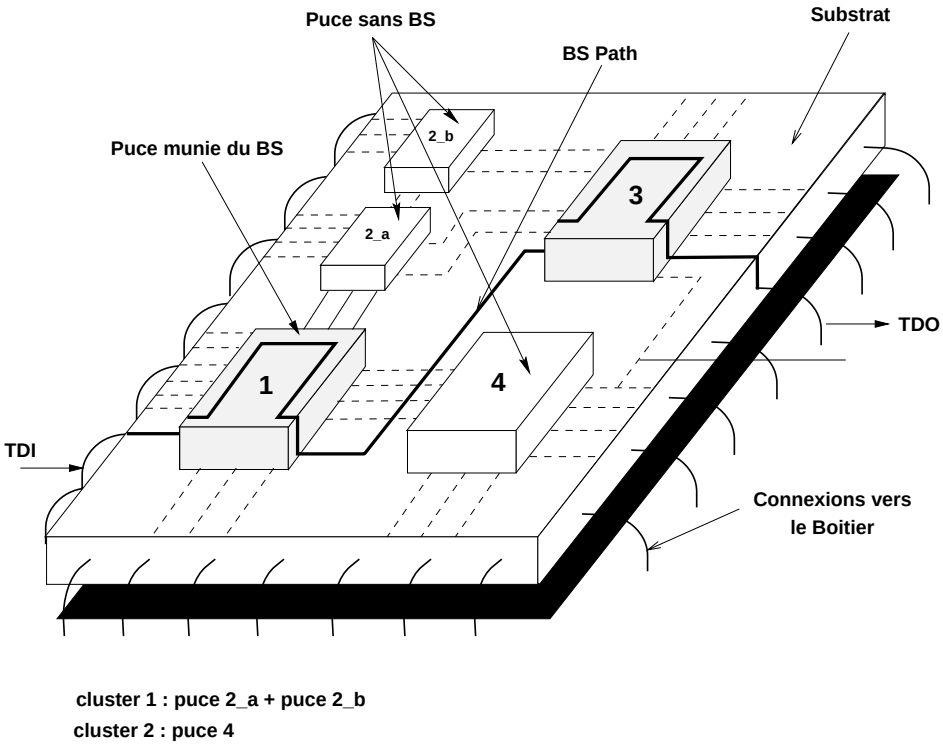

Figure 3.6: MCM Boundary Scan partiel

En supposant que le fabricant de MCM a muni son circuit du standard Boundary Scan [IEEE90], ce qui permet entre autres d'aider au test du substrat, une procédure de test doit pouvoir identifier :

- (1) les conglomérats défectueux

- (2) les fautes d'interconnexion (court−circuits et circuit−ouverts) entre les différentes puces d'un même conglomérat

- (3) les court-circuits entre conglomérats

- (4) les court-circuits entre conglomérats et les puces munies du Boundary Scan

- (5) les fautes d'interconnexion (court−circuits et circuit−ouverts) entres les puces munies du Boundary Scan

- (6) les puces Boundary Scan défectueuses

Une méthodologie pour le test des MCMs munis du Boundary Scan Partiel a été proposée dans [HaW92]. Cependant, le scénario proposé permet seulement d'effectuer le test des fautes de type (5) et (6), alors que pour le diagnostic, il se limite aux conglomérats contrôlant des entrées primaires et délivrant des sorties primaires erronées. Par contre, la méthode que nous proposons ne se limite pas seulement au test et diagnostic des six types de fautes précédemment énumérés, mais elle essaye de le faire de manière simultanée au lieu de le faire séquentiellement (excepté pour les fautes du type (6) où les moyens d'accès sont utilisés pour le test conjoint des puces Boundary Scan et des interconnexions).

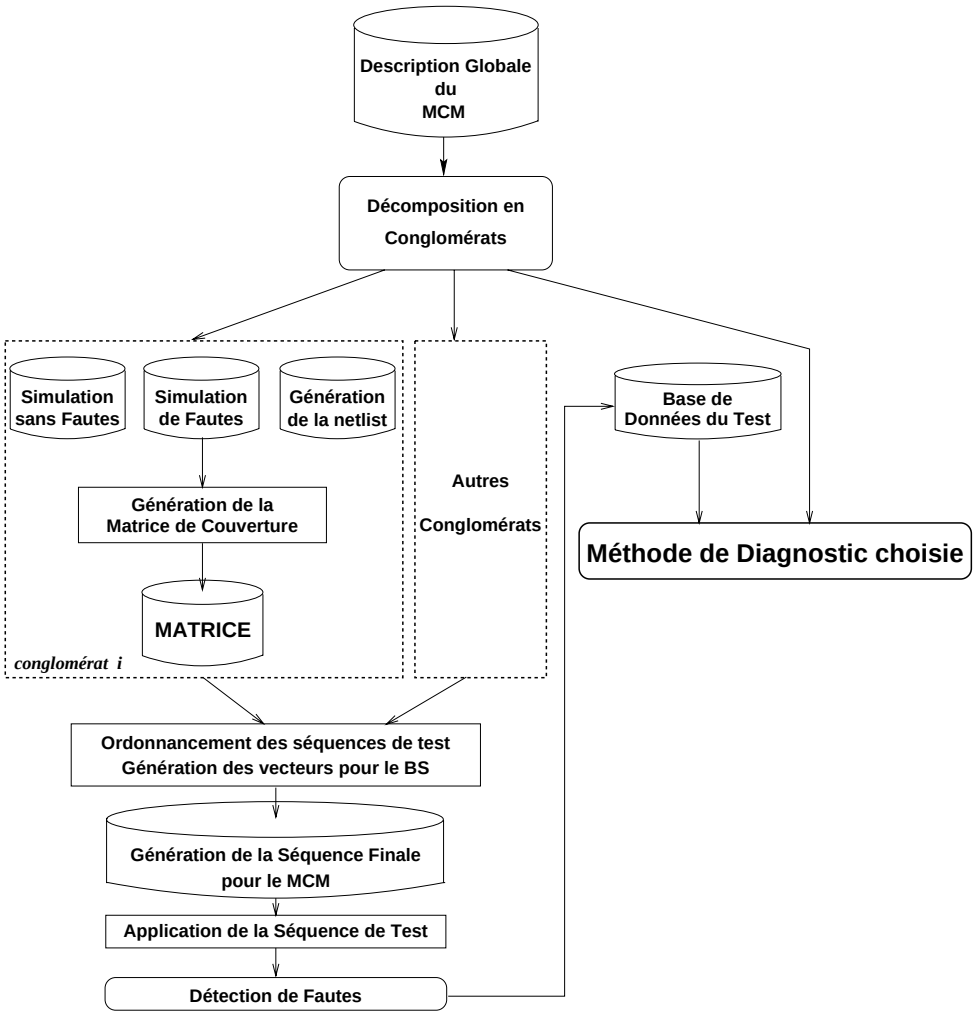

Figure 3.7: Méthodologie Adaptée aux MCMs

Le scénario proposé est une adaptation de la méthode proposée pour les cartes Boundary Scan partiel (chapitre 2). La figure 3.7 résume les différentes tâches à exécuter pour sa mise en œuvre. Les différentes étapes énumérées dans cet organigramme sont identiques à celles décrites pour les cartes mixtes. Les mêmes algorithmes restent applicables.

3.5 Conclusion

La méthodologie que nous avons présentée dans ce chapitre constitue une solution pragmatique [LMT94] au problème du test des MCMs partiellement équipés du standard BS. Elle possède l'avantage de répondre à un besoin réel du marché des MCMs (indisponibilité de puces totalement munies de BS). En plus, elle permet de procéder au test simultané des conglomérats non BS et des interconnexions BS. En outre, elle est indépendante de la technologie utilisée, et peut être utilisée indifféremment pour la validation de prototypes, le test et la maintenance de fin de production.

L'étroite corrélation avec la méthode de diagnostic qui sera présentée dans le prochain chapitre et la disponibilité d'un outil implémentant cette méthodologie (chapitre 6) en fait une application non couteûse qui peut servir de complément à des produits commerciaux existants [Phi93].

Chapitre 4

Approches en vue du Diagnostic

Chapitre 4

Approches en vue du Diagnostic

4.1 Introduction

Quand un système est reconnu défectueux, on se pose naturellement la question de l'origine de la défaillance. Ainsi, la localisation des fautes pose le problème essentiel du diagnostic. En effet, il est très important de faire procéder au diagnostic dans les systèmes électroniques. Au niveau des circuits, ceci permet d'en corriger éventuellement la conception ou d'en améliorer la fabrication afin de minimiser l'occurrence d'exemplaires défectueux. Au niveau des cartes, les fautes constatées sont le plus souvent des fautes de contact : soit des contacts ouverts (rupture de pistes ou soudure sèche) ou des contacts parasites permanents (court−circuit entre pistes, débordement de soudure entre plots ..). Dans ce cas, le diagnostic aura essentiellement pour tâche la localisation rapide et précise de la faute afin de procéder à la réparation. Au niveau système, le diagnostic a pour but de faciliter la maintenance par la diminution de la durée de l'intervention sur site.

Dans le domaine du test, les progrès accomplis au cours de la dernière décennie sont spectaculaires. Ceci a débouché sur l'adoption du standard BS qui permet de faciliter la gestion du test (mais aussi le diagnostic à un certain niveau) de systèmes très complexes. Cependant, même si des avancées significatives ont été réalisées dans

le domaine du diagnostic, beaucoup de travail reste encore à faire.

4.2 Diagnostic de systèmes électroniques : état de l'art

En électronique, deux principales approches ont été adoptées pour le diagnostic jusqu'à ce jour. La première repose sur la génération de Dictionnaire de Fautes (DF) tandis que la deuxième utilise les notions d'Intelligence Artificielle (IA).

4.2.1 Diagnostic à l'aide d'un dictionnaire de fautes

C'est indiscutablement la plus populaire des méthodes de diagnostic en électronique numérique [RiB85]. De nombreux outils de test permettent à leurs utilisateurs d'en générer de manière automatique. Un DF peut être soit une liste de fautes avec chaque vecteur d'entrée et le vecteur de sortie obtenu en présence de chaque faute, soit une liste de vecteurs de test avec les fautes détectées pour chacun de ces vecteurs. Si dans son principe, la génération de DF est simple, en pratique ce n'est plus le cas surtout lorsqu'il s'agit de le faire pour de gros systèmes. En effet, il s'agit de déterminer toutes les fautes testées par chaque vecteur, ce qui est obtenu par simulation sous l'hypothèse de la faute unique. De cette manière, si elle est possible, une simulation complète de toutes les fautes dans un grand système sera très longue et par conséquent coûteuse en temps CPU. A cela, il faut ajouter le problème de l'important espace mémoire nécessaire au stockage du dictionnaire obtenu.

Afin de réduire le temps de simulation et l'occupation mémoire, plusieurs méthodes d'optimisation ont été proposées. Parmi les plus importantes, on citera :

– l'approche qui consiste à générer un dictionnaire SOFE (Stop On First Error) [WaL89]. Dans ce cas, on considère une séquence de test, et les fautes détectées par un vecteur sont immédiatement éliminées de la liste de fautes dès leur première détection (c'est–à–dire que ces fautes ne sont plus simulées). Ceci permet de réduire considérablement le temps de simulation.

– L'utilisation de techniques de compactage [PoR92] [BoK94]. Ces techniques reposent sur le principe d'élimination des informations redondantes contenues dans le DF. Intervenant après la phase de génération du DF, ces techniques n'ont aucun effet sur le temps de simulation, par contre elles permettent de diminuer la taille physique du dictionnaire. En contrepartie, une perte de résolution est souvent constatée.

Parmi les autres limitations liées à l'utilisation des DF, on citera le problème du modèle de fautes utilisé. On rappelle qu'un DF est généré pour une liste de fautes donnée, et donc par rapport à un modèle de fautes bien défini (le modèle du collage logique étant le plus utilisé en général). Par conséquent, le diagnostic de fautes autres que celles modélisées ne pourra se faire car les réponses obtenues sur les sorties du système sous test ne coïncideront pas forcément avec les réponses prévues par la simulation. Et même si c'était le cas, le diagnostic de ces fautes serait erroné.

4.2.2 L'IA au service du diagnostic

Au début des années 80, et après avoir constaté l'évolution rapide de la complexité des systèmes électroniques et les limitations liées à l'utilisation des DF, les travaux de recherche se sont orientés vers l'étude d'une possible contribution que pourrait apporter l'IA dans le domaine de la maintenance. Ces travaux se sont orientés vers le développement de systèmes à base de connaissances (SBC). Ces systèmes sont qualifiés d' "Experts" dans la mesure où ils sont censés reproduire le raisonnement d'un expert humain confronté aux mêmes problèmes. Ils doivent proposer des solutions voisines voire identiques à celles que proposeraient un agent de maintenance guidé par son expérience dans le domaine. Deux types de raisonnements ont été principalement développés pour la réalisation de ces systèmes : le raisonnement de surface et le raisonnement profond.

4.2.2.1 Systèmes à base de raisonnement de surface

Dans ce type de raisonnement plus connu sous sa terminologie anglo-saxonne de ("Shallow Reasoning") [RIC85] [ChM85] [XiS86], les relations entre les symptômes

observés, les tests à exécuter et les décisions à prendre dans chaque cas de défaillance, sont intégrés par les experts dans une base de connaissances. Ensuite, partant d'un ensemble de symptômes observés par l'utilisateur, et en utilisant des relations de *cause à effet* entre pannes et symptômes, le processus de diagnostic tente de construire la chaîne d'inférence filtrant progressivement les éléments de l'ensemble des fautes initialement introduit dans la base des connaissances. En cas de succès, le système donne alors la liste des fautes ayant la plus forte association avec les symptômes observés.

S'ils ont eu beaucoup de succès dans le domaine médical, ces systèmes n'ont pas réussi à percer dans celui de l'électronique. Ceci peut s'expliquer en partie par la difficulté d'énumérer les symptômes et les fautes ainsi que la dérivation de relations empiriques de cause à effet entre les fautes et les symptômes [Dav84]. De plus, une fois réalisés, ces systèmes sont généralement opérationnels uniquement pour une application spécifique donnée [XiS86].

4.2.2.2 Systèmes à base de modèles (raisonnement profond)

Les systèmes de diagnostic qui font appel à cette approche utilisent un raisonnement qualifié de "profond" ("Deep Reasoning") ou ("Model-based Reasoning"). Ce diagnostic s'appuie sur une analyse d'un modèle abstrait du système sous test. Étant donnée une spécification du comportement correct du système, diagnostiquer une faute dans ce système revient à comparer son comportement réel ou observé à celui spécifié.

Les résultats obtenus à l'aide de systèmes basés sur cette approche dépendent de deux points essentiels : la modélisation du bon fonctionnement du dispositif à diagnostiquer et la validité des contradictions détectées entre le dispositif réel et le modèle utilisé [DaK88]. En effet, d'une part la fiabilité des résultats du diagnostic dépend de l'exactitude et de la complétude du modèle utilisé, et d'autre part, l'efficacité des traitements réalisés lors de la mise en œuvre de cette méthode est fortement liée à la quantité d'information impliquée dans le modèle.

Deux approches de modélisation sont utilisées pour un diagnostic à base de modèle :

la première exprime le comportement des composants et du dispositif global sous forme de relation de causalité [PeG87] [Pea88]. La seconde se base sur la description fonctionnelle, comportementale et structurelle du dispositif [Dav84] [KlW87]. Pour illustrer cette approche, nous allons présenter le système PESTICIDE [Mar91]. Cet outil utilise une modélisation basée sur la description. Grâce à la versatilité de la modélisation interne des composants et des fautes dans la base de connaissance, il a été possible d'intégrer cet outil en tant que module de diagnostic dans le système global que nous avons développé, en vue de l'unification du test et du diagnostic dans les cartes Boundary Scan partiel.

4.2.3 Le Système PESTICIDE

PESTICIDE (acronyme de "Prolog−written Expert System as a Tool for Integrated Circuit DEbugging") est un système basé sur la connaissance. Il a été développé au sein du laboratoire TIMA dans le contexte de la localisation de fautes pour des circuits soumis à un test sans contact à l'aide d'un Microscope Electronique à Balayage (MEB), et dont l'architecture globale est présentée en figure 4.1.

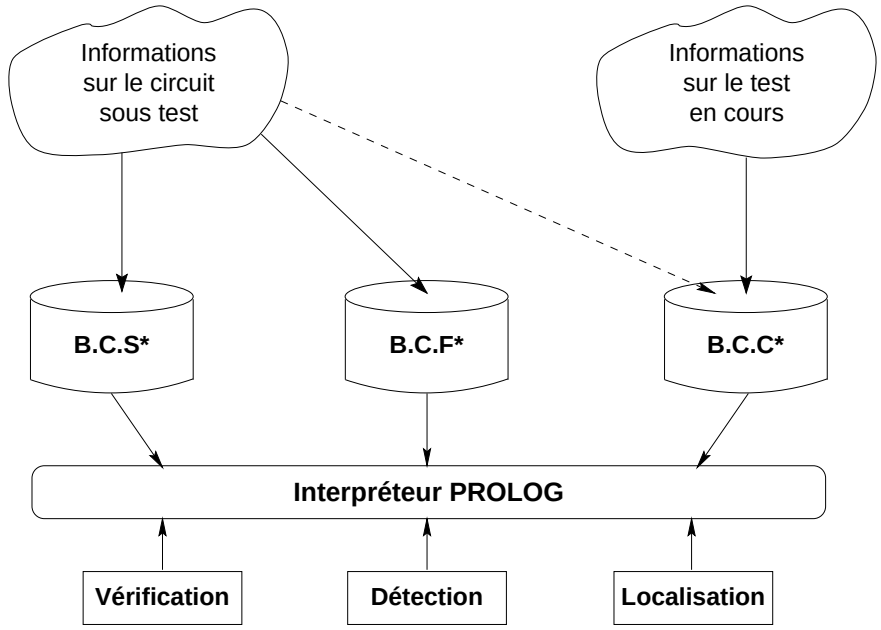

Figure 4.1: Architecture globale de PESTICIDE

4.2.3.1 Modélisation des connaissances

Le modèle utilisé permet la description d'un circuit à l'aide des trois éléments suivants :

– le bloc

– la connexion

– l'interface bloc

Les blocs sont obtenus après le partitionnement du circuit. Un bloc peut correspondre à une porte logique, comme à un module plus complexe. Quant à l'interface bloc, c'est une structure qui permet de relier un bloc à son environnement extérieur c'est à dire aux autres blocs et aux plots du circuit sous test. La connexion correspond à une ligne physique reliant deux blocs à travers leurs interfaces bloc respectives, ou reliant un bloc aux plots du circuit.

