

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE  
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À  
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE  
À L'OBTENTION DE LA MAÎTRISE EN GÉNIE ÉLECTRIQUE  
M.Eng.

PAR  
DANIEL TREMBLAY

OPTIMISATION DES MÉTHODES DE TEST DES CIRCUITS NUMÉRIQUES

MONTREAL, LE 8 FÉVRIER 2007

© droits réservés de Daniel Tremblay

CE MÉMOIRE A ÉTÉ ÉVALUÉ  
PAR UN JURY COMPOSÉ DE :

M. Claude Thibeault, directeur de mémoire  
Département de génie électrique à l'École de Technologie Supérieure

M. Jean-François Boland, président du jury  
Département de génie électrique à l'École de Technologie Supérieure

M. Jacob Davidson, jury externe  
Département d'informatique à l'Université du Québec à Montréal

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC  
LE 17 JANVIER 2007  
À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

# OPTIMISATION DES MÉTHODES DE TEST DES CIRCUITS NUMÉRIQUES

Daniel Tremblay

## SOMMAIRE

Les progrès incessants de la microélectronique sont accompagnés de défis qui se renouvellent constamment et qui touchent tous les aspects de l'intégration à grande échelle, de la conception au test. Au niveau de la conception, cela se traduit par une recherche continue de gain de productivité. Pour le test, cela signifie l'émergence de nouveaux mécanismes de défauts qui finissent par augmenter le coût de cette étape cruciale. Ce travail de recherche a pour but de réduire le temps de test des puces électroniques à l'aide de séquences de test fonctionnelles. Pour ce faire, des signaux génériques ont été utilisés comme sources de génération des vecteurs. Des vecteurs de type balayage ont aussi été ajoutés dans le but de couvrir les sites qui ne l'étaient pas par les vecteurs fonctionnels. L'usage des vecteurs fonctionnels, par rapport aux vecteurs de type balayage, nécessite un temps de test plus court et exige des ressources en mémoire moins élevées au niveau du testeur. Deux types de pannes ont été étudiés dans ce mémoire, soit la panne collé-à, qui vérifie l'intégrité de la structure du circuit, ainsi que la panne transition, qui teste les caractéristiques temporelles du circuit. Pour générer les séquences de test, le logiciel *Matlab* a été utilisé. Ensuite, la conception de modules de simulation de panne fonctionnelle, collé-à et transition a dû être faite et intégrée à un simulateur logique. Ce simulateur, initialement conçu pour tester les pannes de types pont, emploie le simulateur logique Verilog\_XL, pour effectuer les simulations de panne. Les circuits étudiés sont des filtres RIF variant de 4 à 100 étages. Les résultats de simulations de panne montrent que l'utilisation des vecteurs fonctionnels donne une réduction du temps de test jusqu'à 82% pour le modèle de panne collé-à et 85% pour les pannes transitions dans les meilleurs cas. Dans le cas de la réduction de l'espace mémoire du testeur, les réductions sont de 76% pour le modèle collé-à et de 84% pour le modèle transition dans les meilleurs cas. Ces résultats nous confirment que l'utilisation de signaux génériques comme source de stimuli, dans le cas des filtres RIF, est grandement avantageuse par rapport à l'utilisation des vecteurs à balayage seulement.

# OPTIMIZATION OF THE DIGITAL CIRCUITS TEST METHODS

Daniel Tremblay

## ABSTRACT

The goal of this research is to reduce the test time of integrated circuits by using functional tests. This is accomplished through the use of generic signals for the generation of test vectors. Scan vectors have also been added to cover the sites that were not tested by the functional vectors. The use of the functional vectors, in comparison with the use of scan vectors, results in a shorter test time and requires less memory resources on the tester. Two types of faults have been studied in this research, the stuck-at model, to verify the integrity of the structure of the circuit, as well as the transition fault model, to verify the temporal features of the circuit. The *Matlab* software has been used to generate the test sequences. Afterwards, the conception of functional simulation modules for the two fault models had to be made and integrated to our own fault simulator: Sim\_atpg. This simulator, initially conceived to test the bridge fault model, uses the logical simulator Verilog\_XL. The tested circuits were FIR filters from 4 to 100 taps. The results of the simulations show that the use of the functional vectors resulted in a reduction of the test time of up to 82% for the stuck-at model and 85% for the transition model in the best case. Regarding the possible reduction of the tester's memory space, it is up to 76% for the stuck-at model and 84% for the transition model in the best case. These results confirm that the use of generic signals as source of stimulus in the case of the RIF filters, is greatly advantageous as compared to using only scan vectors.

## **REMERCIEMENTS**

Je voudrais remercier sincèrement M. Claude Thibeault, le directeur de ce projet, de m'avoir permis de travailler avec lui. Son aide et sa vaste expérience m'ont permis de pouvoir mener à terme ce mémoire tout en agrandissant le nombre de mes connaissances sur le sujet du test.

L'aide reçu des membres du LACIME m'a aussi permis de mener à bien ce projet, spécialement Yassine Hariri, qui m'a permis d'utiliser son simulateur logique Sim\_atpg ainsi que les nombreuses heures qu'il a passé à m'aider pour la conception des modules de simulation de pannes.

Je voudrais aussi reconnaître le soutien reçu de mes parents et amis, durant ces deux dernières années.

## TABLES DES MATIÈRES

	Page
SOMMAIRE.....	i
ABSTRACT.....	ii
REMERCIEMENTS.....	iii
TABLES DES MATIÈRES.....	iv
LISTE DES TABLEAUX .....	vii
LISTE DES FIGURES .....	viii
LISTE DES ABRÉVIATIONS ET SIGLES.....	x
INTRODUCTION .....	1
CHAPITRE 1      INTRODUCTION AU TEST DE CIRCUIT INTÉGRÉ VLSI.....	4
1.1            Introduction.....	4
1.2            Objectif du test d'un circuit intégré .....	4
1.3            Positionnement du test dans le processus de conception d'une puce	4
1.3.1        Conception à haut niveau.....	5
1.3.2        Conception à bas niveau .....	6
1.4            Description du flot standard de génération des vecteurs de tests .....	7
1.5            Les divers types de test.....	8
1.5.1        Les avantages et les inconvénients reliés à l'utilisation du test fonctionnel .....	8
1.5.2        Avantages et inconvénients du test structurel.....	9
1.5.3        Comparaison entre les tests fonctionnel et structurel .....	10
1.6            Définition d'un nouveau rôle pour le test fonctionnel.....	12
1.7            Conclusion .....	13
CHAPITRE 2      MODÈLES DE PANNES ET GÉNÉRATIONS DES VECTEURS.....	14
2.1            Introduction.....	14
2.2            Modèle de panne.....	14
2.3            Description des modèles de pannes .....	14
2.3.1        Collé-à.....	15
2.3.2        Délai de transition.....	16
2.3.3        Délai de chemin .....	18
2.4            Testabilité.....	19

2.4.1	Contrôlabilité .....	20
2.4.2	Observabilité .....	20
2.5	Méthode de génération de vecteurs de test .....	20
2.5.1	ATPG .....	21
2.5.2	Méthode de génération des vecteurs pour les circuits combinatoires .....	21
2.5.3	Méthode de génération des vecteurs pour les circuits séquentiels ..	23
2.6	Nouvelle approche : l'utilisation d'outil au niveau système .....	27
2.7	Conclusion .....	29
CHAPITRE 3	STIMULIS ET FILTRES .....	30
3.1	Introduction .....	30
3.1	Signaux .....	30
3.2.1	Sinus .....	30
3.2.2	Bruit blanc gaussien .....	31
3.2.3	Impulsion .....	32
3.2.4	Chirp .....	33
3.3	Filtre RIF .....	33
3.3.1	Description du filtre RIF .....	33
3.3.2	Modélisation du filtre RIF sur <i>Matlab/Simulink</i> .....	34
3.3.3	Validation des modèles Matlab et Verilog .....	35
3.4	Nombre de vecteurs par séquence de test .....	37
3.5	Conclusion .....	38
CHAPITRE 4	INFRASTRUCTURE LOGICIELLE .....	39
4.1	Introduction .....	39
4.2	Simulateur <i>sim_atpg</i> .....	39
4.3	Module de panne collé-à .....	40
4.3.1	Détection d'une panne collé-à .....	40
4.3.2	Modification apportées au netlist pour le modèle de panne collé-à ..	41
4.4	Création du module de simulation des pannes transitions .....	42
4.4.1	Détection d'une panne transition sur un site à tester .....	42
4.5	Format des fichiers de résultats .....	45
4.6	Détermination du nombre de vecteurs fonctionnels utiles .....	46
4.6	Ordre d'appel des différents programmes de simulation de panne ..	50
4.7	Génération des vecteurs de type balayage .....	51
4.7.1	Génération de la liste des sites non couverts .....	52
4.8	Vérification du fonctionnement des programmes .....	53
4.8	Conclusion .....	54

CHAPITRE 5	RÉSULTATS ET ANALYSE .....	55
5.1	Introduction.....	55
5.2	Résultats de simulation de panne du modèle collé-à .....	55
5.3	Résultats des estimations de panne du modèle transition.....	62
5.4	Gain en temps de l'utilisation des vecteurs fonctionnels.....	66
5.5	Gain en mémoire sur ATE de l'utilisation des vecteurs fonctionnels .....	69
5.6	Réduction du temps de simulation de panne .....	70
5.7	Conclusion .....	75
CONCLUSION.....		76
RECOMMANDATIONS .....		77
ANNEXE 1 .....		78
RÉFÉRENCES .....		82



## LISTE DES TABLEAUX

	Page
Tableau I	Type de test le plus performant par rapport à un critère de choix ..... 12
Tableau II	Nombre de vecteurs fonctionnels composant ..... 38
Tableau III	Résultats de simulation de pannes collé-à du filtre à 4 étages ..... 56
Tableau IV	Résultats de couverture de pannes collé-à du sinus et du bruit blanc... 57
Tableau V	Résultats de simulation du modèle collé-à avec le signal du bruit ..... 58
Tableau VI	Nombre de vecteurs à balayage pour le modèle collé-à ..... 59
Tableau VII	Augmentation du nombre de vecteurs et de cellules entre chaque filtre..... 61
Tableau VIII	Nombre de sites non couverts pour le bruit blanc..... 62
Tableau IX	Résultats des estimations de couverture de pannes pour les modèles .. 63
Tableau X	Nombre de vecteurs fonctionnels pour les modèles de panne transition..... 64
Tableau XI	Couverture de panne transition avec l'utilisation de vecteurs fonctionnels..... 65
Tableau XII	Gain en temps pour le modèle collé-à avec l'utilisation des vecteurs fonctionnels..... 67
Tableau XIII	Nombre de vecteurs à balayage pour les modèles transition sans et avec l'utilisation du test fonctionnel avec du bruit blanc..... 68
Tableau XIV	Gain en mémoire pour le modèle collé-à avec le signal du bruit blanc..... 69
Tableau XV	Gain en mémoire pour les modèles transition avec le signal du bruit blanc..... 70
Tableau XVI	Temps de chargement et de simulation des vecteurs ..... 73
Tableau XVII	Estimation de la réduction du temps de simulation de panne..... 74

## LISTE DES FIGURES

	Page
Figure 1	Niveaux d'abstraction de la conception d'une puce ..... 5
Figure 2	Flot standard de génération de vecteurs de test ..... 7
Figure 3	Principe de la boîte noire ..... 9
Figure 4	Porte logique Non-Et ainsi que sa table de vérité..... 15
Figure 5	Porte Non-Et collé-à et sa table de vérité ..... 15
Figure 6	Modèle de panne transition TDF ..... 17
Figure 7	Modèle de panne transition IRF..... 18
Figure 8	Circuit affecté par le délai de chemin ..... 19
Figure 9	Circuit d'un additionneur ..... 24
Figure 10	Démonstration de la technique des fenêtres de temps ..... 24
Figure 11	Principe de fonctionnement de l'outil <i>DFTAdvisor</i> ..... 27
Figure 12	Traitement du signal du sinus par le filtre RIF modélisé avec Matlab ..... 31
Figure 13	Signal du bruit blanc gaussien généré par <i>Matlab</i> ..... 32
Figure 14	Modélisation d'un filtre RIF transposé à 4 étages..... 35
Figure 15	Simulation du filtre RIF à 4 étages modélisé en Verilog..... 36
Figure 16	Comparaison entre la sortie des filtres Verilog et Matlab ..... 36
Figure 17	Modification à liste d'interconnexions pour les simulations de panne collé-à ..... 41
Figure 18	Flot de simulation de panne transition..... 43
Figure 19	Contenu des tableaux transitions descendantes avant et après décalage ..... 44
Figure 20	Format du fichier de résultat du modèle collé-à ..... 45
Figure 21	Format du fichier de résultat du modèle transition..... 45
Figure 22	Résultat d'un réarrangement de la séquence de test initiale..... 47

Figure 23	Méthode de réduction de vecteurs avec plusieurs simulations plus petites .....	48
Figure 24	Résultat de simulation collé-à de 2 noeuds fictifs .....	49
Figure 25	Algorithme du programme stuck_opt.....	50
Figure 26	Ordre d'appel des programmes pour les simulations collé-à et transition .....	51
Figure 27	Modifications à la liste des noms des sites à tester.....	52
Figure 28	Format de la liste de panne utilisé par <i>Mentor Graphics</i> .....	53

## **LISTE DES ABRÉVIATIONS ET SIGLES**

DFT	Design For Test
IDDQ	Quiescent current test
TDF	Transition Delay Fault
IRF	Inline Resistance Fault
ATE	Automatic Test Equipment
RIF	Réponse Impulsionnelle Finie
HDL	Hardware Design Language

## INTRODUCTION

L'évolution constante de l'industrie de la microélectronique a plusieurs impacts sur le procédé de conception d'une puce électronique. Par exemple, la réduction de la taille des transistors permet d'en placer une toujours plus grande quantité sur une même surface de silicium. Ces développements technologiques ont aussi une influence sur la détection des défauts pouvant se trouver dans cette puce. Les techniques de test doivent continuellement être adaptées et de nouvelles doivent être utilisées pour faire face à l'apparition de nouveaux types de pannes. Pour une même puce, nous devons utiliser un nombre grandissant de techniques de test, ce qui implique un plus grand nombre de vecteurs de test à appliquer au circuit. L'augmentation du nombre de vecteurs a un impact direct du coût relié au test de cette puce, en allongeant la durée de celui-ci et en nécessitant plus de ressources au niveau du testeur. Afin de réduire ces coûts, l'optimisation de ces vecteurs apparaît comme une voie à explorer.

Le type de test le plus utilisé présentement dans l'industrie est le test structurel. Les avantages majeurs de l'utilisation de ce type de test incluent le temps nécessaire à la génération des vecteurs de test qui est très court ainsi que les couvertures de pannes très élevées. En contrepartie, ce type de test nécessite un temps d'application au circuit important. L'autre type de test qui était grandement utilisé avant l'apparition du test structurel est le test fonctionnel. Ce type de test offre des couvertures de pannes souvent inférieures à celles offertes par le test structurel et requiert un temps de génération des vecteurs considérable. C'est pour ces raisons qu'il a été laissé graduellement de côté.

Ce travail de recherche propose une méthode d'optimisation s'inscrivant à contre-courant des tendances actuelles du test tout en tirant profit des tendances au niveau de la conception. Nous proposons l'utilisation du test fonctionnel comme test initial, complété par l'utilisation de vecteurs structurels. L'utilisation pronée dans le cadre de ce projet est telle qu'elle permet d'atténuer au maximum l'impact des désavantages

du test fonctionnel tout en profitant de ces avantages. Afin de réduire le temps de génération du test fonctionnel, l'utilisation de signaux génériques comme stimuli sera mise de l'avant. Ces signaux, qui sont générés à l'étape du design à haut niveau du circuit, servent à valider les modèles pour s'assurer de leur bon fonctionnement. Leur réutilisation, qui s'inscrit dans les tendances au niveau conception quant à l'utilisation des outils de haut niveau pour la génération de stimuli, permettra d'effectuer un gain important au niveau du temps de génération des vecteurs. L'utilisation des vecteurs fonctionnels vise deux objectifs, soit la réduction du temps de test ainsi que la réduction de la mémoire nécessaire sur un testeur.

Pour mesurer leur efficacité, deux modèles de pannes ont été étudiés, soit le modèle collé-à et le modèle transition. Puisqu'aucun simulateur de panne fonctionnelle pour ces modèles n'était pas à notre disposition, nous avons modifié un simulateur de panne développé à l'ÉTS, Sim\_atpg.

Le premier objectif de ce travail est la conception d'un module de test fonctionnel pour les pannes de type collé-à. Il sera par la suite intégré au simulateur de panne pour nous donner la couverture de panne. Le deuxième objectif est la création d'un module de test fonctionnel pour les pannes de type transition. Le troisième objectif sera le développement d'un module qui sert à trouver le nombre de vecteurs fonctionnels utiles dans chaque séquence de test, afin de réduire à son maximum le temps du test total.

Le mémoire est organisé de la façon suivante. Le chapitre 1 présente l'objectif et le positionnement du test dans le processus de conception d'une puce. Une description sommaire des types de test existants y est faite et les problématiques reliées au test y sont exposées. Le chapitre 2 est consacré à la description des modèles de pannes qui ont été utilisés dans ce mémoire. On y fait aussi mention des méthodes classiques de test d'un circuit. Le chapitre 3 présente les signaux génériques ainsi que le circuit utilisé pour effectuer les simulations de panne. Le chapitre 4 décrit la façon dont les modules

de simulation de pannes collé-à et transition ont été conçus. La méthode de conception du module d'optimisation des vecteurs y est aussi expliquée. Le chapitre 5 contient les résultats de simulation de panne ainsi que les gains en temps et en mémoire obtenus avec notre méthode de travail. Le mémoire se termine par une conclusion ainsi que des recommandations pour des travaux futurs sur le sujet.

