THÈSE DE DOCTORAT DE L'UNIVERSITÉ PARIS VI

Spécialité : INFORMATIQUE

Présentée par : **Abdelhakim KHOUAS**

Pour obtenir le titre de :

DOCTEUR DE L'UNIVERSITÉ PARIS VI

Sujet de la thèse :

SIMULATION DE FAUTES ET OPTIMISATION DES TESTS DE PRODUCTION POUR LES CIRCUITS ANALOGIQUES AVEC PRISE EN COMPTE DES TOLÉRANCES

Soutenue le : **14 septembre 2000**, devant le jury composé de :

M. Adam OSSEIRAN	Rapporteur
M. Michel RENOVELL	Rapporteur
Mme Anne DERIEUX	Examinateur
M. Alain GREINER	Examinateur
M. Jose Luis HUERTAS	Examinateur
M. William HUIN	Examinateur

A Amel
A mes parents
A toute ma famille
A ceux et celles qui m'ont aidé

AVANT-PROPOS

Je souhaite en premier lieu remercier M. Alain Greiner, Directeur du département Architecture du Laboratoire d'Informatique de Paris VI (LIP6), pour m'avoir accueilli au sein de son laboratoire.

J'adresse mes remerciements les plus chaleureux à Mme Anne Derieux, Maître de Conférence à PARIS VI, qui m'a encadré dès mes premiers pas dans mes études et travaux de recherche post-universitaire, et dont les conseils et encouragements m'ont été très utiles dans la préparation de ce travail.

Je remercie sincèrement M. Adam Osseiran, Professeur à l'Ecole d'Ingénieur de Genève (EIG), et M. Michel Renovell, Chargé de Recherche CNRS au LIRMM, qui m'ont fait l'honneur d'être rapporteurs de cette thèse. Je tiens également à remercier M. Jose Luis Huertas, Directeur de L'Institut de Micro-électronique de Séville (IMSE), et M. William Huin, Responsable Test RF et Engineering chez ATMEL-Grenoble, qui ont accepté de juger mon travail en participant à mon jury de thèse.

Je profite de cette occasion pour adresser mes remerciements à tous les membres du département Architecture du laboratoire LIP6, et plus particulièrement mes collègues du bureau 214 qui m'ont permis de passer de bons moments, Marie-Minerve Louerat, Habib Mehrez et Hassan Aboushady pour leur aide et leur soutien, sans oublier mes anciens collègues de l'équipe test : Amar Guettaf, Olivier Florent et El-Housseine Rejouan, qui m'ont initié aux joies du test. J'adresse une pensée spéciale à Mohamed Dessouky pour son aide et son amitié durant ces quatre années de thèse. Mes sincère remerciements vont aussi à Marie-Catherine Hurgues qui m'a aidé à la rédaction de cette thèse et à qui j'exprime toute ma reconnaissance, Nicole Fouque et Régine Guittard pour leur disponibilité et leur sympathie.

Enfin, je remercie ma femme Amel pour toute son aide et son soutien quotidien, et sans qui je n'aurai jamais pu aller jusqu'au bout de cette thèse.

A tous ceux et celles qui m'ont aidé de près ou de loin, à un moment ou à un autre, dans la préparation de ma thèse, je tiens à adresser mes remerciements les plus sincères.

RÉSUMÉ

Face aux progrès accomplis dans le domaine de l'intégration des circuits intégrés (VLSI), les circuits analogiques deviennent plus complexes et plus difficiles à tester, ce qui nécessite de disposer d'outils automatiques pour leur test et leur diagnostic.

Cette thèse présente une nouvelle méthodologie pour la simulation de fautes et l'optimisation automatique des tests de production pour les circuits intégrés analogiques en tenant compte des variations des paramètres des circuits dues aux fluctuations du processus technologique de fabrication des circuits intégrés.

Le simulateur de fautes est un outil indispensable au développement de toute stratégie de test, il permet de valider les techniques de conception en vue du test (DFT), et de réduire les coûts des tests de production. Les deux caractéristiques importantes d'un simulateur de fautes sont : précision et rapidité. Pour répondre à l'exigence de précision dans le monde analogique où les valeurs sont imprécises et avec tolérances, nous avons défini une fonction de probabilité de détection de fautes (PDF) qui permet de quantifier le degré de détection possible d'une faute donnée. Pour la rapidité, nous avons proposé un nouvel algorithme qui utilise des tests d'arrêt pour réduire le temps de simulation de fautes.

Pour les circuits analogiques, les tests dépendent de la nature du circuit à tester. Il est donc impossible de développer un générateur automatique de vecteurs de test pour tous les types de circuits. C'est pourquoi nous avons abordé le problème de l'optimisation automatique d'ensembles de tests pré-existants. Afin de tenir compte des fluctuations du processus de fabrication, une méthode d'optimisation des tests de production basée sur la fonction de probabilité de détection de fautes a été présentée.

Un prototype d'outil de simulation de fautes et d'optimisation automatique des tests de production a été développé pour valider notre approche ; celui-ci utilise le simulateur électrique ELDO. Ce prototype nous a permis de valider, sur plusieurs circuits, notre méthode basée sur les probabilités de détection de fautes, et les résultats de performance obtenus sont très encourageants.

MOTS-CLÉS

VLSI, Test Analogique, Détection de Fautes, Simulation de Fautes, ATPG, Optimisation, Test de Production.

Fault Simulation and Production Test Optimization for Analog Circuits under Parameter Variations

Abstract

As a result of the evolution of VLSI circuit density, analog circuits become more complex and more difficult to test, which requires automatic tools for both circuit test and diagnosis. This thesis presents a new methodology for fault simulation and automatic optimization of production tests of analog integrated circuits taking into account circuit parameter variations due to IC process fluctuations.

The fault simulator is an essential tool for the development of any test strategy. It permits to verify the design for testability technique (DFT), and to reduce the cost of production tests. The two major characteristics of a fault simulator are: accuracy and rapidity. In order to satisfy the accuracy requirement in the analog world where component parameters are imprecise and are usually defined with tolerances, we have defined a fault detection probability function (PDF) which allows to quantify the degree of possible detection of a given fault. For the speed, we have proposed a new algorithm which uses stop rules to reduce simulation time.

Due to the nature of analog circuits, the type of test depend on the circuit under test. It is thus impossible to develop an automatic test pattern generator for all types of circuits. That's why we have studied the problem of automatic optimization of an existing set of tests. In order to take into account process fluctuations, a method of production test optimization based on the fault detection probability function has been presented.

A prototype of a CAD tool intended for fault simulation and automatic production test optimization has been developed so as to be able to verify our approach, the tool uses the electrical simulator *ELDO*. This prototype has allowed us to verify our method, based on the fault detection probability function, on several circuits, and the obtained results are very promising.

Keywords

VLSI, Analog Testing, Fault Detection, Fault Simulation, ATPG, Test Optimization, Process variations, Parameter deviations, Production Testing.

Table des matières

Liste des figures

Liste des tableaux

Introduction

Historiquement, les circuits électroniques étaient exclusivement analogiques et fabriqués avec des composants discrets. Les composants étaient montés sur des cartes imprimées *PCB* " *Printed Circuit Board* " et ces cartes *PCB* étaient testées avec la technique dite " *lit à clous* " ou aussi " *bed of nails* " qui consiste à utiliser une planche sur laquelle sont implantés des clous reliés à un testeur. Cette technique permet un accès à toutes les tensions d'entrée sortie des composants et facilite ainsi le test et le diagnostic des circuits analogiques.

L'évolution de la technologie des circuits intégrés et la miniaturisation des transistors ont permis de développer des circuits beaucoup plus complexes. Les circuits numériques sont devenus prédominants à cause de leur simplicité et de leur faible consommation. Par contre, beaucoup de systèmes électroniques nécessitent une partie analogique ; en effet, comme la nature de l'univers est analogique, en général, les circuits d'interfaces avec le monde extérieur comme les filtres, les convertisseurs analogique-numérique et numérique-analogique sont des circuits analogiques ou mixtes.

L'intégration de ces circuits analogiques a considérablement réduit leur taille et leur coût de revient, mais en contre partie ils sont devenus plus difficiles à tester à cause de la limitation des accès aux seuls nœuds externes du circuit. Durant ces dernière années et suite à l'explosion du marché des télécommunications, multimédias et de l'électronique automobile qui nécessitent des circuits mixtes, les circuits analogiques et mixtes jouent un rôle de plus en plus important sur le marché des circuits intégrés. Aujourd'hui, sur un circuit mixte analogique-numérique, bien que les parties analogiques n'occupent en général que 5% à 30% de la superficie globale du circuit, le coût du test de ces blocs analogiques représente environ 70% du coût global de test.

Dans le premier chapitre de cette thèse, nous introduisons le test analogique et les différences entre le test numérique et le test analogique qui rendent le test analogique plus difficile et plus complexe à réaliser. Dans la première partie du chapitre, nous présenterons les principales difficultés rencontrées dans le test des circuits analogiques, ensuite, nous introduirons brièvement les grandes lignes de notre approche permettant de résoudre une partie des difficultés du test analogique.

Le deuxième chapitre présente un état de l'art des différents domaines du test des circuits analogiques. Nous présenterons les différents travaux publiés dans les domaines suivants :

1. Modélisation de fautes
2. Simulation de fautes
3. Génération des vecteurs de test
4. Conception en vue du test

Le troisième chapitre est consacré aux rappels théoriques et concepts de base. Dans la première partie du chapitre, nous présenterons les rappels théoriques sur les probabilités et les statistiques. Nous donnerons toutes les définitions, les théorèmes et les résultats nécessaires à la compréhension de la suite de la thèse. Comme il n'existe pas de modèles de fautes standards pour les circuits analogiques, la deuxième partie du chapitre sera consacrée à la modélisation de fautes. Nous introduirons les différents modèles et types de fautes que nous utiliserons pour la validation de notre travail.

Dans le quatrième chapitre, nous présenterons notre méthode de simulation de fautes avec prise en compte des tolérances. Dans un premier temps, nous formaliserons le problème de la simulation de fautes pour les circuit analogiques avec prise en compte des tolérances. Nous introduirons les problèmes engendrés par les déviations des paramètres des composants, ensuite, nous définirons la fonction de probabilité de détection de fautes PDF que nous utiliserons dans notre approche. La deuxième partie du chapitre est consacrée à la présentation en détail de notre approche. Dans la dernière partie du chapitre, nous définirons la notion de taux de couverture pour le test des circuits analogiques et nous donnerons les résultats de validation de notre méthode sur plusieurs circuits analogiques en terme d'efficacité (taux de couverture) et en terme de temps de simulation de fautes.

Le cinquième chapitre est une application de la simulation de fautes des circuits analogiques avec prise en compte des tolérances pour l'optimisation des tests de production pour les circuits analogiques. Nous présenterons une méthode d'optimisation automatique des vecteurs de test pour les circuits analogiques qui utilise la fonction de probabilité de détection de fautes PDF pour choisir le meilleur ensemble de vecteurs de test qui permet de réduire le coût du test de production tout en ayant un meilleur taux de couverture.

Dans le dernier chapitre, nous présenterons la mise en œuvre logicielle d'un prototype de simulation de fautes et d'optimisation des ensembles de tests pour les circuits analogiques avec prise en compte des tolérances. Nous expliquerons la mise en œuvre de l'outil *ATSO* " *Automatic Test Pattern Optimization* " et les différents choix adoptés.

Chapitre 1

Problématique

La réalisation d'un circuit intégré passe par deux importantes étapes, une étape de conception du circuit " *Design* " et une étape de fabrication " *manufactoring* " (voir figure 1.1). Dans chacune des deux étapes, nous devons effectuer une phase de test, un test fonctionnel ou un test de validation " *Functional Testing* " pour l'étape de conception et un test de production ou un test structurel " *Structural testing* " pour l'étape de fabrication. Le but du test fonctionnel est de vérifier les spécifications du circuit à fabriquer et si le test fonctionnel ne passe pas, il est important de faire un diagnostic du circuit pour localiser l'erreur de conception. Par contre, le but du test structurel est de détecter les défauts de fabrication pour séparer les bons circuits des circuits défectueux. Comme dans la phase de fabrication en grande série, il n'est plus possible de réparer les circuits défectueux, on n'a pas besoin de faire un diagnostic de tous les circuits rejetés. Par contre, pour améliorer le rendement de la chaîne de fabrication, on peut être amené à faire un diagnostic sur un échantillon de circuits défectueux, mais en utilisant des vecteurs de test spécifiques pour le diagnostic.

Le coût d'un composant défectueux augmente d'un facteur de *10* à chaque phase d'assemblage, un composant défectueux coûte jusqu'à *1000* fois plus si le défaut est détecté chez le client et non juste après fabrication. Il est donc important de détecter les défauts de fabrication le plus tôt possible dans les phases de mise en œuvre du circuit dans le système final. Par contre, le coût du test de production influe considérablement sur le prix de revient des circuits intégrés, il n'est donc pas possible d'appliquer tous les tests possibles pour détecter tous les défauts possibles. Par conséquent, il est très important de trouver un bon compromis entre l'efficacité et le coût global d'une technique de test de production.

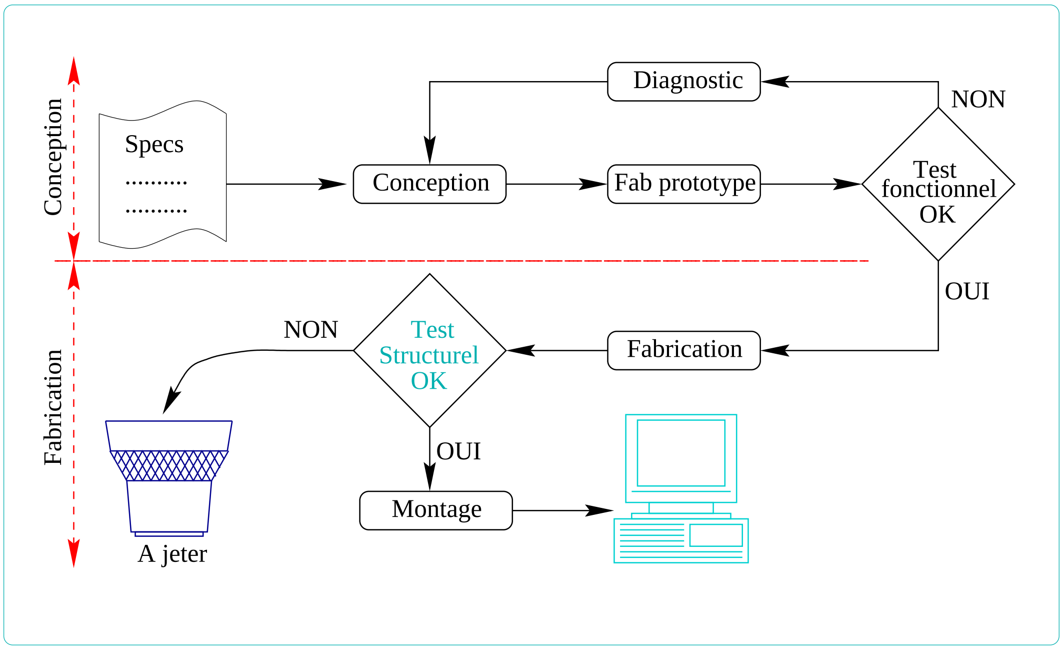

FIG. 1.1: *Les différentes étapes de fabrication d'un circuit intégré.*

1.1 Les problèmes posés par le test des circuits analogiques

Bien que le but du test structurel pour les circuits numériques et pour les circuits analogiques soit le même, il existe des différences importantes entre le test des circuits numériques et le test des circuits analogiques. La difficulté majeure du test des circuits numériques réside dans la complexité et la taille de ces derniers, alors que pour les circuits analogiques nous avons des circuits de taille très faible comparée aux circuits numériques. En contre partie, la majeure difficulté du test des circuits analogiques et mixtes réside dans la nature continue des signaux analogiques, le manque de modèles comportementaux, la diversité des paramètres d'entrée sortie et des types de simulation et la modélisation des défauts de fabrication.

1.1.1 La nature continue des signaux analogiques

Une des différences majeures entre le test des circuits numériques et le test des circuits analogiques est l'infinité des valeurs pour un signal analogique. Pour les circuits numériques, le signal d'entrée ou de sortie ne peut prendre que deux valeurs *0* ou *1*, en conséquence il est plus facile de déterminer précisément les valeurs des entrées à appliquer, et si les sorties mesurées sont correctes ou non. En revanche, il est très difficile d'utiliser la même méthodologie pour le test des circuits analogiques. En effet, comme les signaux analogiques sont continus, il est impossible d'appliquer en entrée du circuit analogique une valeur précise ou de déterminer précisément si les sorties analogiques mesurées sont correctes ou erronées. En conséquence, pour le test analogique il est indispensable de considérer chaque valeur analogique avec son intervalle de tolérance, et pour la comparaison des sorties, le but n'est plus de comparer la sortie mesurée par rapport à une valeur précise comme pour les circuits numériques mais de comparer la sortie par rapport à un intervalle de tolérance souhaité.

1.1.2 Imprécision des simulateurs et des mesures

Les simulations logiques sont des simulations exactes. C'est à dire que lorsque le résultat de la simulation est 1 sur une sortie donnée, alors on est sûr qu'après fabrication du circuit on va bien mesurer la valeur logique 1 à la sortie en question. Pour les circuits analogiques, ceci est complètement différent ; en effet quand un simulateur analogique donne une valeur de $2.5V$ sur une sortie, après fabrication du circuit on peut mesurer une valeur de $2.4V$ ou $2.6V$. Cette imprécision est due au erreurs des modèles des composants utilisés dans les simulateurs analogiques et aux erreurs dues aux instruments de mesure. Il est donc important de prendre en compte ces imprécisions dans la phase de la simulation de fautes pour déterminer si une faute est détectable ou non.

1.1.3 Les variations sur le processus de fabrication

Lorsque on fabrique des milliers de circuits numériques et qu'on mesure une sortie donnée sur tous les bons circuits (qui ne sont pas défectueux), on trouve la même valeur 0 ou 1 pour tous les circuits. Si on refait la même expérience pour les circuits analogiques, on trouve des valeurs différentes. Ces différences sont dues aux variations des paramètres du processus de fabrication d'un circuit à un autre. Par exemple, les circuits qui se trouvent au centre du " *wafer* " vont avoir des mesures différentes des circuits qui se trouvent aux extrémités du " *wafer* ".

Des méthodes de conception et de routage ont été développées pour minimiser l'impact des variations du processus de fabrication sur la fonctionnalité du circuit. Par exemple, les valeurs absolues des composants (capacités, resistances, ...) d'un circuit analogique peuvent varier de 10% de leurs valeurs nominales, mais les variations relatives entre composants peuvent être réduites à 0.1%.

Ces variations dues au processus de fabrication sont petites donc sans conséquence pour les circuits numériques. Pour les circuits analogiques, des variations minimes des valeurs de certains composants peuvent engendrer des variations très grandes en sortie du circuit. Dans ce cas de figure, ces variations engendrent (ce qu'on appelle) les fautes paramétriques (voir section 3.2.2) qui peuvent faire dévier la sortie du circuit de son intervalle de tolérance ou même engendrer un comportement du circuit complètement différent de son comportement initial. Il est aussi possible que ces variations aient un effet compensatoire et donc le comportement du circuit ne sera pas affecté. En résumé, toutes les variations dues au processus de fabrication n'engendrent pas de fautes paramétriques et la simulation de faute est la seule méthode qui permet de déterminer si les variations en question engendrent un comportement fautif du circuit.