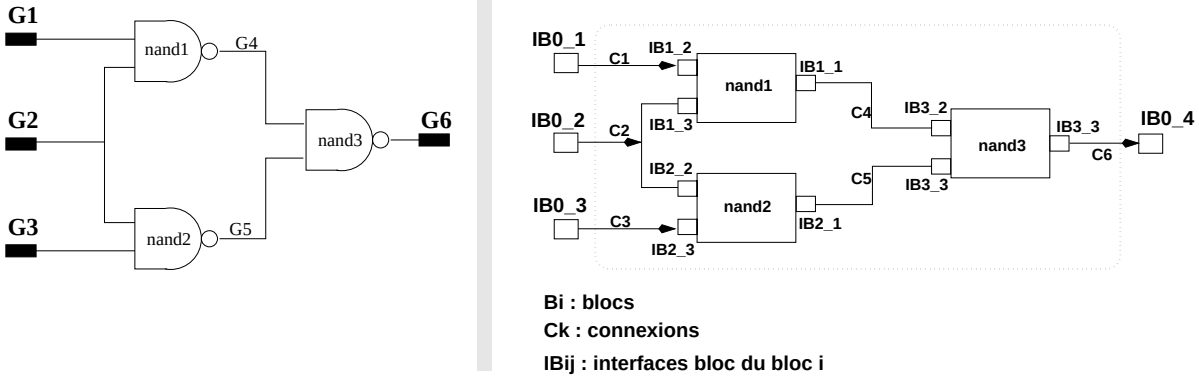

Figure 4.2: Un circuit et son modèle

4.2.3.2 Les Bases de Connaissances

Les trois éléments de base qu'on vient de citer auxquels on ajoute quelques propriétés et hypothèses forment l'essentiel des bases de connaissances de PESTICIDE. Ces bases de connaissances sont au nombre de trois:

– Base de Connaissance Structurelle (BCS). Cette base contient toutes les informations concernant la description structurelle du circuit sous test : liste des blocs, des interfaces bloc et des connexions.
– La Base de Connaissance Fonctionnelle (BCF). Celle ci contient des informations additionnelles telles que les cônes de couverture sur interfaces bloc.
- Une Base de Connaissance Comportementale (BCC) dans laquelle on récolte toutes les données disponibles sur le test en cours. Ces données concernent les interfaces bloc et les hypothèses formulées. La figure 4.3 montre les bases de connaissances relatives au circuit de la figure 4.2.

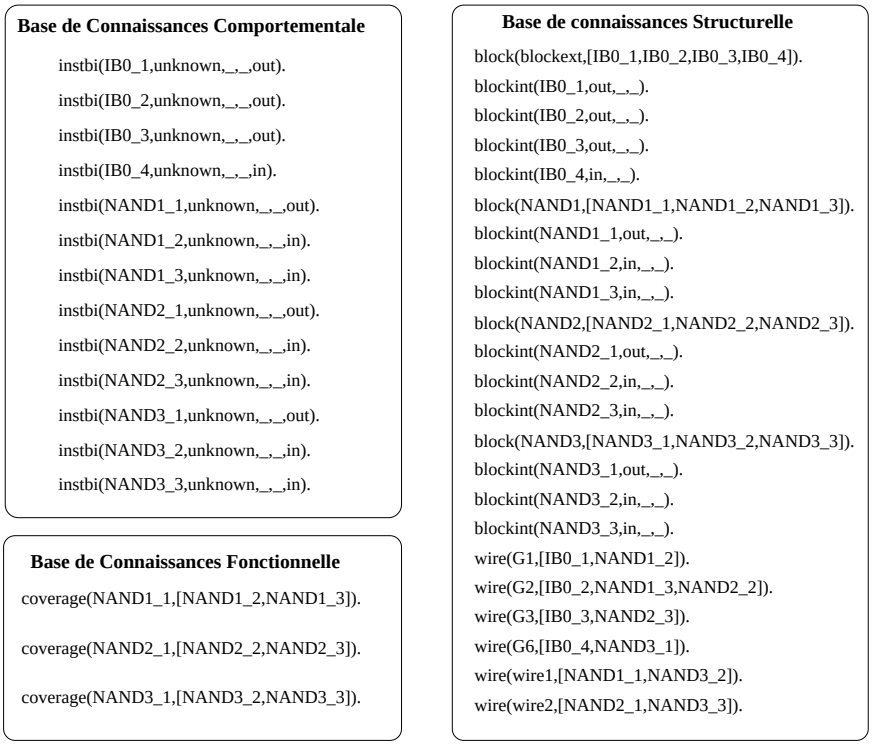

Figure 4.3: Un exemple de Base de Connaissances

4.2.3.3 Détection et localisation de fautes

Le processus de détection et celui de localisation de fautes à travers le circuit sous test constituent les deux tâches principales de PESTICIDE en vue de l'établissement de son diagnostic. Deux types de recherche sont effectués, les recherches de fautes sur les connexions et de fautes dans les blocs. Pour les connexions, il s'agit d'un processus

de détection alors que pour les blocs, il s'agit plutôt d'un problème de localisation. Deux hypothèses de travail sont possibles : une hypothèse de faute simple et une autre de faute multiple.

– Cas de faute simple dans les circuits combinatoires

Pour ce cas, la stratégie se résume par les étapes suivantes :
– Collecter toutes les interfaces bloc dont l'état est déclaré erroné dans la base des connaissances comportementale, ensuite pour chacune de ces interfaces on fait le traitement suivant :
– Examiner le bloc dont elle est une sortie.
– Si aucun des membres du cône de couverture de cette interface bloc n'est erroné alors ce bloc est déclaré défaillant et il manifeste une erreur directe sur la sortie concernée.
– Si l'un au moins des membres du cône de couverture de cette interface bloc est erroné, alors il faut parcourir en amont la chaîne des blocs à travers les connexions, jusqu'aux plots du circuit si nécessaire, pour trouver le bloc défaillant manifestant une erreur propagée sur l'interface bloc en cours de traitement.

– Cas de faute multiple dans les circuits combinatoires

Pour ce cas, la stratégie se résume par les étapes suivantes :
– Recherche de tous les blocs défaillants par l'application de la stratégie "faute simple combinatoire".
– Pour chacun de ces blocs, collecter les interfaces bloc de sortie dont l'état est erroné, et pour chacune de ces interfaces bloc on effectue le traitement suivant :
– On considère les interfaces bloc qui lui sont reliées, celles–ci constituant des entrées de bloc en aval. Chacun de ces blocs sera déclaré présumé défaillant s'il n'est pas déjà classé comme défaillant, et on parcourt à partir de ce bloc la chaîne des blocs en aval en effectuant le même traitement.

4.2.3.4 Amélioration du Diagnostic

PESTICIDE fournit un premier diagnostic en générant les candidats. Si le circuit auquel on a affaire présente un bon profil de diagnostic, cette première étape peut alors s'avérer suffisante. Mais dans le cas ou le circuit est assez complexe, alors il faut songer à améliorer la qualité du diagnostic. Cette amélioration peut être atteinte grâce à l'utilisation d'heuristiques basées sur le concept de "Known−as−good", permettant d'utiliser la connaissance que l'on a des entrées/sorties correctes du circuit sous test. Dans ce cas on peut réduire la liste des candidats en supprimant tous les blocs ne pilotant que des nœuds portant des valeurs correctes. Une autre amélioration peut être obtenue par l'adjonction d'informations supplémentaires dans les bases de connaissances. Ces informations peuvent provenir des résultats d'autres sessions de test ou d'observations effectuées sur le circuit sous test.

4.2.3.5 Exemple d'application

Les résultats présentés dans ce paragraphe sont relatifs au circuit de la figure 2.8. On rappelle que ce circuit représente le s27 des ISCAS89 [BBK89] dont les flip−flops ont été remplacées par un TI BS SCOPE^{TM} OCTAL [TI89].

Une séquence de huit vecteurs de test a été générée à l'aide de SYSTEM HILO. Elle couvre la totalité des fautes de collage, de circuits ouverts et de court−circuits internes. Nous avons utilisé la séquence de test déterminée précédemment sur la carte en supposant l'existence d'un collage à 1 sur le nœud g12. Les résultats obtenus sur les sorties de la carte ont permis la construction de deux bases de connaissances différentes et par la suite de deux sessions de diagnostic dont les conditions sont résumées dans le tableau 4.1.

Session	**Faute**	**Vecteur**	**G10**	**G11**	**G13**	**G17**
S1	g12/s.a.1	V1	ok	ok	erreur	ok
S2	g12/s.a.1	V3	ok	erreur	erreur	erreur

Tableau 4.1: Conditions des différentes sessions

Les résultats de ces sessions de diagnostic sont résumés dans le tableau 4.2. On remarquera que le session S2 donne des résultats médiocres car une grande ambiguïté persiste au niveau de la localisation. Dans la session S1, on a utilisé l'heuristique "Known−as−good" pour la discrimination des candidats. Les résultats se sont alors nettement améliorés.

Session	**Bloc Correct**	**Blocs Candidats**		
		Fautifs		**Présumés Fautifs**
S1	not5	1	nor12	nor13
S2		2	nor12	7 blocs
			not4	13 blocs

Tableau 4.2: résultats des sessions de diagnostic

4.3 Approche Développée en vue du diagnostic

L'importante quantité d'information à gérer pour les gros systèmes (taille des bases de connaissances), le temps d'exécution de l'interpréteur Prolog ainsi que la difficulté d'intégrer efficacement PESTICIDE à un environnement unifié sous Unix nous ont poussé à rechercher une autre solution pour effectuer le diagnostic. Cette solution doit répondre aux critères suivants :

– Simplicité de mise en œuvre

– Nécessité d'appliquation à tout type de systèmes (c'est−à−dire séquentiels et combinatoires).

– Indépendance de tout modèle de fautes.

– Rapidité d'exécution et faible occupation d'espace mémoire

Parmi les solutions possibles, nous avons opté pour une méthode basée sur une approche semi-qualitative. C'est une méthode qui opère en trois phases successives :

- Une première phase responsable de la génération des candidats. Elle repose sur

les informations topologiques extraites de la description du système.

- Une deuxième phase basée sur un raisonnement qualitatif, et dont la tâche consiste à mettre à jour les estimations relatives au degré d'implication de chaque module dans le dysfonctionnement observé.
- Une dernière phase responsable de la localisation du bloc défectueux. Elle utilise un raisonnement semi−qualitatif associant des notions de logique floue.

4.3.1 Génération des candidats

Nous nous plaçons dans le cadre des hypothèses de travail suivantes :

- Le système à diagnostiquer est décrit au niveau des blocs. Ces derniers sont considérés comme des boites noires avec des entrées/sorties. Aucune indication sur la fonctionnalité du bloc n'est requise.
- Nous disposons d'un ensemble de vecteurs de test à appliquer sur les entrées primaires du système, et les réponses correctes relatives à chaque vecteur.

Le processus de diagnostic est invoqué lorsqu'un mauvais fonctionnement du système est constaté. Lors du diagnostic, on peut être confronté à deux types de systèmes : soit des systèmes combinatoires soit des systèmes séquentiels. Dans la méthode que nous proposons, ces deux types de systèmes sont traités globalement de la même façon. Néanmoins, les systèmes séquentiels requièrent un traitement supplémentaire spécifique qu'on indiquera au moment opportun.

4.3.1.1 Cas des systèmes combinatoires

La procédure de génération des candidats utilise un algorithme basé sur la notion de cause à effet. Cet algorithme se déroule en deux phases distinctes :

- Une première phase dite de propagation arrière ("backtracing phase") durant laquelle l'algorithme cherche à mettre à jour les blocs qui ont pu contribuer à engendrer une sortie erronée. Pour cela, et pour chaque sortie affectée, l'algorithme

parcourt le circuit en répertoriant tous les blocs convergeant vers cette sortie. Ces blocs appartiennent tous à ce que l'on désignera par "cône de fanin" ou "coverage cone" [MCl84]. Ces blocs seront déclarés comme "présumés Fautifs", et stockés dans une liste.

- Une deuxième phase dite de propagation avant ("forward propagation") dans laquelle on fait la mise à jour des estimations obtenues lors de la première phase, et en même temps on essaie d'identifier les blocs complètement observables (c'est−à−dire dont les entrées/sorties sont primaires). Ces blocs seront définitivement déclarés comme "Correct" ou "Fautif" en fonction de leur état, et retirés de la liste des blocs "présumé défaillant". Les blocs non affectés et débouchant sur des sorties correctes seront déclarés dans un premier temps comme "Présumé Correct". La mise à jour des estimations se fait de la manière présentée dans le paragraphe 4.3.2.

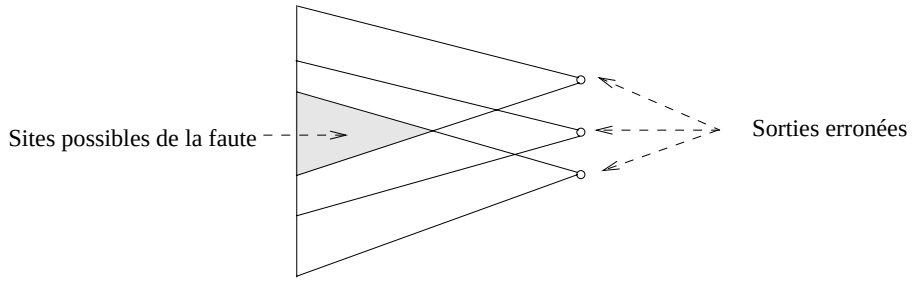

Figure 4.4: Intersection en cas de faute unique

A ce stade de l'algorithme, un choix sur l'hypothèse de faute doit être fait. Si on choisit de se positionner dans l'hypothèse de la faute unique, alors l'étape suivante consiste à chercher l'intersection des ensembles générés (si plusieurs sorties sont erronées). Le bloc fautif appartenant nécessairement à cet ensemble résultant de candidats (figure 4.4). Si la séquence de test appliquée assure une couverture de faute de 100%, alors les blocs restants seront considérés comme "Vraisemblablement correct" . Sinon, ils seront déclarés comme "Présumé Correct".

Par contre, dans le cas où on opte pour l'hypothèse de faute multiple, aucun traitement supplémentaire n'est effectué avant le commencement de la phase suivante.

4.3.1.2 Cas des systèmes séquentiels

Le même algorithme est étendu aux systèmes séquentiels. L'extension est basée sur le fait que l'on doit seulement tenir compte des contraintes supplémentaires relatives aux parties séquentielles du système. En l'état actuel de notre étude, la méthode ne prend en compte que les systèmes séquentiels synchrones. Ces systèmes sont composés de modules combinatoires et de bascules (figure 4.5).

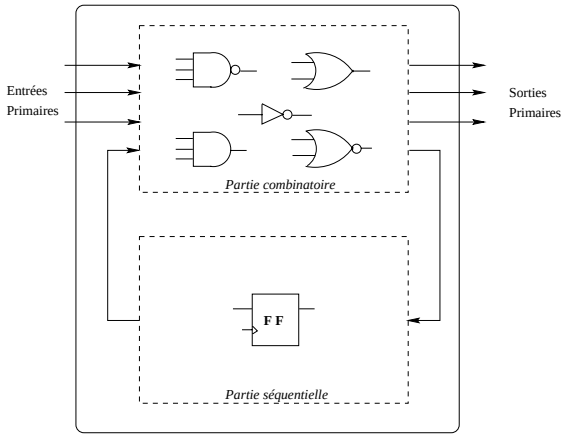

Figure 4.5: Modélisation d'un système séquentiel

Pour ce type de systèmes, le temps est un paramètre très important. En effet, pour propager une valeur à partir d'un nœud donné vers une sortie primaire, on est amené à traverser un certain nombre de bascules. La traversée de chaque bascule nécessite un coup d'horloge. Il est donc clair que l'instant de mesure, en termes de coups d'horloge exécutés depuis le début de la séquence de test, sera une information déterminante pour le processus de diagnostic. En pratique, cette information va servir à éliminer de la liste des candidats, tous ceux dont l'état ne peut être observé sur une sortie primaire au moment de la mesure. Pour cela, nous allons utiliser la notion de profondeur de séquentialité, définie comme le nombre maximal de bascules à traverser pour aller d'un élément donné à une sortie primaire. Néanmoins, dans notre cas la profondeur de séquentialité sera calculée par rapport aux sorties erronées.

4.3.2 Mise à jour des estimations

A la fin de l'étape de génération des candidats, on a donc pour chaque vecteur de test, une liste de candidats avec les estimations qui leur sont associées. La question qu'on se pose naturellement est comment combiner ces diverses sessions de diagnostic (et par conséquent listes de candidats) afin d'obtenir une session unique ?