## **CHAPITRE 1**

### **INTRODUCTION AU TEST DE CIRCUIT INTÉGRÉ VLSI**

#### **1.1 Introduction**

Le premier chapitre a pour but de placer en contexte l'objectif et le positionnement du test dans le processus de conception d'une puce électronique. Les divers types de tests existants y seront discutés et comparés pour ensuite soulever les problématiques qui y sont rattachées. Pour ces problématiques, des plans d'action seront mis en place et des solutions seront proposées pour ensuite être réalisées.

#### **1.2 Objectif du test d'un circuit intégré**

Le test a pour but de s'assurer de la qualité d'un circuit intégré après sa fabrication en s'assurant que les objectifs qui ont été fixés initialement tel que la fonctionnalité, la vitesse, la consommation de puissance et la couverture de panne sont respectées.

#### **1.3 Positionnement du test dans le processus de conception d'une puce**

Le cycle de design d'une puce requiert plusieurs étapes. Afin de mieux détecter les pannes qui pourraient être présentes dans une puce après sa fabrication, il est important de comprendre le processus de conception de celle-ci pour savoir à quel moment ce processus est modifié afin de s'assurer de détecter les pannes et ce le plus rapidement possible.

Le processus de conception consiste généralement à passer d'un niveau d'abstraction élevé vers un niveau plus bas (figure 1). Après le passage d'un niveau d'abstraction à un autre, on effectue la vérification du modèle pour s'assurer qu'il n'y a pas eu de



changement au niveau de la fonctionnalité de la puce ou de ces paramètres de fonctionnement. La vérification est habituellement effectuée à l'aide de bancs d'essai.

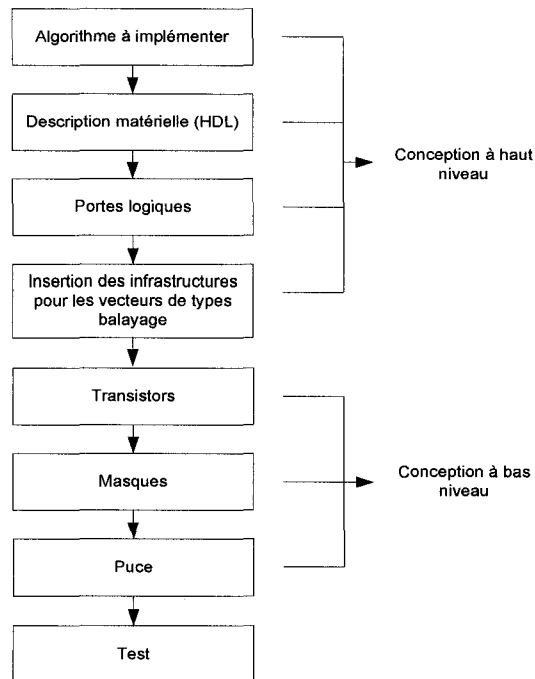


Figure 1 Niveaux d'abstraction de la conception d'une puce

### 1.3.1 Conception à haut niveau

La phase de conception à haut niveau regroupe les différentes étapes où la description de l'algorithme à implémenter est réalisée en un langage de design matériel (HDL). Ce code sera ensuite traduit en portes logiques par un outil de synthèse. Ce dernier utilisera des portes logiques de la technologie avec laquelle on veut fabriquer la puce. L'introduction de la technologie utilisée dans le design permet de connaître les contraintes temporelles reliées à chacune de ces portes logiques. La vérification du design avec les contraintes temporelles est effectuée soit à l'aide d'un simulateur logique ou d'un outil d'analyse statique pour s'assurer que les délais qui pourraient apparaître dans le design n'affectent pas le fonctionnement normal du circuit. Il s'agit ici d'une

vérification préliminaire puisque les délais associés aux interconnexions ne sont pas encore connus. Certains outils de synthèse fournissent également un estimé des délais. On fait ensuite l'introduction dans le circuit d'une ou plusieurs chaînes de balayage. Ces chaînes servent à tester le circuit à l'aide de vecteurs structurels (chapitre 1.5). C'est aussi à ce moment où l'on doit commencer à penser à l'étape du test, soit les modèles de panne à tester ainsi que le type de test que l'on veut utiliser.

### **1.3.2 Conception à bas niveau**

C'est à cette étape que les portes logiques sont transformées en transistors (en associant la représentation logique d'une porte à sa représentation au niveau transistor et masque, contenue dans une librairie de cellules normalisées, par exemple), qu'elles sont placées sur la puce et que les interconnexions entre elles sont effectuées. On place généralement les composantes les plus connectées et utilisées dans le milieu de la puce et celles les moins connectées en périphérie. Des connections trop longues entre les fils peuvent créer des délais trop grands pour que la puce fonctionne correctement. Aussi, on doit espacer au maximum les lignes de métal puisque leur rapprochement provoque l'apparition de capacités parasites, ce qui se traduit par l'ajout de délais supplémentaires dans la puce. Les vérifications qui sont effectuées entre chaque niveau servent à s'assurer que la fonction du circuit n'a pas été altérée et que les délais introduits par le placement et le routage des transistors n'empêchent pas le circuit de fonctionner à sa vitesse nominale. Des délais trop grands nécessiteront un changement de positionnement des transistors se trouvant sur les chemins critiques qui ne respecte pas les exigences temporelles. Lorsque toutes ces étapes ont été franchies avec succès, la puce peut être fabriquée.

#### 1.4 Description du flot standard de génération des vecteurs de tests

Après sa fabrication, le circuit doit être testé pour détecter tous les types de pannes qui auraient pu être injectées dans le circuit pendant sa fabrication. Pour ce faire, il existe trois types de tests possibles: les tests paramétriques, fonctionnels et structurels. Les tests paramétriques visent à s'assurer que des paramètres comme la tension, le courant et la température auxquels le circuit peut fonctionner sont ceux qui ont initialement été prévus. Le test fonctionnel sert à vérifier le bon fonctionnement de la fonction implémentée sur la puce. Le test structurel vérifie l'intégrité de la structure de la puce pour s'assurer que le processus de fabrication s'est bien déroulé. Les deux derniers types de test retiendront notre attention.

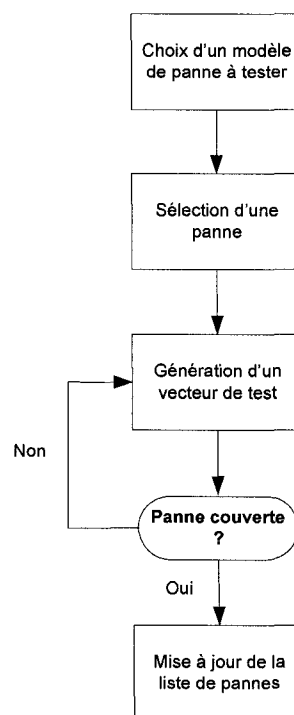


Figure 2 Flot standard de génération de vecteurs de test

Peu importe le type de test (fonctionnels ou structurels) ou de modèle de panne (collé-à, transition, délai de chemin, IDDQ (Nigh et Maly, 1989)) choisi, le flot de génération de vecteurs reste est le même. Il est représenté à la figure 2. On doit premièrement choisir le modèle de panne que l'on veut tester et le type de test que l'on désire lui appliquer. On crée ensuite une liste contenant tous les sites potentiels de pannes du circuit. Pour chaque site, on génère un ou des vecteurs. Si les vecteurs ne peuvent couvrir la panne, on en génère à nouveau. Lorsqu'un de ces vecteurs couvre le site, il est retiré de la liste de sites à tester et on passe au site suivant. Il est possible, pour un site donné, de ne pas trouver de vecteur qui peut le couvrir. Il sera retiré de la liste de pannes et comptabilisée dans les sites non couverts.

## **1.5 Les divers types de test**

Comme présenté dans la section précédente, les tests fonctionnel et structurel requièrent la génération de vecteurs afin de pouvoir être appliqué à un circuit. La génération et l'utilisation de ces vecteurs, pour chaque type de test, ont chacun leurs avantages et inconvénients respectifs. Dans cette section, on en fera l'élaboration pour chaque type de test pour ensuite les comparer, afin de distinguer quel test peut être le plus efficace selon divers critères de choix.

### **1.5.1 Les avantages et les inconvénients reliés à l'utilisation du test fonctionnel**

On peut comparer l'utilisation du test fonctionnel au test d'une boîte noire (figure 3). On envoie des vecteurs sur les entrées, sans s'intéresser à la structure intérieure, et on compare les sorties avec celles qui ont été prévues initialement. S'il y a une différence entre les deux résultats, le circuit est considéré défectueux.

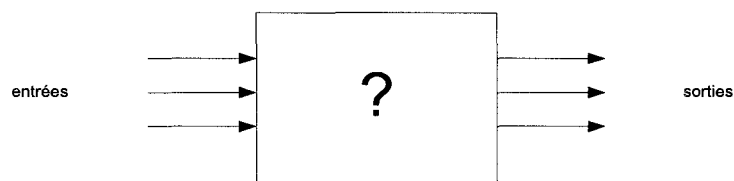


Figure 3 Principe de la boîte noire

Un des grands avantages du test fonctionnel est qu'il peut être fait à la vitesse nominale du circuit, ce qui permet de détecter les pannes liées à des problèmes de délai trop grand. Aussi, son temps d'application est habituellement très court (Maxwell, Hartanto et Bentz, 2000). Il permet en plus de détecter des circuits défectueux que d'autres tests ne peuvent identifier (Maxwell, Hartanto et Bentz, 2000)(Maxwell, Aitken, Johansen et Chang, 1991).

Son temps de développement est par contre très grand (Maxwell, Hartanto et Bentz, 2000)(Tumin, Vargas, Patterson et Nappi, 2001), ce qui pourrait causer des délais dans la mise en marché du produit. Ceci est encore plus vrai lorsque l'on veut atteindre des couvertures de panne de plus de 90% (Tumin, Vargas, Patterson et Nappi, 2001). Avec l'accroissement continu du nombre de transistors dans une puce, il devient plus difficile et plus long de le développer (Lin, Press, Rajske, Reuter, Rinderknecht et Swanson, 2003).

Il a aussi besoin d'être effectué sur un testeur coûteux (West, 1999). Ces coûts sont dus en partie à cause de la haute fréquence à laquelle les testeurs doivent fonctionner.

### 1.5.2 Avantages et inconvénients du test structurel

L'utilisation du test structurel nécessite l'implémentation d'une chaîne de balayage (chapitre 2.5.3.3) dans le circuit transformant le problème du test des circuits séquentiels

en problème combinatoire. Il permet aussi d'augmenter la testabilité (chapitre 2.4) du circuit. L'avantage majeur du test structurel est que le temps de développement des séquences de tests est très court parce qu'elles sont générées automatiquement à l'aide de logiciels commerciaux. Le coût du testeur pour effectuer le test n'est pas élevé, puisqu'il n'a pas à fonctionner à la vitesse nominale du circuit.

L'utilisation du test structurel comporte aussi des désavantages. L'ajout de la chaîne de balayage augmente la surface de silicium de la puce, ajoute des broches au design, peut réduire la vitesse de fonctionnement et le temps d'exécution du test est plus long par rapport au test fonctionnel. L'espace mémoire requis sur un testeur pour emmagasiner les séquences de tests structurels est très grand puisque qu'il doit garder en mémoire, pour chaque vecteur, toutes les valeurs des éléments de mémoire du circuit. Et plus le circuit est grand, plus les besoins en mémoire le sont.

### **1.5.3 Comparaison entre les tests fonctionnel et structurel**

Les tests fonctionnel et structurel ont chacun leurs avantages et inconvénients, mais le test fonctionnel est de plus en plus délaissé pour le test structurel (Tumin, Vargas, Patterson et Nappi, 2001)(Lin, Press, Rajski, Reuter, Rinderknecht et Swanson, 2003). Le temps de développement du test fonctionnel est le problème majeur à son utilisation. Il nécessite d'être fait à la main, comparé au test structurel qui est fait automatiquement. Le temps utilisé à développer les séquences de tests fonctionnels augmente le délai de la mise en marché de la puce (Maxwell, Hartanto et Bentz, 2000), ce qui implique une perte de revenus pour la compagnie qui développe le circuit.

La différence entre la couverture de panne des tests fonctionnels et des tests structurels a déjà fait l'objet d'études. La couverture de panne des tests structurels est plus élevée que la couverture de panne des tests fonctionnels (Maxwell, Hartanto et Bentz, 2000)(Maxwell, Aitken, Johansen et Chang, 1991)(Tumin, Vargas, Patterson et Nappi,

2001). Cependant, dans chacune de ces études, les tests fonctionnels ont permis de détecter des composantes que les tests structurels n'ont pas été en mesure de trouver (Maxwell, Hartanto et Bentz, 2000)(Maxwell, Aitken, Johansen et Chang, 1991).

Malgré cela, est-ce que le nombre de sites couverts dans un circuit est le seul facteur auquel on devrait tenir compte? Maxwell, Aitken, Johansen et Chang (1991) ont montré que le test fonctionnel a de meilleures capacités à détecter les pannes que les tests structurels pour le modèle de panne collé-à. L'étude démontre qu'avec une couverture de panne de 75 % pour le test fonctionnel, le nombre de circuits toujours défectueux était de l'ordre de 2% et pour que le test structurel puisse atteindre un tel niveau, la couverture de panne doit être de 80%. Autrement dit, même avec une couverture de panne plus élevée que le test fonctionnel, le test structurel a laissé passer plus de puces défectueuses que le test fonctionnel.

Le test fonctionnel exige aussi l'utilisation de testeur automatique très coûteux, principalement parce que le test est fait à la vitesse nominale du circuit, ce qui demande au testeur d'être extrêmement précis dans ses mesures de valeur. Avec l'augmentation continue des performances des circuits, on doit être dans la mesure de concevoir des testeurs qui peuvent suivre la cadence d'augmentation de la performance des nouvelles technologies (West, 1998). Puisque le coût des testeurs ne cesse d'augmenter, il est important de pouvoir le maintenir le plus bas possible. L'espace mémoire nécessaire pour les tests structurels est plus grand que pour les tests fonctionnels (Tumin, Vargas, Patterson et Nappi, 2001). Ceci est causé par la taille des vecteurs employés pour le test structurel, qui exige de garder en mémoire les valeurs de chaque bascule composant le circuit, alors que la taille des vecteurs fonctionnels est dépendante du nombre d'entrées du circuit.

Aussi, rappelons que les tests structurels nécessitent des modifications à la circuiterie du circuit en modifiant les bascules en place par des bascules de types balayage,

augmentant leur surface et leur nombre de broches. Un résumé des critères qui nous intéressent est présenté dans le tableau I.

Tableau I

Type de test le plus performant par rapport à un critère de choix

Facteurs à tenir compte lors du choix d'un type de test	Test le plus avantageux
Vitesse à laquelle le test est appliqué	Fonctionnel
Augmentation de la surface	Fonctionnel
Nombre de broches du circuit	Fonctionnel
Vitesse de fonctionnement de la puce	Fonctionnel
Temps d'application du test	Fonctionnel
Utilisation de mémoire de testeur	Fonctionnel
Coût des testeurs	Structurel
Temps de développement des séquences de tests	Structurel

### 1.6 Définition d'un nouveau rôle pour le test fonctionnel

Le test fonctionnel a été pendant plusieurs années le test de prédilection pour les ingénieurs de test. Il a cependant été laissé de côté, petit à petit, à la faveur du test structurel, principalement en raison de son temps de développement, du temps requis pour effectuer les simulations de panne et du coût des testeurs. Nonobstant de ces inconvénients, le test fonctionnel reste intéressant puisqu'il permet de détecter rapidement les pannes, notamment reliées aux problèmes temporels.

Dans Butler et Saxena (2000), il a été proposé de créer un petit ensemble de vecteurs fonctionnels modérément rapides, placé en début de la séquence de test, pour pouvoir prendre avantage du fait qu'ils sont capables de détecter les composantes défectueuses plus rapidement que d'autre type de test. Le but de ce travail de recherche est



d'investiguer cette proposition d'un point de vue de l'optimisation du test. Pour pouvoir mener cette investigation, les éléments suivants sont pris en compte :

- Nous voulons tirer profit des progrès au niveau méthodologie de conception, notamment du point de vue de la réutilisation des vecteurs de simulation générés à un haut niveau d'abstraction, dans le but de faciliter la génération des vecteurs fonctionnels.
- Nous voulons bénéficier de la couverture de panne commune entre différents modèles de panne, afin de réduire davantage les nombres de vecteurs de test.

## **1.7 Conclusion**

Ce premier chapitre avait pour but d'introduire le lecteur aux différentes méthodes utilisées pour tester un circuit après sa fabrication. Les avantages de chaque méthode ont été présentés pour ensuite en ressortir les principales lacunes.

Pour prendre avantage du fait que le test fonctionnel peut détecter des circuits défectueux rapidement, la réutilisation de vecteurs de vérification générés à un haut niveau de conception comme vecteurs de test a été proposée afin de réduire les efforts qui y sont consacrés et donc rendre son utilisation plus intéressante.

## **CHAPITRE 2**

### **MODÈLES DE PANNES ET GÉNÉRATIONS DES VECTEURS**

#### **2.1 Introduction**

Après avoir soulevé les problèmes reliés aux tests fonctionnels, les modèles de pannes utilisés lors de cette étude seront décrits en détail dans ce chapitre. Ces modèles servent à simuler l'effet d'une panne sur le fonctionnement normal du circuit.

Les techniques classiques de test des circuits intégrés seront présentées, tant au niveau des circuits combinatoires et séquentiels. Ces méthodes de test sont à la base de l'approche de test que l'on propose dans ce mémoire.