Il est donc indispensable de modéliser les variations du processus de fabrication pour en tenir compte dans la phase de la simulation de faute si on veut avoir des résultats très précis.

1.1.4 La diversité des modes de simulation

Pour le test des circuits numériques, on utilise toujours des simulateurs de fautes basés sur des simulateurs logiques en utilisant les vues structurelles des circuits, et le problème du choix du type de simulation à utiliser n'existe pas. Par contre pour les circuits analogiques se pose le problème de type d'analyse à effectuer pour la simulation de faute, nous avons le choix entre l'analyse DC " *Direct Current* ", l'analyse AC " *Alternate Current* " et l'analyse transitoire. Bien sûr, le type de simulation dépend du circuit en question, par exemple pour un filtre à capacité commutée, on ne peut pas utiliser une analyse AC (analyse en fréquence), si on veut réduire l'ensemble des fréquences à appliquer au filtre, on doit utiliser une analyse transitoire suivie d'une transformation de Fourrier, alors que pour un filtre continu des simples analyses AC sont suffisantes pour réduire l'ensemble des fréquences. Les types de simulations à effectuer dépendent aussi des types de vecteurs de test à optimiser. Par exemple, pour un circuit donné, si on veut uniquement trouver les meilleures tensions de polarisation qui détectent le maximum de fautes, on est obligé de faire des analyses DC alors que si on veut trouver les meilleures fréquences, on doit faire des simulations AC ou transitoires selon le type et les caractéristiques du circuit à simuler.

1.1.5 La diversité des types et des paramètres d'entrée sortie

Pour les circuits numériques, quelque soit la fonctionnalité du circuit, les signaux d'entrée sortie sont toujours des tensions (Vss et Vdd) et la génération des vecteurs de test revient à trouver uniquement les meilleures tensions à appliquer en entrée du circuit pour pouvoir obtenir en sortie des tension différentes des tensions de référence pour les circuits fautifs.

Par contre, pour les circuits analogiques, le problème de la génération des jeux de stimuli n'est pas aussi facile que pour les circuits numériques. En effet, les types de signaux à appliquer en entrée et à mesurer en sortie des circuits analogiques sont très divers et diffèrent d'un circuit à un autre. Les paramètres d'entrée d'un circuit analogique peuvent être : la tension, le courant, la fréquence, ... etc, et les paramètres de sortie peuvent être : la tension, le courant, le gain, la phase, la tension d'offset, le courant d'offset, le rapport signal/bruit, la réjection en mode commun, ... etc.

1.1.6 La dépendance des stimuli avec la fonctionnalité du circuit

Pour les circuits numériques, on utilise des générateurs automatiques de vecteurs de test (ATPG) pour générer les vecteurs de test structurels en utilisant un modèle de faute donné. L'avantage des ATPGs est qu'ils permettent de trouver des ensembles de vecteurs de test optimisés pour détecter toutes les fautes détectables sans se soucier de la fonctionnalité du circuit.

Pour un circuit numérique, à partir d'un modèle de fautes et un ensemble de fautes donnés, le problème de génération automatique des vecteurs de test est assez simple du point de vue théorique. Comme les signaux d'entrée des circuits numériques ne peuvent prendre que deux valeurs *{0,1}*, pour un circuit donné avec N entrées on a 2^N vecteurs possibles, et le problème de la génération des vecteurs de test revient à réduire l'ensemble des 2^N vecteurs possibles tout en détectant toutes les fautes détectables par l'ensemble des 2^N vecteurs. Pour savoir si une faute est détectée par un vecteur donné, il suffit d'évaluer toutes les sorties du circuit fautif et de les comparer aux sorties du bon circuit, et si au moins une des sorties est différente alors la faute est détectable par le vecteur. Le problème d'évaluation et de comparaison des sorties des circuits numériques est aussi un problème simple car l'ensemble des valeurs de sortie se réduit à $\{0, 1\}$.

Par contre pour les circuits analogiques, les jeux de vecteurs de test à générer pour le test structurel dépendent du circuit à tester. Par exemple, pour les amplificateurs opérationnels (AOP) la génération des vecteurs de test revient à trouver les meilleures fréquences à appliquer en entrée de l'AOP pour détecter toutes les fautes possibles avec un nombre minimum de fréquences possibles. Pour un oscillateur contrôlé par une tension (VCO) " *Voltage Control Oscillator* ", le problème de la génération d'un jeu de vecteurs de test efficace revient à trouver les meilleures tensions à appliquer en entrée du VCO pour détecter toute les fautes possibles avec un nombre minimum de tensions possibles.

Un autre problème à considérer pour les circuits analogiques est celui du choix des sorties à observer. En effet pour une même entrée donnée et une faute donnée, on peut détecter la faute en considérant un paramètre donné en sortie et la même faute ne sera pas détectée en considérant un autre paramètre de sortie. Par exemple, pour un AOP et pour une fréquence donnée, la même faute sera détectée en considérant le gain de l'AOP en sortie et ne sera pas détectée en considérant la phase. Considérer tous les paramètres de sortie pour chaque vecteur de test peut engendrer un temps de simulation déraisonnable et un temps de test très prohibitif sur des équipements de test (ATE) " *Automatic Test Equipement* " très coûteux.

En résumé, contrairement aux circuits numériques où la génération des vecteurs de test revient seulement à trouver les meilleures entrées à appliquer pour détecter toutes les fautes possibles, pour les circuits analogiques la génération des vecteurs de test peut être décomposée en plusieurs étapes :

– Sélection du meilleur mode de simulation.
– Sélection des meilleurs paramètres de sortie à mesurer.
– Recherche du meilleur jeu de vecteurs de test à appliquer pour détecter toutes les fautes possibles.

1.2 Notre approche

Nous présentons dans cette thèse une méthodologie pour la simulation de fautes et l'optimisation automatique des tests de production pour les circuits analogiques avec prise en compte des tolérances et des variations du processus de fabrication.

La génération automatique et l'optimisation automatique sont les deux voies qui permettent d'obtenir un jeu de vecteurs de test efficace pour le test structurel (test de production). Les buts recherchés par la génération automatique et l'optimisation automatique sont les mêmes, c'est d'essayer de trouver un jeu de vecteurs de test le moins coûteux en terme de temps d'exécution sur des équipements de test, et qui détecte le plus de fautes possibles, mais les différences entre la génération et l'optimisation sont très importantes. En effet, dans l'algorithme de génération automatique, le but est de partir d'un ensemble de vecteurs de test vide et d'essayer de trouver des vecteurs de test un par un qui permettent de détecter des fautes, ensuite on utilise des algorithmes de compression pour réduire le nombre de vecteurs de test pour arriver à un jeu de vecteurs de test le plus petit possible et qui permet de détecter toutes les fautes détectables.

Pour l'optimisation automatique, la méthodologie est complètement différente. Le but est de partir d'un jeu de vecteurs de test déjà existant, l'ensemble des vecteurs de test de départ peut être un ensemble exhaustif de tous les tests possibles, un jeu des vecteurs de test fonctionnel, ou peut être donné par le concepteur du circuit ou un expert. Ensuite, on utilise un simulateur de fautes pour classer les tests et éliminer tous les tests redondants et les tests qui ne détectent pas de fautes pour réduire l'ensemble des vecteurs de test de départ à un ensemble plus petit ; par conséquent le temps d'exécution sur des équipements de test est plus rapide.

Pour un circuit numérique, l'optimisation automatique reviendrait à réduire l'ensemble des 2^N vecteurs possibles, avec N le nombre d'entrées du circuit. Pour cela, on prend les vecteurs de test un par un, on utilise un simulateur de fautes pour trouver les fautes détectées par chaque vecteur et on s'arrête lorsque toutes les fautes sont détectées ou que tous les vecteurs ont été simulés. En général, étant donnés la grande complexité des circuits numériques et le nombre croissant des entrées, il est très coûteux en temps de simulation et pratiquement impossible de simuler l'ensemble des 2^N vecteurs. C'est pourquoi, en pratique, pour les circuits numériques, la génération des vecteurs de test se décompose en deux phases, une première phase de génération aléatoire qui utilise le principe d'optimisation et une deuxième phase de génération déterministe qui utilise un algorithme de génération. La phase de génération aléatoire utilise la même méthodologie que l'optimisation automatique, en effet dans la phase de génération aléatoire, on réduit l'ensemble des 2^N à un ensemble beaucoup plus petit et qui détecte une grande partie des fautes à détecter, ensuite on utilise un algorithme de génération déterministe pour détecter les fautes restantes.

Pour les circuits analogiques, comme les jeux de vecteurs de test dépendent de la nature du circuit à tester (voir section 1.1.6), il est impossible de développer un générateur automatique des vecteurs de test pour tous types de circuits, c'est pourquoi nous avons opté pour le développement d'un outil d'optimisation automatique des ensembles de vecteurs de test pré-existants. Mais comme les valeurs analogiques sont imprécises et avec tolérances (voir section 1.1.1), nous avons défini une fonction de probabilité de détection des fautes qui permet de quantifier par un nombre réel appartenant à l'intervalle $[0\ ,\ 1]$ à quel degré la faute est détectable. Nous avons ensuite développé un simulateur de fautes et un outil d'optimisation des tests de production pour les circuits analogiques avec prise en compte des tolérances et des variations du processus de fabrication, en utilisant ces probabilités de détection des fautes.

La méthode générale pour l'optimisation des ensembles de tests de production est la suivante :

– Utilisation d'un module d'injection de fautes automatique qui permet de traduire les défauts physiques au format correspondant au simulateur en utilisant des modèles de fautes.

– Simulation de fautes avec prise en compte des tolérances et détermination des probabilités de détection de chaque faute par chaque test.

– Réduction et ordonnancement de l'ensemble des tests de départ en utilisant les probabilités de détection des fautes données par le simulateur de faute.

Chapitre 2

État de l'art

Un récapitulatif des travaux de recherche publiés avant 1979 sur le test des circuits analogiques est proposé dans [DR79]. A l'époque le problème du test des circuits analogiques se posait pour tester les circuits imprimés qui commençaient à devenir de plus en plus complexes grâce entre autres à l'évolution des circuits MSI (Medium Scale Integrated Circuits) et LSI (Large Scale Integrated Circuits).

Aujourd'hui, avec la forte demande des circuits analogiques et mixtes, et la complexité croissante des circuits, beaucoup de travaux de recherche commencent à être développés dans tous les domaines du test analogique, allant de la modélisation de fautes jusqu'à la conception en vue du test.

Dans ce chapitre nous faisons un tour d'horizon des différentes méthodes publiées dans le domaine de la recherche et des techniques utilisées dans l'industrie. Dans la première section du chapitre, nous allons donner quelques définitions des différents termes utilisés dans le test des circuits analogiques. Dans les autres parties du chapitre, nous nous intéressons aux différents travaux publiés dans les domaines suivants :

– Modélisation de fautes

– Simulation de fautes

– Génération des stimuli

– Conception en vue du test

2.1 Définitions

Comme il n'existe pas de terminologie standard pour les termes utilisés dans le domaine du test analogique, et pour faciliter la lecture de ce manuscrit, voici les définitions des termes importants utilisés :

– *Défaut* : défaut physique qui affecte le *layout* d'un circuit.

– *"Spot defect"* : représente le dépôt supplémentaire ou déficitaire d'un ou plusieurs matériaux sur le "*wafer*", ceci crée le défaut physique.

– *Faute* : Modélisation d'un défaut physique dans le but de simuler l'effet de ce dernier sur le circuit.

– *Faute catastrophique* : Modélisation d'un défaut majeur comme un court-circuit ou un circuit ouvert.

– *Faute paramétrique* : Modélisation des fluctuations de l'environnement de fabrication qui engendrent des variations sur les sorties du circuit.

– *Test fonctionnel* : Vérification des spécifications du circuit. Le but du test fonctionnel est de vérifier le fonctionnement du circuit avant de l'envoyer en production. En général, on effectue le test fonctionnel sur les prototypes des circuits.

– *Test structurel* : Vérification des circuits dans la phase de production en grande série, ceci est aussi appelé test de production. Le but du test structurel est de détecter les défauts de fabrication qui peuvent affecter le *layout*.

– *Diagnostic* : Détermination de la cause du dysfonctionnement d'un circuit.

– *Génération déterministe* : Utilisation d'algorithmes pour trouver des tests qui détectent des fautes données, de façon déterministe.

– *Optimisation d'un ensemble de tests* : Réduction du nombre de vecteurs de test d'un ensemble, tout en détectant les mêmes fautes que l'ensemble de départ. Le but de l'optimisation des tests est de réduire le temps nécessaire à l'application de l'ensemble des tests sur des équipements de test très coûteux, et réduire ainsi le coût du test de production.

– *Taux de couverture* : Le rapport du nombre de fautes détectées par rapport au nombre de fautes globales. Le taux de couverture dépend du modèle de faute utilisé.

2.2 Modélisation de fautes

Un modèle de faute représentatif des défauts réels et simple à utiliser est fondamental pour développer une stratégie de test efficace. En effet, la mesure de l'efficacité d'un ensemble de stimuli en terme de défauts réels détectés est basée sur le modèle de faute utilisé. Si pour un ensemble de tests donnés, on a un taux de couverture de 100%, cela ne veut pas dire qu'on va détecter tous les des défauts physiques. L'efficacité d'un ensemble de test dépend aussi de la représentativité du modèle de faute utilisé, plus le modèle de faute est représentatif de la majorité des défauts physiques, plus on aura de défauts détectés. En général, comme pour les circuits numériques, toutes les modélisations supposent que si la faute existe alors elle est unique. Cependant certaines techniques [HK93b] prennent en compte le cas des fautes multiples, mais elles sont rarement applicables aux circuits actuels car beaucoup trop complexes.

Contrairement aux circuits numériques, les fautes analogiques sont classées en deux catégories : les fautes catastrophiques et les fautes paramétriques [MSD86][MV89] et [Som91] (voir section 3.2). Les fautes catastrophiques sont dues à un dépôt supplémentaire ou au contraire un dépôt déficitaire d'un ou plusieurs matériaux sur le "*wafer*" [Mal87], c'est ce qu'on appelle "*Spot Defect*" . Dans [Syr87], l'auteur étudie les dégradations dues aux "*Spot Defects*" sur les caractéristiques I-V (courant-tension) d'un transistor MOS et montre que les "*Spot Defects*" n'engendrent pas toujours des fautes catastrophiques, mais ils engendrent aussi des fautes paramétriques. Les fautes paramétriques sont dues en général aux fluctuations des paramètres du processus de fabrication [Mil98], mais peuvent aussi être causées par des "*Spot Defects*" [Syr87] et [DSGH99].

Comme les fautes catastrophiques engendrent un fonctionnement complètement différent du circuit, elles sont plus faciles à détecter. En effet comme la plupart des fautes catastrophiques affectent tous les paramètres du circuit, le problème du choix des meilleurs paramètres à mesurer en sortie ne se pose pas (voir section 1.1.5). En général, un simple test DC ou un test en courant "*IDDQ*" peut détecter la majorité des fautes catastrophiques [HRK95] [TS98a] [Som93]. Le test en courant "*IDDQ*" est basé sur la mesure du courant d'alimentation du circuit, en général pour les fautes catastrophiques le courant d'alimentation mesuré est différent du courant d'alimentation du circuit correct. Par contre, les fautes paramétriques engendrent des déviations des paramètres des sorties du circuit et ces déviations peuvent être plus au moins grandes suivant le paramètre considéré, il est donc plus difficile de tester ces fautes. En effet, il ne suffit pas de trouver les stimuli qui activent la faute, mais il faut aussi trouver les meilleurs paramètres qui permettent d'avoir une déviation en sortie du circuit en dehors de la plage de tolérance acceptable [HK93a].

Pour les circuits numériques on peut définir des modèles de faute selon deux niveaux d'abstraction : au niveau logique et au niveau transistor. Par exemple, au niveau logique, nous avons le modèle des collages au niveau porte "*Stuck at Fault*", le modèle des court-circuits "*Bridgings faults*" et le modèle des mintermes [Deb94], au niveau transistor nous avons le modèle des transistors collés [RS89]. De même pour les circuits analogiques, les défauts physiques peuvent être modélisés selon deux niveaux d'abstraction, une modélisation au niveau composant ou structurel et une modélisation au niveau fonctionnel [BHSMK97].

2.2.1 Modélisation au niveau composant ou structurel

La modélisation au niveau composant consiste à modéliser les défauts au niveau des composants. En théorie pour un transistor MOS, le modèle de faute catastrophique comprend 6 fautes possibles, les trois courts-circuits entre les trois nœuds du transistor (Grille, Source et Drain) et les trois circuits ouverts [Aza96]. Mais dans la pratique, les fautes les plus probables sont les courts-circuits grille/drain et source drain et les circuits ouverts sur le drain et la source [BA82] et [JG80].

Pour les composants passifs (résistances et capacités), nous avons 2 fautes catastrophiques par composant, un court-circuit qui est modélisé par une résistance de faible valeur en parallèle avec le composant et un circuit ouvert modélisé par une résistance de valeur très élevée en série avec le composant.

2.2.2 Modélisation au niveau fonctionnel

Le modèle de faute fonctionnel permet de modéliser les effets des défauts physiques au niveau fonctionnel des sous blocs analogiques. Pour la modélisation au niveau fonctionnel, on divise le circuit en plusieurs modules fonctionnels, pour chaque module et chaque faute du module on effectue des simulations analogiques au niveau structurel pour modéliser les effets de la faute au niveau fonctionnel. Après abstraction des fautes du niveau composant au niveau fonctionnel, les modules fonctionnels sont représentés comme des boîtes noires caractérisées par les différences entre les sorties du bon circuit et des circuits fautifs [BHSMK97] et [Vin98].

L'avantage de la modélisation de fautes au niveau fonctionnel est qu'elle permet d'utiliser des simulateurs au niveau comportemental [DLSVV94] et [LCSV93] qui sont beaucoup plus rapides que les simulateurs traditionnels de type SPICE. En plus de la rapidité dans la simulation de fautes, beaucoup de fautes au niveau composant peuvent avoir le même effet au niveau fonctionnel et donc peuvent être représentées par le même modèle, ce qui permet de réduire le nombre de fautes à simuler [VCH+97] et [ZCW96].

La modélisation des circuits et des fautes au niveau fonctionnel est réalisée soit en utilisant des macro-modèles avec un simulateur de type SPICE [PCG94] et [ZCW96] soit en utilisant les langages de description comportemental analogique de type *VHDL-A* [SG97] et [PZCW98]. La modélisation des fautes d'un sous-circuit en utilisant des macro-modèles est réalisée selon les étapes suivantes [PCG94] :

1. Choix des paramètres du sous-circuit à modéliser.
2. Simulation de la faute dans le sous circuit au niveau transistor et calcul des valeurs des paramètres choisis à l'étape 1.
3. Modélisation du sous-circuit fautif au niveau fonctionnel en utilisant des macro-modèles.
4. Simulation de la faute dans le sous-circuit au niveau fonctionnel et calcul des valeurs des paramètres choisis à l'étape 1.
5. Comparaison des résultats obtenus dans les étapes 2 et 4 et calcul de l'erreur entre les deux simulations.
6. Si l'erreur est trop grande, améliorer la modélisation du sous-circuit fautif et recommencer à l'étape 4.

L'inconvénient majeur de cette méthode réside dans la difficulté à générer automatiquement des macro-modèles qui sont assez précis.