Généralement, les estimations sont souvent obtenues après un calcul probabiliste. Plusieurs approches sont possibles pour réactualiser leurs valeurs lorsque de nouvelles informations sont disponibles. Cependant, ces approches nécessitent souvent des calculs très lourds et par conséquent très longs. C'est pour cette raison que nous avons choisi d'utiliser plutôt une méthode qualitative pour réaliser cette opération.

Dans sa nature, le raisonnement humain est qualitatif. Lorsqu'il est confronté à un problème, l'homme essaie de trouver la solution en faisant appel à son expérience. En fait, il se réfère à un modèle mental de nature qualitative constitué par l'expérience acquise au fil du temps. Ce raisonnement sera d'autant plus indispensable que parfois les informations ne sont pas disponibles, ou clairement spécifiées. Cette idée constitue le noyau du raisonnement qualitatif dans le domaine de l'Intelligence Artificielle. Jusqu'à présent, cette approche a été le plus souvent appliquée aux systèmes physiques, ce qui lui a valu d'être parmi les domaines les plus productif ces dernières années [Kui93].

Dans le contexte du raisonnement qualitatif, nous avons défini des estimations qualitatives pour caractériser le degré de défectuosité de chaque élément. Ces estimations couvrent l'intervalle compris entre les états "Correct" et "Fautif". Comme le processus est qualitatif, nous avons découpé cet intervalle selon l'ensemble des estimations suivant :

- **C**orrect (C)
- **V**raisemblablement_**C**orrect (V_C)
- **P**lutôt_**C**orrect (P_C)

- **S**upposé_**C**orrect (S_C)
- **A**mbigu (Amb)
- **S**upposé_**F**autif (S_F)
- **P**lutôt_**F**autif (P_F)
- **V**raisemblablement_**F**autif (V_F)
- **F**autif (F)

Il est clair que cette décomposition purement arbitraire, peut être modifiée en fonction des exigences de l'application. Elle dépendra surtout du degré de granularité désiré pour le diagnostic. Pour faciliter la procédure de modification de cette décomposition, nous avons opté pour une représentation matricielle (figure 4.6). Avec ce type de représentation, changer la décomposition revient à remplacer la matrice par une autre. Le fusionnement des résultats des différentes sessions de diagnostic se déroule donc selon les règles qualitatives définies dans la matrice de la figure 4.6. Cette matrice doit être interprétée de la manière suivante : Si un premier vecteur de test donne à un composant X une estimation i, et un deuxième lui donne une estimation j, alors la nouvelle estimation pour ce composant sera $\mathrm{M}(i,j)$, avec M la matrice et i, j respectivement la ligne et la colonne (ou l'inverse étant donné que la matrice est symétrique). Par exemple, si X est estimé S_C (i=*3*) par un vecteur et S_F (j=*7*) par un autre, le résultat est $\mathrm{M}(S\_C,S\_F) = Amb$. On remarquera que cette matrice comprend deux lignes et colonnes particulières ; il s'agit des lignes et colonnes "Correct" et "Fautif". Elles expriment le fait que "Correct" et "Fautif" sont plus proches de l'affirmation que de l'estimation. Ceci découle du fait qu'elles sont obtenues à partir de mesures (observation directes des entrées/sorties primaires du bloc concerné) et que, hormis le cas où elles auraient été effectuées lors d'un "test escape case" [Ben94], ces mesures sont considérées comme fiables.

Il est à noter que dans ce processus, aucune hypothèse n'a été avancée sur la capacité de diagnostic ("diagnosis power") de chaque vecteur. Cette notion exprime le fait

	Correct	**V_C**	**S_C**	**P_C**	**Amb**	**P_F**	**S_F**	**V_F**	**Fautif**
Correct	Correct	Correct	Correct	Correct	Correct	Correct	Correct	Correct	
V_C	Correct	Correct	V_C	S_C	S_C	S_C	P_C	Amb	Fautif
S_C	Correct	V_C	V_C	S_C	P_C	P_C	Amb	P_F	Fautif
P_C	Correct	S_C	S_C	S_C	P_C	Amb	P_F	S_F	Fautif
Amb	Correct	S_C	P_C	P_C	Amb	P_F	P_F	S_F	Fautif
P_F	Correct	S_C	P_C	Amb	P_F	S_F	S_F	S_F	Fautif
S_F	Correct	P_C	Amb	P_F	P_F	S_F	V_F	V_F	Fautif
V_F	Correct	Amb	P_F	S_F	S_F	S_F	V_F	Fautif	Fautif
Fautif		Fautif	Fautif	Fautif	Fautif	Fautif	Fautif	Fautif	Fautif

Figure 4.6: Règles de combinaison des estimations

qu'un vecteur permet de diagnostiquer plus ou moins facilement un faute. En fait, cette notion prend toute son importance dans l'hypothèse de la faute unique. Un vecteur de test qui détecte un nombre réduit de fautes (donc pas très efficace pour le taux de couverture !) sera efficace lors du diagnostic. Or cette information ne peut être obtenue que par une simulation de fautes, ce qui présume de la disponibilité simultanée de la description au niveau porte ("netlist") du système et de l'outil de simulation. Or ces deux contraintes ne font pas partie de nos hypothèses de travail. De ce fait, le même poids (en terme de capacité de diagnostic) est appliqué à tous les vecteurs. Cependant, si ces informations sont disponibles, leur intégration au processus de combinaison sera toujours possible.

4.3.3 Exemple d'application

Le circuit de la figure 4.7 représente le c17, le plus petit des benchmarks de la série ISCAS'85 [BrF85]. Dans ce circuit, nous avons injecté un collage logique à 1 (s-a-1) sur le nœud G23. On a généré à l'aide de HITEST [Ved94] deux vecteurs de

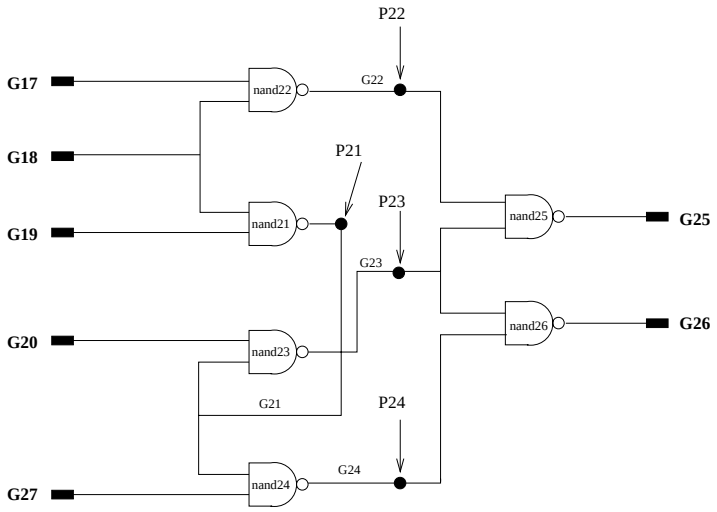

Figure 4.7: ISCAS85 c17 (6 blocs dans le module)

test capables de détecter cette faute. Les conditions de test sont résumées dans le tableau 4.3.

Session	**G25**	**G26**
Session 1	erreur	erreur
Session 2	erreur	ok

Tableau 4.3: Conditions des différentes sessions pour le c17

Le tableau 4.4 montre les résultats de la phase de génération des candidats et la mise à jour des estimations dans l'hypothèse de la faute unique.

Candidat	**nand21**	**nand22**	**nand23**	**nand 24**	**nand25**	**nand26**
Session1	S_F	V_C	S_F	V_C	V_C	V_C
Session2	S_F	S_F	S_F	V_C	S_F	V_C
Résultat	S_F	P_C	S_F	C	P_C	C

Tableau 4.4: Génération des candidats : Hypothèse de faute simple

Le tableau 4.5 donne les résultats obtenus dans l'hypothèse de faute multiple. On remarquera une perte de précision par rapport au résultat précédent. Ceci s'explique par le fait que l'hypothèse de faute simple est plus restrictive et ne correspond pas souvent à la réalité.

Candidat	**nand21**	**nand22**	**nand23**	**nand 24**	**nand25**	**nand26**
Session1	S_F	S_F	S_F	S_F	S_F	S_F
Session2	S_F	S_F	S_F	V_C	S_F	V_C
Résultat	S_F	S_F	S_F	S_C	S_F	S_ C

Tableau 4.5: Génération des candidats : Hypothèse de faute multiple

4.4 Conclusion

Dans ce chapitre, après avoir passé en revue les différentes approches utilisées pour le diagnostic des systèmes électroniques, nous avons développé une méthode basée sur un raisonnement qualitatif pour la génération et la classification des candidats. Cette classification utilise des estimations qualitatives basées sur des quantifications linguistiques, permettant ainsi d'éviter de manipuler des calculs lourds et fastidieux. La fusion des différentes sessions de test est effectuée à l'aide d'une matrice. Ceci permet une simplification du processus et une grande malléabilité quant à la définition de la granularité du diagnostic. En effet, pour la changer, il suffit de remplacer cette matrice par une nouvelle matrice.

La technique proposée repose uniquement sur l'observation des fautes sur les sorties primaires des systèmes à diagnostiquer. Elle n'est pas liée à un modèle de faute particulier et ne suppose aucune information sur la fonctionnalité des blocs internes composant le système. Néanmoins, si on se limite à la phase de génération des candidats, ceci peut s'avérer suffisant pour des petits systèmes. Mais ce n'est généralement pas suffisant pour effectuer un bon diagnostic lorsqu'il s'agit de systèmes un peu plus complexes. La qualité du diagnostic obtenu à ce moment va dépendre de la qualité des vecteurs de test et le profil du système à tester. Afin de remédier à ces insuffisances, nous proposons dans le prochain chapitre une nouvelle stratégie complémentaire susceptible d'améliorer la précision du diagnostic réalisé à l'aide de cette approche.

Chapitre 5

Recherche du meilleur nœud à tester

Chapitre 5

Recherche du meilleur nœud à tester

5.1 Introduction

Quelle que soit la méthode utilisée, les mesures effectuées sur les nœuds primaires du système permettent rarement de remonter directement au site de la faute. Les techniques de diagnostic reposent généralement sur l'établissement d'un chemin de causalité pour la propagation des fautes à l'intérieur du système sous test. Par conséquent, des mesures supplémentaires au niveau des nœuds internes du système sont nécessaires pour raffiner le diagnostic, en diminuant les divergences à l'intérieur du chemin de causalité.

Au niveau des circuits ou des MCMs, ces informations peuvent être obtenues soit par l'intermédiaire des dispositifs de test tels que le Boundary Scan (total ou partiel) s'il est implanté, sinon grâce à des mesures de potentiel effectuées à l'aide du microscope électronique à balayage (dans le cadre d'un test sans contact). Pour les cartes électroniques, les problèmes d'accès sont un peu moins difficiles. Les informations peuvent être alors recueillies soit par l'intermédiaire du dispositif Boundary

Scan s'il existe, soit à l'aide de lit-à-clous si la carte le permet, sinon à l'aide de sondes individuelles reliées au testeur.

Si les moyens d'accès aux informations sont disponibles, il reste néanmoins à résoudre le problème du choix des nœuds à tester. Ce choix doit être effectué de manière pragmatique, car une procédure de choix aléatoire peut se révéler coûteuse et totalement inefficace pour l'amélioration du diagnostic. Pour cette raison, nous proposons dans ce chapitre une méthode basée sur la logique floue qui permet d'indiquer les meilleurs nœuds à sonder pour une convergence rapide vers le (ou les) site(s) de la (ou des) faute(s).

5.2 Raisonnement qualitatif à base de logique floue

La théorie des ensembles flous a été initiée par *Zadeh* en 1965 [Zad65]. Elle avait fait à ce moment l'objet de beaucoup de discussions. Récemment, cette théorie a pris beaucoup d'importance, et a trouvé plusieurs champs d'application. On citera notamment les commandes de processus à base de signaux analogiques ainsi que le contrôle de processus à large spectre de paramètres (par exemple le freinage Anti-patinage plus connu sous le label ABS).

Notre choix pour l'utilisation de la logique floue est motivé par le souci d'éviter toute approche purement probabiliste. Ces approches ont déjà été utilisées dans d'autres systèmes de diagnostic. Elles ont du être délaissées parce qu'elles s'accompagnaient souvent de lourds calculs, d'hypothèses fastidieuses sur les probabilités *a priori* concernant les différents modules du système sous test, ainsi que sur les dépendances statistiques qui les unissent.

Une approche à base de logique floue doit pouvoir aborder le problème du diagnostic à la manière d'un expert, par la réduction des calculs nécessaires, en remplaçant les quantifications numériques par des quantifications linguistiques.

5.2.1 La représentation de *l'imprécis* et de *l'incertain*

Dans le cadre de la logique classique, une proposition est soit *vraie*, soit *fausse*, soit *indéterminée* par rapport à une théorie donnée. Or, dans son raisonnement, l'être humain s'appuie souvent sur des connaissances confuses et des données imprécises, incertaines, voire inexactes. Néanmoins, son raisonnement peut être cohérent et aboutir à des résultats corrects.

Les systèmes électroniques actuels, par leur taille et complexité sont générateurs de grandes quantités d'informations. La nécessité de trouver un moyen pour les traiter en englobant des propositions de plus en plus générales, et de recueillir des données qui sont loin d'être précises et définitives, est devenue pressante. En l'absence de nouvelles théories, la logique floue semble être la solution adéquate.

La logique floue a donc été établie dans le but de traiter *l'imprécis* et *l'incertain*. Mais avant d'aller plus loin dans le développement de cette théorie, il faudrait déjà pouvoir faire la dissociation de ces deux qualificatifs. *Incertain* s'applique à des éléments de connaissance dont la valeur de vérité n'est pas connue de manière exacte. Elle est connue avec plus ou moins d'incertitude : "Il sera probablement élu" donne une information incertaine concernant l'élection de l'individu visé. Quant à *imprécis*, cela s'applique plutôt à des éléments de connaissances dont la valeur de vérité est elle même imprécise : "De Paris à Grenoble, il y a 600 Km environ". En définitive, la différence entre ces deux qualificatifs peut se résumer ainsi : Pour un élément d'information, l'imprécis concerne la valeur tandis que l'incertain est relatif à sa vérité.

5.2.2 Limitations de la logique classique

Différentes approches ont été développées pour prendre en compte ces notions. L'approche probabiliste (et le raisonnement Bayesien) a été parmi l'une des premières solutions adoptées. Mais le modèle probabiliste repose sur la saisie d'informations dispersées mais précises [DuP87]. Par conséquent, dès que la précision fait défaut,

on s'éloigne du domaine de validité de ce modèle. En plus, de toutes les manières, cette approche représente un cadre trop étroit pour pouvoir rendre compte de tous les aspects du jugement incertain. Parmi les autres limitations rencontrées, on citera :
– l'exigence de pouvoir estimer toutes les probabilités *a priori*, chose qui n'est pas toujours possible.
– l'impossibilité de représenter l'ignorance : en probabilité, la relation fondamentale $P(s) + P(\overline{s}) = 1$ signifie que la connaissance de l'événement implique la connaissance totale de son contraire. Dans [HBC91], les auteurs montrent que dans le domaine des systèmes experts, donner la valeur 0.5 à la probabilité conditionnelle $P(H/E)$ peut être interprété comme accorder une confiance à une règle de la forme : *si E alors H*. Dans ce cas, la règle *si E alors* $\overline{H}$ aura la même confiance, ce qui n'est pas forcément l'avis de l'expert. Cet exemple illustre en fait une règle générale qui stipule que, en donnant des probabilités à des événements inconnus mais liés, on tombe rapidement dans ce que l'on appelle le paradoxe de l'ignorance.

Les limites du raisonnement probabiliste ont conduit à la recherche d'autre solutions pour estimer *l'incertain* dans l'intervalle [0,1], et qui permettent en particulier de placer la confiance en des éléments de connaissance à l'intérieur de sous−intervalles de [0,1]. Les axiomes suivants définissent des familles de mesures satisfaisantes :
$G(Faux) = 0$
$G(Vrai) = 1$
Si q est une conséquence logique de p, alors : $G(q) \geq G(p)$
La définition de G a les conséquences suivantes :
$G(p \wedge q) \leq min\,(G(p), G(q))$
$G(p \vee q) \geq max\,(G(p), G(q))$

5.2.3 Le raisonnement flou

La théorie des possibilités qui a été formulée par *Zadeh* en 1977, offre un modèle de quantification du jugement qui permet aussi une généralisation canonique du calcul d'erreurs [DuP87]. Le caractère flou d'une information réside dans l'absence de contour bien délimité de l'ensemble des valeurs affectées aux objets qu'elle concerne.

En logique classique, la notion d'appartenance à un ensemble est exclusive : ou bien $x \in X$ ou bien $x \notin X$. En logique floue, cette notion disparaît pour faire place à une nouvelle notion d'appartenance qui permet une représentation beaucoup plus large.

5.2.3.1 Le degré d'appartenance

Le degré d'appartenance est un concept fondamental dans la logique floue. Toutes les structures de cette logique seront définies à partir de ce concept. Pour expliquer sa signification, nous allons étudier l'exemple de la figure 5.1. Dans cet exemple, Nous définissons l'ensemble $Grand = [80, 100]$. Si on considère le nombre 79, quelle serait sa position vis à vis de cet ensemble ? Nous pouvons seulement affirmer qu'il n'est pas *Grand* ($79 \notin [80, 100]$). Maintenant, considérons le nouvel ensemble $Petit = [60, 79]$, alors à ce moment on peut affirmer que *79 est Petit* ($79 \in [60, 79]$) malgré la faible différence qui le sépare de *Grand*.