#### **2.2 Modèle de panne**

Un modèle de panne est un modèle abstrait qui vise à représenter l'effet d'une défectuosité ou d'une source d'interférence physique dans un circuit. Ces modèles de pannes nous permettent de développer des algorithmes pour simplifier la tâche du test. Dépendamment du modèle de panne choisi, nous devons générer un test en conséquence et effectuer les simulations qui s'y rattachent. Ces vecteurs sont habituellement générés indépendamment de la technologie avec laquelle le circuit a été fabriqué.

#### **2.3 Description des modèles de pannes**

Les modèles de panne utilisés dans ce travail, soit collé-à et transition, seront présenté dans cette section. Leur effet sur le comportement du circuit et les méthodes pour

pouvoir les détecter seront discutées. Le modèle de panne de délai de chemin sera aussi présenté, sans toutefois faire l'attention d'étude future dans ce mémoire.

### 2.3.1 Collé-à

Une panne collé-à est une panne qui affecte un signal qui est de façon permanente à la valeur logique 0 (collé-à 0) ou 1 (collé-à 1). Ce type de panne n'affecte que les interconnexions du circuit. On peut aussi visualiser le modèle de panne collé-à comme étant une connexion entre le nœud fautif et la ligne d'alimentation ou de mise à la terre.

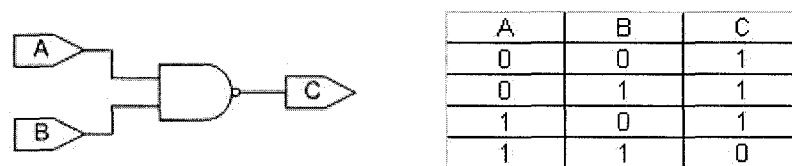


Figure 4 Porte logique Non-Et ainsi que sa table de vérité

La figure 4 représente une porte Non-Et et sa table de vérité lorsque la porte ne contient aucune panne. En forçant la valeur de l'entrée B collé-à 0, ceci revient à la connecter à la mise à la terre.

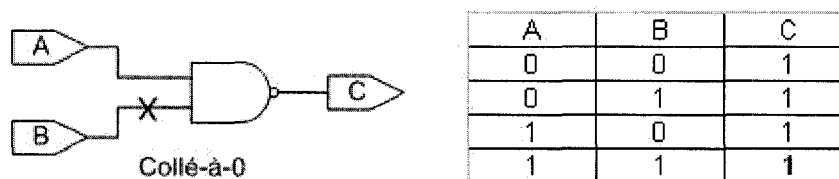


Figure 5 Porte Non-Et collé-à et sa table de vérité

La figure 5 représente la porte Non-Et de la figure 4, mais cette fois avec une panne collé-à 0 sur l'entrée B. Peu importe la valeur appliquée à l'entrée B de la porte, la valeur logique 0 sera toujours transmise à la porte. Ceci provoque un changement dans la table de vérité et lorsque les deux entrées sont à 1, la sortie n'est plus 0, mais 1.

Pour pouvoir tester la panne, on doit envoyer les bonnes valeurs logiques sur les entrées de la porte. L'entrée B doit être à la valeur opposée du modèle collé-à 0, donc à la valeur logique 1. Pour ce qui est de l'entrée A, elle doit être à une valeur permettant de propager l'entrée B vers la sortie, soit la valeur 1.

### **2.3.2 Délai de transition**

Les pannes de types collé-à permettent de vérifier l'intégrité de la structure d'un circuit. Cependant, ceci ne garantit pas que le circuit fonctionne à la vitesse voulue. Il faut donc au moins un autre modèle de panne pour pouvoir vérifier qu'il réponde aux exigences temporelles.

Les pannes de délai de transition (Waicukauski, Lindbloom, Rosen et Iyengar, 1987) sont modélisées de manière ponctuelle au niveau des portes logiques, affectant le délai de propagation au travers de celles-ci. L'hypothèse de base sous-jacente à ce modèle est que les délais supplémentaires causés par ce type de panne est suffisamment important pour faire en sorte que le circuit ne puisse fonctionner correctement à vitesse nominale. Il est possible que ces pannes ne soient pas détectées si le circuit n'est pas testé à sa vitesse nominale.

Il existe 2 types de pannes de délai transition qui peuvent affecter une porte soit, "lent à descendre" et "lent à monter". Une panne de type transition "lent à descendre" signifie que la porte testée ne peut effectuer une transition de la valeur logique 1 vers 0 dans le temps requis. Les délais de transition "lent à descendre" et "lent à monter" ne sont pas

nécessairement identiques pour une même porte logique. Comparé au modèle collé-à, le modèle délai de transition nécessite une paire de vecteurs afin de provoquer une transition sur le site à tester. Comme il est décrit ci-après, il existe 2 manières d'utiliser ce type de pannes, soient de manière complète (avec les 2 transitions, TDF) ou de manière incomplète ( avec une seule transition, IRF).

### 2.3.2.1 Modèle transition TDF

Pour pouvoir couvrir un site avec le modèle TDF (Benware, Lu, Van Slyke, Krishnamurthy, Madge, Keim, Kassab et Rajski, 2004), les deux transitions doivent être couvertes. Ce modèle se comporte comme s'il y avait une résistance indésirable de connectée à la terre ou à la ligne d'alimentation entre deux portes. Elle peut être causée par de l'interférence (crosstalk) ou bien par une source ou drain d'un transistor ouvert (Moore, Gronthoud, Baker et Lousberg, 2000). La figure 6 montre une représentation symbolique du problème.

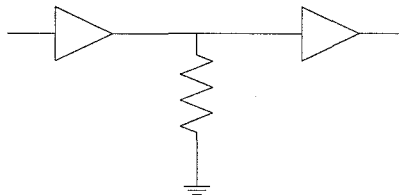


Figure 6      Modèle de panne transition TDF

La résistance agit comme un "pull-down" sur la ligne, ce qui a pour conséquence de rendre lentes les transitions montantes. Si la résistance avait été connectée à la ligne d'alimentation, les transitions descendantes auraient été lentes. On ne peut savoir à l'avance quelle transition serait lente, c'est pourquoi la couverture des deux transitions est nécessaire.

### 2.3.2.2 Modèle transition IRF

Le modèle IRF (Benware, Lu, Van Slyke, Krishnamurthy, Madge, Keim, Kassab et Rajski, 2004) se comporte comme si une résistance fautive était située entre 2 portes (figure 7). Elle peut être, à titre d'exemple, causée par les vias résistifs (Benware, Madge, Lu et Daasch, 2003). Ces vias résistifs apparaissent principalement lors de l'utilisation de technologie 0,18  $\mu\text{m}$  et moins. Le positionnement de la résistance indésirée provoque donc un ralentissement sur les 2 transitions au lieu d'une seule des deux. Nous n'avons donc qu'à tester une des deux transitions pour considérer le site comme étant couvert.



Figure 7      Modèle de panne transition IRF

Les vecteurs générés pour tester le modèle TDF ne sont plus tous utiles et plusieurs d'entre eux peuvent être enlevés de la procédure de test. Aussi, puisque le modèle IRF ne nécessite qu'une seule transition soit couverte comparée au modèle TDF qui exige que les deux transitions soient couvertes, on s'attend à ce que le modèle IRF donne une meilleure couverture de panne.

### 2.3.3 Délai de chemin

Le délai de chemin (Smith, 1985) est similaire au délai de transition puisqu'il s'agit aussi d'une panne au niveau temporel du circuit. Le délai de chemin représente le temps qu'une transition, soit montante ou descendante, prend pour pouvoir traverser un certain chemin. Prenons le circuit de la figure 8 et appliquons-y une transition montante sur l'entrée primaire A. Le temps que la transition met pour pouvoir se rendre à la sortie primaire E ne doit pas excéder un certain temps pour ne pas être affecté par le délai de

chemin. Contrairement au modèle transition, le modèle de délai de chemin ne suppose pas qu'une panne est suffisamment important pour causer un non-respect de la spécification en fréquence. Ce modèle est approprié pour la détection des chemins trop lents en raison des variations du procédé de fabrication.

Le nombre de chemin dans un circuit, exponentiel selon le nombre de portes logiques (Krstic et Cheng, 1998), rend son utilisation beaucoup moins intéressante vu le grand nombre de simulations à effectuer.

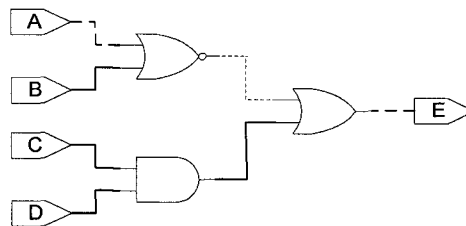


Figure 8 Circuit affecté par le délai de chemin

## 2.4 Testabilité

Afin de nous aider à la tâche de la génération des vecteurs de test pour nos différents modèles de panne décrits précédemment, diverses mesures ont été développées afin de savoir jusqu'à quel niveau un site pouvait être testable. La testabilité d'un circuit est la quantification de la facilité avec laquelle le circuit peut être testé. Avec la grande complexité des circuits présents sur le marché, il est important d'avoir des circuits facilement testables. Un circuit difficilement testable nécessite des tests plus longs, peut avoir des couvertures de pannes plus faibles et entraîne des coûts supplémentaires. Savoir que le circuit n'a pas une bonne testabilité nous permet d'apporter des modifications à la conception du circuit avant qu'il ne soit trop tard et nous évite des coûts supplémentaires.

La testabilité est souvent décrite à l'aide de deux attributs complémentaires, soient la contrôlabilité et l'observabilité.

#### **2.4.1 Contrôlabilité**

La contrôlabilité représente le coût à positionner un noeud du circuit à une valeur prédéterminée à partir des entrées primaires. Les entrées primaires d'un circuit intégré correspondent aux broches d'entrée ainsi qu'à la sortie de certains registres. Ce coût représente habituellement le nombre minimal de noeud dont il faut déterminer la valeur pour pouvoir contrôler un noeud soit à 0 ou 1. Ce coût augmente selon le type de porte, le type de circuit (combinatoire ou séquentiel) et la valeur logique à être imposée sur la ligne.

#### **2.4.2 Observabilité**

L'observabilité représente l'effort nécessaire pour observer une valeur contrôlée à un noeud à partir des sorties primaires. Les sorties primaires correspondent aux broches de sortie d'un circuit ainsi qu'à l'entrée de certains registres. Il s'agit du nombre minimal d'éléments de mémoire entre le noeud à contrôler et une sortie primaire dont on doit déterminer la valeur pour l'observer sur une sortie primaire.

### **2.5 Méthode de génération de vecteurs de test**

Il existe deux types de circuits que les modèles de pannes décrites précédemment peuvent affecter, soit les circuits combinatoires et séquentiels. Pour chaque type de circuit, des méthodes de génération automatique des vecteurs de test (ATPG) ont été développées. Le principe de fonctionnement de l'ATPG sera présenté pour ensuite faire l'élaboration des méthodes de génération des vecteurs respectives aux différents types de circuits.



### 2.5.1 ATPG

La génération automatique de vecteurs de test (ATPG) permet de générer des vecteurs de test déterministes qui ciblent une panne précise. L'utilisation de l'ATPG comme générateur de vecteur de test permet de réduire le temps consacré à cette étape à cause de son automatisation. Plusieurs algorithmes permettent la détection d'une panne. Ils sont généralement constitués des mêmes trois étapes de base, soit :

- l'activation : on impose les valeurs aux entrées de la porte dont on veut tester la sortie à la valeur opposée de la panne pour l'activer selon le concept de contrôlabilité énoncé plus tôt;
- la propagation : la panne est propagée jusqu'aux sorties primaires du circuit selon le concept d'observabilité énoncé plus tôt;
- la justification : les valeurs qui ont été assignées durant les 2 premières phases sont remontées jusqu'aux entrées primaires.

Pour tester une panne de type collé-à dans un circuit combinatoire, la génération d'un seul vecteur de test est suffisant. Dans le cas des circuits séquentiels, à cause des éléments de mémoires ci trouvant, une séquence de vecteurs est nécessaire pour activer et propager la panne jusqu'à une sortie primaire. C'est également le cas pour les pannes de type transition ou chemin de délai dans les circuits combinatoires qui requièrent 2 vecteurs, comme mentionné précédemment.

### 2.5.2 Méthode de génération des vecteurs pour les circuits combinatoires

Pour les circuits combinatoires, les méthodes de l'algorithme D, PODEM et FAN seront présentées, suivies de la méthode de génération pseudo-aléatoire.

### 2.5.2.1 Algorithme D, PODEM et FAN

L'algorithme D de Roth (Roth, Bouricious et Schneider, 1967) fait l'emploi de cubes D qui sont en fait des tables de vérité. Un premier cube D représente les valeurs des entrées du circuit qui vont activer la panne dans le circuit. Un second cube D représente les entrées pour propager la panne vers une sortie primaire. L'algorithme n'offre cependant pas de solution au grand nombre de chemins possibles à emprunter lors de l'étape de la justification jusqu'aux entrées primaires.

L'algorithme PODEM (Goel, 1981) a ensuite été conçu afin de corriger le problème des grands nombre de chemins possibles trouver à l'aide de l'algorithme. Vient ensuite l'algorithme FAN (Fujiwara, 1985), qui réduisait encore plus le nombre de chemins possibles afin d'augmenter la vitesse d'exécution de l'algorithme.

Le problème principal à l'utilisation de ces algorithmes est lorsqu'ils sont utilisés sur de gros circuits, le nombre de chemins possibles à utiliser devient beaucoup trop grands et les efforts nécessaires pour la justification trop importants.

### 2.5.2.2 Pseudo aléatoire

Les tests pseudo aléatoire sont généralement obtenus à l'aide de registres à décalage rebouclé (LFSR). On génère une séquence de test à partir d'une fonction linéaire, soit une addition ou une multiplication, qui est suivie d'une opération modulo, faite à l'aide d'une porte logique Xor. On doit être capable de reproduire la séquence de test après la première utilisation.

La séquence de test générée est en fait déterministe. Pour pouvoir la considérée comme étant pseudo aléatoire, on doit choisir les constantes de la fonction linéaire de façon à rendre la période de la séquence assez longue pour qu'une observation sur une petite

partie de celle-ci la rende pseudo aléatoire. La couverture de panne est alors obtenue à l'aide de simulation et elle est proportionnelle à la longueur totale de la séquence de test (Simeu, Puissochet, Rainard, Tagant et Poize, 1992). Cependant, les tests pseudo aléatoire sont reconnus comme étant inefficace pour détecter les pannes de délais (Dufaza, Viallon et Chevalier, 1995).

### **2.5.3 Méthode de génération des vecteurs pour les circuits séquentiels**

La génération d'un test pour un circuit séquentiel demande plus d'efforts que pour un circuit combinatoire à cause des éléments de mémoire qui y sont présents. Avant d'être testé, les éléments de mémoire du circuit doivent être initialisés dans un état connu. Dans ce qui suit, nous présentons brièvement les méthodes des fenêtres de temps, des simulations à l'aide d'un simulateur concurrent et de la chaîne de balayage. Pour cette dernière méthode, des logiciels commerciaux tel que celui offert par la compagnie *Mentor Graphics*, existent déjà et offrent de très bons résultats.

#### **2.5.3.1 Expansion des fenêtres de temps**

Cette technique requiert de faire une copie du circuit, pour chaque vecteur que nécessitera la séquence de test. L'exemple illustré provient de Bushnell (2000). La figure 9 représente le circuit d'un additionneur. Pour pouvoir tester la panne collé-à 0 indiquée sur la figure 9, les entrées  $A_n$  et  $B_n$  du circuit doivent être à la valeur logique 1. Cependant, l'entrée  $C_n$  du circuit est inconnue puisque la bascule est dans un état inconnu, et pour pouvoir propager l'effet de la panne vers la sortie primaire, la valeur du signal  $C_n$  doit être à 1.

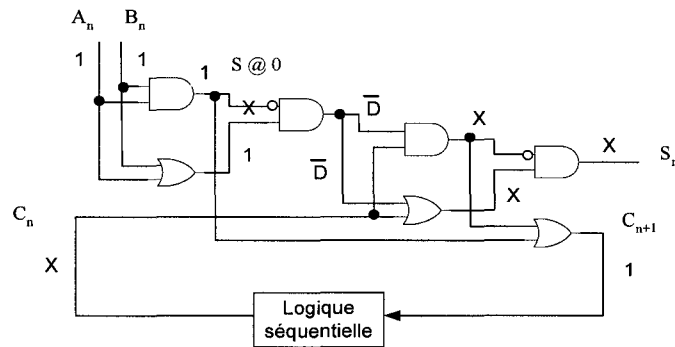


Figure 9 Circuit d'un additionneur

La figure 10 illustre la technique des fenêtres de temps. La fenêtre de droite, soit la fenêtre de temps 0, est celle où on détecte la panne. La fenêtre de gauche, fenêtre -1, servira à initialiser la valeur de  $C_n$ . La panne peut être activée dans chacune des fenêtres, mais cela n'est pas obligatoire. Afin d'obtenir la valeur 1 sur le signal  $C_n$ , les entrées  $A_{n-1}$  et  $B_{n-1}$  devront être placées à la valeur 1. Avec la valeur de  $C_n$  à la valeur logique 1 à l'entrée de la fenêtre de temps 0, la panne pourra être propagé vers la sortie primaire de l'additionneur. Dans ce cas-ci, la séquence de vecteurs sera composée des vecteurs 11 et 11.