Les récentes recherches dans le domaine des langages de description des circuits analogique "*AHDL : Analog Hardware Description Language*" ont donné naissance à des langages comme *VHDL-A* ou *Verilog-A* qui permettent de modéliser le comportement d'un circuit analogique en utilisant des équations et des expressions mathématiques. L'utilisation d'un langage de description "*AHDL*" pour la modélisation des fautes au niveau fonctionnel est appelé "*Induced Behavioural Modeling*" [Vin98]. L'avantage de cette méthode réside dans sa facilité à modéliser le comportement des circuits fautifs grâce à l'utilisation des outils mathématiques et des expressions de contrôle de type "*If Then Else*" d'un langage impératif.

2.2.3 Analyse des fautes par induction *IFA*

L'analyse des fautes par induction *IFA* "*Inductive Fault Analysis*" est une méthode qui permet d'extraire les fautes les plus probables d'un circuit à partir de la vue physique du circuit "*layout*" [DS94]. La méthode d'extraction des fautes les plus probables est basée sur l'utilisation d'un simulateur de rendement comme *VLASIC " VLSI Layout Simulation for Integrated Circuits "* [WD87]. Le simulateur de rendement est un outil qui utilise une approche Monte-Carlo pour calculer les probabilités de chaque faute en injectant aléatoirement un grand nombre de "*spot defect* " dans le "*layout*" [HRBB94], il se déroule en 3 étapes :

1. Génération des "*spot defect*" en utilisant les données statistiques du processus de fabrication. Ces données sont : la distribution de la taille des particules causant les défauts "*Defect Size Distribution*" et la distribution spatiale de ces particules "*Defect Spatial Distribution*".

2. Extraction des fautes au niveau du circuit.

3. Calcul des probabilités d'occurrence et classification des fautes par probabilité d'apparition.

Beaucoup de travaux ont été publiés ces dernières années sur l'utilisation de l'analyse des fautes par induction pour la prédiction des fautes et l'optimisation des "*layout*" pour le test des circuits analogiques [STO95], [SA95], [AZ96] et [OGA+97]. Comme l'utilisation de cette méthode nécessite la vue physique du circuit, elle ne permet la prise en compte du test que très tard dans les phases de conception du circuit ce qui est très néfaste pour le "*Time-To-Market*", C'est pourquoi d'autres méthodes ont été développées ces dernières années dans le but de prédire les fautes réelles d'un circuit avant de disposer de sa vue physique [PRG+98].

2.3 Simulation de fautes

Le but d'un simulateur de faute est de déterminer les effets des défauts sur le comportement du circuit et d'évaluer la qualité des jeux de vecteurs de test en calculant le taux de couverture obtenu selon un modèle de faute donné. Les simulateurs de fautes pour les circuits numériques utilisent des techniques connues comme la simulation de fautes parallèles ou la simulation de fautes concurrentes qui exploitent le fait qu'on utilise des simulations logiques et que les différences entre le bon circuit et le circuit fautif sont minimes. Ces techniques permettent de réduire considérablement le temps nécessaire à la simulation de toutes les fautes. Malheureusement, ces techniques sont difficilement utilisables pour les circuits analogiques [ZBC97].

L'approche la plus utilisée pour la simulation de fautes des circuits analogiques est basée sur l'utilisation d'un simulateur pour les circuits analogiques comme *SPICE* et *ELDO*. La méthode consiste à exécuter les étapes suivantes :

- Simulation du bon circuit
- Introduction d'une faute dans le circuit
- Simulation du circuit fautif
- Comparaison des résultats des deux simulations

Les premiers simulateurs de fautes adoptant cette approche sont *FSPICE* [RCA85] qui utilise le simulateur du domaine public *SPICE*, et *ANAFAULT* [SO93] qui utilise le simulateur commercial *ELDO* de Mentor Graphics. Comme les simulateurs électriques de type *SPICE* sont très lents et que les circuits analogiques deviennent de plus en plus complexes, des simulateurs de fautes spécifiques ont été développés dans le but d'accélérer la simulation de fautes pour certaines classes de circuits.

2.3.1 La simulation de fautes pour les circuits linéaires

Les circuits analogiques linéaires sont les circuits définis par une fonction de transfert entre la sortie et l'entrée du circuit. Les circuits linéaires représentent une grande partie des circuits analogiques utilisés dans les systèmes de vidéo, de communication et de traitement d'images. Les transistors utilisés dans les circuits linéaires sont modélisés par les modèles petits signaux qui sont linéaires autour des points de polarisation.

La linéarité des circuits permet de développer des algorithmes de simulation de fautes efficaces.

Les premiers travaux sur la simulation de fautes pour les circuits analogiques linéaires avec prise en compte des tolérances ont été mis en avant dans [PR82]. La méthode consiste à trouver uniquement les limites inférieure et supérieure de l'intervalle de sortie du circuit correct et des circuits fautifs sans utiliser des simulations Monte Carlo. Le calcul suppose que les extrémités inférieure et supérieure de l'intervalle de tolérance de la sortie du circuit sont obtenues en simulant le circuit avec les limites inférieure et supérieure des intervalles de tolérances des paramètres, ce qui n'est pas vrai pour tous les types de circuits analogiques. Récemment, une approche basée sur les intervalles mathématiques a été présentée dans [TS98b] pour remédier aux limitations de la première approche.

Une autre technique de simulation basée sur la transposition du circuit correct et des circuits fautifs dans le domaine discret Z a été proposée dans [NCA93b]. Le simulateur de fautes analogique *DRAFTS* [NCA93a] développé avec cette méthode permet un gain très considérable en temps de simulation par rapport aux simulateurs électriques, le gain est du à l'utilisation d'un modèle de fautes hiérarchique et des simulations au niveau comportemental. Par contre, la méthode de transposition des fautes dans le domaine Z est très complexe. En effet, des fautes uniques dans le domaine linéaire peuvent nécessiter des fautes multiples dans le domaine Z et certaines classes de fautes ne peuvent pas être simulées car elles sont difficilement transposables dans le domaine Z.

2.3.2 Analyse *DC* pour la simulation de fautes des circuits non linéaires

L'avantage principal du test *DC " Direct Current "* par rapport aux autres tests est qu'il engendre un coût de test moins élevé et un temps d'exécution plus rapide sur les équipement de test. Par contre, la simulation de fautes *DC* est très coûteuse en temps de simulation, la difficulté est due à la résolution des équations non linéaires dans les simulateurs électriques qui peut nécessiter beaucoup d'itérations pour converger. Des recherches ont été effectuées pour accélérer la convergence des simulations *DC* pour les circuits fautifs, plusieurs approches ont été proposées dans [Vin98], [TS98a] et [YZ99].

Dans [TS98a], les auteurs proposent une méthode appelée "*One Step Relaxation*", la méthode utilise le résultat de la première itération *Newton-Raphson* dans la simulation du circuit fautif pour prédire la réponse fautive de chaque faute et ordonnancer toutes les fautes de façon à ce que le résultat de chaque circuit fautif soit utilisé comme la valeur de départ pour la simulation de la faute suivante. Cette technique a été appliquée sur des circuits "*benchmark*" analogiques et a permis de réduire le nombre d'itérations moyen par circuit d'un facteur 4.

Une autre approche pour réduire le nombre d'itérations *Newton-Raphson* dans la simulation de fautes *DC* et transitoire pour les circuits non linéaires a été proposée dans [YZ99]. L'algorithme adopté consiste à appliquer des techniques qui permettent de détecter très tôt dans les itérations les fautes non détectables sans attendre la convergence finale (qui peut être très lente) de la simulation du circuit fautif. Les auteurs ont défini une distance qui permet de quantifier la notion de proximité entre le bon circuit et le circuit fautif. A chaque itération la distance entre le circuit correct et le circuit fautif est recalculée en fonction des résultats de toutes les itérations précédentes et si la distance est inférieure à un certain seuil de distance minimum alors la simulation du circuit fautif est stoppée avant la convergence. L'application de la méthode sur un amplificateur *BJT* à deux étages a permis une réduction du temps de simulation de l'ordre de 50%.

2.3.3 La simulation de fautes à haut niveau d'abstraction

La simulation analogique à haut niveau d'abstraction est rendue possible grâce aux langages de description haut niveau pour les circuits analogiques *AHDL*"*Analog Hardware Description Language*" et aux simulateurs supportant ces langages. Contrairement aux circuits numériques ou les langages de description de type *VHDL* et *Verilog* existent depuis plusieurs années, pour les circuits analogiques ces langages sont très récents et n'ont été normalisés que ces dernières années. Comme exemple de simulateurs analogiques supportants les langages de description, nous avons *Saber* et *Eldo* pour le domaine commercial et *iMACSIM* pour le domaine public [SS91]. L'exploitation des langages de description permet d'accélérer de 10 à 30 fois la simulation de fautes. Le principe consiste à partitionner le circuit au niveau transistor en sous-circuits, simuler les fautes de chaque sous-circuit au niveau transistor et ensuite modéliser les sous-circuits et les fautes au niveau comportemental en utilisant un langage de description comportemental. Ainsi les simulations de fautes des circuits analogiques complexes peuvent être effectuées en utilisant les modèles comportementaux des sous blocs du circuit. Cette méthode a été appliquée avec succès sur une boucle à verrouillage de phase *PLL"Phase Locked Loop"* en utilisant un modèle *HDL-A* pour l'oscillateur contrôlé en tension *VCO "Voltage-Controled Oscillator"* [SG97], sur un circuit audio et un filtre *Leapfrog* contenant six amplificateurs opérationels avec des boucles locales et globales en modélisant les *AOPs* utilisés dans les deux circuits avec le langage *VHDL-AMS* [PZCW98].

2.3.4 La simulation de fautes pour les circuits à capacités commutées

La technique des capacités commutées est une technique analogique à temps discret comme pour la technique des courants commutés, c'est la technique la plus utilisée pour le traitement analogique du signal. L'avantage de cette technique est qu'elle est capable de répondre aux exigences de la grande précision avec un coût modéré [BBL+96]. A cause de la présence des interrupteurs, la simulation d'un circuit à capacités commutées avec un simulateur électrique de type *SPICE* ou *ELDO* n'est possible que dans le domaine du temps par une analyse transitoire. Cela implique des temps de simulation très longs, surtout pour la simulation de fautes qui nécessite de répéter la simulation pour chaque circuit fautif.

Les seuls travaux sur la simulation de fautes pour les circuits à capacités commutées ont été présentés dans [MRO+97] et [MRVH98]. Le simulateur de fautes *SWITTEST* permet d'effectuer des simulations plus rapides dans le domaine des fréquences grâce au simulateur *SWITCAP* [SFT90] et des simulations plus précises dans le domaine transitoire grâce au simulateur électrique *HSPICE*. L'avantage de ce simulateur est qu'il permet l'intégration du test très tôt dans la phase de conception des systèmes à base de circuits à capacités commutées.

Par contre pour les circuits à courant commuté, des techniques de test ont été développées [THB93] et [RABB98], mais elle utilisent des simulateurs électriques de type *SPICE* ou *ELDO*, et à notre connaissance il n'existe pas encore de simulateur de fautes spécifique pour les circuits à courant commuté.

2.3.5 La simulation de fautes pour les circuits mixtes

Les simulateurs mixtes permettent d'analyser en une seule étape les circuits mixtes analogique-numérique sans avoir à simuler séparément les deux parties. Les premiers travaux sur la simulation mixte ont été présentés dans [ADP+90], le simulateur *iS-PLICE3* permet un bon compromis entre la vitesse de simulation et la précision des résultats.

La simulation de fautes pour les circuits mixtes analogique-numérique permet de simuler simultanément la partie analogique du circuit en utilisant un simulateur analogique électrique ou comportemental avec un modèle de faute analogique et la partie numérique avec un simulateur de type *Event-Driven* et un modèle de faute numérique. Le premier simulateur de fautes mixtes développé est *MIXER* [NCA93c], il utilise le même principe que le simulateur de fautes *DRAFTS* pour les circuits analogiques linéaires. L'avantage de ce simulateur de fautes est sa rapidité, mais il présente deux inconvénients majeurs qui sont :

- Le simulateur ne permet de simuler que les circuits mixtes dont la partie analogique est linéaire.

- Le simulateur ne simule que les fautes affectant la partie analogique du circuit, il ne permet pas d'injecter et de simuler des fautes dans la partie numérique du circuit.

Un autre travail intéressant sur la simulation de fautes pour les circuits mixtes est présenté dans [CA95] et [CA96]. La simulation de fautes est basée sur le simulateur industriel *Saber* de chez Analogy. Le simulateur *Saber* permet de modéliser et simuler les circuits analogiques et possède un simulateur *Even-Driven* pour les circuits numériques. L'avantage principal de cette approche est l'utilisation de la modélisation et de la simulation au niveau comportemental, ce qui permet d'accélérer considérablement la simulation de fautes. Le gain en temps de simulation de cette méthode sur un additionneur et un convertisseur analogique numérique *A/D* est de l'ordre de 10 en ayant la même précision qu'avec les simulateurs de fautes au niveau transistor.

2.4 Génération automatique des stimuli

Le but de la génération automatique des vecteurs de test *ATPG "Automatic Test Pattern Generation"* est d'obtenir un ensemble de vecteurs de test le plus optimisé possible qui permet de détecter le maximum de fautes. Pour les circuits numériques, il existe plusieurs algorithmes de type *PODEM* qui permettent de générer des vecteurs de test de façon déterministe selon un modèle de faute donné. La majeure difficulté des *ATPG* pour les circuits numériques réside dans la complexité importante des circuits (plusieurs millions de transistors et plus d'une centaine d'entrées sorties). Par contre pour les circuits analogiques, nous avons des circuits beaucoup plus petits en nombre de composants et d'entrées sorties, mais dont la spécificité ne permet pas d'utiliser des algorithmes de génération automatique comme dans le cas des circuits numériques.

Jusqu'à ces dernières années les vecteurs de test pour les circuits analogiques et mixtes sont générés manuellement. En général, les concepteurs utilisent les vecteurs fonctionnels pour le test structurel, ce qui fait que les circuits analogiques sont soit sous-testés soit sur-testés, ce qui augmente considérablement le coût du test. En plus, la croissance des circuits analogiques ces dernières années a rendu de plus en plus difficile la génération manuelle des vecteurs de test, c'est pourquoi plusieurs travaux commencent à apparaître dans le domaine de la génération des vecteurs de test pour les circuits analogiques et mixtes, mais leur maturité reste très inférieure à celle atteinte par la génération des vecteurs de test pour les circuits numériques.

La génération automatique des vecteurs de test pour les circuits analogiques peut être décomposée en plusieurs domaines : la génération pour le test structurel, l'optimisation des vecteurs de test , la génération pour les circuits mixtes analogique-numérique et la génération pour le diagnostic. Les sections suivantes présentent en détail les différents travaux de recherche qui ont été publiés dans chacun de ces domaines.

2.4.1 Génération automatique des vecteurs pour le test structurel

Le but du test structurel ou de production est de distinguer les circuits corrects des circuits défectueux pendant la phase de fabrication en grande série des circuits. Le coût et la qualité du test de production sont étroitement dépendants du jeu de vecteurs de test choisis, il est donc important d'avoir un ensemble minimum de vecteurs de test pour réduire le coût et que cet ensemble puisse détecter la plupart des fautes pour répondre aux exigences de la qualité. L'objectif de la génération automatique des vecteurs pour le test structurel est de répondre à ces deux exigences.

Une bonne partie des travaux de recherche effectués dans ce domaine concerne la génération des vecteurs de test pour les circuits analogiques linéaires. Les deux techniques proposés dans [NCBA93] et [HKSZ99] permettent de générer un ensemble minimum de fréquences de test qui assurent un taux de couverture maximum. La première approche analyse les fautes une à une et recherche pour chaque faute la fréquence qui donne la différence maximale en sortie entre le bon circuit et le circuit fautif. La deuxième méthode utilise un facteur de testabilité *TTF "Testability Transfer Factor"* pour déterminer les meilleures fréquences. Une méthode de calcul du facteur *TTF* a été déterminée pour la majorité des composants analogiques, le facteur *TTF* est ensuite utilisé pour formuler les équations d'observabilité et de contrôlabilité de chaque nœud du circuit. La matrice des équations ainsi formulées est résolue pour déterminer les mesures de testabilité de chaque nœud en fonction de la fréquence, ces mesures de testabilité sont ensuite utilisées pour déterminer les degrés de détectabilité de chaque faute en fonction de l'intervalle de fréquence utilisé ce qui permet de déterminer et classer les meilleures fréquences à appliquer au circuit sous test.

Pour [ACK96] et [ACK99] la méthode proposée permet de générer les meilleures fréquences pour les fautes paramétriques en tenant compte des variations du processus de fabrication. Dans [ACK96] la génération des test est formulée comme un problème non linéaire d'optimisation, résolu avec la méthode de programmation *SQP "Sequential Quadratic Programming"* disponible dans *MATLAB*. Par contre la technique de génération proposée dans [ACK99] est basée sur le langage *CLP "Constraint Logic Programming"*. *CLP* est un langage de programmation logique capable de résoudre des systèmes d'équations linéaires et d'inégalités, son utilisation avec les opérations de l'arithmétique des intervalles permet de prendre en compte des tolérances sur les paramètres du circuit et de choisir les meilleures fréquences pour lesquelles il est utile de faire le test du circuit.

Une autre approche présentée dans [PC99] permet de prendre en compte des variations du processus de fabrication dans la génération automatique des vecteurs de

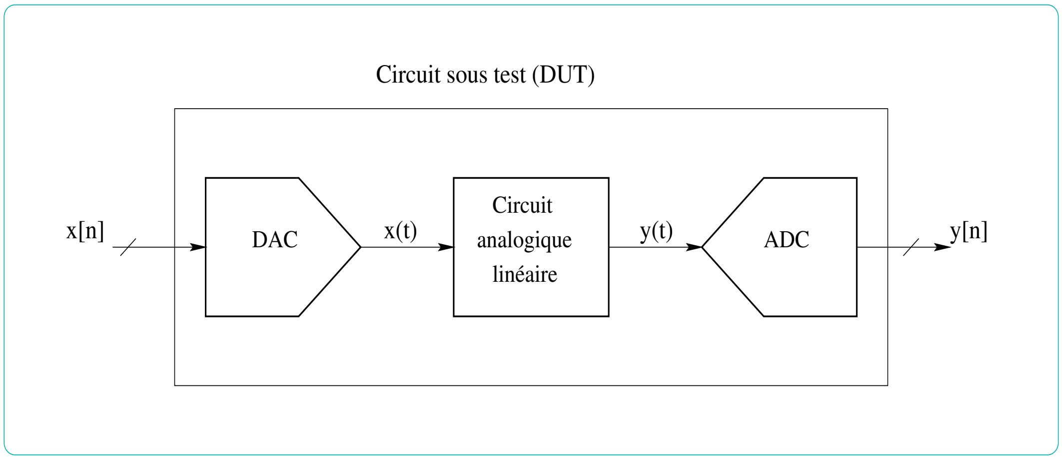

FIG. 2.1: *Technique de test numérique pour un circuit analogique.*

test pour les circuits analogiques linéaires. La technique proposée considère le circuit analogique comme un circuit numérique et utilise deux convertisseurs numérique analogique *DAC* et analogique numérique *ADC* à l'entrée et à la sortie du circuit sous test *DUT " Device Under Test "* (voir figure 2.1). Le circuit sous test peut ainsi être testé avec les techniques de test numérique.

2.4.2 Optimisation automatique des vecteurs de test

La formulation du problème d'optimisations des vecteurs de test consiste en : soit un ensemble de vecteurs de test donné et un ensemble de fautes à détecter, le coût du test dépend du nombre de vecteurs de test à appliquer et de l'ordre dans lesquels ces vecteurs sont appliqués. En effet plus on a de vecteurs plus le temps de test est grand. Comme le test d'un circuit fautif est arrêté dés que la faute est détectée, il est avantageux en terme de temps de test, de placer les vecteurs qui détectent le plus de fautes au début du test. Le but des techniques d'optimisation des tests est de réduire le nombre de vecteurs dans l'ensemble de test initial et de classer les vecteurs sélectionnés pour réduire le coût du test, tout en conservant le même taux de couverture que celui de l'ensemble de départ [Mil98].