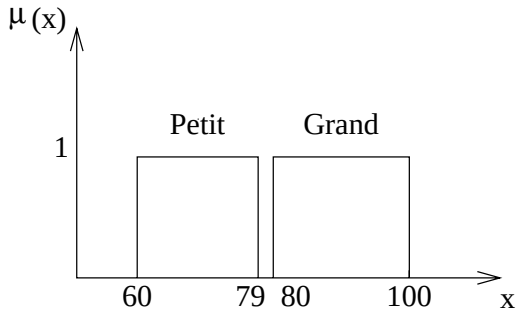

Figure 5.1: Intervalle classique

En logique floue, 79 sera considéré comme *Petit* avec un degré d'appartenance égal à 1, et comme *Grand* avec un degré d'appartenance égal à 0.9 . De ce fait, on vient de définir *l'intervalle flou* (figure 5.2). Cette représentation en intervalles flous est beaucoup plus générale que celle en intervalles classiques. Elle permet de tenir compte des informations présentes autour des bornes inférieures et supérieures de l'intervalle classique.

Par conséquent, un ensemble flou A sera défini sur un domaine T par la donnée

d'une fonction μ_A telle que :

$$\mu_A : T \longmapsto [0,1]$$
$$t \in T \longrightarrow \mu_A(t)$$

où $\mu_A(t)$ *représente le degré d'appartenance de* t *à* A

On définit le *support* d'un ensemble flou A, comme l'ensemble des éléments avec un degré d'appartenance supérieur à zéro. De la même manière, on définit le *noyau* comme l'ensemble des éléments dont le degré d'appartenance est égal à 1.

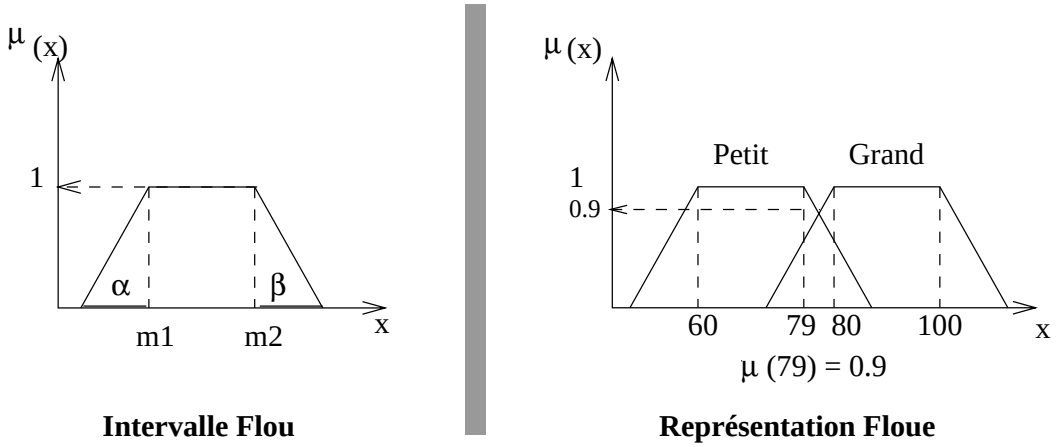

Figure 5.2: Intervalle et représentation Flous

5.2.4 Quantités, intervalles et nombres flous

On désigne par *Quantité floue* Q , un ensemble flou sur les réels, l'application définie par : $\mu_Q : \Re \longmapsto [0,1]$. Cette application est supposée normalisée sauf indication contraire. On désigne par *valeur modale de* Q, tout nombre réel m appartenant au *noyau* de Q. Un intervalle flou est une quantité floue convexe c'est−à−dire dont la fonction d'appartenance est quasi−concave [DuP87] :

$$\forall u, v, \forall w \in [u,v],\ \mu_Q(w) \geq min\ (\mu_Q(u), \mu_Q(v))$$

qui peut être écrite sous une forme plus connue de la convexité :

$$\forall u, v, \lambda \in [0,1],\ \mu_Q(\lambda u + (1-\lambda)v) \geq min\ (\mu_Q(u), \mu_Q(v))$$

Une quantité floue est convexe si et seulement si ses $\alpha-$coupes sont convexes, c'est−à−dire sont des intervalles (bornés ou non). Les intervalles fermés sont généralisés par des intervalles flous dont la fonction d'appartenance est semi−continue supérieurement (s.c.s) c−à−d, par définition, dont les $\alpha-$coupes sont des intervalles fermés. Les sous−ensembles compacts de $\Re$ (fermés et bornés) sont généralisés par les quantités floues s.c.s, à support compact. On appellera nombre flou un intervalle s.c.s. à support compact et de valeur modale unique. Si M est un nombre flou de valeur modale m, M est une représentation possible de *environ* m [DuP87]. Dans le cas d'un intervalle flou, l'ensemble des valeurs modales est un intervalle.

L'intervalle flou permet d'avoir une bonne représentation des quantités imprécises ; on choisira le support de façon à être sûr que la quantité cernée n'en sortira pas, et le noyau contiendra les valeurs les plus plausibles. Par conséquent, la détermination d'une fonction d'appartenance peut être définie par rapport aux quantités floues à représenter.

5.2.5 Calcul pratique des intervalles flous

En pratique, un intervalle flou sera défini par un quadruplet $[m1, m2, \alpha, \beta]$ (figure 5.2), avec m1 la valeur modale inférieure, m2 la valeur modale supérieure, α etβ respectivement les limites inférieure et supérieure. De la même manière, un nombre réel m peut être défini par l'intervalle $[m, m, 0, 0]$, un intervalle classique $[a, b]$ par $[a, b, 0, 0]$, un nombre flou m par $[m, m, \alpha, \beta]$. Ceci permet de représenter uniformément un nombre réel, un intervalle classique, un nombre flou, et un intervalle flou. Par conséquent, on peut envisager les opérations arithmétiques suivantes :

Soit $M = [m1, m2, \alpha, \beta]$ et $N = [n1, n2, \gamma, \delta]$, deux intervalles flous :

- $M \oplus N = [m1 + n1, m2 + n2, \alpha + \gamma, \beta + \delta]$; Addition Floue
- $M \ominus N = [m1 - n2, m2 - n1, \alpha + \delta, \beta + \gamma]$; Soustraction Floue

Le multiplication est définie de manière fonctionnelle comme expliqué dans [DuP87], et elle est approchée par [MoM95] :

- $M \otimes N = [min(S), max(S), abs\ (min(S') - min(S)), abs\ (max(S') - max(S))]$,
  pour $S = (m1 * n1, m1 * n2, m2 * n1, m2 * n2)$ et
  $S' = ((m1 - \alpha) * (n1 - \gamma), (m1 - \alpha) * (n2 + \delta), (m2 + \beta) * (n1 - \gamma), (m2 + \beta) * (n2 + \delta))$,
  avec le but de trouver les nouvelles extrémités.

La division sera approchée de la même manière [MoM95] par :

- $M \oslash N = [min(S), max(S), abs(min(S') - min(S)), abs(max(S') - max(S))]$,
  pour S = (m1/n1,m1/n2,m2/n1,m2/n2) et $S' = ((m1-\alpha)/(n1-\gamma), (m1-\alpha)/(n2+\delta), (m2+\beta)/(n1-\gamma), (m2+\beta)/(n2+\delta))$, avec $n1$ et $n2$ non nul et de même signe.

5.3 Meilleur point à tester : une approche qualitative floue

Jusqu'à présent, la méthode que nous proposons permet de synthétiser les différentes sessions de diagnostic en une session unique. Par conséquent, partant de plusieurs listes de candidats avec leurs estimations, nous obtenons à la fin du processus une liste unique. A ce stade de la méthode, et mis à part le cas des petits systèmes ou bien le cas où l'ingénieur de maintenance juge suffisantes les informations mises à sa disposition, une importante ambiguïté persiste concernant l'origine de la ou des fautes qui induisent le dysfonctionnement observé du système sous test.

Afin de remédier à ce problème et d'améliorer l'efficacité de la méthode développée, nous proposons une solution consistant en une étape supplémentaire responsable de la détermination de l'ensemble des nœuds à tester pour localiser la ou les sites potentiels de fautes. Le choix de ces nœuds doit se faire selon des critères bien déterminés de façon à diminuer le nombre de nœuds et par suite le coût de cette opération en termes de temps et d'équipements nécessaires.

Au cours de cette étape supplémentaire, une fonction de coût est calculée pour

chaque nœud afin d'estimer l'importance de l'information qui serait recueillie sur ce dernier. Cette fonction utilise à cet effet trois paramètres :

- Le cône de Fanout : C_{fout} (figure 5.3). On désigne par ce terme l'ensemble des composants qui sont susceptibles d'être influencés par le nœud considéré.
- Le cône de Fanin : C_{fin} (figure 5.3). C'est l'ensemble des composants qui peuvent exercer une influence sur le nœud considéré.
- L'entropie floue : il s'agit de la mesure de l'entropie des probabilités des candidats.

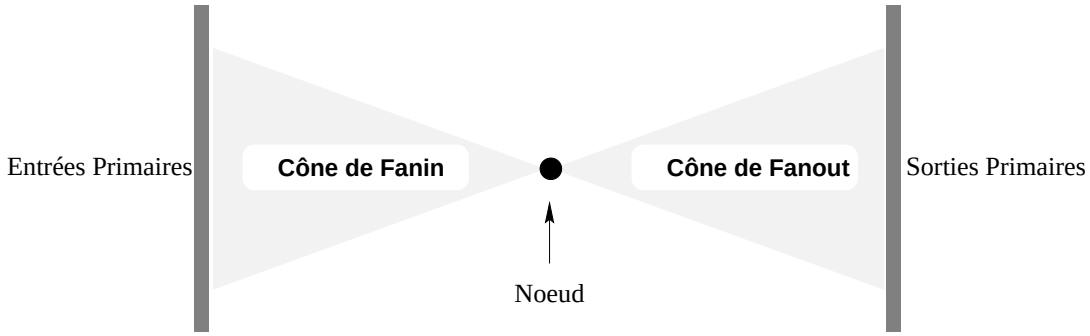

Figure 5.3: Cône de Fanin et Cône de fanout

Le choix de ces paramètres est motivé par les raisons suivantes :

– En ce qui concerne l'entropie, étant donnée une étape particulière dans le processus de diagnostic, on est amené à évaluer l'apport de chaque mesure effectuée sur le système pour déterminer la mesure à effectuer au cours de l'étape suivante. Dans ce cas, avec les nombreuses propriétés importantes [BoD86] qu'elle possède, la mesure de l'entropie des probabilités constitue une bonne indication comme elle l'a déjà démontré dans d'autres applications. Pour les composants estimés "*Vraisemblablement_Correct*" ou "*Vraisemblablement_Fautif*", l'entropie sera minimale. Par contre, pour les éléments dont l'estimation est "*Ambigu*", l'entropie sera maximale et par conséquent la quantité d'information apportée par la mesure de ce nœud n'aura pas beaucoup d'effet sur le degré d'ambiguïté courant. Comme les données

manipulées sont floues, et dans le but d'éviter l'influence des cardinaux des ensembles considérés, on calcule *l'entropie moyenne floue* comme expliqué dans le paragraphe 5.3.2. L'ensemble présentant l'entropie minimale sera considéré comme le meilleur, puisque présentant une ambiguïté minimale.

– Le choix du *cône de fanin* comme paramètre structurel s'explique par le fait que, si la mesure d'un nœud appartenant au chemin de propagation de la faute montre qu'il est correct, alors on peut en déduire que la faute se trouve quelque part dans l'ensemble de ses successeurs (cône de fanout), ce qui permet de réduire l'ambiguïté sur son cône de fanin.

– Le choix du *cône de fanout* s'explique de manière analogue à celle du cône de fanin. Si une mesure effectuée sur un nœud donne une valeur erronée, ceci indique que la faute se situe quelque part dans l'ensemble de ses prédécesseurs (cône de fanin), et par suite l'ambiguïté est atténuée sur le cône de fanout.

En plus, le fait de considérer simultanément ces deux cônes (fanin et fanout) permet d'équilibrer leurs influences respectives. En effet, considérer uniquement le cône de fanout revient à favoriser les composants voisins des entrées primaires. A l'inverse, considérer uniquement le cône de fanin revient à favoriser les éléments voisins des sorties primaires. Or les nœuds situés au milieu des chemins sont souvent les plus intéressants à sonder (quelles que soient les conditions, on éclaire 50% des éléments). Pour cette raison, nous définissons le paramètre *Degré d'influence* (D_{inf}), une combinaison linéaire de C_{fin} et de C_{fout} de la manière suivante :

$$\boxed{D_{inf} = \alpha * C_{fin} + \beta * C_{fout}}$$

α, β sont les coefficients de pondération. Ils permettent de donner un poids plus ou moins important aux éléments qui leur sont associés. Le choix de ces coefficients sera étudié dans le paragraphe 5.3.2

5.3.1 L'entropie floue

Dans [NeR75], les auteurs déclarent que "Un événement est considéré aléatoire quand son occurrence n'est pas exactement déterminée. Un événement peut être considéré comme flou quand il est intrinsèquement imprécis. Néanmoins, le flou et l'aléatoire ne sont pas incompatibles, et dans plusieurs cas, ils peuvent être présents simultanément. Pour cette raison, la notion de *probabilité floue* est loin d'être vide de sens, et on peut alors définir une entropie relative à un ensemble flou".

Dans [Kos91], l'auteur évoque ce problème sous un angle "géométrique". Il donne par la suite des explications très détaillées sur l'entropie floue.

Pour notre application, le module sous test est considéré comme un ensemble de composants dont chacun est muni d'une estimation de son état en terme de probabilité floue. Pour calculer cette entropie, nous avons adopté la définition intrinsèque de *Shannon* qu'on a adaptée au calcul flou.

Soit S un ensemble de n composants caractérisés par leurs estimations floues. L'entropie floue de cet ensemble se calcule de la manière suivante :

$$\boxed{Ent(S) = \oplus_{i=1}^{n} F_i \otimes Log_2(1 \oslash F_i)}$$

Avec F_i l'estimation de défectuosité du composant i, et $\oplus$, $\otimes$ et $\oslash$ représentent respectivement l'addition, la multiplication et la division floues (paragraphe 5.2.5). Les fonctions Logarithme et calcul de l'inverse sont les suivantes [BoD86] :

Pour $m = [m1, m2, \alpha, \beta]$

* $log_2 m = [log_2(m1), log_2(m2), log_2(m1/(m1-\alpha)), log_2(m2+\beta)/m2]$ avec $\alpha > 0$
* $1/m = [1/m2, 1/m1, \beta/(m2 * (m2+\beta)), \alpha/(m1 * (m1-\alpha))]$ avec $m1 > 0$ ou $m2 < 0$

5.3.2 Choix du point à tester

Afin de déterminer le meilleur point à tester, nous devons évaluer l'importance topologique de chaque nœud dans le processus de diagnostic. Pour cela, nous disposons de deux informations concernant chaque nœud : l'entropie et le degré d'influence.

Le problème du choix du meilleur point à tester peut être formulé de la manière suivante : étant donnés n points (chaque point représente un ensemble de candidats de cardinal k), on calcule son entropie floue moyenne et son degré d'influence. Donc, pour chaque point, on obtient deux vecteurs d'information suivants :
$Ent_{moy}(i) = \frac{1}{k} * Ent_{i\ (i=1,k)}$ et $Df_{j\ (j=1,k)}$.

Supposons maintenant que $E = min\ \{Ent_{moy}(i)_{(i=1,n)}\}$ et $D = max\ \{Df(j)_{(j=1,n)}\}$. Il est alors clair que le point optimal $P_{opt}\ (E, D)$ représente le meilleur point à tester car il possède en même temps, un fort degré d'influence et une faible entropie.

Mais en pratique, il est rare de tomber sur le point idéal vérifiant simultanément ces deux conditions. Dans ce cas, le meilleur point serait celui qui s'en rapproche le plus. Reste à trouver un moyen pour chercher ce point.

5.3.2.1 La distance de Hamming

Une solution possible pour la recherche du point le plus proche du point optimal est *la distance de Hamming.*

Soit $P_{opt}\,(E, D)$ et les deux vecteurs $Ent_{i\ (i=1,n)}$ et $Df_{j\ (j=1,n)}$. Pour ces deux vecteurs, on calcule la distance suivante :

$$D_{ham}\ (Pk, P)\ =\ \sqrt{(Ent(k) - E)^2 + (1 - \frac{Df(k)}{D})^2} \quad \text{pour chaque } k \in [1, n]$$

Le point donnant la distance minimale correspondra au point recherché. Si pour le même i, on a $Ent(i) = E$ et $Df(i) = D$, alors la distance obtenue sera nulle, et il s'agira à ce moment du point optimal. On notera au passage que les termes de cette fonction sont tous normalisés ($Ent_{moy} < 1$ par définition et $\frac{Df_i}{D} \leq 1$).

Cette normalisation permet d'obtenir un résultat borné ce qui facilite la détection des problèmes de calculs.

5.3.2.2 La fonction de coût

Le calcul d'une fonction de coût à minimiser est aussi une solution possible pour la recherche du meilleur nœud à tester. Nous savons déjà, grâce à la théorie de l'information et celle des décisions que l'entropie est un bon paramètre d'estimation. Si à ce dernier, on ajoute des informations d'ordre structurelles (par conséquent spécifiques au système sous test), on devrait arriver à construire une fonction de coût efficace. Comme nous désirons minimiser l'entropie et maximiser le degré d'influence, nous avons choisi une fonction qui a la forme suivante :

$$\boxed{Cost\,(Ent, Df) = \alpha * Ent \; + \; \beta * \tfrac{1}{Df}}$$

Avec α et β des coefficients de pondération.

Ces coefficients de pondération peuvent avoir des importances diverses. Différents poids peuvent leur être affectés en fonction de l'importance à donner à chaque paramètre de la fonction. A ce niveau, la logique floue est particulièrement adaptée, car on n'est pas tenu de donner des valeurs numériques précises à chaque coefficient. Il suffit simplement de définir des classes de poids qu'on adapte à la granularité désirée.