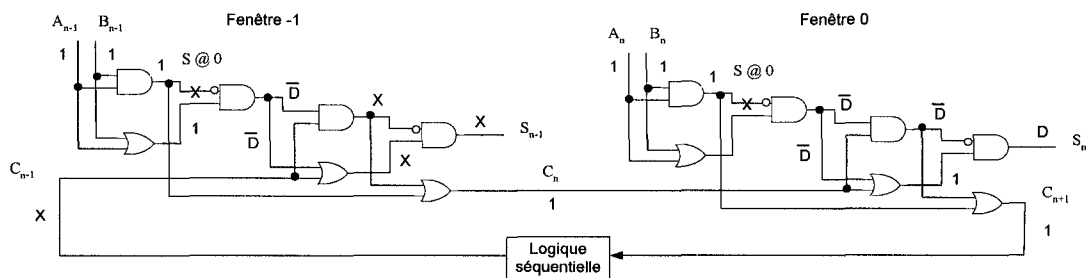


Figure 10 Démonstration de la technique des fenêtres de temps

La technique d'expansion des fenêtres de temps requiert, pour chaque fenêtre, de garder en mémoire une copie de la structure du circuit entier. Plus le circuit sera grand, plus les

besoins au niveau mémoire seront importants, ce qui rend son utilisation peu intéressante.

### **2.5.3.2 Génération de séquences de vecteurs basés sur les simulations**

La génération des séquences se fait à l'aide d'un générateur, généralement pseudo-aléatoire. On se sert ensuite d'un simulateur de panne concourant (Schuler, Ulrich, Baker et Bryant, 1975). On simule ensuite les vecteurs sur le circuit à tester, qui est initialisé dans un état connu et le même pour chaque simulation, pour un certain modèle de panne. On change ensuite les états du circuit correspondant au vecteur ayant détecté le plus de pannes. On recommence tant que l'on n'est pas satisfait de la couverture de panne. Cette méthode requiert plusieurs simulations sans avoir de garantie sur le résultat final, ce qui rend son utilisation moins intéressante.

### **2.5.3.3 Chaîne de balayage complète ('full-scan')**

Le principe de la chaîne de balayage complète est de créer un ou plusieurs registres à décalage avec toutes les bascules qui se trouvent dans le circuit à tester, afin de pouvoir en contrôler et observer les valeurs. Pour ce faire, on doit remplacer les bascules déjà en place par des bascules de type balayage. Plusieurs types de bascules à balayage ont été proposés, Celui qui domine actuellement est le type dit 'mux-scan'. Les bascules de ce type ont comme particularité d'avoir un multiplexeur supplémentaire sur leur entrée, qui permet aux bascules soit de fonctionner normalement, ou de se transformer en registre à décalage pour le mode de test. Les bascules en effet aussi toutes interconnectées entre elles pour faire la création du ou des registres à décalage. Quand les bascules sont en mode de registre à décalage, le circuit séquentiel se transforme alors en circuit combinatoire. On peut donc y appliquer les techniques du test combinatoire.

On peut placer dans les registres les valeurs voulues en décalant ces valeurs à partir d'une broche d'entrée prévue à cet effet. Les valeurs peuvent aussi être observées en les décalant vers la sortie prévue à cet effet. Toutes les bascules peuvent être contrôlées et observées dans un temps équivalent aux nombres de bascules dans le registre à décalage le plus long. Lors de l'ajout de la chaîne de balayage, certaines règles doivent être respectées :

- n'utiliser que des bascules D;
- avoir au moins une entrée disponible pour le test;
- les horloges qui contrôlent les chaînes de balayage doivent être contrôlées à partir d'une entrée primaire;
- une horloge ne peut alimenter une bascule en donnée.

Pour chaque panne dans le circuit, le programme d'ATPG générera la séquence de bits qui doivent être insérés dans le registre à décalage ainsi que la réponse attendue par le circuit sans panne. Pour charger le registre à décalage avec les valeurs voulues, on applique un coup d'horloge par bit. Une fois les valeurs chargées, le circuit est reconfiguré en mode normal et exécute un ou 2 cycles dans ce mode, puis il retourne en mode de test pour récupérer les nouvelles valeurs contenues dans les registres. Fait intéressant, la récupération de ces valeurs se fait lors du chargement du vecteur suivant. Si le résultat de sortie diffère de celui prévu, on considère le circuit comme étant défectueux.

#### **2.5.3.4 Outil pour la génération de vecteur de type balayage**

Des outils commerciaux existent déjà pour faciliter la génération des vecteurs de test pour un circuit utilisant les chaînes de balayage. Ceux disponible à l'ETS sont offerts par la compagnie *Mentor Graphics*. L'outil est composé de deux logiciels soit *DftAdvisor* et *Fastscan*. L'outil *DftAdvisor* (figure 11) sert à faire l'introduction de la ou

des chaînes de balayage dans le circuit. On peut choisir une implémentation complète ou partielle des chaînes de balayage.

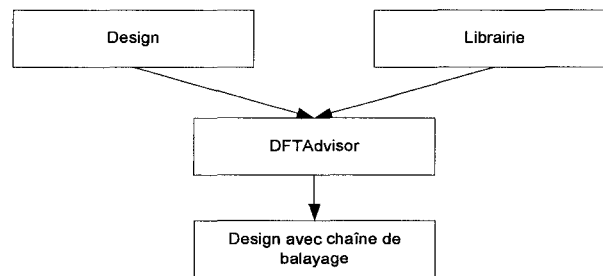


Figure 11 Principe de fonctionnement de l'outil *DFTAdvisor*

Le second outil, *Fastscan*, sert à la génération des vecteurs de test selon le modèle de panne choisi. Les modèles de panne disponibles à l'ETS sont les modèles collé-à, transition et IDDQ. L'outil commence par générer des vecteurs de façon pseudo-aléatoire. Dans le cas où la couverture obtenue est insuffisante, l'outil génère des vecteurs déterministes pour cibler les pannes restantes. On obtient ainsi notre séquence de vecteurs. Il est aussi possible d'optimiser cette séquence de test. Pour ce faire, l'outil va prendre la séquence initialement générée et va appliquer ces vecteurs au circuit en commençant par le dernier vecteur de la séquence et en remontant au premier vecteur. On peut lui demander de recommencer la procédure jusqu'au moment où il a réduit le nombre de vecteur au minimum.

## 2.6 Nouvelle approche : l'utilisation d'outil au niveau système

Comme vu précédemment, les méthodes actuelles pour générer les séquences de test d'un circuit séquentiel, excepté la chaîne de balayage, nécessite un très grand nombre de ressources pour être utilisés. Ceci ne nous incite donc pas à les employer.

Cependant, il existe un moyen de générer des séquences de test fonctionnel sans nécessiter un grand effort machine. Lorsque la modélisation du système à concevoir a été effectuée à l'aide de l'outil *Matlab*, une séquence de simulation a déjà été générée par l'outil pour vérifier la fonctionnalité du modèle.

Mais qu'arrive-t-il si on utilise cette séquence comme stimulus pour effectuer le test du circuit? La séquence de simulation nous servirait comme séquence de test pour effectuer l'estimation de la couverture de panne du circuit des différents modèles de pannes. La réutilisation de la séquence de simulation nous permettra de réduire considérablement le temps consacré à sa génération par rapport à une séquence de test générée manuellement.

L'outil Matlab nous permet de choisir quel type de stimulus désiré, par exemple, des signaux génériques tels que le sinus, le bruit blanc gaussien, le chirp et l'impulsion. Ce genre de signaux est souvent utilisé pour effectuer la validation et la vérification de circuit de télécommunications avant leur mise en marché. En générant une séquence de test fonctionnel à l'aide d'un signal générique avec Matlab, on réduit considérablement l'effort consacré au développement de la séquence de test. Les signaux génériques auraient quand même été utilisés avant la mise en marché, donc pourquoi ne pas les utiliser au début du processus de test.

Aussi, en générant une séquence de test fonctionnel, donc qui nécessite un temps de test très court, on répond à la proposition de Butler et Saxena (2000) qui était d'utiliser une séquence de test fonctionnel au début du processus de test pour détecter les pannes le plus rapidement possibles.

Pour mener cette investigation, des circuits de type filtres RIF ont été choisis puisqu'ils sont fréquemment utilisés dans des applications de DSP. Ils se démarquent aussi par une structure très simple représentant un arbre d'additionneur.



## **2.7 Conclusion**

Dans ce chapitre, les modèles de pannes qui serviront à mener l'investigation de la nouvelle approche du test fonctionnel ont été présentés. Ces modèles serviront à simuler différentes pannes sur nos circuits d'étude soit des filtres RIF.

Les méthodes classiques de test de circuits combinatoires et séquentiels ont été présentées. Les méthodes basées sur l'utilisation de chaînes de balayage sont les plus utilisées. Ce sont ces méthodes que nous voulons compléter avec une séquence de vecteurs fonctionnels, en tirant profit de la réutilisation des vecteurs de vérification.

## **CHAPITRE 3**

### **STIMULIS ET FILTRES**

#### **3.1 Introduction**

Ce chapitre a pour but de présenter la structure des filtres RIF qui a été utilisée pour effectuer les estimations de couverture de pannes. Aussi, les stimuli qui ont été employés pour la création des séquences de test y seront présentés. Pour terminer, la méthode utilisée pour calculer le nombre de vecteurs fonctionnels par séquence de test y sera décrite.

#### **3.1 Signaux**

Pour effectuer la validation des filtres RIF, certains signaux génériques sont fréquemment utilisés tels que le sinus, le bruit blanc gaussien, l'impulsion ainsi que le chirp. Ce sont ces mêmes signaux qui seront réutilisés comme vecteurs de test.

##### **3.2.1 Sinus**

Le sinus est principalement caractérisé par son amplitude et sa fréquence. Le signal de sinus généré est représenté à la figure 12 (signal du haut).

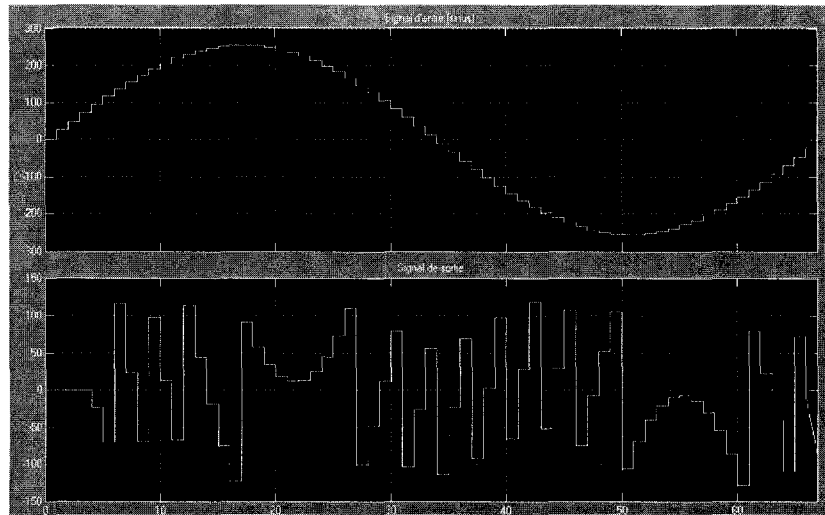


Figure 12 Traitement du signal du sinus par le filtre RIF modélisé avec Matlab

On remarque que la forme de la sortie du filtre (signal du bas, figure 12) est distordue. Ceci est dû au fait que la sortie du filtre a été tronquée à 8 bits, pour tous les filtres, ce qui implique une réduction dans la qualité des résultats de sorties des filtres. La troncation avait pour but de réduire la complexité des filtres.

### 3.2.2 Bruit blanc gaussien

Le bruit blanc est un signal dont la densité spectrale de puissance est la même sur tout l'axe des fréquences et dont la fonction de densité de probabilité est de loi Normale (Distribution Gaussienne). Le terme blanc fait analogie avec la lumière blanche, dont sa puissance est distribuée également sur l'axe des fréquences. Le signal du bruit blanc gaussien est représenté à la figure 13 (signal du haut).

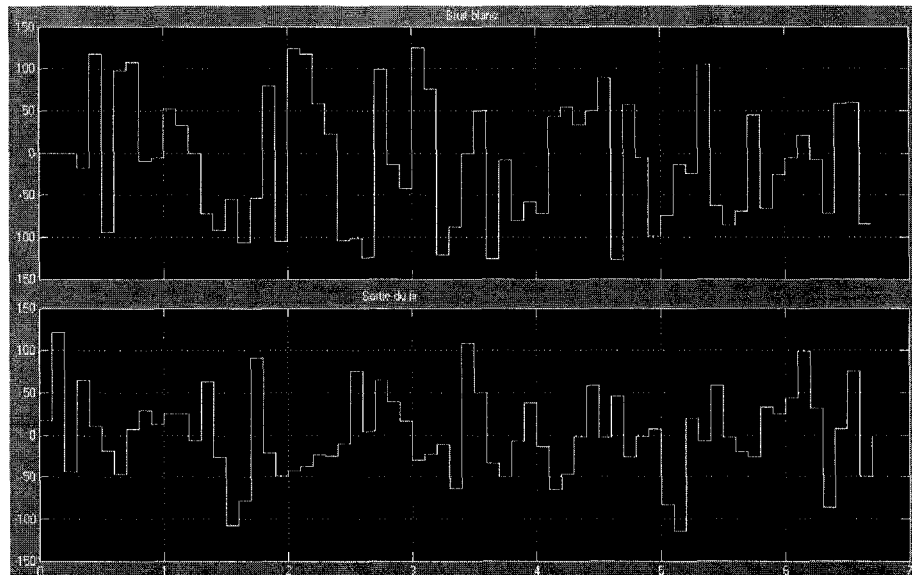


Figure 13 Signal du bruit blanc gaussien généré par *Matlab*

Le signal du bruit blanc gaussien ne se répète pas à intervalle régulier comme le signal du sinus. Pour créer nos vecteurs de test à partir de ce signal, nous avons choisi les valeurs à partir du début de la séquence générée par Matlab et ce en collectionnant suffisamment de valeurs pour créer le nombre de vecteurs nécessaire pour composer notre séquence de test.

### 3.2.3 Impulsion

L'impulsion de Dirac est un signal nul en tout point sauf à l'origine. Appliqué à l'entrée d'un système, on obtient en sortie la réponse impulsionnelle, qui caractérise ce même système. Lorsqu'il est appliqué à un filtre RIF, on obtient en sortie les coefficients du filtre. Ceci est donc une bonne méthode pour valider rapidement la fonctionnalité d'un filtre RIF. Pour compléter la séquence de vecteurs du signal de l'impulsion, des vecteurs de valeur nulle (tous les bits à 0) seront ajoutés pour permettre à l'impulsion de traverser les filtres complètement.

### 3.2.4 Chirp

Le chirp est un signal de sinus dont la fréquence augmente ou diminue dans le temps. La fréquence peut augmenter de façon linéaire ou exponentielle dans le temps. Pour calculer la fréquence minimale du chirp à utiliser, le temps nécessaire au signal pour traverser le filtre est estimé par l'équation 3.1 :

$$F_{\min} = \frac{1}{nb.bascules * fréquence.scan.shift} \quad (3.1)$$

où *nb.bascules* représente le nombre de bascules du registre à balayage et *fréquence.scan.shift* la fréquence de balayage du même registre.

## 3.3 Filtre RIF

La description de la fonction des filtres RIF, l'implémentation et la validation des modèles Matlab et Verilog sera présentée dans la section suivante. Ces types de circuit ont été choisis à cause de la simplicité de leur structure et la facilité avec laquelle des filtres de taille différente peuvent être générés.

### 3.3.1 Description du filtre RIF

Un filtre numérique est un dispositif qui effectue des calculs sur une suite de nombres. Le filtre à réponse impulsionnelle finie (RIF) est le dispositif avec lequel toutes les simulations de pannes ont été effectuées. Il s'agit d'un filtre linéaire discret invariant dans le temps constitué par l'interconnection d'éléments de délai, d'additionneurs et de multiplicateurs. On peut exprimer la sortie du filtre  $y(n)$ , comme le produit de convolution de l'entrée du filtre  $x(n)$  et de la réponse impulsionnelle  $h(n)$  (équation 3.2).

$$y(n) = \sum_{k=0}^{M-1} h(k)x(n-k) \quad (3.2)$$

où  $M$  représente le nombre de coefficients. Il possède un nombre fini de coefficients, donc il travaille sur un horizon fini. Les filtres RIF sont toujours stables.

### 3.3.2 Modélisation du filtre RIF sur *Matlab/Simulink*

Les filtres RIF ont été modélisés à l'aide du logiciel *Matlab/Simulink* et *System Generator*. *System Generator* est un outil de conception à haut niveau qui sert à concevoir des modules de DSP en utilisant les FPGA. C'est ce même outil qui génère nos bancs d'essai utilisés par notre simulateur logique pour effectuer l'estimation de couverture de pannes. Le banc d'essai généré était composé de deux fichiers, un comprenant les vecteurs d'entrée et l'autre les vecteurs de sortie. Ces valeurs sont respectivement lues à partir des convertisseurs à l'entrée et à la sortie du filtre. Afin d'éviter d'alourdir le mémoire, la notation *Matlab* seul sera utilisé au lieu de *Matlab/Simulink*. L'outil *Matlab* fonctionne en utilisant des nombres en format double. Cependant, les blocs utilisés pour modéliser le filtre sont en format point fixe. C'est pourquoi des convertisseurs de nombre ont été ajoutés à l'entrée et à la sortie du filtre. La figure 14 représente le filtre RIF transposé à 4 étages modélisé.