La majorité des travaux de recherche sur la génération automatique des vecteurs pour le test de production des circuits analogiques, avec prise en compte des déviations sur les paramètres, utilisent des techniques d'optimisation [DS94] [GWS99] [VdPG97] [ST98] et [DSGH99]. Les méthodes présentées dans [GWS99] et [VdPG97] utilisent des simulations Monte Carlo pour simuler les perturbations dues au processus de fabrication, l'inconvénient majeur de cette technique est le temps *CPU* excessif que nécessite ce type de simulation. Dans [DS94], l'auteur propose une méthode d'optimisation des tests pour le test des fautes catastrophiques en mode *DC*, la méthode est basée sur la simulation pire cas "*Worst Case Simulation*" du bon circuit et des circuits fautifs. Cette technique a été généralisée dans [DSGH99] pour les différents types de test et permet un gain de 3 en moyenne par rapport à la technique Monte Carlo.

L'approche proposée dans [ST98] pour le test des circuits linéaires permet de déterminer les meilleures fréquences de test en tenant compte des tolérances sur les composants. Cette méthode utilise la simulation de fautes avec prise en compte des tolérances définie dans [TS97], elle consiste à représenter le circuit linéaire avec les variations de ces paramètres, comme un système d'équations linéaires d'intervalles. La résolution de ces équations linéaires d'intervalles permet d'obtenir les sorties du bon circuit et des circuits fautifs avec leur intervalle de tolérance, ce qui permet de choisir efficacement les fréquences à appliquer pour le test de production.

Une autre méthode d'optimisation des tests pour les circuits analogiques avec prise en compte des variations du processus de fabrication a été développée dans [HK93b]. La méthode est basée sur l'analyse de sensibilité "*Sensibility Analysis*" des composants du circuit et permet de simuler des fautes multiples. La technique a été intégrée dans le premier outil de test industriel pour les circuits analogiques et mixtes *LIMsoft* [HSM$^+$96] de chez *OPMAXX*.

2.4.3 Génération des vecteurs de test pour les circuits mixtes analogique-numérique

Les techniques récentes de conception des circuits intégrés permettent l'intégration des fonctions numériques et analogiques sur le même circuit. En contre partie, le test de ces circuits mixtes n'est pas une tâche facile, elle est plus complexe que le test d'un circuit purement numérique ou purement analogique. La difficulté dans la génération des tests pour les circuits mixtes est due à l'interface analogique-numérique et en particulier à la contrôlabilité et l'observabilité des fautes de la partie numérique à travers la partie analogique. A cause de ces difficultés, certaines techniques de test utilisent des méthodes de conception en vue du test pour découper le circuit en deux parties, l'une analogique et l'autre numérique, de manière à pouvoir accéder séparément aux entrées sorties de chaque bloc. Une des implémentation de cette technique est l'utilisation du bus de test standard *IEEE1149.4* pour les circuits analogiques et mixtes (voir section 2.5.2).

A notre connaissance, il n'existe pas beaucoup de travaux publiés dans le domaine de la génération des vecteurs de test pour les circuits mixtes analogique-numérique sans modification du circuit et les quelques méthodes qui existent sont en général limitées à certain type de circuits comme les convertisseurs analogique-numérique [Max89]. Une autre méthode de génération est présentée dans [ABK95], mais l'application de cette technique est aussi limitée, car elle ne considère que la propagation des fautes analogiques à travers la partie numérique et seulement pour certaines catégories de circuits mixtes.

2.4.4 Génération des vecteurs de test pour le diagnostic

Le but du test structurel est de distinguer les bon circuits des circuits fautifs, alors que le but du diagnostic est de distinguer les composants défectueux des composants corrects du circuit défectueux. Le test sert à détecter les défauts et le diagnostic à localiser l'origine des défauts dans les circuits défectueux. Le diagnostic permet d'améliorer la conception des circuits intégrés dans le but d'augmenter le rendement des circuits corrects dans la phase de fabrication en grande série.

Contrairement au test structurel ou le but de la génération des vecteurs de test est de trouver un ensemble minimum de vecteurs de test pour réduire le coût du test de production, l'objectif de la génération pour le diagnostic est d'obtenir un ensemble de vecteurs de test qui permettent de distinguer les circuits défectueux entre eux. Il existe deux méthodes générales pour le diagnostic des circuits intégrés nommées *SBT " Simulation Before Test "* et *SAT " Simulation After Test "* [Mil98]. La première approche est basée sur l'utilisation d'un dictionnaire et d'un simulateur de fautes et les circuits défectueux sont diagnostiqués en comparant les réponses simulées avec les réponses mesurées [PR82]. Par contre, la deuxième approche utilise les réponses mesurées du circuit défectueux pour estimer les paramètres incorrects ou les valeurs des composants défectueux [FMPS99] et [CC99].

Contrairement aux circuits numériques, la génération des vecteurs de test pour le diagnostic des circuits analogiques n'est encore qu'à ses débuts. Un des travaux récents dans le domaine est présenté dans [MLC96b] et [MLC96a], l'algorithme de génération proposé est à base de dictionnaire de fautes *(SBT)* et permet de générer les fréquences qui assurent un diagnostic maximum. La méthode n'est valable que pour les circuits analogiques linéaires.

2.5 Conception en vue du test

Avec la complexité croissante des circuits intégrés, le test et le diagnostic des circuits devient de plus en plus difficile. En effet, durant ces dernières années, la phase de conception a été considérablement raccourcie grâce à l'utilisation des outils de *CAO* puissants, alors que le test a continué à être réalisé principalement en fin de cycle de conception. De ce fait, les coûts de développement du test sont devenus les plus élevés du coût total du développement d'un circuit [PZ97] et [Rob97].

La conception en vue du test *DFT " Design For Testability "* permet de prendre en compte les problèmes du test très tôt dans l'étape de conception des circuits et éviter ainsi de faire des choix rendant le test difficile. le but de la conception en vue du test est donc de réduire le coût total du test et améliorer ainsi la fiabilité des circuits. Pour les circuits numériques, la plupart des concepteurs sont aujourd'hui convaincus de la nécessité d'intégrer la testabilité dans la conception de leurs circuits et plusieurs outils commerciaux d'aide à la conception en vue du test sont disponibles. De nos jours, l'application d'un certain type de techniques de conception en vue du test *DFT*, tels que le chemin de test *SCAN-PATH*, le test intégré *BIST* et le bus de test des frontières *Boundary-Scan* est obligatoire pour garantir la fiabilité des circuits et des systèmes. Ces techniques de conception en vue du test nécessitent l'ajout des structures de test qui ne serviront qu'à tester le circuit, ces structures de test engendrent un surcoût en terme de surface de silicium et peuvent avoir un impact sur les performances du circuit.

Pour les circuits analogiques et mixtes, les techniques de conception en vue du test ont commencé à émerger ces dernières années et notamment pour le test intégré *BIST* ou plusieurs techniques ont été développées . La spécificité des circuits analogiques ne permet pas la transposition directe des techniques de conception en vue du test développées pour les circuits numériques, et la diversité des modes de test et des paramètres de mesure des circuits analogiques (voir section 1.1.4 et 1.1.5) rendent très difficile et très complexe l'ajout des structures de test. En outre, l'ajout des structures de test pour les circuits analogiques peut engendrer des dégradations considérables des performances du circuit. Dans les sections suivantes nous allons présenter les différentes techniques qui ont été proposées pour la conception en vue du test pour les circuits analogiques.

2.5.1 Le test intégré analogique

Le test intégré est une technique de conception en vue du test qui consiste à intégrer au sein même de l'unité à tester les différents modules de génération de vecteurs de test, d'analyse des signatures et du contrôle de test (voir figure 2.2. Le test intégré permet d'exécuter le test à vitesse réelle et de diminuer la complexité des équipements automatiques de test *ATE*, ce qui permet de réduire le coût total du test. En plus de la rapidité et de l'efficacité, le test intégré permet la hiérarchisation des solutions de test et peut être réutilisé à différents niveaux (puce, carte et systeme) et à différents stades du cycle de vie du produit tels que sa fabrication, son installation et son exploitation [Rob97].

Le test intégré pour les circuits analogiques et mixtes *MAD BIST " Mixed Analog Digital Built-In Self-Test* " est la technique de conception en vue du test la plus appropriée et la plus utilisée en industrie. Un brevet pour le test d'un convertisseur analogique numérique *ADC* avec la technique du *BIST* a été déposé en 1991 par les laboratoire AT&T Bell [Vin98]. Au même moment la première technique de test intégré pour les circuits mixtes a été publiée dans [Ohl91]. Cette technique appelé *HBIST* pour "*Hybrid Built-In Self-Test*" utilise une méthode de *BIST* classique pour la partie numérique. Pour la partie analogique la technique utilise les mêmes registres à décalages avec des convertisseurs *DAC* et *ADC* pour convertir les signaux (voir figure 2.3). Une autre approche appelé *T-BIST* pour "*Translation Built-In Self-Test*" a été proposée dans [SK93]. La méthode présentée consiste à vérifier que les paramètres testés sont dans leur intervalle de tolérance, elle est basée sur la conversion de chaque paramètre mesuré en une tension, ensuite cette tension est comparée aux deux tensions de référence représentant les limites supérieure et inférieure de l'intervalle de tolérance.

Contrairement aux circuits numériques où on utilise des *LFSRs* et *MISRs* pour la

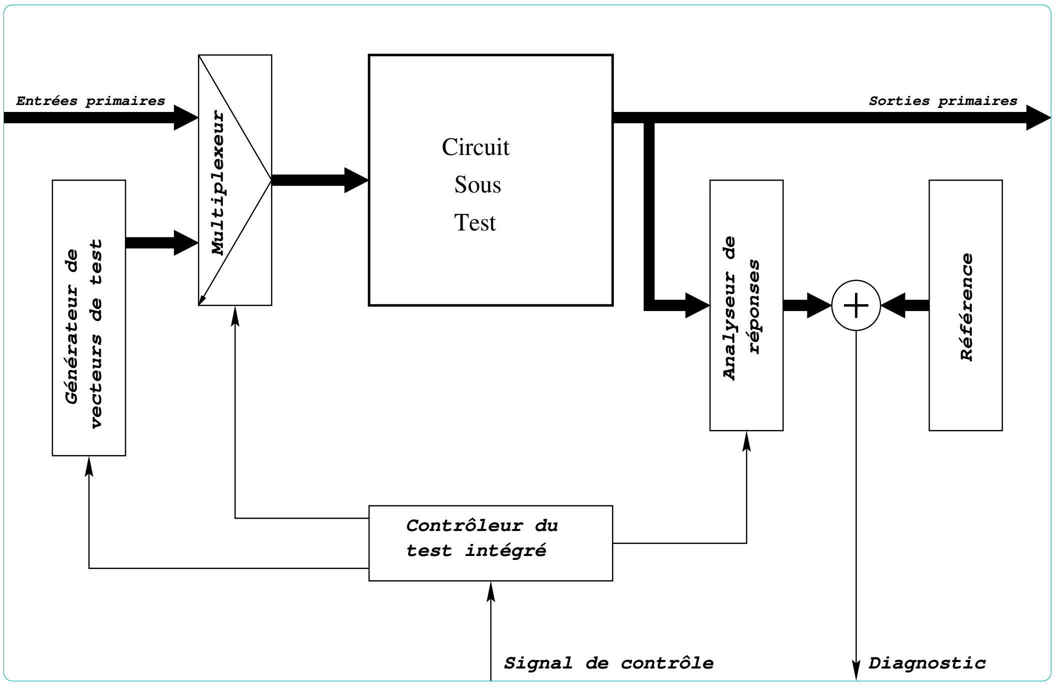

FIG. 2.2: *Architecture générique du test intégré.*

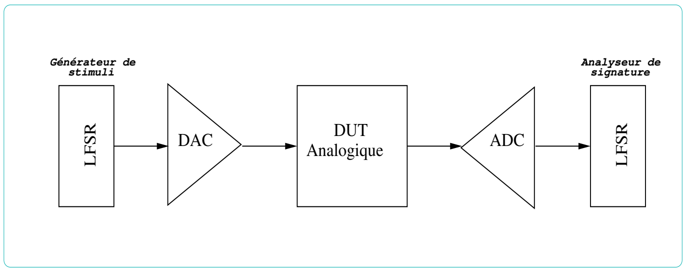

FIG. 2.3: *Architecture du HBIST "Hybrid Built-In Self-Test".*

génération des vecteurs de test et l'analyse des réponses, pour les circuits analogiques la générations interne des différents type de signaux (pulse, rampe ou sinusoïdale) et l'analyse des réponses analogiques sont des problèmes plus complexes et nécessitent la conception de circuits spécifiques. Différents circuits de générations de signaux analogiques pour le test intégré ont été proposés dans [HR95] [LRM$^+$98] et [DR99]. Les différentes techniques de générations proposées utilisent soit des oscillateurs programmables soit des registres à décalage *LFSRs* suivis de convertisseur numérique analogique pour convertir les signaux numériques en sortie du *LFSR* en signaux analogiques.

Pour la compression des réponses et l'analyse de signature analogique, plusieurs travaux de recherche ont été publiés. Dans [NCA94] la méthode proposée utilise un intégrateur numérique pour la compression des réponses, cette approche permet de résoudre le problème des imprécisions des signaux analogiques en considérant les tolérances. L'approche présentée dans [PC95] utilise des convertisseurs *ADC* et *DAC* pour obtenir une interface numérique du circuit analogique et ensuite utilise la technique du *BIST* numérique en utilisant des registres à décalage *LFSR*. L'idée principale de la technique présentée dans [RAB97] est de remplacer l'additionneur discret utilisé pour l'analyse de signature des circuits numériques par un intégrateur analogique de réponses transitoires. L'intégrateur pour les circuits à capacités commutées proposé dans cette approche est à base d'amplificateur opérationnel et permet d'analyser en parallèle plusieurs nœuds internes du circuit.

L'utilisation du test intégré pour les circuits analogiques et mixtes commence à intéresser de plus en plus d'entreprises de conception de circuit intégrés. Des implémentations du *BIST* ainsi que des outils commerciaux d'aide à l'insertion du *BIST* pour les circuits analogiques et mixtes commencent à être développés et utilisés par plusieurs entreprises. La première chaîne d'outil d'aide à l'insertion du test intégré *BISTMaxx* de chez *OPMAXX* permet d'insérer du *BIST* pour différents types de circuits analogiques et mixtes (*VCO*, *ADC* et *DAC*) [FA97] et [AK97]. De même *Logic Vision* propose des solutions pour les tests intégrés des convertisseurs *ADC* et des boucles à verrouillage de phase *PLL*. Ces solutions sont présentées dans les deux offres *adcBIST* et *pllBIST* intégrant les outils automatiques pour faciliter la synthèse des circuits intégrant le *BIST*.

2.5.2 Bus de test *Boundary Scan IEEE 1149.4*

Le bus de test *Bounadary Scan* est une norme qui vise à simplifier le test des circuits sur une carte et à procurer une interface normalisée pour mettre en œuvre les tests des circuits et des cartes. Le but du *Boundary Scan* est de résoudre le problème d'isolement des différents circuits de la carte pendant le test et aussi de permettre le test des interconnexions entre les différents circuits. Pour les circuits numériques la norme *IEEE 1149.1* a été finalisée en 1990 [Par92], cette norme prévoit l'ajout au circuit de quatre ou cinq plots (le plot de remise à zero *TRST* est optionnel), du *Scan Path* sur toute sa périphérie et d'un contrôleur d'accès aux ports du circuit sous test "*TAP Contoller*" (voir figure 2.4).

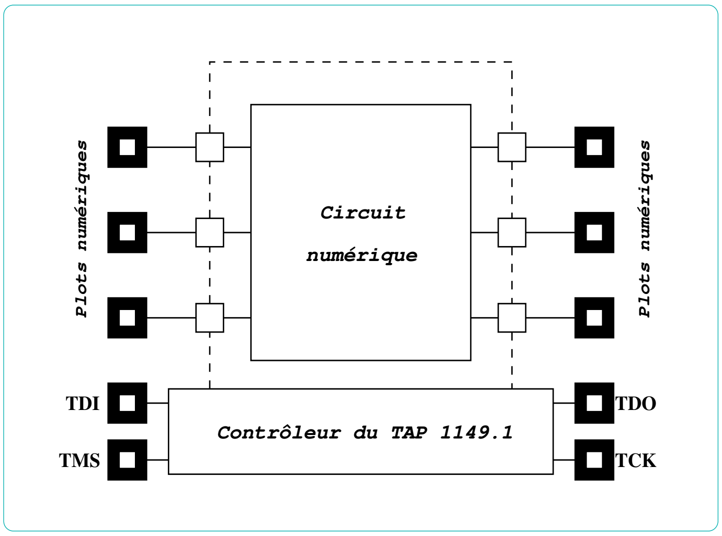

FIG. 2.4: *IEEE 1149.1: Architecture du Boundary-Scan pour les circuits numériques.*

Durant ces cinq dernières années un groupe de travail *IEEE* international regroupant les compagnies et les instituts de recherche travaille pour le développement d'un bus de test pour les circuits analogiques et mixtes qui soit compatible avec la norme *IEEE 1149.1*. Ces travaux ont donné naissance à la norme *IEEE 1149.4* qui est une extension de la norme *IEEE 1149.1* pour les circuits analogiques et mixtes [Sun95], [Rob97] et [Vin98]. La nouvelle norme *IEEE 1149.4* prévoit l'ajout à la norme *IEEE 1149.1* les éléments suivants (voir figure 2.5):

1. Un ensemble de ports de test analogique appelé *ATAP* pour "*Analog Test Access Port*" qui comprend au moins deux plots *AT1* et *AT2*. La norme prévoit un fil de bus unique appelé aussi *AT1* pour acheminer le signal d'entrée analogique sur *AT1* au nœud souhaité du circuit. De même pour le bus *AT2* qui permet d'acheminer la réponse du nœud souhaité vers la sortie analogique *AT2*.

2. Deux bus de test analogiques " *on-chip* " *AB1* et *AB2*. L'utilisation de ces bus permet d'éviter les problèmes dus aux capacités parasites, aux courants de fuite et aux interférences pour les hautes fréquences et les circuits très sensibles.

3. Un module de frontière analogique appelé *ABM* " *Analog Boundary Module* " pour chaque plot analogique. Ces modules analogiques doivent être inclus dans le registre *Boundary Scan* global contenant les modules de frontière de tous les plots d'entrées sorties numériques.