5.3.2.3 Choix des coefficients

Qu'il s'agisse du degré d'influence (Df) ou de la fonction de coût ($Cost\,()$), le choix des coefficients de pondération aura un grand impact sur le résultat du calcul final.

Pour notre application, nous avons choisi de décomposer l'intervalle [0,1] en trois poids (intervalles) flous : Faible, Moyen, et Fort (figure 5.4). Ce choix est totalement arbitraire, et peut être rapidement modifié si cela s'avère nécessaire.

Pour ce qui concerne le degré d'influence (Df), nous avons fait le choix de donner le poids "Moyen" pour le cône de fanout et le poids "Faible" pour le cône de fanin.

$$Df = Moyen \otimes C_{fin} + Faible \otimes C_{fout}$$

Le choix de donner un poids plus important au cône de fanout est motivé par le souci d'accorder plus d'importance à l'observabilité (le cône de fanout débouche sur les sorties primaires, par conséquent observables) qu'à la contrôlabilité (le cône de fanin prend naissance aux entrées primaires, par conséquent totalement contrôlables).

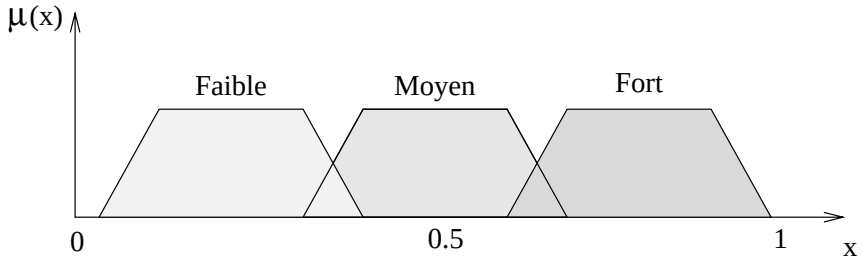

Figure 5.4: Poids flous des paramètres

En ce qui concerne la fonction de coût, il existe deux façons possibles de choisir les poids à affecter aux coefficients des paramètres de cette fonction :

– une manière "statique" qui consiste à faire un choix arbitraire comme dans le cas du degré d'influence. Mais si dans ce dernier cas, le choix était guidé par des critères de contrôlabilité et d'observabilité, ce n'est plus le cas ici.

– une manière "dynamique" dans laquelle les poids sont automatiquement déterminés en fonction de la quantité d'information apportée par chaque paramètre. On calcule pour cela l'écart absolu moyen (ε) des entropies moyennes des points de test. Cet écart est donné par la relation suivante :

$$\varepsilon = \frac{1}{n} * \sum_{i=1}^{n} \mid E_i - \bar{E} \mid$$

où n désigne le nombre de points, E_i l'entropie moyenne du point i, et $\bar{E}$ la valeur moyenne des entropies moyennes du système.

Si cet écart est jugé significatif, ceci implique que l'entropie apporte suffisamment d'information pour la discrimination des points, et elle aura par conséquent un poids

"Fort" tandis que le degré d'influence aura le poids "Moyen". Dans le cas inverse, l'entropie aura le poids "Moyen" et le degré d'influence aura le poids "Fort".

5.4 Exemples d'application

Pour illustrer et en même temps évaluer l'efficacité de la démarche que nous avons développée dans ce chapitre, nous allons l'appliquer sur des exemples.

5.4.1 Application à un système combinatoire

Pour ce cas, nous allons traiter successivement le s27 de la série des benchmarks séquentiels ISCAS'89 [BBK89] dans une configuration Scan Complet, et le c432nr de la série des benchmarks combinatoires ISCAS'85 [BrF85].

La configuration Scan Complet du s27 permet d'obtenir un module combinatoire composé de 10 blocs avec 7 entrées et 4 sorties primaires (figure 5.5).

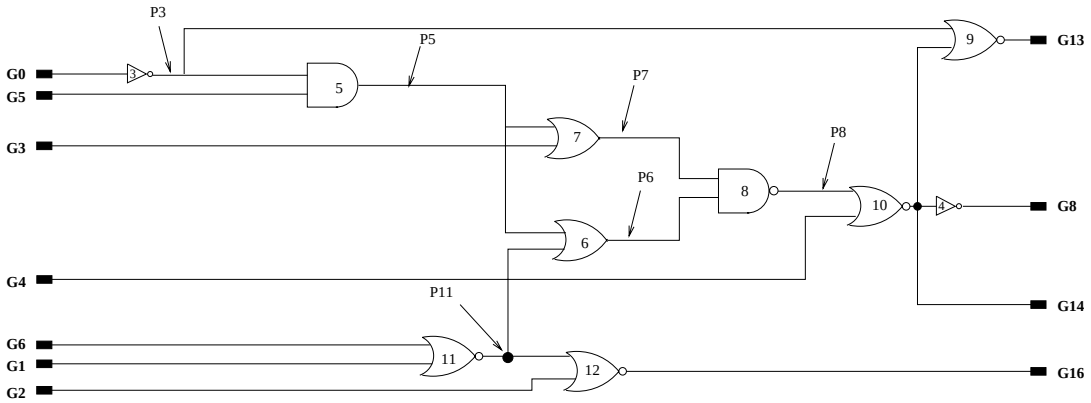

Figure 5.5: s27 scan complet (10 blocs dans ce module)

Nous avons injecté une faute du type collage logique à 1 (Stuck-at-1) sur le nœud G15 (point P11 sur la figure 5.5). Nous avons ensuite généré un ensemble de vecteurs de test permettant une couverture de 100% des fautes de collage. Deux de ces vecteurs permettent de détecter la faute injectée sur le nœud G15. Le premier vecteur exhibe cette faute sur la sortie G16, tandis que le deuxième le fait sur les sorties G16, G14 et G8.

Deux sessions de diagnostic sont donc exécutées. Le tableau 5.1 montre les résultats de l'étape de la génération et de la mise à jour des candidats.

Candidat	not3	not4	and5	or6	or7	nand8	nor9	nor10	nor11	nor12
Session 1	S_C	C	S_C	S_C	S_C	S_C	S_C	V_C	S_C	S_F
Session 2	S_F	C	S_F	S_F	S_F	S_F	S_C	V_F	S_F	S_F
Résultat	Amb	C	Amb	Amb	Amb	V_C	Amb	Amb	V_F	V_F

Tableau 5.1: Génération des candidats pour le s27 Scan

Au cours de la deuxième étape, nous avons calculé l'entropie et le degré d'influence pour chaque nœud du circuit. Les résultats obtenus sont résumés dans le tableau 5.2. Les résultats obtenus sont satisfaisants. Pour trouver la faute, il fallait mesurer le point P11 pour découvrir que le bloc nor11 était fautif, vu que sa sortie aurait été erronée alors que ses entrées étaient correctes. Notre stratégie de recherche du meilleur point à tester a précisément indiqué que le point P11 devait être testé en premier, ce qui conduisait à trouver directement le site de la faute.

Nœud à tester	**Entropie moyenne**	**Degré D'influence**	**Fonction de Coût**	**Classement Obtenu**
P3	0.627	2.247	0.635	sixième
P5	0.627	2.505	0.621	cinquième
P6	0.481	3.046	0.483	deuxième
P7	0.627	2.528	0.620	quatrième
P8	0.530	3.844	0.502	troisième
P11	0.046	1.998	0.187	premier

Tableau 5.2: Recherche du meilleur point à tester pour le s27 Scan

Le deuxième essai porte sur le c432nr (figure 5.6). Ce circuit est composé de 157 portes, 36 entrées et 7 sorties primaires. Nous avons injecté une faute du type collage logique à 1 (Stuck-at-1) sur le nœud G107 (point P107 sur la figure 5.6). Nous avons ensuite généré un ensemble de vecteurs de test permettant une couverture de 100% des fautes de collage. Trois de ces vecteurs permettent de détecter la faute injectée sur le nœud G107.

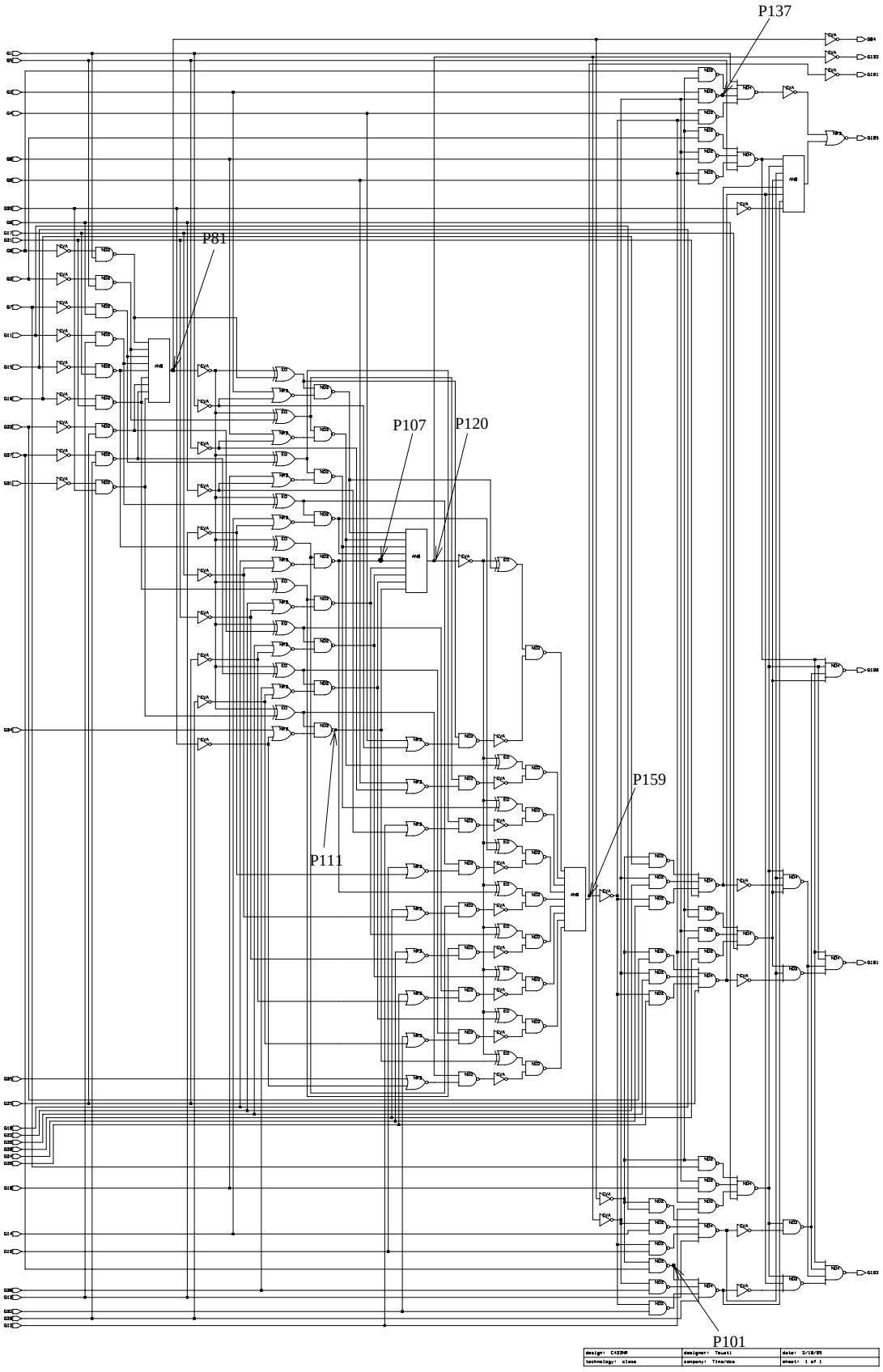

Figure 5.6: Module c432nr (157 blocs)

Les résultats obtenus avec les procédures de diagnostic et de localisation sont résumés dans le tableau 5.3. Nous remarquons qu'avec seulement des informations limitées aux entrées/sorties primaires, et malgré l'importance relative de la taille du système (157 blocs), la stratégie de localisation a parfaitement fonctionné (convergence rapide vers le site de la faute). On notera également l'intérêt de l'utilisation de la distance de *Hamming* qui permet dans ce cas une meilleure discrimination que la fonction de coût.

Nœud à tester	Entropie moyenne	Degré D'influence	Fonction de Coût	Distance de Hamming	Classement Obtenu
P81	0.25	36.97	0.240	0.18	troisième
P101	0.29	12.38	0.300	0.36	sixième
P107	0.25	27.85	0.244	0.25	quatrième
P111	0.25	27.85	0.244	0.25	quatrième
P120	0.25	44.63	0.241	0.12	deuxième
P137	0.27	31.50	0.260	0.22	cinquième
P159	0.25	61.86	0.239	0	premier

Tableau 5.3: Recherche du meilleur point à tester pour le c432nr

5.4.2 Application à un système séquentiel

Pour illustrer la méthode sur un système séquentiel, nous allons traiter le module s27 auquel nous avons apporté une légère modification pour la clarté de l'explication.

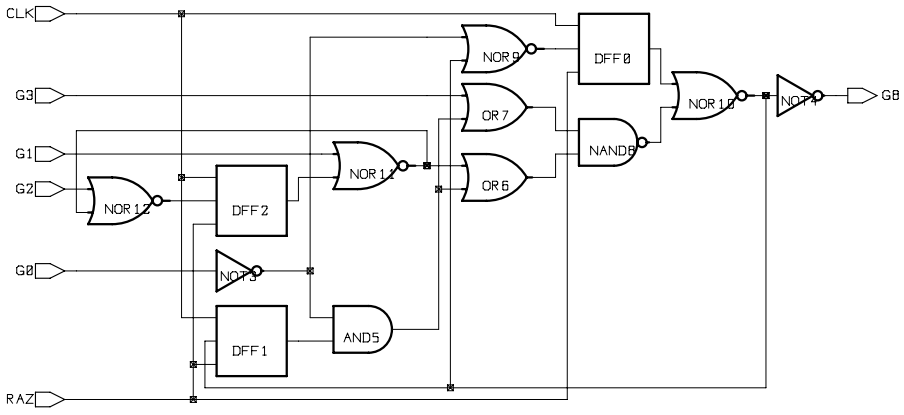

Figure 5.7: Module s27 original

En effet, dans sa version originale (figure 5.7), ce circuit ne comprend qu'une seule sortie primaire, ce qui n'est pas vraiment le cas idéal pour le diagnostic.

La figure 5.8 montre le circuit modifié, sur lequel l'expérience a eu lieu. Nous avons rajouté une sortie supplémentaire et brisé une boucle de rétroaction.

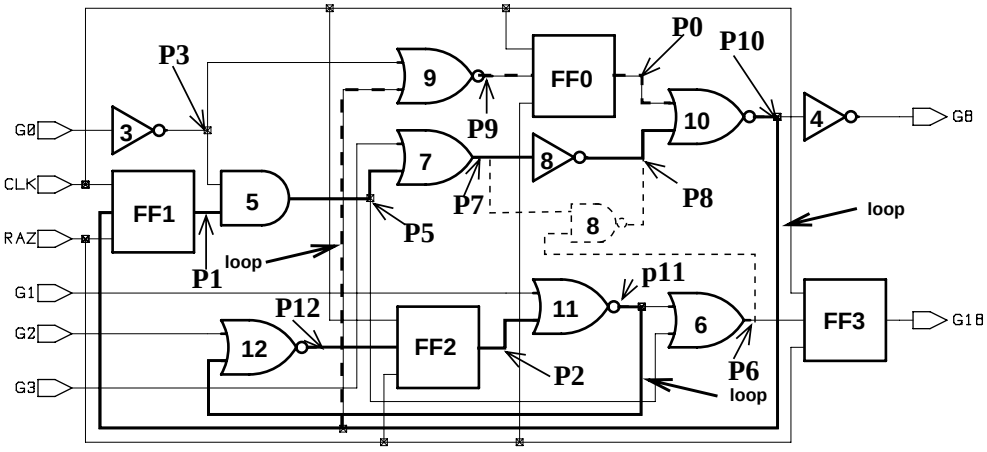

Figure 5.8: Module s27 modifié : 14 blocs dans le module

Nous avons injecté un collage logique à 1 sur le nœud G13 (P9). Dans l'ensemble de vecteurs de test que nous avons généré pour ce module, un seul vecteur permet de détecter cette faute. La détection se fait sur la sortie G8.

Une seule session de diagnostic est exécutée. Les résultats de la phase de génération des candidats sont donnés dans le tableau 5.4. Dans le tableau 5.5, nous présentons les résultats de la procédure de recherche du meilleur nœud à tester.

Candidat	FF0	FF1	FF2	FF3	not3	not4	and5	or6	or7	not8	nor9	nor10	nor11	nor12
Résultat	S_F	S_F	S_C	S_C	S_F	S_F	S_F	S_C	S_F	S_F	S_C	V_F	S_C	S_C

Tableau 5.4: Génération des candidats pour le s27 modifié

Pour les modules séquentiels, nous avons à résoudre le problème des boucles séquentielles. Comme on peut le constater dans le tableau 5.5, tous les nœuds appartenant à une boucle (en gras sur la figure 5.8) ont des caractéristiques identiques (cône de fanin,

cône de fanout et les estimations de fautes). Nous avons donc besoin d'au moins une information supplémentaire pour les différencier. A cet effet, nous utilisons la notion de profondeur de séquentialité comme paramètre de distinction entre ces candidats. Nous considérons que les éléments dont la profondeur de séquentialité correspond au nombre de coups d'horloges effectués au cours du test sont plus à même de contenir la ou les fautes, et que par conséquent ils doivent être sondés en premiers.