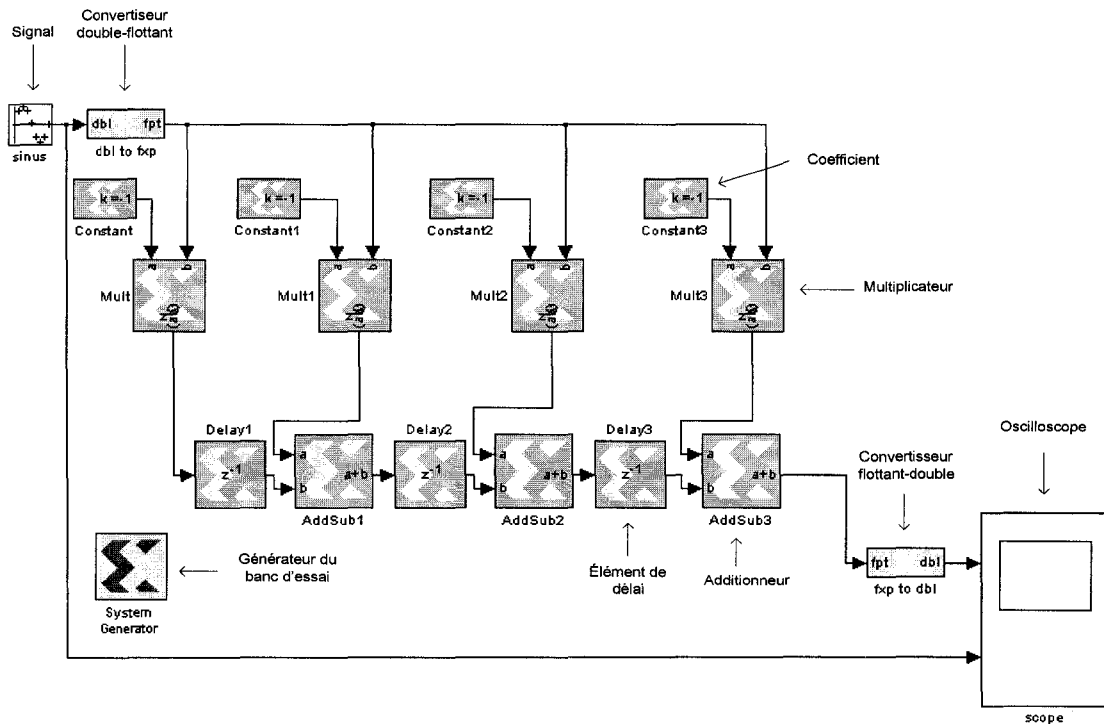


Figure 14 Modélisation d'un filtre RIF transposé à 4 étages

### 3.3.3 Validation des modèles Matlab et Verilog

Après avoir été modélisé avec l'outil Matlab, les filtres ont été conçus en langage Verilog. On leur applique ensuite les vecteurs d'entrée générés par Matlab. On compare la sortie obtenue du filtre en Verilog avec celle qui avait été obtenue avec le modèle Matlab. Si les sorties des filtres sont identiques, les modèles sont équivalents.

La figure 15 représente le résultat de simulation des vecteurs générés par Matlab sur le filtre en format Verilog. Comme on peut l'observer, les signaux de sortie des filtres Matlab (figure 12) et Verilog (figure 15) sont semblables. Cependant, une observation visuelle n'est pas suffisante pour confirmer que les deux filtres sont identiques.

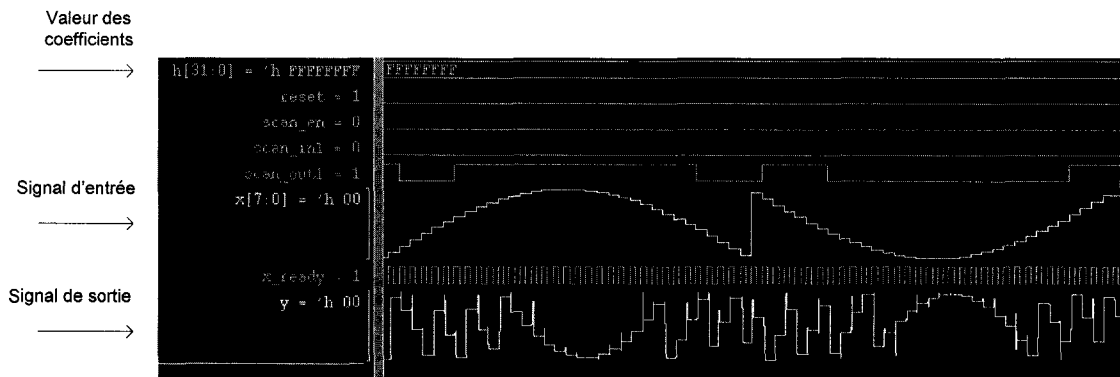


Figure 15 Simulation du filtre RIF à 4 étages modélisé en Verilog

Le simulateur logique Sim\_atpg est utilisé pour comparer les valeurs de sortie du filtre en Verilog avec les valeurs de sorties obtenues avec le modèle *Matlab*. Le résultat de la comparaison donne le résultat de la figure 16.

Debut	
OK	1
OK	2
OK	3
OK	4
OK	5
OK	6
OK	7
fin	

Figure 16 Comparaison entre la sortie des filtres Verilog et Matlab

où :

- OK : les sorties des filtres sont identiques;
- Mismatch : les sorties des filtres sont différentes.



Si pour tous les vecteurs appliqués au circuit on obtient de résultat ‘OK’ on conclut que les deux filtres sont identiques. Le numéro du vecteur est inscrit à côté de chaque sortie dans le but de savoir quel vecteur aurait provoqué une erreur.

### 3.4 Nombre de vecteurs par séquence de test

Le nombre de vecteurs fonctionnels composant la séquence a été choisi pour être équivalent au temps d’un vecteur de type balayage. Le choix d’utiliser un seul vecteur scan a été fait empiriquement. Puisque notre étude est basée sur les résultats de Butler et Saxena (2000), nous allons utiliser les valeurs de fréquence d’horloge utilisée dans leur étude :

- fréquence de balayage du registre de test (*fréquence.scan.shift*) : 10 MHz;
- temps de capture de la chaîne de scan : 35 ns.

Nous devons aussi tenir compte du nombre de bascules constituant le circuit. On en vient à proposer l’équation (3.3) suivante pour trouver le nombre de vecteurs fonctionnels qui peuvent être appliqués durant l’espace d’un vecteur scan :

$$nb.vect.fonct = \frac{nb.bascules \times fréquence.scan.shift}{temps.capture} \quad (3.3)$$

Le tableau II contient le nombre de vecteurs fonctionnels composant chaque séquence pour chacun des filtres utilisés selon l’équation 3.3.

Tableau II

Nombre de vecteurs fonctionnels composant  
les séquences de test

Filtre	Nb. bascules	Nb. vecteurs
4	24	68
7	48	137
15	112	320
25	192	548
50	392	1120
100	792	2262

Nous devons aussi tenir compte de l'initialisation de nos circuits avant de procéder à l'étape du test. Donc, pour chaque séquence de vecteurs, le premier vecteur sera un vecteur d'initialisation ou 'reset'.

### 3.5 Conclusion

Ce chapitre avait pour but de présenter les signaux utilisés pour effectuer les simulations de panne. Ces signaux seront échantillonnés pour pouvoir être utilisé par un simulateur de panne.

Le nombre de vecteurs utilisés pour chaque taille de filtre différente a été calculé. Il s'agit ici du nombre de vecteurs correspondant au temps d'un vecteur à balayage et ne représente pas le nombre de vecteur optimal.

## **CHAPITRE 4**

### **INFRASTRUCTURE LOGICIELLE**

#### **4.1 Introduction**

Après avoir décrit les signaux utilisés comme stimuli ainsi que les filtres d'intérêt dans le chapitre précédent, nous nous intéressons aux programmes qui seront mis à profit pour : 1) estimer la couverture de panne obtenue en utilisant les vecteurs fonctionnels; 2) réduire le temps de test. L'utilisation, le fonctionnement et l'ordre d'appel de chacun de ses programmes seront expliqués dans ce chapitre.

Nous débutons ces descriptions par un simulateur de pannes, Sim\_atpg, développé précédemment pour la simulation des pannes de type court-circuit et qui a dû être modifié pour cibler les types de pannes d'intérêt pour ce projet.

#### **4.2 Simulateur sim\_atpg**

Le simulateur de panne sim\_atpg (Hariri , 2002) est un simulateur de pannes maison développé à l'ÉTS. Il commence par créer une liste des nœuds valides à partir de la liste d'interconnexions, c.-à-d. tous les nœuds qui ne sont pas des entrées ou sorties et qui ne sont pas des lignes d'alimentation ou de mise à la terre. Ensuite, il utilise un simulateur logique commercial (Verilog\_XL) pour effectuer les simulations lorsqu'une panne est injectée dans un circuit et vérifier si le comportement du circuit en est affecté.

La structure du simulateur sim\_atpg s'est avérée parfaitement compatible avec la simulation des pannes de types collé-à et transition. Cette structure contient trois modules principaux, soit : le module de création de la liste des nœuds à simuler, le module d'injection de la panne et le module de simulation logique. Si une erreur dans la

sortie du circuit est détectée lors de la simulation logique (avec pannes), le nœud sera considéré comme couvert pour la panne injectée. La conception des modules prenant en charge ces deux nouveaux modèles de panne à considérer sera nécessaire et ces modules seront par la suite intégrés au simulateur.

### **4.3 Module de panne collé-à**

Afin d'effectuer l'estimation de couverture de panne collé-à, certaines modifications doivent être apportées à la liste d'interconnexions (netlist) du circuit à tester pour tenir compte de l'effet de la panne sur le nœud à ciblé. Ces modifications seront décrites ainsi que la méthode pour détecter les pannes à la sortie du circuit.

#### **4.3.1 Détection d'une panne collé-à**

Rappelons que pour vérifier qu'un nœud est couvert par le modèle de panne collé-à, on doit modifier la description du circuit afin de forcer la valeur de ce site, soit à 0 ou 1. On génère ensuite des vecteurs de test afin d'envoyer la valeur logique opposé à la valeur de panne sur le nœud testé. On observe le résultat obtenu aux sorties primaires (incluant les entrées des bascules formant la ou les chaînes de balayage) du circuit et on le compare avec la réponse du circuit sans panne. Si les résultats sont les mêmes, la panne n'est pas couverte. S'ils ne le sont pas, la panne est couverte. Cette comparaison entre la réponse du circuit sans panne et du circuit sous test est effectuée par le simulateur *sim\_atpg*. Cette même méthode avait été utilisé pour la validation des modèles *Matlab* et *Verilog* précédemment (chapitre 3.3.3).

### 4.3.2 Modification apportées au netlist pour le modèle de panne collé-à

Les listes d'interconnexions des filtres sont en langage Verilog. Ce langage nous permet de forcer la valeur que l'on désire sur un nœud avec la commande 'assign'. Par exemple, si l'on désire collé-à 1 le nœud 'cdsNet3024', la commande est :

```
> assign cdsNet3024='b1;
```

Cette commande est être ajoutée à la liste d'interconnexions du filtre pour chaque simulation de panne que l'on effectue. La figure 17 illustre les modifications qui sont apportées, pour chaque simulation, à la liste d'interconnexions pour les simulations collées-à.

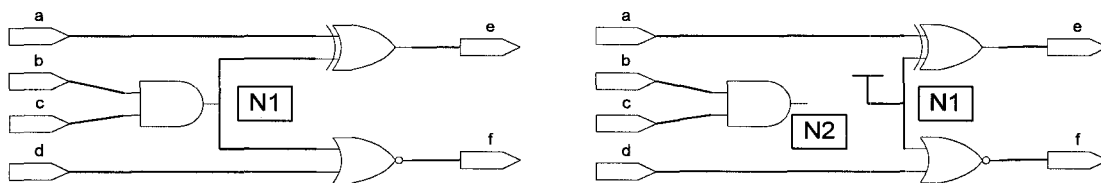


Figure 17 Modification à liste d'interconnexions pour les simulations de panne collé-à

Supposons que l'on veut tester le nœud N1 du circuit de gauche de la figure 17 pour le modèle collé-à 1. Les étapes suivantes sont effectuées :

- on coupe le nœud N1 de la porte qui l'alimente;
- on renomme la sortie de la porte qui alimentait le nœud N1 avec le nom d'un nœud fictif (ex. N2);
- on connecte le nœud N1 à la ligne d'alimentation.

Le circuit de droite de la figure 17 représente la topologie du circuit qui sera simulé. Lorsque la simulation est terminée, on revient au schéma initial. Cette modification doit être apportée à la liste d'interconnexions pour chaque nœud à tester. Si l'on désire tester le modèle de panne collé-à 0, le N1 sera connecté à la mise à terre.

#### **4.4 Création du module de simulation des pannes transitions**

Pour effectuer l'estimation de couverture de panne des modèles transitions TDF et IRF, un autre module a dû être conçu. Contrairement au modèle de panne collé-à, l'estimation de la couverture de panne transition ne requiert aucune simulation supplémentaire puisque les résultats obtenus des simulations du modèle de panne collé-à sont réutilisés à cette fin. Le principe de fonctionnement du module de couverture de transition est expliqué ci-après.

##### **4.4.1 Détection d'une panne transition sur un site à tester**

Rappelons que pour qu'un site soit couvert pour une transition, on doit être capable de provoquer et d'observer cette transition, soit montante et/ou descendante, par 2 vecteurs consécutifs. Pour se faire, on doit trouver une paire de vecteurs consécutifs qui couvrent le site collé-à 0 suivi de collé-à 1 dans le cas de la transition descendante et de collé-à 1 suivi de collé-à 0 pour une transition montante. On doit donc effectuer les simulations de panne collé-à 0 et 1 pour un certain site avant de pouvoir vérifier s'il est couvert transition IRF et/ou TDF. L'algorithme du programme se trouve à la figure 18. Il sera expliqué en détail pour le cas d'une transition descendante.

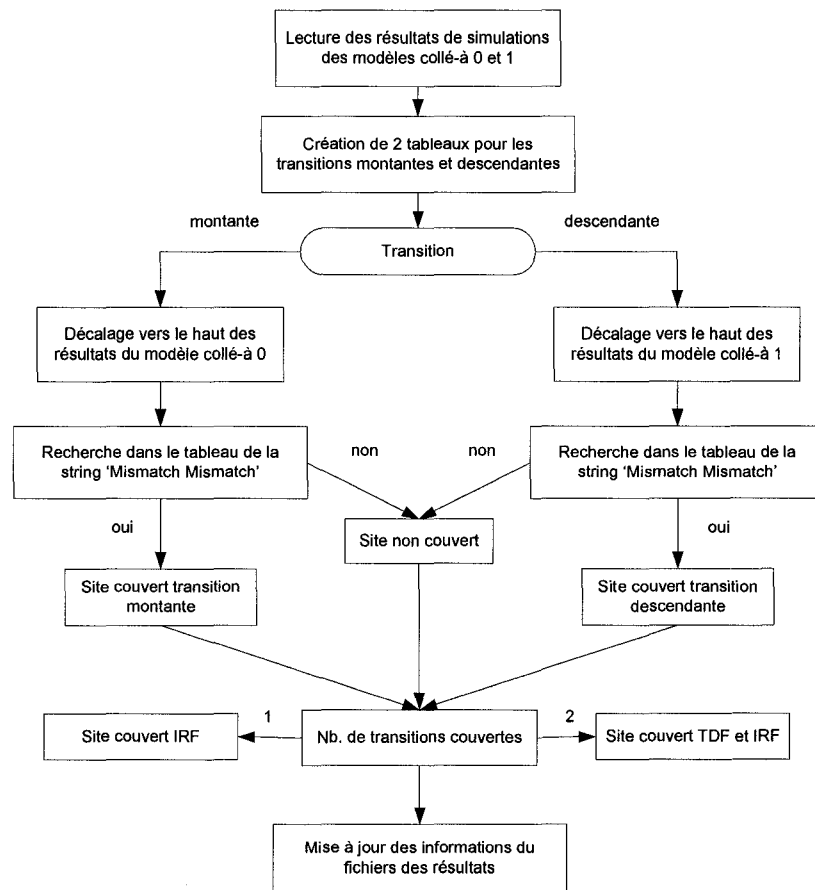


Figure 18 Flot de simulation de panne transition

Fonctionnement du programme :

- Le programme commence par effectuer la lecture des résultats des simulations collées-à 0 et 1 et les sauvegarde dans 2 tableaux, qui correspondent aux 2 transitions à couvrir (montante et descendante).

#### Pour les transitions descendantes

- Pour former les paires de vecteurs consécutifs, on décale d'une ligne vers le haut les résultats de la simulation collé-à 1 comme montré à la figure 19.

<i>Avant décalage</i>		<i>Après décalage</i>	
collé-à 0	collé-à 1	collé-à 0	collé-à 1
OK 1	OK 1	OK 1	OK 2
OK 2	OK 2	OK 2	Mismatch 3
OK 3	Mismatch 3	OK 3	...
Mismatch 4	...	Mismatch 4	OK 49
...	OK 49	...	OK 50
OK 50	OK 50	OK 50	

Figure 19 Contenu des tableaux transitions descendantes avant et après décalage

- Pour qu'un site soit couvert pour une transition, on recherche sur la même ligne la chaîne 'Mismatch Mismatch', ce qui signifie que le site est couvert collé-à 0 et 1 par 2 vecteurs consécutifs. Si la chaîne est trouvée, le site est considéré couvert et les numéros des vecteurs sont gardés en mémoire, dans le but de connaître le nombre de vecteurs fonctionnels optimal pour obtenir la couverture de panne maximale (chapitre 4.5).

#### Pour les transitions montantes

- on effectue les mêmes tâches que pour les transitions descendantes sauf que c'est la colonne qui contient les résultats de simulations du modèle collé-à 0 qui seront décalés vers le haut;
- lorsque nous possédons les résultats des 2 transitions, si un site est couvert par une ou l'autre des transitions, on dit qu'il est couvert par le modèle IRF et s'il est couvert par les 2 transitions, le site est couvert par les modèles IRF et TDF;
- on met à jour le contenu du fichier de résultat et on recommence la procédure pour le site suivant.