4. Un circuit d'interface pour les bus de test appelé *TBIC* pour " *Test Bus Interface*

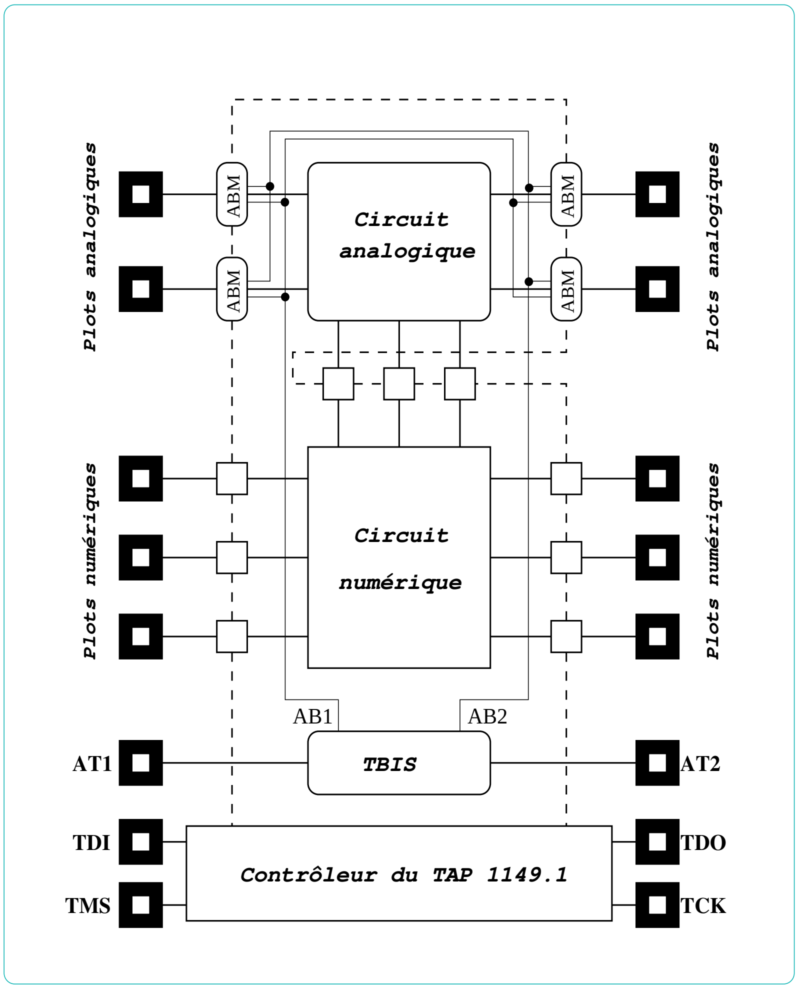

FIG. 2.5: *IEEE 1149.4: Architecture du Boundary-Scan pour les circuits mixtes.*

Circuit ". Le module *TBIC* est composé de quatre interrupteurs qui permettent d'interconnecter les bus " *on-chip* " *AT1* et *AT2* avec les bus " *off-chip* " *AB1* et *AB2*. Ces interrupteurs permettent aussi de déconnecter les bus " *on-chip* " *AB1* et *AB2* des circuits inutiles pour les mesures spécifiques, ce qui permet de minimiser les courants de fuite , les capacités parasites des bus et les interférences avec les autres circuits.

Les avantages de l'utilisation de la norme *IEEE 1149.4* sont multiples, et les deux plus importantes pour lesquels le groupe de normalisation a développé ce standard sont :

1. Réduire le coût des tests au niveau carte et des tests d'interconnexions pour les cartes contenant des circuits analogiques et mixtes en supprimant le test " bed-of-nails ".

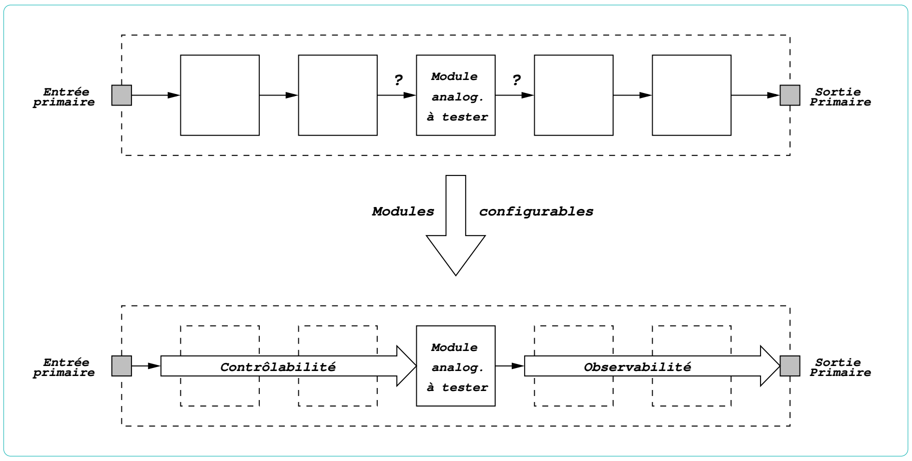

FIG. 2.6: *Technique des modules configurables.*

2. Réduire le temps de développement des tests pour les circuits analogiques et mixtes en permettant le développement d'outils d'automatisation et de réutilisation " re-used " qui sont potentiellement moins coûteux à utiliser que les solutions de test internes développées par chaque fabricant.

2.5.3 Les modules configurables ou le *Scan Path* analogique

Pour les circuits numériques, le *Scan Path* est une technique de modification systématique des circuits séquentiels synchrones qui permet de couper les rebouclages pendant la phase de test. La modification consiste à créer un mode test ou les éléments mémorisants du circuit (à l'exception des mémoires) constituent un registre à decalage accessible directement de l'extérieur du circuit. Ces points mémoires deviennent ainsi à la fois contrôlables et observables à partir des entrées sorties primaires rendant plus facile le test du ciruit.

Comme pour le *Scan Path*, le but des techniques à base de modules configurables est d'assurer la contrôlabilité et l'observabilité complète de certains nœuds importants du circuit en modifiant les chemins déjà existants dans le circuit de manière à établir un chemin sensible de propagation. Il s'agit donc de contrôler l'entrée et d'observer la sortie de chaque module à travers les modules précédant et suivant du circuit (voir figure 2.6) [Aza96].

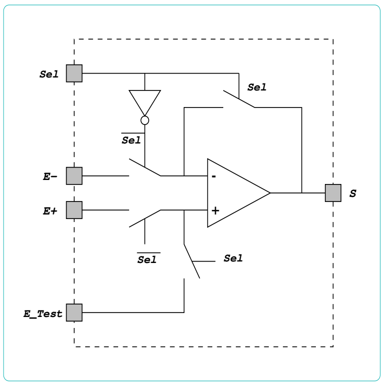

FIG. 2.7: *Schéma de l'amplificateur opérationnel configurable.*

Une première technique proposée dans [Som90] permet de propager un signal à travers certains modules du circuit. Cette approche fournit une contrôlabilité et une observabilité partielle puisque la propagation d'un signal s'effectue avec une modification du signal par le gain. Une autre approche a été proposée récemment dans [Aza96] et [RAB96], les auteurs proposent une méthode de conception en vue du test dédiée aux circuits analogiques constitués de modules simples cascadés, chaque module étant construit autour d'un amplificateur opérationnel et de composants passifs. La méthode proposée consiste à remplacer l'*AOP* classique par un *AOP* configurable qui fonctionne en deux modes : le mode normal où l'*AOP* réalise sa fonction d'amplification et le mode test où l'amplificateur est activé en mode suiveur. Dans le mode suiveur, l'entrée de l'*AOP* est recopiée vers la sortie de l'*AOP*, ce qui rend l'amplificateur transparent. Le fonctionnement de l'*AOP* reconfigurable est identique au fonctionnement des bascules avec *Scan-Path* pour les circuits numériques. L'*AOP* configurable est réalisé en rajoutant des interrupteurs *MOS* autour d'un amplificateur opérationnel classique (voir figure 2.7). Comme les *AOPs* configurables sont plus coûteux que les *AOPs* classiques, une méthode de réduction du nombre d'*AOP* configurables dans les circuits a été présentée dans [RAB98].

2.6 Conclusion

Dans ce chapitre, nous avons présenté les différents travaux publiés dans le domaine du test des circuits analogiques. Dans la première section du chapitre, nous avons introduit quelques définitions des termes importants pour faciliter la lecture du manuscrit.

Pour les circuits analogiques, il existe deux catégories de fautes : les fautes paramétriques et les fautes catastrophiques. Pour réduire le nombre de fautes à simuler, des méthodes d'analyse de fautes par induction *IFA* ont été développées. Ces méthodes d'*IFA* permettent d'extraire les fautes les plus probables d'un circuit à partir de sa vue physique "*layout*". Plusieurs travaux ont été publiés dans le domaine de la modélisation de fautes, mais contrairement aux circuits numériques, il n'existe pas encore de modèles de fautes standard pour les circuits analogiques.

Pour la simulation de fautes, l'un des problèmes majeurs est la prise en compte des tolérances dues aux variations du processus de fabrication. Il existe plusieurs travaux qui ont été publiés mais qui ne considèrent que certains types de circuits comme les circuits linéaires ou certains types d'analyse comme l'analyse *DC*. Les approches proposées ne permettent pas de définir avec précision l'efficacité des ensembles de tests, c'est pourquoi nous avons choisi de développer une méthodologie de simulation de fautes basée sur les probabilités de détections de fautes qui permettent de calculer avec précision l'efficacité des ensembles de tests.

De même que pour la simulation de fautes, la génération automatique des vecteurs de tests pose le problème de la prise en compte des tolérances sur les paramètres des circuits analogiques. De plus, pour les circuits analogiques, les vecteurs de test dépendent de la nature du circuit à tester. Il est donc impossible de développer un générateur automatique de vecteurs de test pour tous les types de circuits, c'est pourquoi nous avons attaqué le problème de l'optimisation automatique des ensembles de tests déjà pré-existants.

En ce qui concerne les techniques de conception en vue du test *DFT*, l'ajout de structures de test pour les circuits analogiques est plus complexe que pour les circuits numériques. En effet, les circuits analogiques sont beaucoup plus sensibles que les circuits numériques. Malgré cette complexité, ces techniques de *DFT* commencent à être développées et notamment le test intégré *BIST*, ce qui nécessite de développer des simulateurs de fautes efficaces pour pouvoir évaluer avec précision les améliorations apportées par ces techniques de *DFT*.

Chapitre 3

Concepts de base

Un des problèmes majeurs du test des circuits analogiques est la nature imprécise des paramètres des circuits analogiques. L'imprécision des signaux analogiques est due aux variations aléatoires du processus de fabrication, au bruit des générateurs de signaux et aux imprécisions des instruments de mesure. Les méthodes de probabilités et statistiques permettent de prendre en compte ces variations dans la phase de simulation de fautes et l'optimisation des vecteurs de test. Dans la première partie de ce chapitre, nous allons introduire les éléments de probabilités et statistiques nécessaires à la compréhension de la suite de la thèse.

La simulation de fautes et l'optimisation des vecteurs de tests nécessitent l'utilisation d'un modèle de fautes représentatif des défauts physiques qui peuvent affecter le circuit. Dans la deuxième partie du chapitre nous allons présenter les différents modèles de fautes existant pour les circuits analogiques et ceux que nous avons adoptés pour la validation de notre travail.

3.1 Probabilités et statistiques

Pour avoir des simulations précises et des vecteurs de test efficaces, il faut inclure dans les simulations les déviations des paramètres du circuit dues aux fluctuations du processus de fabrication. Malgré les ordinateurs puissants et leur grande capacité de traitement des données, il est impossible de simuler toutes les déviations possibles du processus de fabrication. Il ne nous reste donc que l'outil statistique pour étudier et maîtriser les conséquences des variations aléatoires du processus de fabrication.

3.1.1 Définition de la probabilité

On considère l'ensemble E des éventualités possibles résultant d'une épreuve (expérience, observation ou simulation), chacune de ces éventualités étant appelée événement élémentaire. Un événement quelconque est défini comme un sous ensemble A de E contenant tous les événements élémentaires de E composant l'événement A. La probabilité attachée à un événement A est un nombre $P(A)$ compris entre 0 et 1, obéissant à certaines règles axiomatiques semblables à celles d'une répartition d'une masse totale unitaire sur l'ensemble des éléments de E. En particulier :

1. L'événement de l'ensemble $\emptyset$ (qui ne se produit jamais) a une probabilité $P(\emptyset) = 0$.

2. L'événement E (qui se produit toujours) a une probabilité $P(E) = 1$.

3. $\forall A \subseteq E$ nous avons, $0 \leq P(A) \leq 1$.

4. $\forall A, B \subseteq E$ nous avons, $P(A \cup B = P(A) + P(B)$ si $A \cap B = \emptyset$ (axiome d'additivité).

Le problème de l'attribution de probabilités à un ensemble d'événements peut être résolu dans un certain nombre de cas de la façon suivante :

1. Si les événements élémentaires sont en nombre fini et qu'ils jouent un rôle symétrique, on répartit de façon égale la probabilité $P(E) = 1$ entre eux. La probabilité d'un événement A quelconque est alors égale au rapport du nombre d'événements élémentaires dans A au nombre total d'événements élémentaires possibles.

2. Si les événements élémentaires sont en nombre fini et qu'ils ne jouent pas un rôle symétrique, on peut procéder à une série de répétitions de l'épreuve : la fréquence de chaque événement est une estimation de sa probabilité.

3. Si les événements sont en nombre infini, on peut définir sur cet ensemble une densité de répartition de probabilité (voir section 3.1.2) analogue à une densité de masse.

3.1.2 Les distributions de fréquence ou densités de probabilités

Dans le cas où le nombre d'événements élémentaires de l'ensemble E est infini, on ne peut pas attribuer une probabilité non nulle à chaque événement élémentaire car la

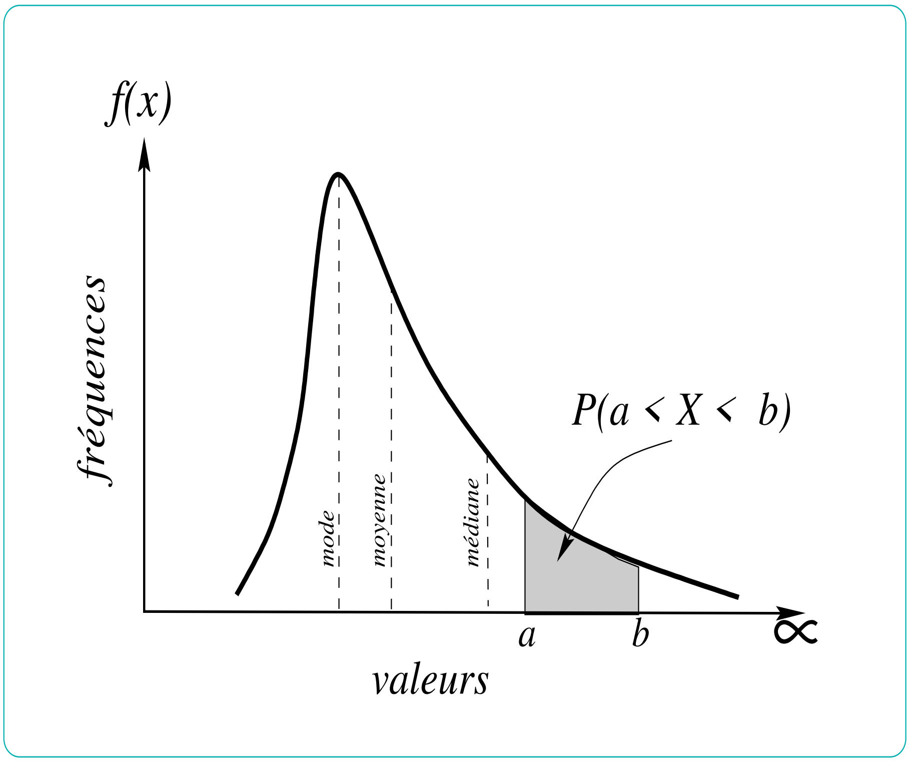

FIG. 3.1: *Calcul des probabilités à partir des distributions de probabilités.*

somme totale de toutes les probabilités sera égale à l'infini. La probabilité de chaque événement élémentaire ne pourrait être que nulle. Il convient donc de trouver une formulation plus générale des probabilités.

La solution de ce problème consiste à définir une fonction de distribution de fréquence des événements quelconques, et non plus des seuls événements élémentaires. Chaque événement A est caractérisé par un intervalle $[a, b]$, la fonction de densité de probabilité permet de calculer la probabilité de l'événement A en calculant l'aire située sous la courbe de densité de probabilités entre les valeurs a et b (voir figure 3.1).

Nous noterons en général X une variable aléatoire et $x_1, x_2, \cdots x_n$ chacune des valeurs qu'elle peut prendre. La probabilité d'un événement A défini par X appartenant à l'intervalle $[a, b]$ est donc donnée par l'intégrale suivante :

$$\boxed{P(A) = P(a < X < b) = \int_a^b f(x)\,dx} \tag{3.1}$$

On remarque que la probabilité de l'événement E est donné par la surface globale (intégrale) de la fonction de distribution qui doit donc être égale à 1 selon les axiomes définissant les probabilités (voir section 3.1.1).

Il existe deux types de mesures pour caractériser les distributions :

1. Les mesures de position
2. Les mesures de dispersion

Le but de ces mesures est de définir des méthodes mathématiques permettant de mesurer objectivement certaines tendances ou caractéristiques d'un ensemble de données. On cherche donc à résumer les distributions de fréquence par quelques valeurs numériques caractéristiques.

3.1.3 Les mesures de position ou de tendance centrale

Les mesures de position permettent de caractériser l'ordre de grandeur des variables aléatoires. Elles indiquent globalement où se situent principalement les valeurs prises par la variable statistique. Les caractéristiques de position les plus fréquentes sont les mesures de tendance centrale permettant de caractériser la valeur centrale de la variable. Comme les variables statistiques ont tendance à se concentrer autour d'une certaine valeur, il est intéressant de déterminer cette valeur centrale. Il existe 3 mesures de tendance centrale qui sont : la moyenne arithmétique, la médiane et le mode.

La moyenne ou l'espérance mathématique

La mesure de tendance centrale la plus importante et la plus utilisée pour caractériser les distributions de probabilités d'une variable aléatoire X est la moyenne μ aussi appelée l'espérance mathématique et notée $E(X)$.

Pour le cas où les valeurs possibles sont discontinues et en nombre fini, la moyenne de la variable aléatoire X prenant les valeurs $x_1, x_2, \cdots x_n$ avec les probabilités $p_1, p_2, \cdots p_n$ est donnée par l'expression :

$$\boxed{\mu = E(X) = \sum_{i=1}^{n} p_i x_i} \tag{3.2}$$

L'expression 3.2 s'étend sans difficulté au cas d'une variable continue X, définie par une loi de distribution $f(x)$ sur l'intervalle $[a, b]$ avec l'expression suivante :

$$\boxed{\mu = E(X) = \int_a^b x f(x) dx} \tag{3.3}$$

La médiane

La médiane d'une variable statistique X est, par définition, la valeur numérique telle qu'il y a au plus 50% des valeurs de la variable qui lui soient inférieures et au plus 50% des valeurs de la variable qui lui soient supérieures (voir figure 3.1).

Le mode

Le mode est la mesure de tendance la plus simple à calculer mais la moins utilisée. Pour une variable X, le mode est défini comme la valeur qui a été observée le plus souvent ou encore la valeur pour laquelle la fréquence d'occurrence (la fonction de distribution) est maximum (voir figure 3.1).

3.1.4 Les mesures de dispersion

La tendance centrale d'une distribution est une caractéristique essentielle mais ne peut pas suffire à bien caractériser la distribution ; en effet il est important de savoir de quelle manière les valeurs s'en écartent. On peut dire qu'une forte concentration des valeurs de la variable autour d'une valeur centrale donne à cette valeur une signification accrue.

Par exemple, si on considère deux techniques de conception d'amplificateurs opérationnels " *AOPs* ", on mesure les gains d'une série de chaque type d'amplificateur et on trouve la même valeur moyenne qui est de $70dB$. Le problème est que le premier type d'amplificateur donne des valeurs qui s'étalent de 50 à $90dB$ alors que les mesures du deuxième type d'amplificateur donnent des valeurs plus compactes qui varient de 65 à $75dB$. Si, dans le cahier des charges, l'*AOP* est considéré bon pour un gain ≥ 65, on voit facilement que le rendement pour le deuxième type d'amplificateur sera meilleur. Le gain moyen des amplificateurs est important, mais la dispersion autour de cette moyenne l'est tout autant.