Nœud à tester	**Entropie moyenne**	**Degré D'influence**	**Fonction de Coût**	**Prof_Seq**		**Classement**
				G8	**G18**	**Obtenu**
P0	0.269	6.579	0.260	0	0	second
P1	0.269	6.579	0.260	0	1	second
P2	0.573	2.773	0.566	-	1	cinquième
P3	0.269	2.995	0.315	0	0	troisième
P5	0.269	6.579	0.260	0	1	second
P6	0.370	6.499	0.342	-	1	quatrième
P7	0.269	6.579	0.260	0	0	second
P8	0.269	6.579	0.260	0	0	second
P9	0.269	6.579	0.260	1	1	premier
P10	0.269	6.579	0.260	0	0	second
P11	0.573	2.773	0.566	-	1	cinquième
P12	0.573	2.773	0.566	-	2	cinquième

Tableau 5.5: Recherche du meilleur point à tester pour le s27 modifié

5.5 Conclusion

Vu le niveau de complexité atteint par les systèmes VLSI actuels, les méthodes de diagnostic reposant uniquement sur des mesures effectuées sur les entrées/sorties primaires ne peuvent plus permettre une localisation précise du site de la ou des fautes. Souvent des mesures supplémentaires sont nécessaires, que ce soit par l'intermédiaire d'équipements spéciaux (microscope électronique, sondes mécaniques, etc) ou par l'intermédiaire des dispositifs de conception en vue d'une meilleure testabilité (notamment le Scan Path pour les circuits et le Boundary Scan pour les cartes ou les MCMs).

Dans ce chapitre, nous avons présenté une méthode de recherche automatique des meilleurs nœuds à tester. Elle vient compléter l'approche de diagnostic présentée dans le chapitre précédent dans le but d'améliorer le processus de localisation des fautes. Elle repose sur l'analyse de la structure du système sous test ainsi que sur les informations issues de la première étape de diagnostic.

La fonction de coût définie pour estimer l'importance de chaque nœud est facile à mettre en œuvre, et ne demande pas de calculs fastidieux. Les résultats que nous avons montré dans ce chapitre sont plutôt prometteurs [TMM96]. Davantage de résultats seront présentés dans le dernier chapitre, ce qui montre l'efficacité de l'ensemble de l'approche de diagnostic développée.

Chapitre 6

Présentation de l'outil TANDEM

Chapitre 6

Présentation de l'outil TANDEM

6.1 Introduction

Dans ce chapitre, nous allons décrire l'outil TANDEM (pour Test ANd Diagnosis Embedded Modules), qui constitue la mise en œuvre de la méthodologie visant l'unification du test et du diagnostic. Cet outil a été développé au sein du laboratoire TIMA dans le cadre de cette thèse. Il a été réalisé en deux temps :

– durant la première étape, nous nous sommes principalement occupé à définir une méthode permettant d'automatiser la mise au point de vecteurs de test pour les fautes d'interaction (chapitres 2 et 3 de ce manuscrit). Car si les ATPGs disponibles actuellement sur le marché permettent d'obtenir de bons résultats pour les fautes de collage, il en est autrement pour ce qui concerne les court−circuits notamment. Alors souvent, quand on a besoin d'un bon taux de couverture pour ce type de fautes, on est obligé de passer par la simulation de fautes de nouveaux vecteurs générés à l'origine pour des fautes de collages. Néanmoins, cette solution reste seulement applicable pour les systèmes de faible complexité.

– au cours de la deuxième étape, nous avons cherché à mettre au point une nouvelle approche pour le diagnostic des systèmes complexes (chapitres 4 et 5 de ce manuscrit).

Mais étant donné que notre but est de réaliser un système complet dédié à l'unification de test et du diagnostic, il fallait trouver une méthode qui pouvait être facilement intégrée à l'environnement mis au point au cours de la première phase de ce travail. Cette intégration passe obligatoirement par le partage des informations contenues dans la base de données de l'outil.

6.2 Architecture globale du système

La figure 6.1 représente l'architecture globale de l'outil. Comme on peut le constater, le système se compose de quatre blocs indépendants :

- le noyau de l'outil.
- le bloc d'interface avec les outils externes de test
- le bloc des modules de test
- le bloc des modules de diagnostic

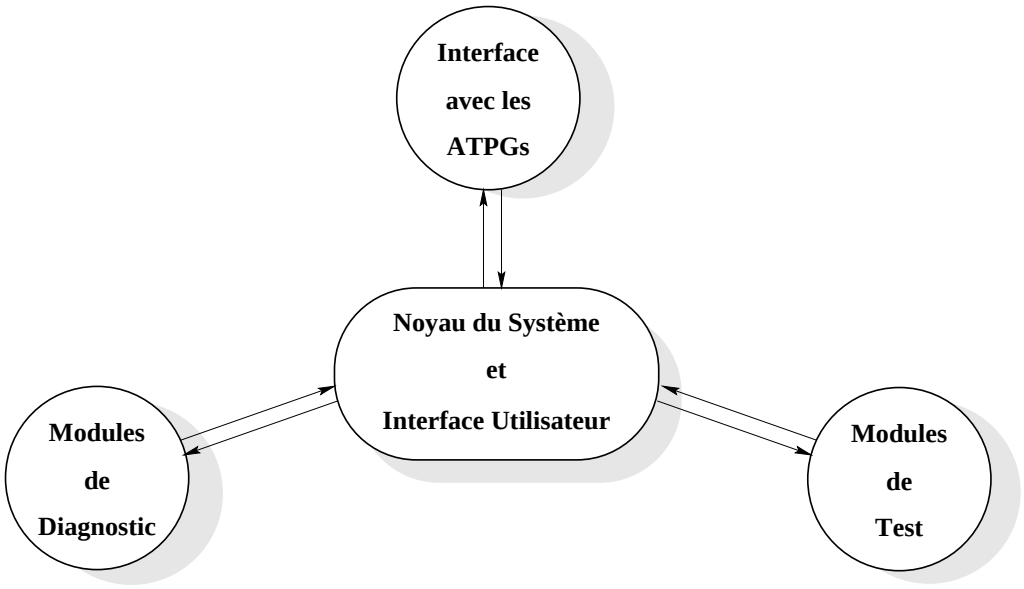

Figure 6.1: Architecture globale du système

La programmation de ces blocs a été réalisée en langage C dans l'environnement graphique X-WINDOWS sous système UNIX.

6.3 Présentation des différent modules du système

Dans ce paragraphe, nous allons donner quelques informations sur la composition de chaque bloc ainsi que sur les fonctions qu'il exécute.

6.3.1 Description du noyau de l'outil

Le noyau représente la partie centrale de l'outil. Il est composé de deux parties :
– La première concerne la base de données qui contient les structures de données internes manipulées par les différents modules de l'outil. Elle comprend la représentation des entités à traiter (circuits, cartes, ...) ainsi que les informations reçues des outils externes de test.
– La partie interface graphique utilisateur (figure 6.2). Elle est responsable de l'acquisition des commandes à exécuter, et l'affichage sur l'écran des résultats obtenus.

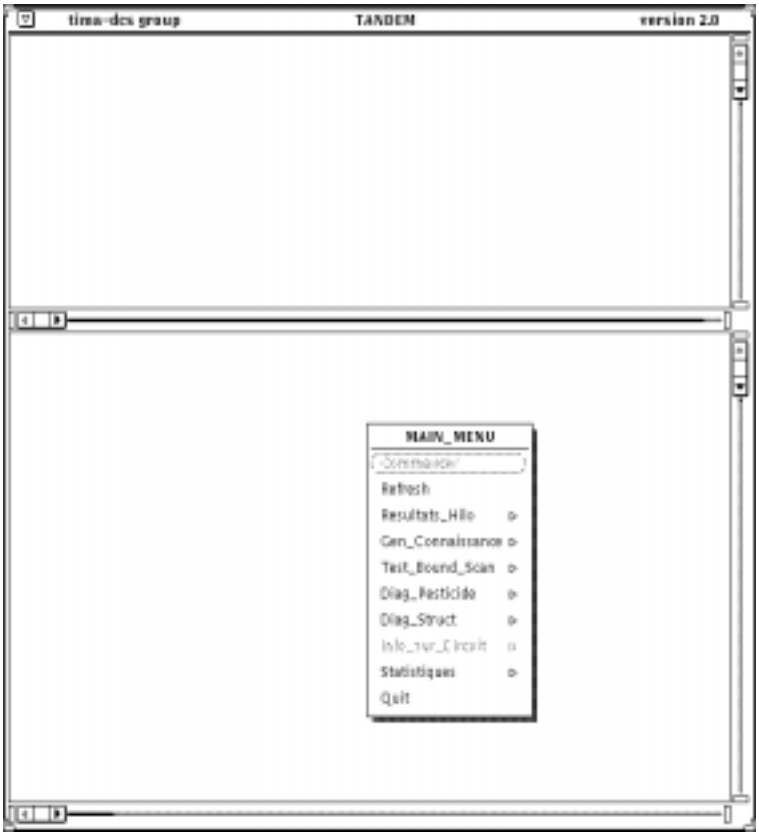

Figure 6.2: Interface utilisateur

6.3.2 Description de l'interface avec les ATPGs

L'utilisation de l'outil débute par le chargement des informations dans sa base de données. Ces informations concernent la description des élément à traiter (netlists), les vecteurs de test disponibles et la couverture de fautes qu'ils produisent.

Afin de pouvoir communiquer avec un large spectre d'outils, notre système supporte quatre types de netlists :

– La description *GHDL* (Genrad Hardware Description Language). C'est une description propre à System Hilo. Avec ce type de représentation, on peut utiliser HITEST [Rob83] [Ben84], l'ATPG de System Hilo pour générer des vecteurs de test, et HIFAULT, le simulateur de fautes de System Hilo pour effectuer des simulations de fautes et générer par la même occasion des dictionnaires de fautes qu'on utilise pour remplir la base de données.

– La description *VERILOG* au niveau porte. Ceci permet l'interfaçage avec les outils de CADENCE Design Systems (VERIFAULT et Test_Synthesizer).

– La description *VHDL* au niveau porte. De plus en plus d'outils utilisent ce standard de description. Dans notre cas, ceci nous permet de communiquer avec l'environnement de SYNOPSYS, et de pouvoir générer ultérieurement des "netlists" au format Boundary Scan Description Language (BSDL) [Par92].

– La description *NDL* (Netlist Description Language). C'est un format de type VERILOG utilisé par les outils TestGen [NiP91] et FaultSim [NCP92], le générateur et le simulateur de fautes de SUNRISE TEST SYSTEMS.

Par conséquent, nous avons programmé des traducteurs permettant de convertir ces différents langages de description au format interne de notre outil. Au stade actuel de son développement, la base de donnée est construite à partir des informations recueillies du simulateur de fautes HIFAULT. Ceci implique de passer nécessairement par un format GHDL. Dans le cas ou l'on désire utiliser un autre simulateur de fautes, il suffit de réaliser le traducteur correspondant.

Le diagramme de la figure 6.3 résume les étapes nécessaires pour la création de la base de données.

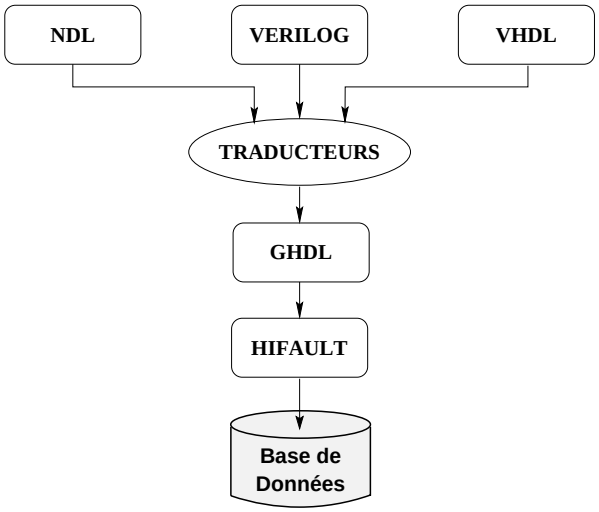

Figure 6.3: Génération de la base de données

6.3.3 Description du bloc de test

Le bloc de test comprend deux modules indépendants : un module pour la génération des vecteurs de test pour les fautes d'interaction entre conglomérats, et un module pour la génération de vecteurs de test par ordonnancement pour les conglomérats composés de modules montés en cascade.

6.3.3.1 module de test des fautes d'interaction

Ce module regroupe les différentes procédures utilisées lors de l'étape de génération des séquences de test pour les fautes d'interaction. Cette étape débute par une phase d'acquisition des informations concernant la carte, et le remplissage de la base de données. La deuxième phase concerne la construction de la matrice de couverture. La troisième phase est responsable de la génération des *D-set* et $\overline{D}$*-set* initiaux. La génération des matrices des candidats est réalisée dans la quatrième phase. Dans la cinquième phase, on génère les *D-set* et $\overline{D}$*-set* finaux, et en dernier on met au point la séquence finale de test.

Figure 6.4: Module de test BS

Toutes ces étapes sont réalisées directement à partir de l'interface utilisateur (figure 6.4)

6.3.3.2 module de test des circuits en cascade

En invoquant ce module, l'outil récupère les informations concernant les modules à assembler. Il construit alors la matrice de couverture de chacun d'entre eux. Ensuite, il demande à l'utilisateur la configuration d'interconnexion souhaitée. En sortie, il donne les vecteurs de test générés ainsi que des informations sur les problèmes de contrôlabilité et d'observabilité rencontrés (vecteurs amont non propagés complètement ou bien vecteurs aval n'ayant pas pu être reconstitués) [Mor95].

6.3.4 Description du Module de diagnostic

Le module de diagnostic est en réalité composé de deux blocs : un bloc d'interface avec PESTICIDE (paragraphe 4.2.3) et le bloc de diagnostic propre à TANDEM

(paragraphe 4.3)

6.3.4.1 Bloc d'interface avec PESTICIDE

Ce bloc est responsable de la génération automatique des diverses bases de connaissances nécessaires pour démarrer le diagnostic avec PESTICIDE. A partir des informations stockées dans la base de données de TANDEM, et les résultats des sessions de test effectuées, TANDEM génère tous les fichiers requis par PESTICIDE.

6.3.4.2 Bloc de diagnostic de TANDEM

Ce bloc regroupe les procédures de diagnostic et de localisation élaborées dans le cadre de cette thèse. Dans un premier temps, l'outil extrait les informations accumulées pendant les différentes sessions de test, et propose après traitement les candidats potentiels (figure 6.5).

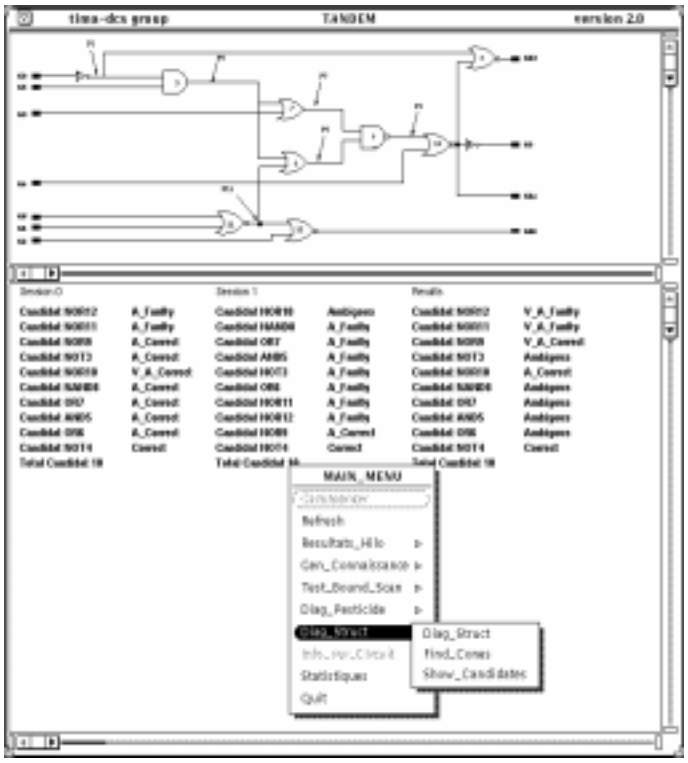

Figure 6.5: Diagnostic de fautes avec TANDEM

La deuxième phase s'inscrit dans un processus de localisation. Après traitement, une classification des meilleurs nœuds à sonder est établie (figure 6.6).

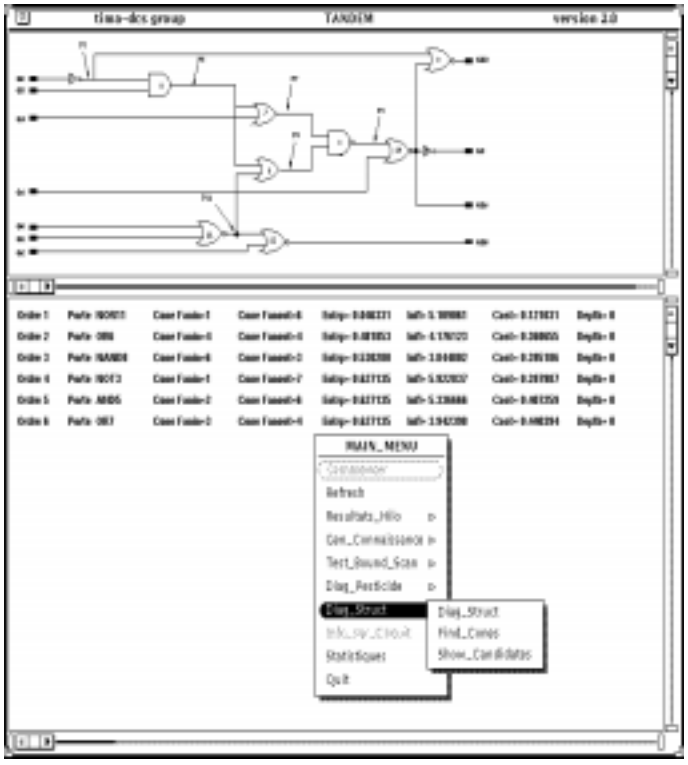

Figure 6.6: Localisation des fautes avec TANDEM

6.4 Conclusion

Dans ce chapitre, nous avons présenté l'outil permettant d'unifier le test et le diagnostic des cartes mixtes. Nous avons aussi décrit son environnement global et les possibilités de le coupler à d'autres outils de test existant. Dans le chapitre suivant, nous allons donner les principaux résultats obtenus à l'aide de cet outil.