Comme expliqué précédemment, l'estimation de couverture de panne transition ne requiert aucune simulation supplémentaire, mais plutôt l'exécution de quelques



opérations, ce qui réduit considérablement le temps consacré à l'estimation de la couverture de panne transition.

#### 4.5 Format des fichiers de résultats

Les résultats de simulation sont enregistrés dans des fichiers en format texte. Les résultats pour les simulations de panne collé-à sont représentés à la figure 20. Dans ce cas-ci, il s'agit des résultats du filtre à 4 étages avec le signal du bruit blanc.

```

Nombre total des noeuds                = 933
Nombre total des noeuds valides        = 827
Nombre total des sites collé-à        = 1654
Nombre total des sites collé-à couverts = 1618 ---> Couverture
collé-à = 0.978235
Nombre total des sites collé-à 1 couverts = 807 ---> Couverture collé-
à 1 = 0.975816
Nombre total des sites collé-à 0 couverts = 811 ---> Couverture collé-
à 0 = 0.980653

```

Figure 20 Format du fichier de résultat du modèle collé-à

Pour le modèle de panne transition, le format du fichier résultat est représenté à la figure 21. Il s'agit encore des résultats du filtre à 4 étages avec le signal du bruit blanc.

```

Nb_sites_transitions_valides = 827
Nb_sites_transitions_TDF_couverts = 747
Nb_sites_transitions_IRF_couverts = 784
Nb_vecteurs_transitions_stuck_opt = 33
Nb_vecteurs_transitions_TDF_opt = 59
Nb_vecteurs_transitions_IRF_opt = 53

```

Figure 21 Format du fichier de résultat du modèle transition

où :

- Nb\_vecteurs\_transitions\_stuck\_opt : nombre de vecteurs fonctionnels optimaux pour atteindre la couverture de panne maximale pour le modèle collé-à;

- `Nb_vecteurs_transitions_TDF_opt` : nombre de vecteurs fonctionnels optimaux pour atteindre la couverture de panne maximale pour le modèle transition TDF;
- `Nb_vecteurs_transitions_IRF_opt` : nombre de vecteurs fonctionnels optimaux pour atteindre la couverture de panne maximale pour le modèle transition IRF;

#### **4.6 Détermination du nombre de vecteurs fonctionnels utiles**

Rappelons que le choix d'utiliser le nombre de vecteurs fonctionnels que l'on peut appliquer pendant l'espace d'un vecteur scan a été fait empiriquement. On s'attend donc à ce que la séquence de vecteurs ne soit pas tout à fait optimale. De ce fait, certains vecteurs constituant la séquence de test pourraient être appliqués au circuit inutilement ce qui prolongerait le temps de test et n'améliorerait pas la couverture de panne. Certaines techniques ont été étudiées dans le but de connaître le nombre de vecteurs fonctionnels utiles pour réduire le temps de test. Les options envisagées sont :

- **Option 1 : Compression de la séquence de tests.** Lors de la compression d'une séquence de vecteurs, le programme qui l'effectue essaie de réarranger l'ordre des vecteurs ou d'en enlever dans le but de trouver ceux qui détectent le plus de sites et de les appliquer en premier. Ainsi, certains vecteurs ne seront plus utiles et on pourra réduire la séquence. Cependant, cette façon de faire n'est pas applicable dans notre cas. Le réarrangement de la séquence initiale détruirait l'intégrité du signal qui a été échantillonné pour effectuer le test. La figure 22 illustre cette situation.

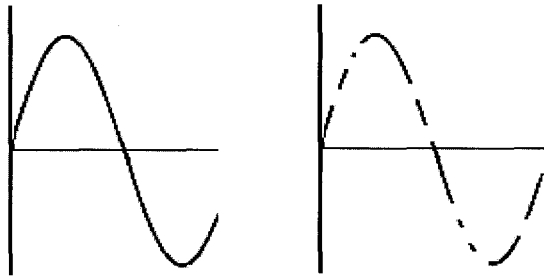


Figure 22 Résultat d'un réarrangement de la séquence de test initiale

• Option 2 : Création de sous-séquences de vecteurs. On décompose la séquence initiale en sous-séquences de 10 vecteurs, avec une incrémentation de 10 entre chacune d'elles (10, 20, 30, ... vecteurs). On effectue la simulation de panne de chacune d'entre elles et on compare la couverture de panne obtenue avec celles de la séquence initiale (figure 23). Lorsque les couvertures de panne seront les mêmes, on aura trouvé le nombre de vecteurs fonctionnels pour obtenir une couverture de panne maximale. Cette méthode de travail, qui a été appliquée dans le cas des filtres à 4 et 7 étages, comporte plusieurs problèmes.

- 1) On doit faire la simulation de panne de la séquence initiale (séquence complète).
- 2) On doit effectuer la simulation de panne de chaque sous-séquence de vecteurs.
- 3) On n'obtient pas le nombre précis de vecteurs fonctionnels optimaux, mais seulement qu'une approximation (soit 10, 20, 30, ... vecteurs).

Le nombre élevé de simulations à être effectuées pose un problème majeur. Il est vrai que dans le cas du filtre à 4 étages, où les simulations sont effectuées assez rapidement, cela n'est pas problématique. Mais dans le cas de filtre très gros, le très grand nombre de simulations à être effectuées va empêcher d'obtenir un résultat dans un temps convenable.

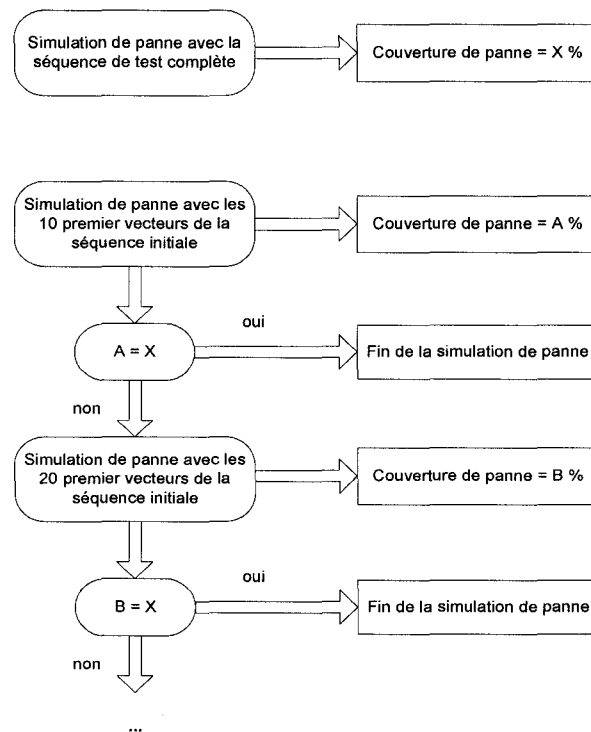


Figure 23 Méthode de réduction de vecteurs avec plusieurs simulations plus petites

- Option 3 : Détection du premier ‘Mismatch’ lors de la simulation. Qu’un site soit couvert par 1 ou plusieurs vecteurs n’est pas vraiment important. Ce que l’on veut, c’est qu’il le soit le plus rapidement possible, pour qu’on puisse passer au site suivant.

Supposons 2 nœuds fictifs, soit N1 et N2, qui subissent une simulation collé-à 0 par une séquence de vecteurs fonctionnels comportant 100 vecteurs. Le résultat de la simulation se trouve à la figure 24.

N1	N2
OK 1	OK 1
OK 2	OK 2
...	Mismatch 3
Mismatch 45	...
OK46	OK 100
...	
Mismatch 67	
...	
OK 100	

Figure 24      Résultat de simulation collé-à de 2 noeuds fictifs

Dans le cas du nœud N1, le premier Mismatch se produit au vecteur 45. Donc, pour que le site N1 soit couvert collé-à 0, seulement les 45 premiers vecteurs sont nécessaires. Pour le nœud N2, cela nécessite que 3 vecteurs pour être couvert.

En utilisant une séquence qu'à 3 vecteurs, on ne couvre que le nœud N2, mais en utilisant les 45 premiers vecteurs de la séquence, les nœuds N1 et N2 sont couverts, d'où le principe de détecter les premiers Mismatch de chaque nœud. Parmi tous ces premiers Mismatch, celui qui se produira le plus tardivement sera le nombre de vecteurs fonctionnels optimaux pour le filtre en question. Le même principe s'applique dans le cas des modèles transitions pour trouver le nombre de vecteurs fonctionnels optimaux.

Un programme, appelé 'stuck\_opt' et écrit en langage C, a été conçu dans le but d'effectuer cette tâche. Il est appelé par le simulateur sim\_atpg, après chaque simulation collé-à 0 et 1. L'algorithme du programme se trouve à la figure 25.

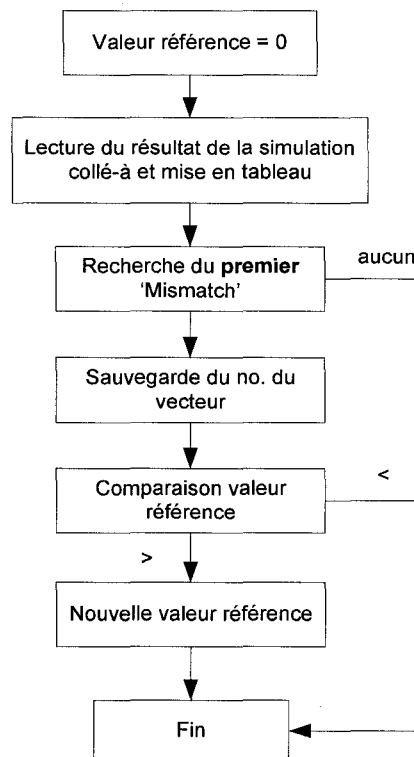


Figure 25 Algorithme du programme stuck\_opt

Des programmes ont été conçus dans le but de trouver le nombre de vecteurs fonctionnels utiles selon les options 2 et 3. L'option 2 a vite été abandonnée puisque le nombre de simulations supplémentaires à effectuer ne justifiait pas le gain en temps de test obtenu. L'option 3 s'est montrée comme étant la plus efficace, ne demandant aucune simulation supplémentaire, mais seulement l'exécution de quelques opérations sur les fichiers de résultats, n'augmentant pas le temps de test de façon significative.

#### 4.6 Ordre d'appel des différents programmes de simulation de panne

Comme vu précédemment, pour vérifier qu'un site est couvert par les modèles de pannes transitions, on doit déjà connaître les résultats de simulation de panne des modèles collé-à 0 et 1.

Pour ce qui est de trouver le nombre de vecteurs fonctionnels optimaux, le programme doit être appelé après chaque simulation du modèle collé-à 0/1. Donc, l'ordre dans lequel seront appelés tous ces programmes se trouve à la figure 26.

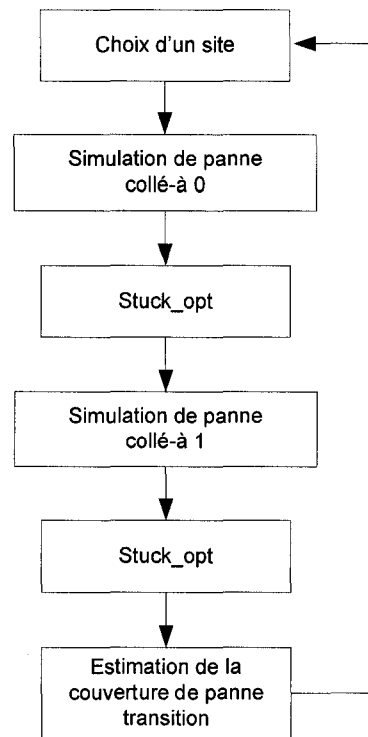


Figure 26 Ordre d'appel des programmes pour les simulations collé-à et transition

#### 4.7 Génération des vecteurs de type balayage

Pour pouvoir obtenir une couverture de panne maximale, nous avons proposé d'utiliser des vecteurs de type balayage pour couvrir les sites que les vecteurs fonctionnels n'auraient pas couverts. Pour ce faire, les outils de Mentor Graphics ont été utilisés. L'exemple proposé est fait avec le modèle de panne collé-à. Le même principe s'applique aux modèles de panne transition.

#### 4.7.1 Génération de la liste des sites non couverts

Lors des simulations de pannes, Sim\_atpg génère une liste des sites qui n'ont pas été couverts par les vecteurs fonctionnels. Cependant, cette liste n'est pas directement interprétable par les outils de Mentor Graphics et certaines modifications doivent y être apportées (figure 27).

```
cdsNet2619 /U835/op
cdsNet6669 /U764/op
cdsNet1857 /U661/op
cdsNet7591 /U566/op
cdsNet2711 /U496/op
cdsNet8679 /U738/op
cdsNet8959 /U714/op
cdsNet146  /U669/op
```

Figure 27 Modifications à la liste des noms des sites à tester

L'outil Sim\_atpg travaille avec le nom des nœuds du circuit (colonne de gauche de la figure 27), alors que les outils de *Mentor Graphics* fonctionnent avec le nom de la porte ainsi que de la broche auquel le nœud est attaché (colonne de droite de la figure 27). Un programme a été conçu et intégré au simulateur Sim\_atpg pour pouvoir effectuer cette conversion. Avec ces informations en main, on peut créer notre liste de sites non couverts pour la génération de vecteurs de type balayage. La figure 28 représente le format de la liste final utilisé par *Mentor Graphics*.



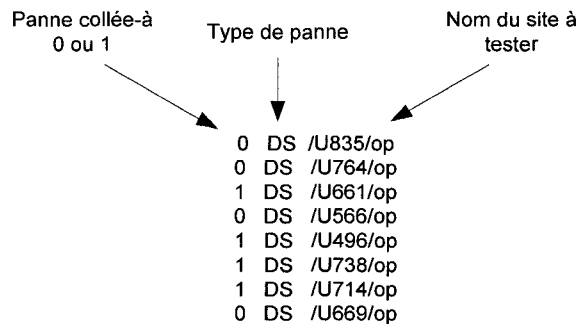


Figure 28 Format de la liste de panne utilisé par *Mentor Graphics*

Pour le modèle de panne transition, les étapes sont les mêmes sauf qu'on doit signifier à l'outil de générer des vecteurs pour le modèles transition au lieu du modèle collé-à.

#### 4.8 Vérification du fonctionnement des programmes

Divers programmes ont été conçus, mais on doit s'assurer qu'ils fonctionnent correctement. Il n'existe présentement pas de logiciels équivalents à ceux que nous avons faits pour vérifier la validité de nos résultats. Dans le cas de chaque programme, ils ont premièrement été appliqués au filtre à 4 étages, puisqu'il ne nécessitait qu'une séquence à 68 vecteurs.

Les résultats des simulations de pannes étaient vérifiés à la main, pour un certain nombre de nœuds, et comparée au résultat que donnait le simulateur. Par exemple, si le simulateur de panne collé-à nous indiquait que le site était couvert, on vérifiait dans le fichier résultat à la main qu'il l'était bien. Lorsque nous avons confirmé le fonctionnement des programmes au filtre à 4 étages, ils ont été appliqués sur les filtres plus gros.

Cette façon de faire, soit à la main, était lente, mais a permis d'approuver le fonctionnement des programmes.

#### **4.8 Conclusion**

Ce chapitre avait pour but de décrire le fonctionnement des programmes qui ont été conçus dans le but d'effectuer les simulations de couverture de pannes des modèles collé-à et transition à l'aide de vecteurs fonctionnels.

Les programmes ont tous été faits en langage C pour pouvoir être intégré facilement au simulateur `sim_atpg`. Le simulateur `sim_atpg`, utilisé pour les simulations de pannes, a été conçu par Yassine Hariri. Il a aussi été d'une très grande aide pour la création du module de simulation de panne collé-à et a intégré tous les modules dans le simulateur `sim_atpg`.

L'ordre dans lequel les programmes doivent être appelés a été présenté et correspond à la méthodologie de test avec les vecteurs fonctionnels que l'on propose. Des méthodes qui ont pour but de réduire le temps de test ont également été présentées.

Dans le chapitre suivant, les résultats obtenus à l'aide de nos modules de simulations de pannes seront présentés et discutés.

## **CHAPITRE 5**

### **RÉSULTATS ET ANALYSE**

#### **5.1 Introduction**

Dans ce chapitre, nous présenterons au départ les résultats de simulation de pannes obtenus à l'aide des modules de simulation des modèles collé-à et transition qui ont été ajoutés au simulateur `sim_atpg`. Différents stimuli de type fonctionnel sont utilisés pour générer ces premiers résultats.

Suite à ces résultats, des vecteurs de type scan seront ajoutés dans le but d'obtenir une meilleure couverture de panne. Nous pourrons ensuite calculer le gain en terme de temps et d'espace mémoire sur ATE par rapport à l'utilisation de vecteurs de type scan seulement.

Pour terminer, une méthode de réduction du temps de simulation de panne sera présentée. Cette méthode sera appliquée au filtre RIF à 100 étages.

#### **5.2 Résultats de simulation de panne du modèle collé-à**

Le filtre RIF à 4 étages est celui qui a été soumis au plus grand nombre de simulations de panne, soit avec les 4 signaux de base présentée au chapitre 3 (bruit blanc, chirp, impulsion, sinus) ainsi qu'avec 3 valeurs de coefficients différents. Puisqu'aucune valeur de référence n'est existante au niveau des valeurs de coefficients optimaux du filtre, les 2 valeurs extrêmes, soient "00" et "FF" ont été empiriquement choisis, ainsi qu'une autre valeur "55", qui représente une alternance entre 0 et 1 en binaire. L'objectif ici est d'identifier les meilleures combinaisons (signal, coefficients) possibles pour réduire les efforts de simulation avec les filtres plus gros.