Il existe plusieurs mesures de dispersion : parmi ces mesures, nous avons l'étendue, l'écart moyen et l'écart-type. Dans la suite de cette section, nous allons uniquement introduire l'écart-type qui sera utilisé dans la suite du manuscrit.

L'écart-type (σ)

La mesure de dispersion la plus utilisée est l'écart-type noté σ. Dans le cas des variables discontinues et finies, σ est défini par l'expression :

$$\boxed{\sigma = \sqrt{VAR(X)} = \sqrt{\sum_{i=1}^{n} p_i(x_i - \mu)^2}} \tag{3.4}$$

De même que pour la moyenne, la définition de l'écart-type s'étend facilement pour le cas d'une distribution continue $f(x)$, en remplaçant la somme par l'intégrale et la probabilité p_i par $f(x)dx$. En final nous obtenons :

$$\boxed{\sigma = \sqrt{VAR(X)} = \sqrt{\int_a^b (x - \mu)^2 f(x)dx}} \tag{3.5}$$

Théorème de Bienaymé-Tchébycheff

Soit X une variable aléatoire de distribution quelconque, de moyenne μ et d'écart-type σ. Alors pour tout nombre $k > 0$, on a :

$$\boxed{P(\mu - k\sigma \leq X \leq \mu + k\sigma) > 1 - \frac{1}{k^2}} \tag{3.6}$$

De cette inégalité, on peut affirmer que les intervalles $[\mu-2\sigma, \mu+2\sigma]$ et $[\mu-3\sigma, \mu+3\sigma]$ contiennent toujours respectivement au moins 75% et 88.9% des valeurs de la variable X et pour obtenir plus de 95% des valeurs, il faut prendre $k > 4,5$. Dans la section suivante, nous montrerons que pour la distribution normale qui est la plus répandue, nous avons des résultats plus précis.

3.1.5 La loi normale

La loi de distribution, dite loi normale ou de *Laplace Gauss*, est la loi qui régit les variations de nombreux paramètres physiques. Elle est applicable lorsque les dispersions de la variable sont dues à l'influence de nombreux paramètres indépendants les uns des autres et dont les effet s'additionnent (théorème central limite). Les variables influençantes peuvent avoir des distributions normales, mais aussi quelconques.

Définition

On appelle loi normale, une loi de distribution des probabilités définie par une fonction de densité de probabilité de la forme :

$$\boxed{f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp \frac{-(x-\mu)^2}{2\sigma^2}} \tag{3.7}$$

avec :
- μ : l'espérance ou la valeur moyenne des valeurs de la variable
- σ : l'écart-type des valeurs de la variable

On peut immédiatement voir que la loi normale est une loi symétrique autour de la valeur moyenne μ (si l'on remplace $x - \mu$ par $\mu - x$, $f(x)$ ne change pas). On peut aussi voir que la loi normale est complètement définie par son espérance μ et son écart-type σ. On peut montrer facilement que l'écart-type σ d'une distribution normale est la distance entre l'axe de la courbe et le point d'inflexion (voir figure 3.2).

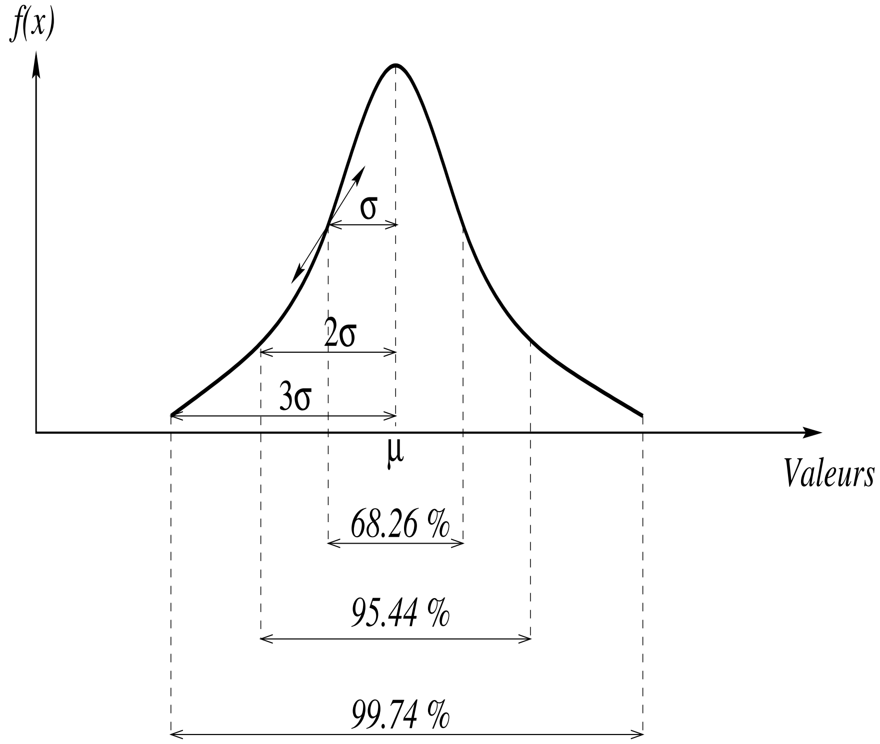

FIG. 3.2: *Distribution normale ou loi de Laplace Gauss.*

Loi normale réduite

Dans le cas particulier ou l'espérance ($\mu = 0$) et l'écart-type ($\sigma = 1$), la distribution de la loi normale devient :

$$\boxed{z(x) = \frac{1}{\sqrt{2\pi}} \exp \frac{-x^2}{2}} \tag{3.8}$$

et on dit qu'on a affaire à la loi normale réduite Z. On montre qu'on peut passer aisément d'une variable normale quelconque X de moyenne μ et d'écart-type σ, à une variable normale réduite ϵ par un changement de variable linéaire de la forme $\epsilon = \frac{X-\mu}{\sigma}$.

Caractéristique de la loi normale

Pour une loi normale, les différentes mesures de tendance (moyenne, médiane et mode) sont toutes égales à μ. Bien que la fonction de densité ne soit jamais nulle, elle prend une valeur très petite en tout point x distant de plus de 3σ de la moyenne μ. Pour une distribution normale, la probabilité qu'une variable aléatoire X soit à l'intérieur de l'intervalle $[\mu - k\sigma, \mu + k\sigma]$ pour différentes valeurs de k est donnée dans le tableau suivant :

intervalle considéré	probabilité
$[\mu - 0.5\sigma, \mu + 0.5\sigma]$	38.29 %
$[\mu - \sigma, \mu + \sigma]$	68.26 %
$[\mu - 1.5\sigma, \mu + 1.5\sigma]$	86.63 %
$[\mu - 2\sigma, \mu + 2\sigma]$	95.44 %
$[\mu - 2.5\sigma, \mu + 2.5\sigma]$	98.75 %
$[\mu - 3\sigma, \mu + 3\sigma]$	99,74 %

Théorème centrale limite

La moyenne ou l'espérance mathématique d'un échantillon de taille n extrait d'une population quelconque de moyenne μ et d'écart-type σ est distribuée selon une loi pratiquement normale de moyenne μ et d'écart-type $\frac{\sigma}{\sqrt{n}}$ quand la taille de l'échantillon est suffisamment grande.

Pour une population de départ de distribution normale, le théorème centrale limite est valable pour tout n. Pour les distributions continues rencontrées dans la pratique courante, plus la taille de l'échantillon est grande et plus la loi se rapproche de la loi normale, et on peut considérer qu'à partir d'environ $n = 30$, la moyenne d'un échantillon est distribuée de façon sensiblement normale.

Une des utilisations pratiques du théorème centrale limite ou de convergence est de résoudre le problème des fluctuations d'échantillonnage d'une moyenne (voir section 3.1.6). D'après le théorème centrale limite et les caractéristiques de la loi normale réduite, on peut montrer que la variable $\frac{\overline{x}-\mu}{\frac{\sigma}{\sqrt{n}}}$ suit une loi normale réduite notée Z. En conséquence, on peut démontrer que si on admet un pourcentage d'erreur sur les résultats qu'on note aussi le risque α (voir section 3.1.6), la moyenne $\overline{x}$ d'un échantillon de taille n reste comprise dans un intervalle symétrique de dimension $|\epsilon|\frac{\sigma}{\sqrt{n}}$ autour de son espérance μ, $|\epsilon|$ correspond à la valeur pour laquelle la probabilité que la variable normale réduite soit en dehors de l'intervalle $[-|\epsilon|, +|\epsilon|]$ est égale à α.

3.1.6 Estimation de la moyenne et de l'écart-type à partir d'un échantillon

Dans la pratique, on distingue la moyenne et l'écart-type théoriques d'une distribution de probabilités notés respectivement μ et σ de la moyenne et l'écart-type estimés ou observés sur un échantillon notés respectivement $\bar{x}$ et s. Il existe deux types d'estimation d'un paramètre inconnu : ponctuelle et par intervalle de confiance.

Estimation ponctuelle

L'estimation ponctuelle est choisie de façon à ce qu'elle converge vers la valeur du paramètre, qu'elle soit sans biais c'est à dire que son espérance mathématique soit égale à la valeur du paramètre et que sa variance soit minimum. Ainsi à partir d'un échantillon de taille n, les estimations de la moyenne μ et de l'écart-type σ sont données par :

$$\boxed{\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}} \tag{3.9}$$

$$\boxed{s = \sqrt{\frac{(\sum_{i=1}^{n} (x_i - \bar{x})^2}{n-1}}} \tag{3.10}$$

Intervalle de confiance

Le risque $\alpha \in [0, 1]$ d'une estimation correspond à la probabilité que l'estimation qu'on fait d'un paramètre soit fausse. Donc, si on a une estimation à un risque α, alors la probabilité que l'estimation soit vraie est égale à $(1 - \alpha)$.

L'estimation d'un paramètre par un intervalle de confiance au risque $\alpha \in [0, 1]$, consiste à trouver un intervalle qui a une probabilité de $(1 - \alpha)$ de contenir la vraie valeur du paramètre. Pour un échantillon de taille n, l'intervalle de confiance de la moyenne μ est défini par $\overline{x} \pm |\epsilon|\frac{s}{\sqrt{n}}$ pour les grands échantillons ($n > 30$), et si l'échantillon est de petite taille ($n < 30$), par $\overline{x} \pm |t|\frac{s}{\sqrt{n}}$ ou t est la valeur maximale de la fonction t_{n-1} de *Student* d'ordre $(n-1)$ correspondant au risque α choisi. Dans ce cas de petite taille, l'intervalle de confiance n'est valable que dans le cas où on suppose que la distribution des événements étudiés suit la loi normale. Dans le tableau 3.1, nous avons donné les valeurs de ϵ et t de *Student* pour différentes valeurs du risque α. Le paramètre t de *Student* dépend du risque α et de la taille n de l'échantillon.

Le risque α	La probabilité correspondant au risque α	La valeur de $\|\epsilon\|$	La valeur de $\|t\|$	
			$(n = 5)$	$(n = 10)$
0.20	80%	1.282	1.533	1.383
0.10	90%	1.645	2.132	1.833
0.05	95%	1.960	2.776	2.262
0.01	99%	2.576	4.604	3.250

TAB. 3.1: *Les valeurs des paramètres $|\epsilon|$ et $|t|$ pour différentes valeurs du risque α.*

3.2 Modélisation de fautes

L'étude des défauts physiques présents dans les circuits intégrés nécessite une modélisation. Un modèle de fautes permet de représenter les défauts physiques qui peuvent affecter le " *layout* " d'un circuit pour pouvoir simuler leur conséquence sur le comportement du circuit. Un bon modèle de fautes doit être simple à utiliser et doit représenter fidèlement le comportement des défauts physiques du circuit. La qualité d'un ensemble de tests est déterminée par le taux de couverture et le modèle de fautes utilisé, plus le modèle de fautes est représentatif de la majorité des défauts physiques, plus on aura de défauts détectés. Dans la suite de la thèse, nous supposons que si la faute existe alors elle est unique.

Le modèle de fautes le plus utilisé pour les circuits numériques est le modèle des collages " *Stuck at Fault* ". La puissance du modèle des collages réside dans sa simplicité d'utilisation et dans sa capacité à détecter la majorité des défauts physiques. En effet même si beaucoup de défauts ne peuvent pas être représentés par le modèle des collages, ils peuvent être détectés par ce modèle [T.W86]. Les fautes dans les circuits intégrés analogiques sont classées en deux catégories :

- Les fautes catastrophiques appelées aussi " *Hard Fault* ".
- Les fautes paramétriques appelées aussi " *Soft Fault* ".

3.2.1 Les fautes catastrophiques

La définition des fautes catastrophiques diffère d'un auteur à un autre. Pour certain auteurs, les fautes catastrophiques sont des fautes qui correspondent à des défauts aléatoires " *Spot Defect* ", une particule de poussière sur un masque photolithographique entraînant des déformations locales comme les court-circuits et les circuits ouverts. En revanche, pour d'autres auteurs les fautes catastrophiques sont des fautes qui engendrent un fonctionnement du circuit complètement différent du fonctionnement normal, même si l'origine de cette faute n'est qu'une petite variation d'un paramètre du circuit. Dans la suite de la thèse, nous avons adopté la première définition car elle permet de différencier les types des fautes non pas par rapport au fonctionnement du circuit mais suivant l'origine de la faute. En outre, pour les circuits analogiques, il n'est pas facile de définir la limite entre une petite déviation et une grande déviation pour pouvoir classer sans ambiguïté une faute selon la deuxième définition.

Les défauts physiques affectant les circuits varient d'un procédé de fabrication à un autre. Une étude des fautes les plus courantes sur un procédé *nMOS* à grille d'aluminium a donné les résultats suivants [JG80] :

Type de défectuosité	Fréquence %
Court-circuit Alu	39
Circuit ouvert	14
Court-circuit diffusion	14
Circuit ouvert diffusion	6
Court-circuit Alu-Substrat	2
Pas de défectuosité observable	10
Divers (macroscopiques)	15

TAB. 3.2: *Répartition des défectuosités dans les circuits nMOS*

Dans [BA82], l'auteur propose une classification des fautes en fonction de leur probabilité d'apparition dans les circuits *MOS*. Les résultats de l'étude sont donnés dans le tableau 3.3. Une autre étude plus récente [HRK95] basée sur l'analyse des fautes par induction *IFA* a montré que les défauts les plus courants dans les circuits intégrés *MOS* sont : les court-circuits et les circuits ouverts.

Classe	Types de défectuosité %
Très probable	Court-circuit entre la grille et le drain Court-circuit entre la grille et la source
Probable	Circuit ouvert sur le drain Circuit ouvert sur la source
Peu probable	Court-circuit entre la grille et le substrat Grille flottante

TAB. 3.3: *Classification des fautes catastrophiques dans les transistors MOS*

3.2.2 Les fautes paramétriques

Comme pour les fautes catastrophiques, pour certains auteurs, les fautes paramétriques sont des fautes dues aux fluctuations des paramètres du processus de fabrication qui en général n'engendrent pas un comportement complètement différent du circuit mais causent des déviations des sorties du circuit qui sont en dehors des intervalles de tolérance. Pour d'autres auteurs, les fautes paramétriques sont les fautes qui engendrent des déviations des sorties du circuit en dehors des intervalles de tolérance. En utilisant la deuxième définition, cela reviendrait à dire que même si l'origine de la faute est un court-circuit dû à une particule de poussière, si le comportement du circuit diffère seulement de son comportement initial mais reste globalement le même, alors la faute est considérée comme paramétrique. Pour les mêmes raisons que précédemment, nous utiliserons la première définition pour la suite de la thèse.

Comme les fautes paramétriques engendrent des déviations des paramètres de sorties du circuit et que ces déviations peuvent être plus au moins grandes suivant le paramètre considéré, il est donc plus difficile de tester ces fautes. En effet, il ne suffit pas de trouver seulement les stimuli qui activent les fautes, mais il faut aussi trouver les meilleurs paramètres qui permettent d'avoir des déviations en sortie du circuit en dehors des intervalles de tolérance.

3.2.3 Processus de fabrication des circuits analogiques

Le processus de fabrication des circuits intégrés utilise des réactions chimiques et des procédés physiques qui sont très dépendants de l'environnement, très sensibles aux variations de température, aux vibrations et aux différentes variations des outils de fabrication. Des petites fluctuations de l'environnement de fabrication peuvent affecter un circuit analogique et créer des fautes catastrophiques ou paramétriques. L'énumération des différents paramètres du processus de fabrication diffère d'un auteur à un autre [LS85] [MSV94] [DSGH99] et [LGA99], les plus importants sont :

- L'épaisseur d'oxyde t_{ox} " *oxide thickness* ".
- La mobilité en surface μ_{0n} et μ_{0p} " *surface mobility* ".
- Le dopage en substrat " *substrate doping* ".

- Les tensions de seuil à polarisation zéro v_{ton} et v_{top} " *zero-bias threshold voltage* ".
- Le Décalage des valeurs de la largeur du poly " *lithographic offset in poly width* ".
- Le Décalage des valeurs de la largeur de diffusion " *lithographic offset in diffusion line* ".
- Variation de la géométrie du transistor δL et δW.

Comme les paramètres du processus de fabrication sont dues aux fluctuations de l'environnement de fabrication, nous supposerons que ces paramètres sont indépendants et distribués suivant une loi normale.

3.2.4 Injection des fautes

Pour les fautes catastrophiques, le modèle de faute utilisé dans cette thèse pour le transistor *MOS* est celui de la figure 3.3 [THB93]. Il permet de représenter les quatre fautes les plus courantes dans un transistor *MOS* qui sont :

- Court-circuit grille source *(GSS) " Gate Source Short "*
- Court-circuit grille drain *(GDS) " Gate Drain Short "*
- Circuit ouvert sur le drain *(DOP) " Drain Open "*
- Circuit ouvert sur la source *(SOP) " Source Open "*

Pour les composants passifs (résistances et capacités), nous avons 2 fautes catastrophiques par composant, un court-circuit modélisé par une résistance de faible valeur ($Rs = 1\Omega$) en parallèle avec le composant et un circuit ouvert modélisé par une résistance de valeur très élevée ($Ro = 10M\Omega$) en série avec le composant.