Chapitre 7

Expérimentations et résultats

Chapitre 7

Expérimentations et résultats

7.1 Introduction

Pour évaluer l'efficacité de notre outil, nous avons procédé à des essais sur des circuits. Ces essais concernent les divers modules de l'outil. Nous avons choisi à cet effet de réaliser les tests sur les circuits utilisés usuellement pour la validation des méthodes de test. Il s'agit des benchmarks de la session ISCAS'85 pour les systèmes combinatoires, et ceux de la session ISCAS'89 pour les modules séquentiels.

Circuit	# portes	# entrées	# sorties	Couverture (%)
C17	6	5	2	100
C432NR[1]	157	36	7	100
C499	202	41	32	100
C880	383	60	26	100
C1355NR[1]	546	41	32	100
C1908NR[1]	878	33	25	100
C2670NR[1]	961	157	63	100
C3540NR[1]	1620	50	22	100
C5315	2307	178	123	100
C6288	2406	32	32	100
C7552NR[1]	3397	206	107	100

[1] NR : Non Redondant

Tableau 7.1: Caractéristiques des circuits combinatoires utilisés

En fait, nous assimilons ces circuits à des cartes électroniques (chaque porte correspondrait en fait à un bloc de la carte).

Circuit	#EP	#SP	#DFF	#Portes	SETA+Hilo %	SUNRISE %
s27	4	1	3	10	100	100
s208	11	2	8	96	100	100
s298	3	6	14	119	95.59	95.86
s344	9	11	15	160	98	98
s349	9	11	15	150	97.84	97.78
s382	3	6	21	158	99.73	97.08
s386	7	7	6	159	97.97	97.97
s444	3	6	21	181	98.78	98
s510	19	7	6	211	100	99.76
s526n	3	6	21	193	94.27	xxx
s641	35	24	19	379	90.30	90.45
s820	18	19	5	289	97.92	97.27
s832	18	19	5	287	97.74	97.08
s953	16	23	29	395	99.85	99.85
s1196	14	14	18	529	100	100
s1238	14	14	18	508	99.07	99.04
s1488	8	19	6	653	99.85	98.03
s1494	8	19	6	647	99.70	99.61

Tableau 7.2: Caractéristiques des circuits séquentiels utilisés

Les principales caractéristiques de ces circuits ainsi que le taux de couverture obtenu pour chaque circuit sont résumés dans les tableaux 7.1 et 7.2. Pour les circuits combinatoires, les vecteurs ont été générés avec HITEST [Ved94]. Pour les circuits séquentiels, nous avons utilisé deux outils : l'ATPG séquentiel SETA [CGP91] développé à l'Institut Polytechnique de Turin (*Politecnico di Torino*), et l'ATPG TestGen [NiP91] de Sunrise. Ces deux outils utilisent deux approches différentes pour la génération de vecteurs de test [Tou92]. Le premier est basé sur la représentation des circuits séquentiels par des automates (MEF) opérant en mode fondamental à partir d'un état initial connu (d'où la nécessité d'un signal de "reset" sur les bascules). Quant au deuxième, il utilise plutôt une approche topologique. La génération des vecteurs est basée sur la représentation du circuit séquentiel en un circuit pseudo-combinatoire, itératif dans le temps ("Iterative Array Model"). Si les résultats en taux de couverture de fautes et en temps CPU obtenus par TestGen sont très bons, en revanche cet

ATPG produit des vecteurs de test sous forme de séquence unique, ordonnée, dont la décomposition en vecteurs indépendants (nécessaire pour l'application de notre méthodologie) s'avère fastidieuse, et engendre une augmentation importante (à cause des répétitions) du nombre de vecteurs de test. Par contre, les vecteurs générés par SETA se présentent sous forme de séquences indépendantes, commençant chacune par un signal de remise–à–zéro, ce qui convient parfaitement à notre application. Par conséquent, en ce qui concerne la phase de génération des séquences de test BS, on utilisera en priorité les vecteurs générés par SETA. Pour la phase de diagnostic, les deux types de vecteurs (SETA, Sunrise) sont utilisés.

7.2 Résultats du module de test par ordonnancement

Pour tester ce module (chapitre 2, paragraphe 2.3.1), nous avons réalisé des conglomérats à l'aide de circuits mis en cascade. Le tableau 7.3 montre la configuration utilisée et les caractéristiques des systèmes créés. Pour chaque carte, on présente le nombre de connexions communes (#Conn.), le nombre d'entrées primaires (#EP) et de sorties primaires (#SP).

Système en Cascade				**Module Amont**			**Module Aval**		
Nom	#Conn.	#EP	#SP	Nom	#Portes	#Vect.	Nom	#Portes	#Vect.
C0	2	8	2	c17	6	6	c17	6	6
C1	3	190	64	c432nr	157	39	c2670nr	961	43
C2	3	190	64	c432nr	157	39	c2670nr	961	43
C3	3	190	64	c432nr	157	39	c2670nr	961	43
C4	3	62	25	c880	383	28	c17	6	6
C5	2	63	26	c17	6	6	c880	383	28
C6	2	94	31	c880	383	28	c432nr	157	39
C7	3	93	30	c880	383	28	c432nr	157	39
C8	4	189	63	c432nr	157	39	c2670nr	961	43
C9	3	93	30	c432nr	157	39	c880	383	28

Tableau 7.3: Configuration des systèmes traités: cartes C0 à C9.

Les cartes C1, C2 et C3 sont composées des mêmes circuits, mais configurées de manières différentes (nœuds d'interconnexion différents). Le tableau 7.4 présente les taux de couverture obtenus avec TANDEM ainsi que ceux obtenus avec HITEST dans

les mêmes conditions (ces circuits étant mis à plat dans ce cas). Il présente aussi une comparaison des résultats entre le nombre de vecteurs de test générés par chacun des deux outils et le temps de CPU utilisé à cet effet. Ces résultats ont été obtenus sur une station de travail SUN-SPARC10.

7.2.1 Analyse des résultats obtenus

Dans l'ensemble, les résultats obtenus sont satisfaisants. TANDEM génère souvent un nombre légèrement supérieur de vecteurs pour un taux de couverture égal ou légèrement inférieur par rapport à l'ATPG. Dans tous les cas, le temps CPU est minime avec TANDEM, très largement inférieur à celui nécessité par HILO.

Système	Taux de Couverture		# Vecteurs		Temps CPU (s)	
	HILO	TANDEM	HILO	TANDEM	HILO	TANDEM
C0	100%	100%	9	9	0.30	0.00
C1	99.9%	99%	56	64	21.36	0.42
C2	100%	100%	54	57	20.7	0.04
C3	100%	100%	46	50	18.19	0.02
C4	100%	78.2%	24	6	3.85	0.08
C5	100%	99.8%	26	28	3.94	0.00
C6	99.7%	93.2%	43	53	11.83	0.04
C7	99.5%	91.2%	55	66	13.28	0.18
C8	99.9%	96.6%	52	43	19.25	0.08
C9	99.8%	99.8%	42	44	10.39	0.04

Tableau 7.4: Comparaison des résultats avec l'ATPG.

Le plus mauvais résultat a été obtenu pour l'exemple du système C4. Cependant, ce résultat s'explique facilement : le circuit aval (C17) possède seulement 6 vecteurs de test. Cet ensemble réduit de vecteurs ne permet pas de satisfaire les contraintes d'observabilité sur les nœuds d'interconnexion. Pour les systèmes C6 et C7, nous avons obtenu une couverture relativement inférieure à celle de l'ATPG. Dans ces deux cas, les vecteurs disponibles pour les circuits aval ne permettent pas de vérifier les conditions d'observabilité.

Quant au système C8, l'interconnexion des deux circuits se fait par l'intermédiaire de

4 nœuds. Ceci entraîne un durcissement au niveau des contraintes d'observabilité (possibilité d'avoir 2^4 vecteurs différents à propager), ce qui rend plus difficile l'application de la technique d'ordonnancement et se traduit par un taux de couverture inférieur à celui de l'ATPG.

En conclusion, la qualité des résultats obtenus dépend fortement de celle des vecteurs disponibles pour chaque circuit, et en tout état de cause, elle ne peut que décroître avec l'augmentation du nombre d'interconnexions. Néanmoins, dans notre contexte de travail, c'est-à-dire la génération d'une séquence globale de test pour les fautes d'interaction dans des cartes partiellement Boundary Scan, où la description interne des circuits n'est pas toujours disponible, notre méthode reste la seule applicable.

7.3 Résultats du module de test BS

Nous présentons dans ce paragraphe les résultats obtenus par le module mettant en œuvre les algorithmes de calcul des *D-set* et $\overline{D}$*-set* présentés au chapitre 2.

Circuit	Init_PTV	**D_set**	$\overline{D}$**_set**	**Nb_X**	**Final_PTV**	% additionnel
c17	6	4	4	2	10	66.6
c432nr	34	19	26	3	48	41.1
c499	52	32	22	3	57	9.6
c880	28	20	18	5	43	53.5
c1908nr	73	36	61	12	109	49.3
c1355nr	88	32	81	7	120	36.3
c2670nr	42	40	38	2	80	86
c6288	42	4 0	0	–	–	–
c7552nr	66	55	44	4	103	56

Tableau 7.5: Résultats sur les circuits combinatoires

7.3.1 Cas de "clusters" combinatoires

Nous avons utilisé les ISCAS'85 que nous avons assimilés à des conglomérats. Le tableau 7.5 donne les résultats obtenus avec TANDEM. L'ordonnancement des vecteurs de test pour les conglomérats et le calcul des vecteurs de test pour les connexions BS

ont permis d'obtenir des séquences finales de test permettant la détection des fautes du type 1, 2, 3 et 4 (paragraphe 2.2). En contrepartie, un surcoût dû à la répétition des vecteurs de test est constaté. Il varie de 6 à 86% (la moyenne est de 49 %) en nombre de vecteurs par rapport aux vecteurs initialement générés par l'ATPG.

Le test des fautes de type 5 peut démarrer avec l'utilisation des X contenus dans les BS STV (colonne 5 du tableau 7.5).

7.3.2 Résultats sur les "clusters" séquentiels

Nous avons utilisés les ISCAS'89 que nous avons assimilés à des conglomérats. Le tableau 7.6 donne les résultats obtenus avec TANDEM.

Circuit	Init_PTV	**D_set**	$\overline{D}$**_set**	**Nb_X**	**Final_PTV**	% additionnel
s27	7	4	3	3	10	42.8
s208	36	19	13	14	46	27.7
s298	21	11	10	8	29	38.1
s344	21	12	4	8	24	14.2
s382	24	10	12	8	30	25
s386	38	20	11	13	44	15.7
s444	19	10	13	3	26	36.8
s510	42	20	22	13	53	26.2
s526n	32	15	15	8	38	18.7
s641	74	25	28	34	87	17.5
s444	42	20	22	13	53	26.2
s820	102	42	40	13	126	23.53
s832	105	40	40	44	127	20.95
s953	83	34	32	38	104	25.3
s1196	122	74	80	18	172	40.9
s1488	100	52	25	37	114	14
s1494	92	49	25	31	105	14.1

Tableau 7.6: Résultats sur les circuits séquentiels

7.4 Résultats du module de Diagnostic

Nous avons testé le module de diagnostic de TANDEM sur les ISCAS'85 pour les systèmes combinatoires et les ISCAS'89 pour les séquentiels. Pour chaque circuit,

nous avons injecté une faute, ensuite nous avons démarré le processus de diagnostic à partir des vecteurs de test qui mettent en évidence cette faute. Pour évaluer les performances de l'outil, nous avons défini le paramètre *efficacité* que nous calculons de la manière suivante :

$$Eff = 100 \times \frac{N-(k-1)}{N}$$

Avec N le nombre total de composants du système et k le rang du site de la faute (sortie du bloc fautif) dans la liste des points de test proposés.

Circuit	**# total de blocs**	**Bloc Fautif**	**Rang du Bloc Fautif**	**Efficacité en (%)**
c17	6	NAND23	1	100
c432nr	171	NAND71	34	80.7
c880	383	NOR73	11	97.3
c2670nr	961	AND243	21	97.9

Tableau 7.7: Résultats pour quelques ISCAS'85

Le tableau 7.7 présente les résultats obtenus sur les modules combinatoires. Ces résultats sont dans l'ensemble assez satisfaisants.

Circuit	**# total de blocs**	**# de Bascules**	**Bloc Fautif**	**Prof. de Séquentialité**	**Long. du Vecteur**	**Rang du Bloc Fautif**	**Efficacité en (%)**
s298	133	14	NOT22	1	2	36	73.6
s344	175	15	DFF11	2	3	29	84
s510	217	6	OR99	2	27	100	54.3
s1196	547	18	AND159	1	2	158	73.6

Tableau 7.8: Résultats pour quelques ISCAS'89

Le tableau 7.8 présente les résultats obtenus sur les modules séquentiels. Bien que moins bons que ceux obtenus pour les modules combinatoires, ces résultats sont dans

l'ensemble assez encourageants. On notera cependant le cas du s510 pour lequel l'efficacité se situe légèrement au dessus de 50%. Ce résultat s'explique en partie par la qualité des vecteurs de test disponibles. Les séquences sont longues et par conséquent la profondeur de séquentialité ne joue pas son rôle de discrimination des candidats.

De toutes manières, en ce qui concerne les circuits séquentiels, une amélioration demeure possible. Elle devra surtout porter sur le traitement des boucles séquentielles.

Chapitre 8

Conclusion

Chapitre 8

Conclusion

Dans ce travail, nous avons proposé une méthodologie qui représente une solution pratique pour un problème d'actualité : le test et le diagnostic des MCMs et des cartes électroniques munis du Boundary Scan Partiel.

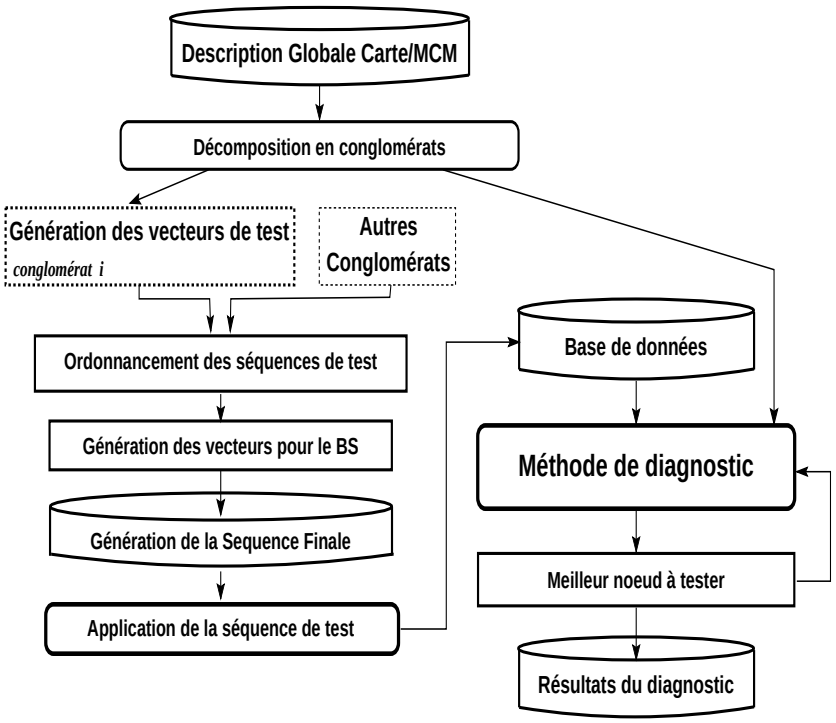

Figure 8.1: Synoptique générale du système

La solution proposéet dont les différentes étapes sont résume dans la figure 8.1 se distingue par plusieurs aspects, parmi lesquels :

- la prise en compte de la réalité industrielle actuelle qui ne permet pas encore de disposer de systèmes totalement Boundary Scan.
- la capacité de tester simultanément les conglomérats et les interconnexions Boundary Scan.
- L'utilisation des mêmes séquences de test pour la détection et le diagnostic de fautes.
- la possibilité d'utiliser deux approches différentes pour effectuer le diagnostic ; l'utilisation de l'approche à base de connaissances avec l'outil PESTICIDE, ou bien l'approche qualitative à base de logique floue implantée sur l'outil TANDEM.
- son étroite corrélation avec les outils de test commerciaux.
- sa facilité de mise en œuvre grâce à l'outil TANDEM et sa simplicité d'utilsation grâce à son interface graphique conviviale.

Les résultats obtenus sont très satisfaisants, et des perspectives d'évolution sont apparues surtout au niveau de l'amélioration du diagnostic. En ce qui concerne les tâches futures, nous citerons pour ce qui est de la partie test le découpage automatique du système en conglomérats.

Pour ce qui est du diagnostic, une amélioration des critères de discrimination des candidats dans les systèmes séquentiels est à trouver. L'implantation de la fonction de coût dynamique avec réglage automatique de la granularité ("self-tuning") reste à réaliser.