Le tableau III contient les résultats de simulation du filtre RIF à 4 étages pour le modèle collé-à.

Tableau III

Résultats de simulation de pannes collé-à du filtre à 4 étages

	Couverture de pannes %		
Coefficient	"00"	"55"	"FF"
Bruit blanc	43.29	85.85	97.82
Chirp	43.29	85.79	95.71
Impulsion	42.19	51.59	60.22
Sinus	43.29	85.85	97.28

En première lecture des résultats, on remarque que :

- le filtre RIF avec ses coefficients à "FF" obtient les meilleures couvertures de panne pour chaque signal;
- le bruit blanc est le signal qui offre la meilleure couverture de panne, suivi de près par le sinus.

Lorsque la valeur des coefficients est à "00", ceci implique qu'après les multiplicateurs, les valeurs dans l'arbre d'additionneur du filtre se retrouveront toutes à la valeur 0. Il est donc très difficile par la suite de contrôler et observer un nœud, ce qui explique la couverture de panne très faible. À l'opposé de ce résultat, les coefficients composés seulement de 1 offre la meilleure couverture de panne puisque les multiplicateurs agissent comme composante passe tout et qu'un seul changement sur l'entrée du filtre produit un changement sur la sortie. Il est donc plus facile, par la suite, de contrôler et observer les nœuds qui composent l'arbre d'additionneur.

La bonne couverture offerte par le signal du bruit blanc s'explique par le fait que ce signal peut être considéré comme étant un signal pseudo-aléatoire. Et comme discuté au chapitre 2, les signaux pseudo-aléatoires offrent une bonne couverture de panne pour le genre de circuits considérés dans ce projet. Pour les signaux du chirp et sinus, même si ces signaux offrent de bonnes couvertures de panne, elles sont quand même inférieures à celle du bruit blanc. Le signal de l'impulsion n'étant constitué que d'un seul vecteur, ceci explique sa couverture de panne très basse.

Par conséquent, les simulations futures seront effectuées qu'avec un seul coefficient, soit "FF" puisque les 2 autres coefficients utilisés donnaient de moins bons résultats. Pour la même raison, les signaux du chirp et de l'impulsion seront laissés de côté. Le prochain ensemble de résultats vise à déterminer le choix final quant au signal à appliquer pour les très gros filtres.

Tableau IV

Résultats de couverture de pannes collé-à du sinus et du bruit blanc

Nb. d'étages	Couverture de pannes %	
	Sinus	Bruit blanc
4	97.28	97.82
7	97.56	98.02
15	98.15	98.15

Le tableau IV contient les résultats de simulations de pannes collé-à du sinus et du bruit blanc, pour des filtres à 4, 7 et 15 étages.

On remarque que le bruit blanc offre dans les 3 cas une couverture de pannes légèrement supérieure ou égale à celle du sinus.

Par conséquent, afin d'utiliser le plus efficacement possible le temps de simulation à notre disposition et puisque le bruit blanc offre en général une meilleure couverture de panne que le sinus, le bruit blanc sera le seul signal utilisé dans les simulations des filtres à 25, 50 et 100 étages.

Tableau V

Résultats de simulation du modèle collé-à avec le signal du bruit

Nb. étages	Couverture de pannes %	Nb. vecteurs fonctionnels utiles	Prop. vect. fct. p/r à la séquence initiale %
4	97.82	33	48.53
7	98.02	86	62.77
15	98.15	93	29.06
25	94.86	201	36.68
50	94.31	417	37.23
100	92.17	867	38.33

où :

- Prop. vect. fct. p/r à la séquence initiale : Proportion de vecteurs fonctionnels utiles par rapport à la séquence initiale pour effectuer la couverture de pannes du filtre, c.-à-d. le nombre de vecteurs contribuant à abaisser le nombre de pannes non-couvertes sur le nombre de vecteurs fonctionnels dont le temps d'application correspond au temps d'un vecteur de type balayage.

Le tableau V contient les résultats des simulations de pannes du modèle collé-à pour le signal du bruit blanc ainsi que le nombre de vecteurs fonctionnels qui ont été nécessaires pour obtenir la couverture de panne, et ce pour toutes les tailles de filtres RIF.

À la vue de ces résultats, on remarque que :

- la couverture de panne augmente très légèrement du filtre à 4 étages jusqu'au filtre à 15 étages et pour ensuite tomber du filtre à 25 étages jusqu'au filtre à 100 étages;
- en général, le nombre de vecteurs fonctionnels utiles est légèrement supérieur au tiers de la séquence initiale;
- la proportion de vecteurs fonctionnels utiles connaît une forte hausse dans le cas du filtre à 7 étages.

Normalement, on s'attend à ce que la couverture de panne diminue lorsque la complexité des circuits augmente. Cependant, du filtre à 4 étages jusqu'aux filtres à 15 étages, elle augmente. Une analyse du nombre de vecteurs à balayage collé-à nous permet d'expliquer ce résultat.

Tableau VI

Nombre de vecteurs à balayage pour le modèle collé-à

Nb. Étages	Nb. vect. balayage aléatoires	Nb. vect. balayage déterministes	Proportion vect. balayage aléatoire	Couverture de panne fonctionnelle %	Couverture de panne balayage %
4	68	16	80.95	97.82	100
7	86	13	86.87	98.02	100
15	123	14	89.78	98.15	100
25	135	40	77.14	94.86	100
50	168	122	57.93	94.31	100
100	190	267	41.57	92.17	100

Le tableau VI contient les résultats de simulation de panne du modèle collé-à avec l'utilisation des vecteurs à balayage seulement.

On remarque que du filtre 4 étages jusqu'au filtre à 15 étages, la proportion du nombre de vecteurs à balayage aléatoires augmente, pour ensuite diminuer jusqu'au filtre à 100 étages.

Les couvertures de panne offertes par les vecteurs à balayage étant toutes à 100%, il n'est pas possible de tirer de conclusion par rapport au nombre de vecteurs à balayage aléatoires et leur couverture de panne. Cependant, les augmentations et diminutions dans la proportion du nombre de vecteurs à balayage aléatoires correspondent aux variations des couvertures de pannes obtenues avec les vecteurs fonctionnels seulement.

À la vue de ces résultats, on peut conclure que plus la proportion du nombre de vecteurs à balayage aléatoire est importante, plus on peut s'attendre à obtenir une couverture de panne élevée en utilisant des vecteurs aléatoires.

Pour ce qui est de la proportion de vecteurs fonctionnels utiles élevée du filtre à 7 étages, leur forte variation ne peut être reliée au nombre de vecteurs à balayage aléatoires et déterministes appliqué au circuit puisqu'il n'y a pas de relation qui peut être établie entre les résultats. Une analyse au niveau du nombre de vecteurs fonctionnels utiles et du nombre de cellule de chaque filtre a été faite dans le but d'investiguer le problème. Les résultats sont présentés dans le tableau VII.



Tableau VII

Augmentation du nombre de vecteurs et de cellules entre chaque filtre

Nb. étages	Nb. vect. fonctionnels utiles	Nb. Cellules	Augmentation relative du nb. vecteurs utiles %	Augmentation relative du nb. cellules %
4	33	901	-	-
7	86	1640	161	85
15	93	3520	8.1	114.6
25	201	6470	116	83.8
50	417	13100	107	102.5
100	867	26400	108	101.5

En observant les résultats du tableau VII, on observe une très grande diminution dans la proportion de vecteurs fonctionnels utiles pour le filtre à 15 étages. Dans le cas des cellules, on remarque une fluctuation de filtre en filtre, où aucune conclusion ne peut être tirée.

Un retour sur les résultats obtenus lors de l'utilisation de sous séquence de test présenté à la section 4.5 permet de clarifier la situation. Les résultats obtenus lors de la simulation de panne avec les sous séquences sont présentés dans le tableau VIII.

Tableau VIII

Nombre de sites non couverts pour le bruit blanc  
avec l'utilisation de sous séquences  
pour le filtre à 7 étages

Nb. vect. fct.	Nb. de sites non couverts
10	517
20	152
30	88
40	70
50	70
60	70
70	69
80	69
90	60

On remarque que :

- de 40 à 80 vecteurs, un seul site est couvert;
- de 80 à 90 vecteurs, 9 sites sont couverts.

Ces résultats nous indiquent qu'une certaine partie du filtre est plus difficile à tester et explique le résultat obtenu dans le tableau III. Si on ne tient pas compte de ces sites plus problématiques, soit en utilisant une séquence de 40 vecteurs, qui représente 29% de la séquence initiale, on obtient un résultat comparable aux autres filtres.

### 5.3 Résultats des estimations de panne du modèle transition

Le tableau IX présente les résultats des estimations des couvertures de panne des modèles transition, soit TDF et IRF. Le bruit blanc a été utilisé comme stimuli. Les

résultats obtenus avec le modèle collé-à, présentés au tableau III, y apparaissent également pour fins de comparaison.

Tableau IX

Résultats des estimations de couverture de pannes pour les modèles

Nb. étages	Couverture de pannes %		
	TDF	IRF	Collé-à
4	90.33	94.8	97.82
7	94.73	95.52	98.02
15	95.84	96.03	98.15
25	90.06	90.14	94.86
50	89.43	89.85	94.31
100	87.15	87.6	92.17

On observe que :

- le modèle IRF donne une meilleure couverture de panne que le modèle TDF. Puisqu'une seule panne doit être couverte sur les 2 possibles sur chaque site, les conditions à remplir sont plus faciles à atteindre;
- en général, la couverture de panne diminue avec la complexité des circuits;
- la couverture de panne des modèles transition est moins élevée que pour le modèle collé-à.

La couverture de panne transition des filtres est élevée contrairement à ce qui est répertorié dans la littérature comme discuté au chapitre 2. C'est la structure du filtre RIF qui permet d'obtenir d'aussi bon résultat. Rappelons que cette structure est composée d'un arbre d'additionneurs et de multiplicateurs. Elle ne contient aucune boucle de retour ce qui permet aux données qui parcourent le filtre de passer très facilement de l'entrée vers la sortie. Ces données sont donc très faciles à contrôler et observer

contrairement, par exemple, à une structure de mémoire ou de microprocesseur dont les données peuvent voyager dans les deux sens sur certains bus, ce qui rend le travail de contrôle et d'observation des données plus difficile. Lorsque les coefficients des multiplicateurs du filtre sont tous à 1, on peut modifier les valeurs des nœuds constituant l'arbre d'additionneur du filtre très facilement. Il est donc plus facile par la suite de trouver des paires de vecteurs pour tester les nœuds du circuit. Cependant, on peut supposer qu'en appliquant les vecteurs fonctionnels à une structure différente que celle du filtre RIF, les résultats seraient moins élevés.

Tableau X

Nombre de vecteurs fonctionnels pour les modèles de panne transition

Nb. étages	Nb. vecteurs fonctionnels utiles		Prop. vect. fct. p/r à la séquence initiale %	
	TDF	IRF	TDF	IRF
4	59	53	86.76	77.94
7	109	107	79.56	78.1
15	298	290	93.12	90.62
25	528	341	96.35	62.23
50	705	555	62.95	49.55
100	1254	1017	55.44	44.96

Le tableau X contient le nombre de vecteurs fonctionnels qui ont été nécessaires pour obtenir la couverture de panne des modèles de panne transition.

On remarque que :

- le modèle IRF nécessite moins de vecteurs fonctionnels que le modèle TDF;
- pour les gros filtres (25, 50 et 100 étages), la proportion de vecteurs fonctionnels utiles diminue face à l'augmentation de la taille des filtres.

Plus la taille des filtres augmente, meilleurs sont les résultats avec le modèle transition IRF. Comme dans le cas du modèle collé-à, le nombre de vecteurs fonctionnels utiles se stabilise pour les gros filtres. On suppose donc qu'en général, seulement 50% des vecteurs fonctionnels de la séquence initiale sont nécessaires.

Comparons maintenant la couverture pour la panne transition obtenue à partir des 3 scénarios suivants : 1) l'application des vecteurs fonctionnels (bruit blanc) seulement, 2) l'application des vecteurs transition à balayage (de Fastscan) et, 3) la combinaison des vecteurs fonctionnels et à balayage. Le tableau XI contient ces différentes couvertures de pannes.

Tableau XI

Couverture de panne transition avec l'utilisation de vecteurs fonctionnels seulement, l'utilisation de vecteurs à balayage seulement, et l'utilisation des 2 types de vecteurs combinés

Nb. étages	Couverture de panne transition avec les vecteurs fct. seul %	Couverture de panne transition avec les vecteurs scan seul %	Couverture de panne transition avec vecteurs fct. et scan %
4	97.82	99.27	100
7	98.02	99.27	100
15	98.15	98.74	100
25	94.86	99.46	100
50	94.31	98.78	100
100	92.17	99.5	100

On remarque :

- les couvertures de pannes transition sont plus élevées avec l'utilisation des vecteurs à balayage seuls;

- les couvertures de pannes transition sont toutes à 100 % lorsque l'on combine les deux types de vecteurs.

Ces résultats nous indiquent que l'utilisation de la méthode de test proposée dans ce mémoire nous permet d'obtenir une couverture de panne à 100 % dans tous les cas étudiés, couverture qui n'a pas été atteinte par chacun des types de vecteurs utilisés séparément.

#### 5.4 Gain en temps de l'utilisation des vecteurs fonctionnels

Pour calculer le gain maximal en temps de notre méthode, on utilise l'équation 5.1.

$$G_{\max} = \frac{NVBT - (NVBAF + (NVFO/NVFT))}{NVBT} \quad (5.1)$$

où :

- $G_{\max}$  : Gain maximal;
- NVBT. : Nombre de vecteurs à balayage nécessaires pour couvrir le filtre sans l'utilisation des vecteurs fonctionnels (approche traditionnelle);
- NVBAF : Nombre de vecteurs à balayage nécessaires pour couvrir les sites non couverts par le test fonctionnel;
- NVFO : Nombre optimal de vecteurs fonctionnels, i.e. le nombre minimum de vecteurs fonctionnels pour obtenir la même couverture que la séquence initiale de vecteurs fonctionnels dont le temps d'application égal celui d'un vecteur à balayage;
- NVFT : Nombre total de vecteurs fonctionnels de la séquence initiale.

Le nombre de vecteurs à balayage sans l'utilisation du test fonctionnel (NVBT), le nombre de vecteurs à balayage avec l'utilisation du test fonctionnel (NVBAF) ainsi que le gain maximale ( $G_{\max}$ ) sont donnés au tableau XII. Le gain obtenu sans optimisation du nombre de vecteurs fonctionnels,  $G_{\text{sopt}}$ , apparaît également dans ce tableau. Ce

gain, qui est obtenu en utilisant le nombre total de vecteurs fonctionnels dont la durée correspond à celle d'un vecteur à balayage, est calculé en remplaçant le terme NVFO par le terme NVFT dans l'équation (5.1). La comparaison entre ces deux gains va permettre d'évaluer la pertinence d'optimiser le nombre de vecteurs fonctionnels.

Tableau XII

Gain en temps pour le modèle collé-à avec l'utilisation des vecteurs fonctionnels

Nb. étages	NVBT	NVBAF	Gmax %	Gsopt %	Gmax - Gsopt %
4	76	17	76.99	76.32	0.67
7	87	15	82.04	81.61	0.43
15	127	23	81.66	81.1	0.56
25	169	47	71.97	71.6	0.37
50	276	93	66.26	65.94	0.32
100	430	168	60.84	60.7	0.14

La différence entre le gain maximal et le gain obtenu avec l'utilisation d'un seul vecteur à balayage est très faible. Cette faible différence est normale puisque le gain maximal dans le meilleur des cas, ne nécessite l'utilisation que de 29.06 % de la séquence initiale de vecteurs fonctionnels. Et comme déjà discutés précédemment, les vecteurs fonctionnels nécessitent un temps d'application au circuit très court par rapport au vecteur à balayage.

Tableau XIII

Nombre de vecteurs à balayage pour les modèles transition sans et avec l'utilisation du test fonctionnel avec du bruit blanc

Nb. étages	Sans fct.	Avec fct.		Gain maximal %	
		TDF	IRF	TDF	IRF
4	149	34	26	76.6	82.03
7	176	37	30	78.53	82.51
15	259	41	39	83.81	84.59
25	357	86	86	75.64	75.74
50	516	76	76	85.15	85.22
100	768	153	151	80.01	80.28

Les résultats présentés au tableau XIII représentent le gain maximal en temps de l'utilisation des vecteurs fonctionnels.

En comparant les résultats des tableaux XII et XIII, on remarque que le gain obtenu par l'utilisation des vecteurs fonctionnels est plus grand, dans la plupart des cas, avec le modèle transition que le modèle collé-à. Les modèles transition étant composé d'une paire de vecteurs, ils nécessitent plus de temps à être appliqué à un circuit qu'un vecteur collé-à qui ne requiert qu'un seul vecteur. Chaque gain de vecteur transition nous donne donc une réduction de 2 vecteurs à balayage par rapport à un vecteur à balayage pour chaque vecteur collé-à enlevé.



### 5.5 Gain en mémoire sur ATE de l'utilisation des vecteurs fonctionnels

À la suite de l'obtention du nombre de vecteurs fonctionnels et à balayage qui sont nécessaires à la couverture de panne de nos filtres, on peut calculer la réduction au niveau de l'espace mémoire requis pour emmagasiner nos vecteurs de test.

Tableau XIV

Gain en mémoire pour le modèle collé-à avec le signal du bruit blanc

Nb. Étages	NBB	NBFB	Gain %
4	1824	672	63.16
7	4176	1408	66.3
15	14224	3320	76.66
25	32448	10632	67.23
50	108192	37480	65.36
100	340560	139992	58.89

où :

- NBB : Nombre de bits nécessaire pour emmagasiner les vecteurs à balayage sans l'utilisation des vecteurs fonctionnels;
- NBFB : Nombre de bits de mémoire requis pour les vecteurs fonctionnels et les vecteurs à balayage ajoutés.