Les fautes paramétriques sont modélisées par des variations de valeurs des paramètres des composants (comme la valeur de la résistance ou de la capacité) en-dehors de leur intervalle de tolérance. Par contre, pour les transistors, nous avons plusieurs paramètres du processus de fabrication qui interviennent dans la modélisation du transistor. Comme pour la simulation des circuits analogiques, on utilise des modèles de transistors définis par plusieurs paramètres, les fautes paramétriques sont modélisées par des déviations en-dehors des intervalles de tolérance des paramètres les plus importants du modèle utilisé. Ces paramètres peuvent être : la longueur L, la largeur W,

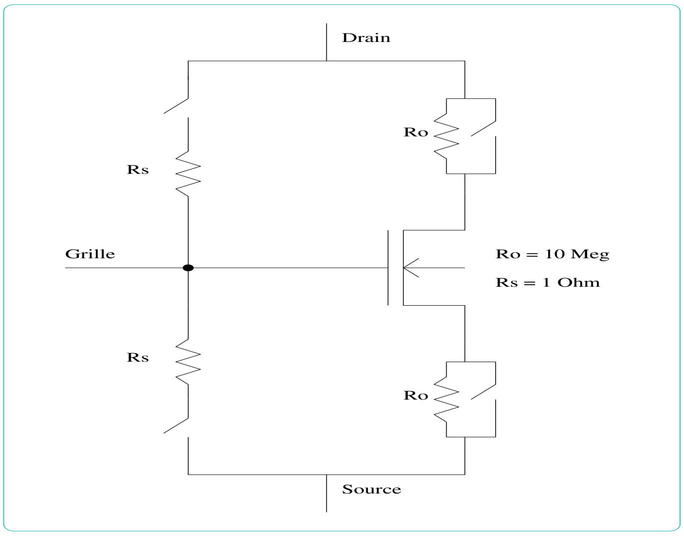

FIG. 3.3: *Le modèle de fautes catastrophiques pour un transistor MOS.*

la tension d'offset V_{th} et le gain en courant β. Même si on suppose que les distributions des paramètres du processus de fabrication sont normales, les distributions des paramètres du modèle de transistor utilisé pour la simulation ne sont pas forcement normales. En effet, les paramètres du modèle de transistor sont des combinaisons des paramètres du processus de fabrication et ces combinaisons peuvent être non linéaires. Les distributions statistiques et les intervalles de tolérance des paramètres du modèle du transistor sont en général donnés par le fondeur.

3.3 Conclusion

Notre travail de thèse est basé sur l'étude statistique des déviations des paramètres des circuits analogiques. Ces déviations sont dues aux variations du processus de fabrication et aux fluctuations de l'environnement de fabrication des circuits. Dans la première partie du chapitre, nous avons fait un rappel des différentes définitions et théorèmes que nous sommes amenés à utiliser dans la suite de la thèse.

Pour le test des circuits analogiques, il n'existe pas encore de modèle de fautes standard comme c'est le cas pour les circuits numériques. C'est pourquoi, la deuxième partie du chapitre est consacrée à l'introduction des modèles et des types de fautes que nous allons utiliser pour valider notre approche de simulation de fautes et d'optimisation des tests de production avec prise en compte des tolérances.

Chapitre 4

Simulation de fautes avec prise en compte des tolérances

La simulation de fautes est indispensable pour le test des circuits intégrés. En effet, elle permet de déterminer l'efficacité des ensembles de tests choisis et de vérifier si les techniques de conception en vue du test *DFT* adoptées sont efficaces. Pour les circuits numériques, la simulation de fautes est largement adoptée par les concepteurs et plusieurs techniques (comme la simulation de fautes parallèle, déductive et concurrente) ont été développées pour accélérer les simulations. Aujourd'hui, la principale difficulté de la simulation de fautes pour les circuits numériques vient de la complexité croissante des circuits à simuler. C'est pourquoi les nouvelles techniques de simulation de fautes et de test sont à base du découpage structurel du circuit [Flo98].

Pour les circuits analogiques, la taille et la complexité des circuits reste petite par rapport aux circuits numériques mais les principales difficultés de la simulation de fautes des circuits analogiques sont dues au manque de modèles de simulation comportementaux, à la diversité des paramètres d'entrée sortie et des types de simulation, au manque de modèle de fautes efficace et aux variations du processus de fabrication(voir section 1.1). Le phénomène des variations du processus de fabrication est l'un des problèmes majeurs du test des circuits analogiques. En effet, les variations du processus de fabrication engendrent des déviations sur les paramètres des composants des circuits analogiques, et prendre en compte ces tolérances dans la phase de simulation de fautes est l'une des principales difficultés de la simulation de fautes pour les circuits analogiques.

Dans ce chapitre, nous proposons une nouvelle méthode de simulation de fautes pour les circuits analogiques avec prise en compte des tolérances sur les paramètres du circuit. Dans la première partie du chapitre, nous formaliserons la problématique de la simulation de faute pour les circuits analogiques avec prise en compte des tolérances. Après la formalisation, nous présenterons les problèmes dus aux variations des paramètres des circuits et les différentes approches proposées pour prendre en compte ces variations dans la phase de simulation de fautes. Ensuite, nous introduirons la notion de probabilité de détection des fautes *PDF* utilisée dans notre approche. La dernière partie du chapitre est consacrée à la présentation de notre méthode ainsi qu'aux résultats de validation sur divers circuits.

4.1 Formulation

Avant d'exposer la problématique de la simulation de fautes pour les circuits analogiques avec prise en compte des tolérances, nous allons introduire quelques définitions et notations pour mieux comprendre le problème et les approches proposées. Dans la suite de ce chapitre, nous allons considérer que nous avons un circuit analogique avec les éléments suivants :

- L'ensemble de vecteurs de test à simuler : l'utilisateur doit définir l'ensemble

  $$\mathbf{T} = \{T_1, T_2, ...T_i...T_n\}$$

  des n vecteurs de test dont on veut mesurer l'efficacité avec le simulateur de fautes . Chaque test T_i est déterminé par un ensemble de valeurs à appliquer aux entrées.

- L'ensemble de fautes : en plus de l'ensemble des tests, on définit l'ensemble

  $$\mathbf{F} = \{F_1, F_2, ...F_j...F_m\}$$

  des m fautes que l'on souhaite détecter.

- L'ensemble des paramètres : l'ensemble des paramètres du circuit sera noté :

  $$\mathbf{P} = \{P_1, P_2, ...P_k...P_l\}$$

  Pour les composants passifs (résistance et capacité), ces paramètres sont les valeurs des composants ; et pour les transistors ces paramètres peuvent être : la longueur L, la largeur W, la tension d'offset V_{th} et le gain en courant β. Chaque paramètre P_k est défini par sa valeur nominale notée P_{k_N} qui est la valeur par défaut du paramètre, et par son intervalle de tolérance noté $[P_{k_L}, P_{k_R}]$. En général, les valeurs nominales des paramètres sont données par le concepteur du circuit et les intervalles de tolérances sont extraits des fichiers technologiques donnés par le fondeur du circuit.

- L'ensemble des paramètres de sortie : comme pour un circuit analogique, il existe plusieurs sorties qu'on peut analyser (voir 1.1.5) ; il est important de préciser les paramètres de sortie que l'on souhaite mesurer pour détecter les fautes. Lorsqu'on parlera de la mesure d'une sortie donnée, cela voudra dire la mesure des paramètres de la sortie. On dit que le test T_i détecte la faute F_j si et seulement si il existe au moins un paramètre de sortie qui donne des résultats différents pour le bon circuit et le circuit fautif. Pour faciliter la compréhension et simplifier la mise en œuvre de l'algorithme de simulation, on supposera que chaque circuit ne contient qu'une seule sortie à considérer que l'on notera S. Dans le cas de plusieurs paramètres de sortie, il suffit de prendre le meilleur paramètre et notre raisonnement sera toujours le même puisque la condition de détectabilité exige un seul paramètre.
- Pour chaque test $T_i \in \mathbf{T}$, on notera $V_{i,ref}$ la valeur simulée de la sortie S du circuit correct en prenant les valeurs nominales des paramètres. Par souci de simplification, quand on a un seul vecteur de test, on remplacera $V_{1,ref}$ par V_{ref}.
- Pour chaque test $T_i \in \mathbf{T}$ et chaque faute $F_j \in \mathbf{F}$, on notera $V_{i,j}$ la valeur simulée de la sortie S du circuit fautif contenant la faute F_j (qu'on notera aussi la faute j) en prenant les valeurs nominales des paramètres. De même, quand on a un seul vecteur de test, on remplacera la valeur $V_{1,f}$ correspondant à la faute f par V_f.

4.2 Problématique de la simulation de fautes analogique

Le but de la simulation de fautes est de déterminer suivant un modèle de faute donné les fautes détectées par un ensemble de test. Les trois étapes d'un simulateur de fautes analogique sont (voir figure 4.1) :

1. Injection de fautes
2. Simulation du bon circuit et des circuits fautifs

3. Comparaison des résultats de simulation

4.2.1 Injection de fautes

Le module d'injection permet de traduire les fautes que l'on souhaite simuler au format du simulateur en utilisant les modèles de fautes définis dans le fichier de modèles. En général, la liste des fautes peut être donnée par le concepteur du circuit ou extraite du " *layout* " du circuit en utilisant les méthodes d'*IFA " Inductive Fault Analysis "* (voir 2.2.3). Pour les circuits de petite taille, on peut aussi construire la liste des fautes en analysant la netlist du circuit et en considérant toutes les fautes possibles du circuit (tous les courts-circuits, les circuits ouverts et les fautes paramétriques). A la fin de l'étape d'injection, on obtient la netlist du circuit correct et des circuits fautifs en vue de leur simulation.

Il existe deux possibilités différentes pour l'injection de fautes ; la première consiste à générer le fichier correspondant au bon circuit et un fichier de netlist pour chaque circuit fautif, et la deuxième solution consiste à générer un seul fichier de netlist contenant le bon circuit et tous les circuits fautifs. La faisabilité de la deuxième solution dépend du simulateur analogique utilisé, sachant que les simulateurs *SPICE* et *ELDO* permettent la modification des paramètres du circuit avec la commande *ALTER*. L'avantage principal de la première solution est qu'elle permet facilement de distribuer les simulations des circuits fautifs sur plusieurs machines, alors que la deuxième solution permet de réduire considérablement l'espace disque sur la machine.

4.2.2 Simulation

Le module de simulation permet d'utiliser un simulateur analogique pour simuler le bon circuit et les circuits fautifs générés par le module d'injection de fautes. Contrairement aux circuits numériques, où il existe des simulateurs spécifiques pour la simulation de fautes qui n'utilisent pas de simulateurs logiques externes, pour les circuits analogiques, il n'existe pas encore de noyau de simulation spécifique pour la simulation de fautes analogique. En conséquence, tous les simulateurs de fautes analogiques utilisent un simulateur fonctionnel externe, et le simulateur de faute ainsi réalisé est

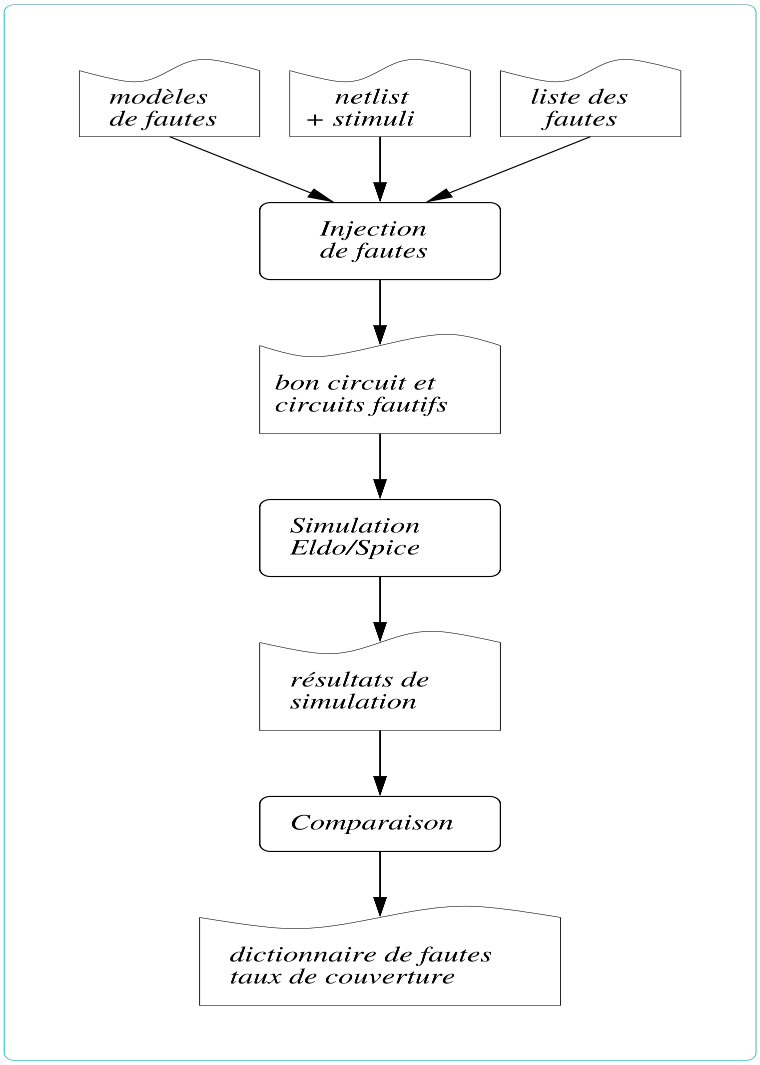

FIG. 4.1: *La simulation de fautes pour les circuits analogiques.*

en général très dépendant du simulateur analogique qu'il utilise. Ainsi, le simulateur de faute *SWITTEST* [MRVH98] pour les circuits à capacités commutées est basé sur le simulateur analogique *SWITCAP* ; *ANAFAULT* [SO93] est basé sur *ELDO* et *FSPICE* [RCA85] est basé sur *SPICE*. Par conséquents, la rapidité de simulation et la précision des résultats du simulateur de fautes analogique dépendent essentiellement du simulateur fonctionnel utilisé.

4.2.3 Comparaison des résultats de simulation

L'étape de comparaison des résultats de simulation est aussi appelée " *Post Processing* ", cette étape finale permet d'analyser les résultats de simulation et de déduire les fautes qui sont détectables ou non détectables par chaque vecteur de test. Le module de comparaison permet donc de générer le dictionnaire de fautes et de calculer le taux de couverture de l'ensemble de vecteurs de test simulés.

En général, le module de comparaison utilise un algorithme qui permet de comparer les valeurs simulées du bon circuit et des circuits fautifs. Selon les paramètres considérés, comme le bruit, les tolérances des composants et les erreurs de mesure et suivant la précision que l'on souhaite avoir, on utilise des algorithmes de comparaison plus ou moins complexes. L'algorithme de comparaison le plus simple consiste à calculer la différence entre les valeurs simulées du bon circuit et du circuit fautif, si cette différence est plus grande que le seuil de tolérance fixé par l'utilisateur alors la faute est détectable, sinon la faute est non détectable. L'inconvénient de cet algorithme est qu'il ne représente pas fidèlement les déviations des circuits analogiques dues au bruit et aux variations des paramètres des composants, ce qui occasionne souvent des erreurs dans les décisions prises pour la détectabilité ou non des fautes. Dans la suite de ce chapitre, nous présenterons les erreurs de comparaison dues aux déviations des paramètres du circuit ainsi que des algorithmes de comparaison plus complexes et plus précis et qui prennent en compte ces déviations.

4.2.4 Problématique des résultats de simulation

L'utilisation directe des résultats des simulateurs analogiques peut engendrer plusieurs erreurs de décision, ces erreurs sont dues aux:

– Vecteurs de test à appliquer sur les circuits qui, en général, sont imprécis.

– Paramètres des composants des circuits analogiques qui, en général, sont donnés avec tolérance.

– Modèles des transistors utilisés pour les simulations qui sont imprécis.

Pour bien comprendre les erreurs qui peuvent résulter des variations du processus de fabrication, prenons l'exemple de simulation d'un amplificateur opérationnel *AOP* et considérons la mesure du gain noté G. Soit G_{ref} la valeur simulée du gain avec les

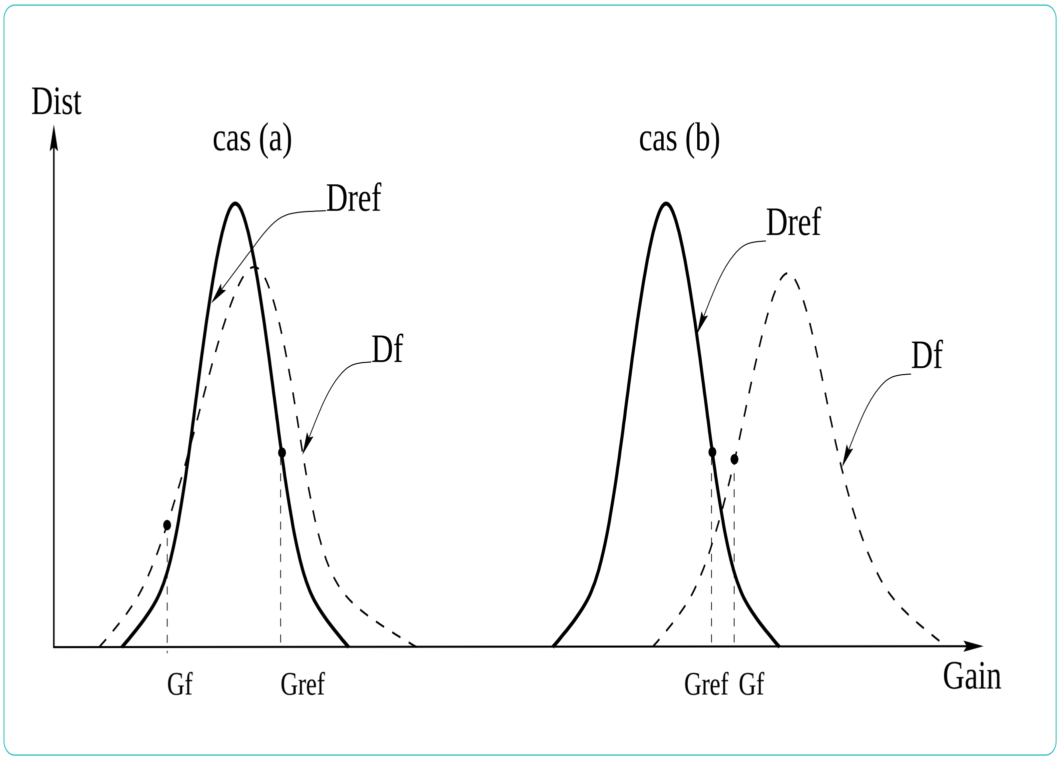

FIG. 4.2: *Comparaison des résultats de simulation du gain d'un AOP avec prise en compte des distributions.*

valeurs nominales des paramètres du bon circuit, et G_f la valeur simulée du gain avec les valeurs nominales des paramètres du circuit fautif contenant la faute f. D_{ref} et D_f sont les distributions de probabilités des valeurs que peut prendre G pour le circuit correct et le circuit fautif respectivement (voir figure4.2).

Si on compare simplement les deux valeurs G_{ref} et G_f (cela suppose qu'on ne tient pas compte des déviations des paramètres du circuit), on peut déduire dans le cas (a) que la faute f est détectable car la valeur du gain pour le circuit fautif G_f est loin de la valeur du gain G_{ref} du bon circuit (donc les deux circuits sont très différents), alors que dans le cas (b) la faute est non détectable car G_f est proche de G_{ref} (donc les deux circuits sont semblables).

En revanche, si maintenant, on oublie les valeurs simulées G_{ref} et G_f du gain, et qu'on regarde seulement les distributions D_{ref} et D_f des valeurs du gain pour le bon circuit et le circuit fautif, on constate que les conclusions que l'on peut tirer sont complètement différentes. En effet, dans le cas (a), on peut dire que la faute f est non détectable (au lieu de détectable) car les deux distributions D_f et D_{ref} sont presque complètement recouvrantes. Dans le cas (b), on peut dire que la faute f est non détectable (au lieu de détectable) car les deux distributions D_f et D_{ref} sont presque disjointes.

En conclusion, pour la simulation de fautes avec prise en compte des tolérances, on a besoin de déterminer les distributions des valeurs simulées pour le bon circuit et les circuits fautifs. Comme l'évaluation exacte des distributions nécessite des simulations *Monte Carlo* coûteuses en temps de simulation, nous avons essayé de trouver des méthodes qui nous permettront soit de déduire approximativement les distributions soit de déduire des résultats sans utiliser les distributions .