Ce travail a débouché sur la réalisation d'un outil ouvert à plusieurs langages de description au niveau netlist. Par conséquent, il peut être utilisé en complément à

des outils commerciaux tel que le PM 3790 BSD de Fluke & Philips, pour lequel on précise que pour les conglomérats, le diagnostic se limite à l'indication des entrées contrôlant le conglomérat et les sorties fautives.

Outre ces développements à court terme, les perspectives à long terme s'inscrivent dans le cadre général de la recherche dans le domaine du test et du diagnostic des systèmes hétérogènes en cours au laboratoire TIMA. La figure 8.2 montre les trois principales tâches couvertes, et la façon dont les résultats de ce travail y participent. Pour ce qui concerne les systèmes mixtes analogiques-digitaux, le but envisagé est le développement d'un environnement unifié pour le test et le diagnostic de ces systèmes. Les travaux en cours concernant les systèmes analogiques rejoignent déjà les résultats de notre travail au niveau de la recherche du meilleur nœud à tester : la même stratégie s'applique avec succès aux systèmes analogiques [MMT96] pour faciliter et améliorer le diagnostic.

Pour ce qui concerne les systèmes mixtes matériel−logiciel, les travaux dans le cadre de leur conception conjointe en vue du test s'appuient sur l'utilisation des standards de la famille IEEE 1149.*, et notamment le standard Boundary Scan pour assurer la testabilité des interfaces de communication entre les parties matérielles et logicielles comme entre les modules d'un système matériel. Les méthodologies que nous avons développées pourront être également utilisées pour la validation de prototypes de tels systèmes.

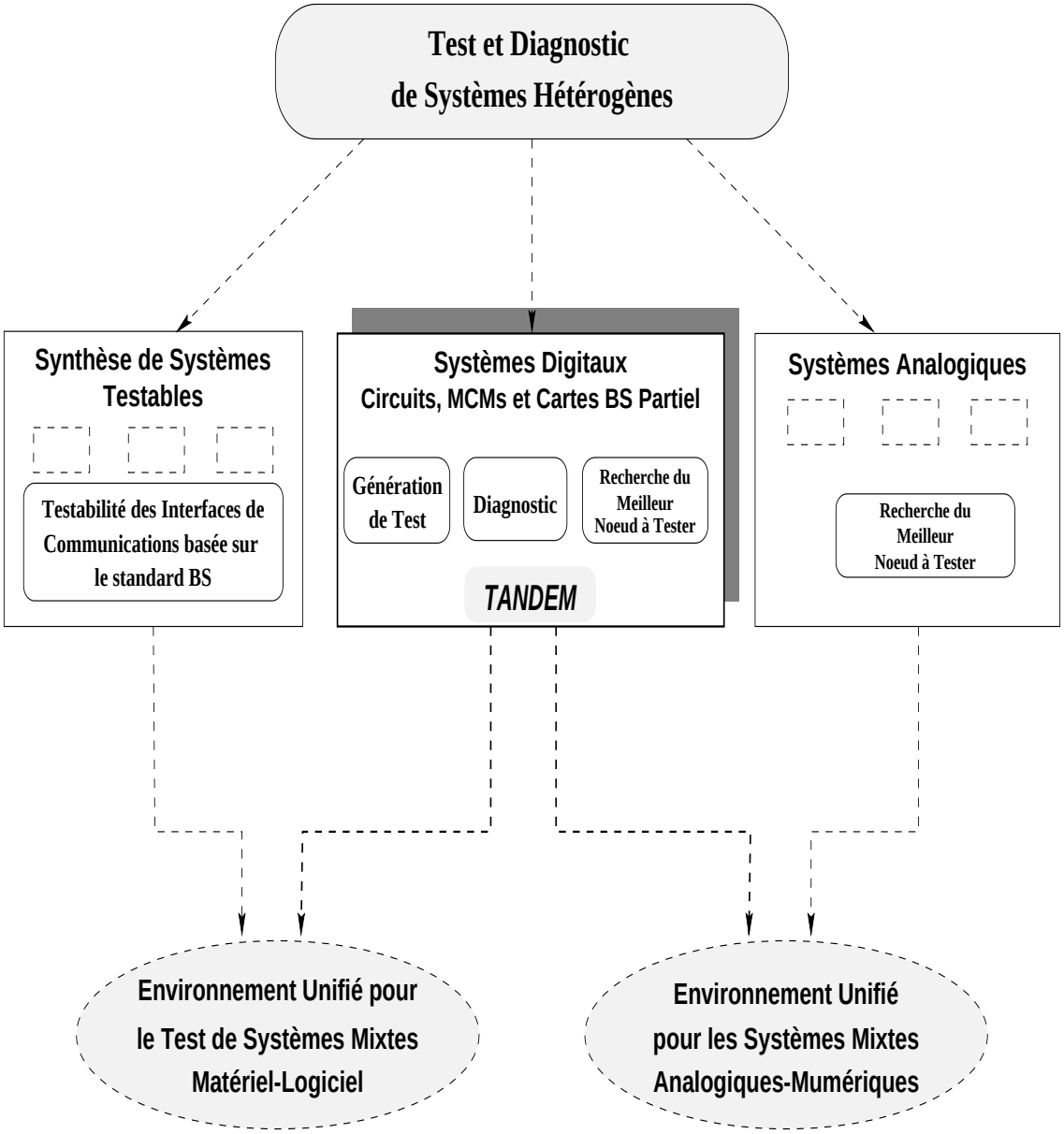

Figure 8.2: Réalisations et perspectives

Bibliographie

Bibliographie

[BBK89] F. Brglez,D. Bryan and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits", *Proc. IEEE ISCAS*, 1989, pp. 1929-1934.

[BeO91] R. G. Bennetts and A. Osseyran, "IEEE Standard 1149.1-1990 on boundary scan : History, litterature survey, and current status", *Journal of Electronic Testing : Theory and Applications (Special issue on Boundary Scan)*, 2 (1):11-25, Mars 1991.

[Ben84] M. T. Bending, "Hitest : A Knowledge-Based Test Generation System", *IEEE Design & Test of Computers*, Vol. 1, N°. 2, 1984, pp. 83-92.

[Ben94] B. Bennets, "Progress in Design for Test: A Personal View", *IEEE Design and Test of Computers*, Spring 1994, pp. 53-59.

[BoD86] Piero P. Bonnisson and Keith S. Decker; "Selecting uncertainty calculi and granularity: An example in trading-off and complexity", in Uncertainty in AI, L.N Kanal and J.F Lemmer (ed), Elservier Science Publishers B.V. (North Holland), pp. 217-247, 1986.

[BoK94] V. Boppana and W. Kent Fuchs, "Fault Dictionary Compaction by Output Sequence Removal". *Proc. Intl. Conf. on Computer-Aided Design*, Nov 1994, pp. 576-579.

[BrF85] F. Brglez and H. Fujiwara, "A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran", *Proc. Intl. Test Conference*, 1985, pp. 785-794.

[CGP91] P. Camurati, M. Gili, P. Prinetto and M. Sonza Reorda, "An application of Automata Theory to sequential ATPG", *IEEE European Test Conference*, 1991.

[ChM85] B. Chandraskan et R. Milne, "Special section on reasoning about structure, behavior and function - introduction", SIGART Newsletter 1, 93, jul. 1985.

[Cou94] B. Courtois, "CAD and Testing of ICs and Systems : where are we going?", Research Report, Laboratoire TIMA, 46 Avenue Félix Viallet- 38031 Grenoble Cedex France, Septembre 1994.

[DaK88] J. M. David, J. P. Krivine et al., "Structuration du raisonnement et explications", *Revue D'Intelligence Artificielle*, Vol. 4, 1988, pp. 77-88.

[DaS83] R. Davis and H. Shrobe, "Representing structure and behavior of digital hardware". *IEEE Computer journal*, October 1983, pp. 75-82.

[Dav84] R. Davis "Diagnostic reasoning based on structure and behavior". *Artificial intelligence*, Vol 24, pp. 347-410, 1984.

[Gul92] M. Gullet, "Monolithic or MultiChip", *ASIC & EDA*, May 1992, pp. 24-26.

[DuP87] D. Dubois and H. Prade; "Theorie des Possibilités", MASSON, Paris, 1987.

[For84] K. D.Forbus, "Qualitative Process Theory", *Artificial Intelligence* Vol 24, 1984.

[Gil95] L. Gilg, "Known Good Die", *IEEE Multi-Chip Module Conference*, Tutorial #4, 1995.

[HAR89] A. Hassan, V. K. Agrawal, J. Rajski and B. N. Dostie, "Testing of Glue Logic Interconnects using Boundary Scan Architecture", *International Test Conference*, 1989, pp. 700-711.

[HBC91] J-P. Haton, N. Bouzid, F. Charpillet, M.C. Haton, B. Lâasri, H.Lâasri, P. Marquis, T. Mondot, & A. Napoli "Le Raisonnement en Intelligence Artificielle", *Inter Edition*, 1991.

[HRA88] A. Hassan, J. Rajski et V. K. Agrawal, "Testing and Diagnosis of Interconnects using Boundary Scan Architecture", *International Test Conference*, 1988, pp. 254-265.

[HaW92] J. K Hagge and R. J. Wagner, "High-Yield Assembly of Multi-Chip Modules through Known-Good IDs and Effective Test Strategies", *Proceedings of the IEEE*, Vol. 80, N. 12, December 1992, pp. 1965-1994.

[Han89] P. Hansen, "Testing Conventional Logic and Memory Clusters using Boundary Scan Devices as Virtual ATE Channels", *International Test Conference*, 1989, pp. 166-173.

[IEEE90] IEEE Standard 1149.1-1990, "IEEE Standard Acess Port and Boundary Scan Architecture", IEEE Standards Board, 345 East 47th Street, New York, NY 10017-2394, USA, 1990.

[JaY89] N. Jawarla et C. Yau, "A New Framework for Analyzing Test Generation and Diagnosis Algorithms for Wiring Interconnects", *International Test Conference*, 1989, pp. 63-70.

[JTB91] R. W. Johnson, R. K. F. Teng and J.W. Balde. Ed. "MultiChip Modules". *IEEE Press Selected Reprint Series*. 1991.

[Kau74] W. Kautz, "Testing for faults in Wiring Networks", *IEEE Transactions on Computers*, Vol. C-23, N° 4, April 1974, pp. 358-363.

[KlW87] J. De. Dekleer and B.C Williams. "Diagnosing multiple faults", *Artificial intelligence*, Vol 32, pp. 97-130, 1987.

[Kos91] B. Kosko "Neural Networks and Fuzzy Systems", Prentice Hall, 1991.

[Kui93] B. Kuipers, "Reasoning with qualitative models", *Artificial Intelligence*, vol 59, 1993.

[Lan91] D. Lang, "Cost Effectiveness of nCHIP's MCM Technology", *MultiChip Module Workshop*, Santa Cruz, March 1991, pp. 16-23.

[LMT94] M. Lubaszewski, M. Marzouki and M. H. Touati, "A Pragmatic Test and Diagnosis Methodology For Partially Testable MCMs", *MultiChip Module Conference*, 94, pp. 108-113.

[MCMC92] *Proceedings of the MultiChip Module Conference*, Santa Cruz, March 1992.

[MCMC93] *Proceedings of the MultiChip Module Conference*, Santa Cruz, March 1993.

[MCl84] E. J. MacCluskey,"Verification testing–a pseudo-exhaustive test technique", *IEEE Trans. on Computers*, vol c-33, pp. 541-546, June 1984.

[MLC91] M. Marzouki, J. Laurent and B. Courtois, " Coupling electron-beam probing with knowledge-based fault localization", *22nd IEEE Intl. Test Conference*, October 1991, pp. 238-247, Nashville (TN), USA.

[MLT93] M. Marzouki, M. Lubaszewski and M. H. Touati, "Unifying Test and Diagnosis of interconnects and logic Clusters in Partial Boundary Scan Boards", *Proc. Intl. Conf. on Computer-Aided Design*, Nov 1993, pp. 654-657.

[MMT96] F. Mohamed, M. Marzouki and M. H. Touati, "FLAMES : A Fuzzy Logic ATMS and Model-based Expert System for analog Diagnosis", *European Design and Test Conference*, Paris, France, Mars 1996.

[MaV91] M. Marzouki and F. L. Vargas, "Knowledge-based debugging of ASICs : real case study and performance analysis", *Proc. Intl. Conf. on Computer-Aided Design*, Nov 1991.

[Mar91] M. Marzouki, "Model-Based Reasoning for Electron-Beam Debugging of VLSI Circuits", *Journal of Electronic Testing: Theory and Applications*, 2, 385-394, 1991.

[MoM95] F. Mohamed, M. Marzouki, "Test and Diagnosis of Analog Devices : When Fuzziness can lead to Accuracy", *Journal of Electronic Testing : Theory and Applications*, Parution prochaine.

[Mor95] F. Morgado, "Génération de vecteurs de test par ordonnancement", *Rapport de DEA*, TIMA-INPG, Juin 1995, Grenoble.

[NCP92] T. Niermann W. T. Cheng and J. H. Patel, "PROOFS : A Fast, Memory-Efficient Sequential Circuit Fault Simulator", *IEEE Transactions on Computer-Aided Design*, Vol. 11, N°. 2, 1992, pp. 198-206.

[NeR75] C. V. Negoit and D. A. Ralsecu; " Applications of fuzzy Sets to Systems Analysis", 1975 Birkhuser Verlag, Basel und Stuttgart.

[NiP91] T. Niermann and J. H. Patel, "HITEC : A Test Generation Package for Sequential Circuits", *European Design Automation Conference*, 1991.

[Par92] K. P. Parker, " *The Boundary Scan Handbook*, Kluwer Academic Publishers, 1992.

[PeG87] J. Pearl and H. Geffner, "An improved constraint propagation algorithm for diagnosis", *International Joint Conference on AI*, Milano (Italy), 1987, pp. 1105-1111.

[Pea88] J. Pearl, " A constraint-propagation approach to probabilistic reasoning", *machine Intelligence and Pattern Recognition n° 4 : Uncertainly in Artificial Intelligence*. Editors L. N. LAKANAL & J. F. LEMMER, North Holland, 1988.

[Phi93] Philips Electronics, "Product Data Sheet, PM 3790 BSD Boundary Scan Diagnostics", 1993.

[PoR92] I. Pomeranz and S. M. Reddy, "On the Generation of small Dictionaries for Fault Location", *Proc. Intl. Conf. on Computer-Aided Design*, Nov 1992, pp. 272-279.

[RIC85] Richardson and al, "Artificial Intelligence in Maintenance : Synthesis of Technical Issues", Report N° AFHRL-TR-85-7, Air Force Systems Command. Air Force Human resources Laboratory, Brooks Air Force Base. Texas 78235, 1985.

[RiB85] J. Richmann and K. R. Bowden, "The Modern Fault Dictionary", International Test Conference, 1985, pp. 696-702.

[RoD90] G. D Robinson and J. G Deshayes, "Interconnect Testing of Boards with Partial Boundary Scan",International Test Conference, 1990, pp. 572-581.

[Rob83] G. D. Robinson, "HITEST-Intelligent Test Generation", *International Test Conference*' 1983, pp. 311-323.

[ShL91] Q.shen & R.Leitch; "Diagnosing Continuos Dynamic Systems Using Qualitative Simulation", *Proc. of the Fifth National Conf. on Control*, 1991.

[SzY93] Z. Shen, R.R.Yager, S. Ovchinnkov, R.M Tong and H.T. Nguyen; "Fuzzy Sets and Their Applications": Selected Papers by L.A.Zadeh, *Artificial Intelligence* Vol 61, pp. 351-358, 1993.

[TFH92] M. V. Tegethoff, T. E. Figal et S. W. Hird, "Board Test DFT Model for Computer Products", International Test Conference, 1992, pp. 367-371.

[TI89] Texas Instruments Inc., "Product Preview, SN54BCT8244, SN74BCT8244 Scan Test Device with Octal Buffer", 1989.

[TMM96] M. H. Touati, F. Mohamed and M. Marzouki, "System Fault Diagnosis based on a Fuzzy-Qualitative Approach", *European Design and Test Conference*, Paris, France, Mars 1996.

[Tou92] M. H. Touati, "Implantation et Validation de la méthode de Simplification de Graphe sur les Benchmarks Séquentiels", Rapport de DEA, LIRMM-Université de Montpellier II, Juillet 1992.

[TuY89] R. E. Tullos and C. W. Yau, "Test program pseudocode", *European Test Conference*, Paris, 1989, pp. 106-111.

[Ved94] Veda Design Automation, "System Hilo 4, user manual", 1994.

[WaL89] J. A. Waicukauski and E. Lindbloom, "Failure Diagnosis on Structured VLSI", IEEE Design and Test of Computers, 1989, pp. 49-60.

[Wag87] P. Wagner, "Interconnect Testing with Boundary Scan", International Test Conference, 1987, pp. 52-57.

[Woo91] O. C. Woodward Sr, "Voltage Contrast Electron Beam Tester for Testing Unpopulated MCMs", *Hybrid Circuit Technology*, February 1991, pp. 20-27.

[XiS86] Z. Xiang et S. N. Srihari, "A Strategy for diagnosis based on empirical and model knowledge", journées des systèmes experts, Avignon (France), pages 835-848, 1986.

[Zad65] L. A. Zadeh, "Fuzzy Sets", *Information and Control*, pp. 338-353, 1965.

[Zor92] Y. Zorian, "A Universal Testability Strategy for Multi-Chip Modules Based on BIST and Boundary Scan", International Conference on Computer Design, 1992, pp. 59-66.

[Zor95] Y. Zorian, "Design and Test of MCMs", *European Design and Test Conference*, TUTORIAL #D, Paris 1995.