Le tableau XIV contient le gain en mémoire que l'on obtient en utilisant les vecteurs fonctionnels. Excepté le filtre à 15 étages qui présente le meilleur gain, tous les autres filtres obtiennent des résultats similaires, soit autour de 65% et sont légèrement inférieurs pour le filtre à 100 étages dans le cas du bruit blanc.

Tableau XV

Gain en mémoire pour les modèles transition avec le signal du bruit blanc

Nb. Étages	NBB	NBFB		Gain %	
		TDF	IRF	TDF	IRF
4	7152	2104	1672	70.58	76.62
7	16896	4424	3736	73.82	77.89
15	58016	11568	11056	80.06	80.94
25	137088	37248	35752	72.83	73.92
50	404544	65224	62424	83.88	84.57
100	1216512	252384	247320	79.25	79.67

Les gains en mémoire pour les modèles transition se retrouvent dans le tableau XV.

On observe que :

- le modèle IRF offre un gain légèrement plus élevé que le modèle TDF;
- les modèles transition donnent des gains plus élevés que le modèle collé-à.

## 5.6 Réduction du temps de simulation de panne

La notion de réduction du temps de test a déjà été abordée dans le chapitre 4 avec la réduction du nombre de vecteurs constituant la séquence de test. Cependant, avec la très grande taille du filtre à 100 étages, la simulation pour estimer la couverture de panne était beaucoup trop longue. Une nouvelle solution s'imposait. L'option envisagée est la suivante :

- diviser la liste des nœuds à simuler en plusieurs plus petites dans le but de partager le travail qui serait normalement fait par un seul ordinateur par plusieurs machines. Une

première estimation du temps nécessaire pour effectuer la simulation du filtre à 100 étages avait été faite alors que la simulation avait été lancée pour une période d'une semaine. On extrapole les résultats obtenus pour déduire le temps total qui revenait à environ 8 mois.

La liste principale des sites a été divisée en 5 listes plus petites. On pouvait donc s'attendre à obtenir un temps de simulation d'environ 1,6 mois par simulation. Cependant, l'estimation initiale a été effectuée sur la machine la plus puissante à notre disposition et les 3 autres ordinateurs que nous pouvions utiliser étaient beaucoup plus vieux et donc beaucoup plus lents.

Le temps total de simulation pour le filtre à 100 étages a finalement été d'environ 3 mois. Cependant, aucune mesure précise n'a été prise et ce ne sont que des approximations.

Ces résultats nous poussent à nous demander si notre simulateur de panne est optimisé pour le travail qu'il doit effectuer. La réponse à cette question est non. À la base, le simulateur de panne était un simulateur logique qui a été modifié pour effectuer des simulations de panne. Pour réduire le temps de simulation total de façon significative, des modifications devraient y être appliquées comme :

- concevoir un autre simulateur de panne : l'utilisation du simulateur logique Verilog-XL nécessite son appel pour chaque simulation, ce qui consomme un temps important. Pour chaque simulation, il doit lire la liste d'interconnexions et ensuite il effectue la simulation. Après la simulation, il ne garde pas en mémoire cette liste. Un logiciel qui pourrait garder en mémoire ces informations après chaque simulation réduirait le temps de simulation;
- arrêter la simulation de panne lorsque le site testé est couvert par la séquence de test (détection du premier Mismatch) : présentement, notre simulateur effectue la

simulation de tous les vecteurs composant une séquence de test, même si le site testé est couvert très tôt par la séquence. Ceci est causé par l'utilisation du simulateur Verilog-XL, qu'on ne peut arrêter en pleine simulation, même lorsque le site est couvert. Cette dernière proposition peut être réalisée à l'aide d'un simulateur de panne en série qui vérifie après chaque simulation de vecteur, si le site est couvert ou non. Cette proposition revient à utiliser le principe de détection du premier Mismatch proposé au chapitre 4.6, mais que cette fois la simulation arrêterais après avoir découvert un site défectueux.

- réduire le nombre de sites à tester : cette méthode, proposée par Bose et Agrawal (2006), est appelée le « upper-bounding ». Elle consiste à identifier et éliminer lors de l'étape de la simulation logique tous les sites qui sont garantis de ne pouvoir être détectés par les vecteurs appliqués au circuit. Tous les sites qui ne pourront être couverts seront retirés de la liste de pannes à simuler. Ensuite, les probabilités de détection des pannes restantes sont estimées et celles qui offrent une probabilité de détection sous un certain seuil sont aussi retirées de la liste de pannes à simuler.

Pour pouvoir estimer le gain que les deux premières améliorations proposées pourraient nous apporter, des mesures ont été prises au niveau du temps de chargement de l'outil et de la liste d'interconnexions ainsi qu'au niveau du temps de simulation des vecteurs. Les résultats sont présentés dans le tableau XVI.

Tableau XVI

## Temps de chargement et de simulation des vecteurs

Nb. étages	Temps de chargement / panne sec	Temps de simulation vect./ panne sec	Temps de chargement total h	Temps de simulation vect. total h	Temps de simulation et chargement total h
4	22	1.1	10.1	0.5	10.6
7	23	1.9	19.4	1.6	21
15	26	5	47.2	9.1	56.2
25	28	13	93.1	43.2	136.3
50	36	49.1	246.3	336	582.3
100	48	189.3	590.4	2328	2918.4

où :

- Temps de chargement / panne : temps nécessaire au simulateur logique Verilog\_XL à s'ouvrir et d'effectuer la lecture de la liste d'interconnexions. Cette valeur est une approximation et a été obtenue par observation;
- Temps de simulation vect./ panne : temps nécessaire au simulateur logique Verilog\_XL pour effectuer la simulation des vecteurs sur le filtre. Elle est donnée par Verilog\_XL à la fin de la simulation;
- Temps de chargement total : temps total, lors d'une simulation de panne sur un filtre, consacré au chargement des outils. Il est obtenu en multipliant le nombre de panne à simuler par le temps de chaque simulation;
- Temps de simulation vect. total : temps total, lors d'une simulation de panne sur un filtre, consacré à la simulation de vecteurs fonctionnels. Il est obtenu en multipliant le nombre de panne à simuler par le temps de chaque simulation;
- Temps de simulation et chargement total: temps partiel pour effectuer la simulation de panne du filtre avec la séquence de vecteurs fonctionnels initiale. Le temps nécessaire au simulateur de panne Sim\_atpg pour modifier la liste d'interconnexions pour chaque

panne et le temps nécessaire aux modules de simulations de panne collé-à et transition pour effectuer leur travail ne sont pas pris en compte.

Avec ces résultats, on peut estimer, qu'en apportant les changements proposés au simulateur dans cette section, que l'on pourrait réduire le temps total de simulation de panne. En ne prenant compte d'un seul chargement des outils de simulation ainsi qu'une séquence de test équivalente à 50% de la séquence initiale, comme proposé dans la section 5.3, on obtient les réductions en temps de simulation de panne présentée dans le tableau XVII.

Tableau XVII

Estimation de la réduction du temps de simulation de panne

Nb. étages	Temps de chargement / panne sec	Temps de simulation vect./ panne sec	Temps de simulation vect. total sec	Temps de simulation de panne total h	Réduction du temps de simulation et chargement total %
4	22	0.55	909.7	0.25	97.56
7	23	0.95	2886.1	0.8	96.15
15	26	2.5	16355	4.55	91.92
25	28	6.5	77805	24.6	84.14
50	36	24.55	604764	168	71.15
100	48	94.65	4191291	1164	60.11

Avec l'application des deux modifications proposées, on obtient des gains très intéressants. Ils permettent de se retrouver avec des temps de simulation plus réalistes pour pouvoir utiliser notre méthode de test.

## 5.7 Conclusion

L'application de vecteurs fonctionnels en début de séquence de tests a permis d'obtenir des gains très intéressants tant au niveau du gain en temps qu'en gain au niveau mémoire d'un ATE. Le test fonctionnel ayant été laissé de côté particulièrement à cause du temps nécessaire à son développement, l'utilisation d'outil comme *Matlab* a permis de réduire ce temps de façon significative. Les gains obtenus à l'aide des séquences générées à l'aide de *Matlab* nous permettent de conclure que leur utilisation offre un gain considérablement important pour justifier leur usage.

La méthode de réduction du temps de test de détection du premier 'Mismatch' présentée au chapitre précédent a permis de déduire que pour les simulations collé-à, seulement le tiers de la séquence initiale est nécessaire pour effectuer les simulations de panne, alors que pour les modèles de pannes transition cette proportion atteint la moitié.

L'utilisation de plusieurs ordinateurs pour effectuer les simulations du filtre à 100 étages a été bénéfique. Mais, pour réduire encore ce temps, un plus grand nombre de machines à notre disposition aurait été nécessaire. Certaines modifications qui pourraient être appliquées au simulateur ont été proposées afin de réduire encore plus le temps de simulation. La mise en œuvre de ces modifications ainsi que l'exploration d'autres méthodes d'accélération font partie de travaux futurs.

## CONCLUSION

Ce travail de recherche avait pour but de réduire le temps de test et la quantité de mémoire requise sur les testeurs de circuits intégrés avec l'utilisation de vecteurs fonctionnels et de signaux génériques. Butler et Saxeja (2000), par une étude empirique, ont démontré que les vecteurs fonctionnels détectaient plus rapidement les pannes que les vecteurs structurels. Nous avons donc repris cette affirmation et l'avons appliquée à des filtres RIF de différents nombres d'étages. L'utilisation d'outil comme *Matlab* a permis de générer les signaux génériques très rapidement, ce qui a permis de réduire significativement le temps de génération de vecteurs des tests fonctionnels. Des modules de simulation de pannes de type collé-à et transition ont été conçus et implémentés dans un simulateur de panne conçus à l'ÉTS. Les gains obtenus en temps par l'utilisation des signaux génériques, complémentés des vecteurs de type balayage, atteignent respectivement 82 % et 85 % pour les modèles collé-à et transition, alors que les gains en mémoire atteignent respectivement 76% et 84%. Avec ces résultats, on peut conclure que les objectifs fixés au départ de ce projet ont été atteints.

Cependant, notre méthode de test exigeait un test de simulation très grand pour connaître les couvertures de panne. Pour résoudre le problème, un programme a été conçu dans le but de connaître le nombre de vecteurs utiles de chaque séquence. Il a permis de trouver qu'en moyenne, les séquences de test peuvent être le tiers de la séquence initiale pour le modèle collé-à et de la moitié pour le modèle transition. Ce résultat permettra, pour de futures simulations, de réduire considérablement le temps de simulation.

Ce projet a fait l'objet d'une publication : *Redefining the Role of Functional Testing* à l'occasion de la conférence IEEE NEWCAS 2006.



## **RECOMMANDATIONS**

Le travail de recherche a seulement été appliqué avec un seul type de circuit soit les filtres RIF. Pour pouvoir savoir son efficacité à plus grande échelle, un plus grand nombre de types de circuit devrait être étudié. Ceci nous permettra de déterminer sur quel type de circuit on peut utiliser notre méthode de test et sur lesquelles elle ne sera pas efficace. De plus, le simulateur de pannes devra être optimisé afin de réduire davantage le temps de simulation. La vérification pratique des résultats obtenus sur ATE serait intéressante. Cependant, l'accès à ce type de machine demeure restreint.

## **ANNEXE 1**

Résultat de simulation de couverture de panne avec le signal du sinus

### Résultats de simulation du modèle collé-à pour le filtre RIF

	Couverture de pannes %	Nb. vecteurs fonctionnels	Nb. vect. fct. p/r à la séquence initiale %
4	97.28	58	85.29
7	97.56	84	61.31
15	98.15	113	35.31

### Résultats de simulation de panne du filtre RIF pour les modèles de panne transition

	Couverture de panne %		
Nb. étages	TDF	IRF	Collé-à
4	91.54	93.83	97.28
7	92.03	93.09	97.56
15	94.34	95.75	98.15

### Nombre de vecteurs fonctionnels pour les modèles de panne transition

	Nb. vecteurs fonctionnels		Nb. vect. fct. p/r à la séquence initiale %	
Nb. étages	TDF	IRF	TDF	IRF
4	60	59	88.24	86.76
7	122	114	89.05	83.21
15	306	194	95.63	60.63

Gain en temps pour le modèle collé-à avec l'utilisation des vecteurs fonctionnels

Nb. étages	Avant fct.	Après fct.	Gain optimal %	Gain 1 vct. scan
4	76	17	76.51	76.32
7	87	17	79.76	79.31
15	127	23	81.61	81.1

Nombre de vecteurs scan pour les modèles transition avant et après l'utilisation du test fonctionnel

Nb. étages	Avant fct.	Après fct.		Gain %	
		TDF	IRF	TDF	IRF
4	149	31	29	78.6	79.95
7	176	49	41	71.65	76.23
15	259	45	43	82.26	83.16

Gain en mémoire pour le modèle collé-à

Nb. Étages	Nb. bits scan	Nb. bits fct + scan	Gain %
4	1824	872	52.19
7	4176	1488	64.37
15	14224	3480	75.53

## Gain en mémoire pour les modèles transition

		Nb. bits fct+scan		Gain %	
Nb. Étages	Nb. bits scan	TDF	IRF	TDF	IRF
4	7152	1968	1864	72.48	73.94
7	16896	5680	4848	66.38	73.3
15	58016	12528	11184	78.41	80.72

## RÉFÉRENCES

- Benware, B., Lu, C., Van Slyke, J., Krishnamurthy, P., Madge, M., Keim, M., Kassab, M., Rajski, J. (2004), *Affordable and Effective Screening of Delay Defects in ASICs using the Inline Resistance Fault Model*, Int. Test Conf., pp. 1285 – 1294.
- Benware, B., Madge, R., Lu, C., Daasch, R. (2003), *Effectiveness Comparisons of Outlier Screening Methods for Frequency Dependent Defects on Complex ASICs*, VLSI Test Symposium, pp. 39 – 46.
- Bose, S., Agrawal, V. D. (2006), Fault Coverage Estimation for Non-Random Functional Input Sequences, Int. Test Conf, paper 19.3.
- Bushnell, M. L. (2000), *Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits*, Édition Springer.
- Butler, K. M., Saxena, J. (2000), *An Empirical Study on the Effects of Test Type Ordering on Overall Test Efficiency*, Int. Test Conf., pp. 408-416.
- Dufaza, C., Viallon, H., Chevalier, C. (1995), *BIST Hardware Generator for Mixed Test Scheme*, European Design and Test Conf., pp. 424 – 430.
- Fujiwara, H. (1985), *FAN: A Fanout-Oriented Test Pattern Generation Algorithm*, Proc. International Symp. On Circuits and Systems, pp. 671-674.
- Goel, P. (1981), *An implicit enumeration algorithm to generate tests for combinational logic circuits*, IEEE Trans. Comput., Vol. C-30, No. 3, pp. 215-222.
- Hariri, Y. (2002), *Amélioration de la méthode de diagnostic basée sur les signatures probabilistes de  $\Delta IDDQ$* , Mémoire de maîtrise, École de technologie supérieure.

Krstic, A., Cheng, K. T. (1998), *Delay Fault Testing for Vlsi Circuits*, Édition Springer.

Lin, X., Press, R., Rajski, R., Reuter, P., Rinderknecht, T., Swanson, B., Tamarapalli, N. (2003), *High-Frequency At-Speed Scan Testing*, IEEE Design & Test, Vol. 20, No. 5, pp.17-25.

Maxwell, P., Aitken, R.C., Johansen, V., Chang, I. (1991), *The Effect of Different Test Sets on Quality Level Prediction : When is 80% Better Than 90%?*, Int. Test Conf., pp.358-364.

Maxwell, P., Hartanto, I., Bentz, L. (2000), *Comparing Functional and Structural Tests*, Int. Test Conf., pp.400-407.

Moore, W., Gronthoud, G., Baker, K., Lousberg, M. (2000), *Delay Fault Testing and Defects in Deep Sub-micron ICs-does Critical Resistance Really mean anything*, Proc. Int. Test Conf., pp. 95 – 104.

Nigh, P., Maly, W. (1989), *Test Generation for Current Testing*, Proceedings of 1st European Test Conference, pp. 194-200.

Roth, J.P., Bouricious, W.G., Schneider, P.R. (1967), *Programmed algorithms to compute tests to detect and distinguish between failures in logic circuits*, IEEE Trans. Electroni. Comput., Vol. EC-16, No.10, pp. 567-580.

Schuler, D.M., Ulrich, E.G., Baker, T.E., Bryant, S.P. (1975), *Random Test Generation Using Concurrent Logic Simulation*, Proc. 12th Design Automation Conf, pp. 261-267.

Simeu, E., Puissochet, A., Rainard, J.L., Tagant, A.M., Poize, M. (1992), *A new tool for random testability evaluation using simulation and formal proof*, VLSI Test Symposium, pp. 7 – 9.

Smith, G.L. (1985), *Model for Delay Faults Based Upon Paths*, Proceedings IEEE Int. Test Conf., pp. 342-349.

Tumin, K., Vargas, C., Patterson, R., Nappi, C. (2001), *Scan vs. Functional – A Comparative Effectiveness Study on Motorola's MMC2107*, Int. Test Conf., pp. 443-450.

Waicukauski, J.A., Lindbloom, E., Rosen, B.K., Iyengar, V.S. (1987), *Transition Fault Simulation*, IEEE Design & Test of Computers, pp. 32-38.

West, B.G. (1998), *Functional ATE CAN Meet the Challenges*, Int. Test Conf. pp. 1155.

West, B.G. (1999), *Accuracy Requirements in At-Speed Functional Test*, Int. Test Conf., pp. 780-787.