4.3 Les différentes approches proposées

Il existe principalement trois méthodes qui permettent de prendre en compte les tolérances dues aux variations du processus de fabrication dans la simulation de fautes des circuits analogiques. Ces trois méthodes sont :

1. L'analyse Monte Carlo
2. L'arithmétique des intervalles
3. La logique floue

4.3.1 La méthode Monte Carlo

L'utilisation de l'approche Monte Carlo pour la prise en compte des tolérances des composants des circuits dans la simulation de fautes analogique est la méthode la plus précise mais la plus coûteuse en temps de simulation. Cette approche consiste à effectuer pour le bon circuit et tous les circuits fautifs plusieurs simulations, en faisant varier les paramètres du circuit dans leur intervalle de tolérance [BHSMK97] et [RVMN99]. Les simulations Monte Carlo permettent de calculer les intervalles de sorties du bon circuit et des circuits fautifs. Les fautes sont ensuite déclarées détectables ou non détectables en comparant les intervalles I_{ref} du circuit correct avec les intervalles I_{f_i} des circuits fautifs. L'organigramme de la simulation d'un circuit fautif contenant une faute f est donné dans la figure 4.3, le paramètre *Nmax* représente le nombre de simulations

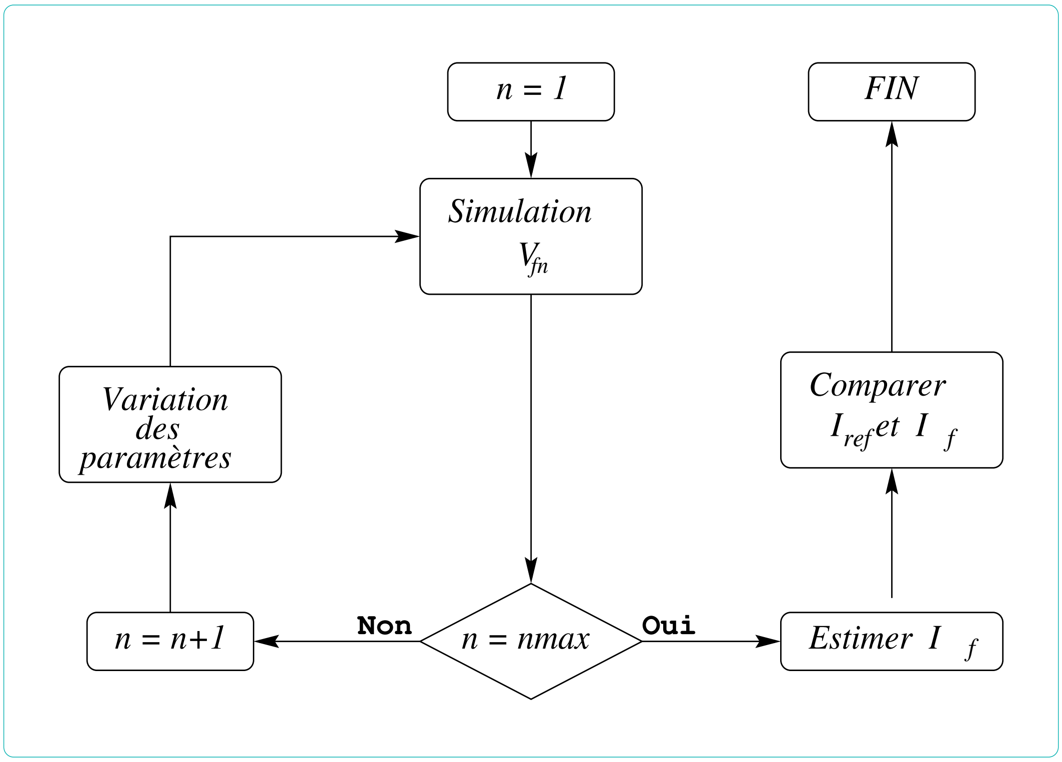

FIG. 4.3: *Organigramme de la simulation de fautes avec prise en compte des tolérances utilisant l'approche Monte Carlo.*

nécessaires pour l'évaluation de la distribution D_f du circuit fautif. La comparaison d'intervalle permet d'avoir une meilleure précision que la seule comparaison des valeurs simulées, mais cette méthode reste moins précise dans le cas ou les deux intervalles à comparer ne sont pas complètement disjoints.

Une des améliorations de l'analyse Monte Carlo a été présentée dans [SSI97], l'algorithme proposé est basé sur le calcul de la distance métrique appelée $DIST$. Cette distance est calculée en fonction des réponses nominales du bon circuit et des circuits fautifs. La réponse nominale correspond à la valeur simulée du circuit obtenu en prenant les valeurs nominales des paramètres du circuit. La fonction $DIST$ est donnée par la formule suivante :

$$\boxed{DIST_f = \frac{|V_{ref} - V_f|}{S_{ref}}} \tag{4.1}$$

où V_{ref} et V_f sont respectivement les réponses nominales du bon circuit et du circuit fautif contenant la faute f, et S_{ref} est l'écart-type estimé des réponses du bon circuit obtenu avec des simulations Monte Carlo. Durant la phase de simulation de fautes, si pour une faute donnée f_i, la distance $DIST_{f_i}$ est supérieure à un seuil donné par l'utilisateur alors la faute f_i est détectable ; sinon, on effectue les simulations Monte Carlo pour calculer la distribution D_{f_i} de la réponse du circuit fautif et on la compare à la distribution D_{ref} de la réponse du bon circuit. L'inconvénient de cette méthode est que le seuil de référence est choisi arbitrairement par l'utilisateur et si la valeur simulée du circuit fautif est proche des limites de la distribution du circuit correct, la faute est déclarée complètement détectable alors qu'elle peut être partiellement indetectable. Notre approche est une amélioration de cette méthode pour améliorer la précision dans la détection de fautes et diminuer le temps de simulation de fautes.

4.3.2 La méthode des intervalles

Les programmes de simulation de type *SPICE* utilisés dans les simulateurs électriques sont basés sur la résolution des équations mathématiques. Pour les analyses transitoires, la forme la plus générale des équations décrivant un circuit est :

$$\boxed{F(\dot{x(t)}, x(t), u(t)) = 0 \;\; avec \;\; x(0) = X_0} \tag{4.2}$$

où $x(t) \in \mathcal{R}^n$ est le vecteur inconnu, $u(t) \in \mathcal{R}^n$ le vecteur des sources indépendantes du circuit et X_0 est le vecteur de condition initiale. Pour le cas des simulations *AC* en fréquence, en général on utilise l'analyse nodale notée *MNA* " *Modified Nodal Analysis* ". L'analyse nodale est basée sur la résolution des équations formulées par la loi de *Kirchoff* en courant noté *KCL* " *Kirchoff's Current Law* ". Pour un circuit avec n nœuds et m branches, l'équation à résoudre est sous forme matricielle et est de la forme :

$$\boxed{\mathcal{A}i = i_0} \tag{4.3}$$

où $\mathcal{A} \in \mathcal{R}^{n*m}$ est la matrice décrivant le circuit, $i \in \mathcal{R}^m$ est le vecteur des courants dans les branches du circuit et $i_0 \in \mathcal{R}^m$ est le vecteur des sources de courant indépendantes du circuit.

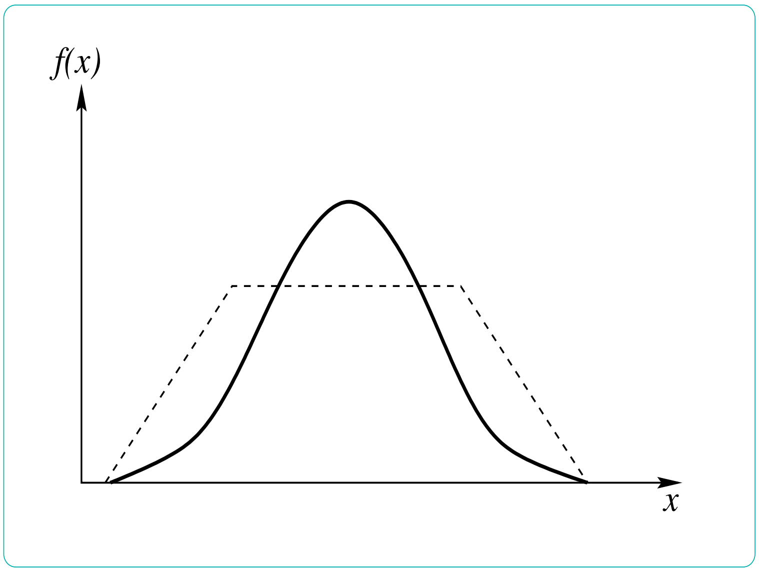

FIG. 4.4: *Similitude entre un intervalle flou et une distribution de valeurs.*

La méthode des intervalles consiste à effectuer des simulations avec les intervalles de tolérance $[P_{k_L}, P_{k_R}]$ des paramètres du circuit et non pas avec les valeurs nominales P_{k_N}. Les algorithmes de simulation à base d'intervalles utilisent l'arithmétique des intervalles qui définit toutes les opérations de base (addition, soustraction, multiplication et division) sur des intervalles mathématiques. Les deux inconvénients majeurs de cette méthode sont :

1. L'explosion des intervalles en sortie : par exemple, si on utilise l'arithmétique des intervalles sur la fonction $f(x) = 1 - x + x^2$ avec $x = [0, 2]$, on a $1 - x = [-1, 1]$, $x^2 = [0, 4]$ et $f(x) = [-1, 1] + [0, 4] = [-1, 5]$. Alors que l'analyse de la fonction $f(x)$ avec précision montre que pour $x \in [0, 2]$ on a $f(x) \in [\frac{3}{4}, 3]$. Ceci démontre la caractéristique particulière de l'arithmétique des intervalles : l'intervalle résultant des opérations peut être une surestimation de l'intervalle réel, ceci est du au fait que les opérations sur les intervalles ne prennent pas en compte les corrélations qui peuvent exister entre les intervalles.

2. Cette méthode est utilisable uniquement pour les analyses AC : les seuls travaux publiés dans ce domaine [TS97] et [ST98] ne considèrent que les circuits linéaires. En effet, il existe des méthodes de résolution d'un système d'équations linéaires d'intervalles qui sont utilisées dans l'analyse nodale pour résoudre l'équation 4.3 en remplaçant la matrice A par la matrice d'intervalle correspondante et les vecteurs i et i_0 par les vecteurs d'intervalle correspondants. Par contre, pour les ana-

lyses transitoires et *DC*, on utilise les techniques de *Newton-Raphsen* pour résoudre l'équation non linéaire 4.2 ; il n'existe pas encore de méthode pour appliquer ces techniques sur des équations d'intervalles.

4.3.3 La logique floue

La méthode des intervalles flous est très identique à la technique des intervalles, la seule différence entre les deux méthodes est que pour la méthode des intervalles flous, les paramètres des circuits sont définis par des intervalles flous au lieu des intervalles normaux. Le choix des intervalles flous est dû à la similitude qui existe entre un intervalle flou et une distribution (voir figure 4.4), ce qui rend les intervalles flous plus réels que les intervalles normaux. Cette approche a été utilisée dans [Moh98] pour le test et le diagnostic des circuits analogiques, mais l'implémentation d'un noyau de simulation à base de la logique floue reste très difficile à réaliser pour les circuits linéaires et pratiquement impossible pour les circuits non linéaires.

4.4 Probabilité de détection des fautes

Le problème avec la simulation de fautes analogique est que lorsque l'on compare les résultats de deux simulations avec prise en compte des tolérances, on n'a plus le choix entre seulement deux solutions : faute détectée ou non détectée. En effet, en comparant la distribution de la sortie du circuit fautif D_f avec la distribution du circuit de référence D_{ref} (voir figure 4.5), la conclusion peut être :

1. La faute f est complètement détectable

2. La faute f est complètement non détectable

3. La faute f est partiellement détectable (ou partiellement non détectable)

C'est pourquoi il est important de définir une fonction de probabilité qui permet de déterminer avec précision à quel point (degré) une faute est détectable.

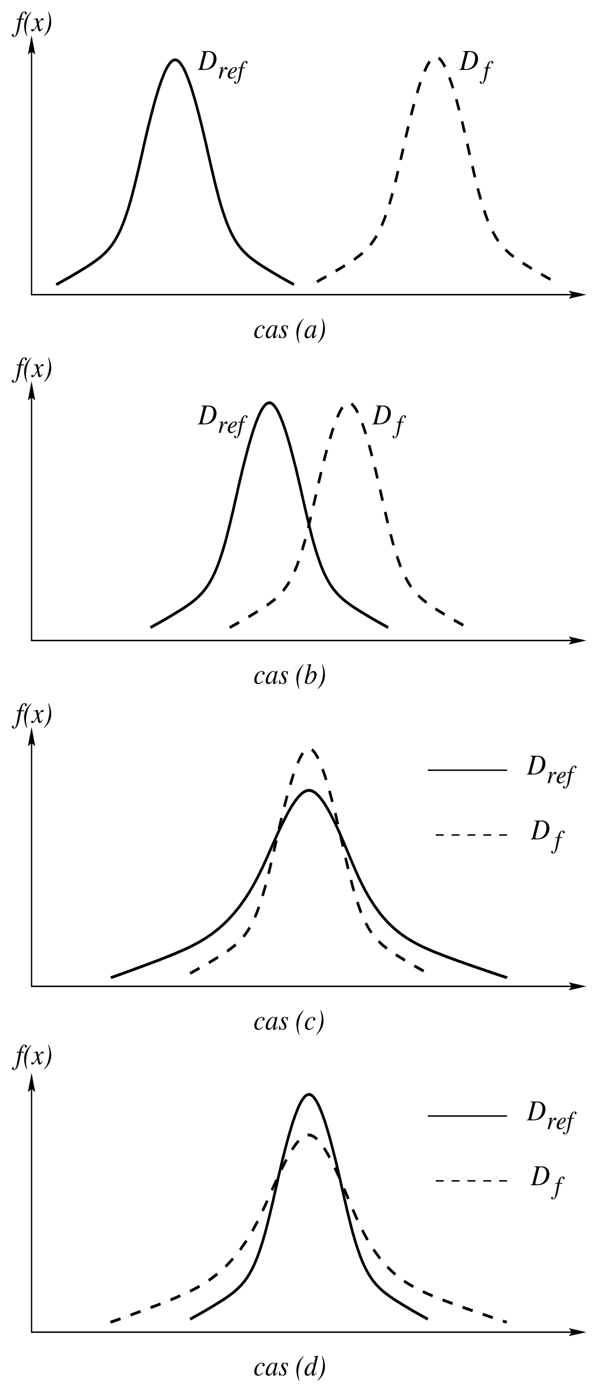

FIG. 4.5: *Comparaison de distributions pour la détection de fautes : (a) faute détectable, (b) faute partiellement détectable, (c) faute non détectable et (d) faute partiellement détectable.*

Le risque α	La probabilité correspondante	Le paramètre k	
		Loi normale	Loi quelconque
0.20	80%	1.282	2.236
0.10	90%	1.645	3.162
0.05	95%	1.960	4.472
0.01	99%	2.576	10.00

TAB. 4.1: *Les valeurs du paramètre k pour différentes valeurs du risque α et pour une distribution normale et quelconque.*

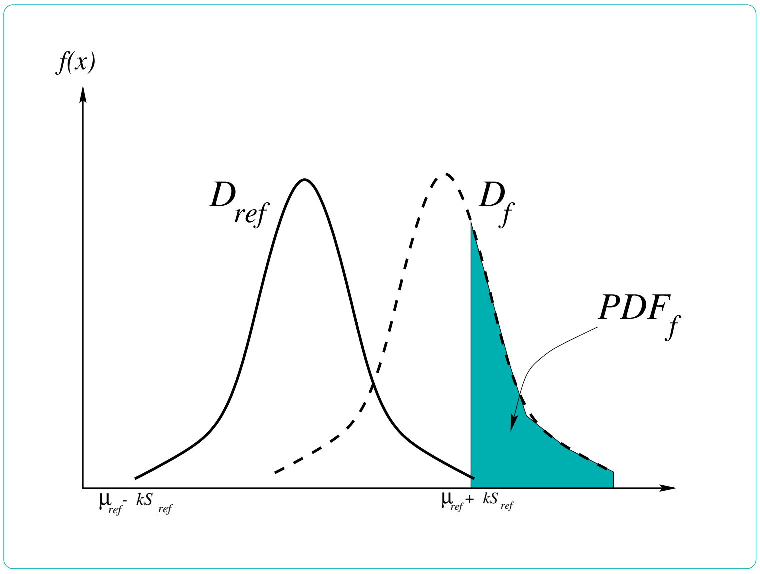

FIG. 4.6: *La probabilité de détection de fautes PDF.*

4.4.1 Définition de la probabilité de détection de fautes PDF

Soit D_{ref} la distribution estimée de n valeurs simulées de la sortie du circuit correct et μ_{ref} et S_{ref} la moyenne et l'écart-type estimés des n valeurs simulées. Selon le risque α que nous souhaitons avoir, nous définissons le paramètre k correspondant au risque α tel que la probabilité que les valeurs de la distribution D_{ref} soient à l'intérieur de l'intervalle de confiance $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$ est égale à $(1 - \alpha)$ (voir section 3.1.6). Si D_{ref} est une distribution normale ou si $n \geq 30$ alors k correspond au paramètre $|\epsilon|$ de la loi normale définie dans la section 3.1.6, sinon pour une distribution quelconque, le paramètre k correspond au paramètre du théorème de *Bienaymé-Tchébycheff* énoncé dans la section 3.1.4 . Le tableau 4.1 récapitule les valeurs des paramètres k pour les différentes valeurs du risques α, et selon la distribution, normale ou quelconque.

La probabilité de détection d'une faute f de distribution D_f qu'on notera PDF_f, est définie par l'aire de la courbe D_f se trouvant à l'extérieur de la distribution D_{ref} du bon circuit (voir figure 4.6). Cette probabilité permet de quantifier par un nombre appartenant à $[0, 1]$ à quel degré la faute f est détectable.

4.4.2 Calcul de la fonction PDF

Pour un risque α donné, nous considérons que toutes les valeurs des distributions D_{ref} du bon circuit et D_f du circuit fautif sont respectivement comprises dans les intervalles $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$ et $[\mu_f - kS_f, \mu_f + kS_f]$. Donc, la probabilité de détection d'une faute est donnée par l'aire de la courbe D_f se trouvant à l'extérieur de l'intervalle $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$. La fonction de probabilité de détection de la faute f notée PDF_f est donnée par :

$$\boxed{PDF_f = \int_{-\infty}^{\mu_{ref} - kS_{ref}} D_f(x)\,dx + \int_{\mu_{ref} + kS_{ref}}^{+\infty} D_f(x)\,dx} \tag{4.4}$$

Comme on a $\int_{-\infty}^{+\infty} D_f(x)\,dx = 1$ alors on a :

$$\boxed{PDF_f = 1 - \int_{\mu_{ref} - kS_{ref}}^{\mu_{ref} + kS_{ref}} D_f(x)\,dx} \tag{4.5}$$

Cas de la distribution normale

Dans le cas où on suppose que la distribution D_f est normale de moyenne μ_f et d'écart-type S_f, alors on remplace dans l'équation 4.4 D_f par $\frac{1}{\sqrt{2\pi}S_f} \exp\left(-\frac{(x-\mu_f)^2}{2S_f^2}\right)$ et on obtient :

$$\boxed{PDF = 1 - \int_{\mu_{ref} - 2S_{ref}}^{\mu_{ref} + 2S_{ref}} \frac{1}{\sqrt{2\pi}S_f} \exp\left(-\frac{(x-\mu_f)^2}{2S_f^2}\right) dx} \tag{4.6}$$

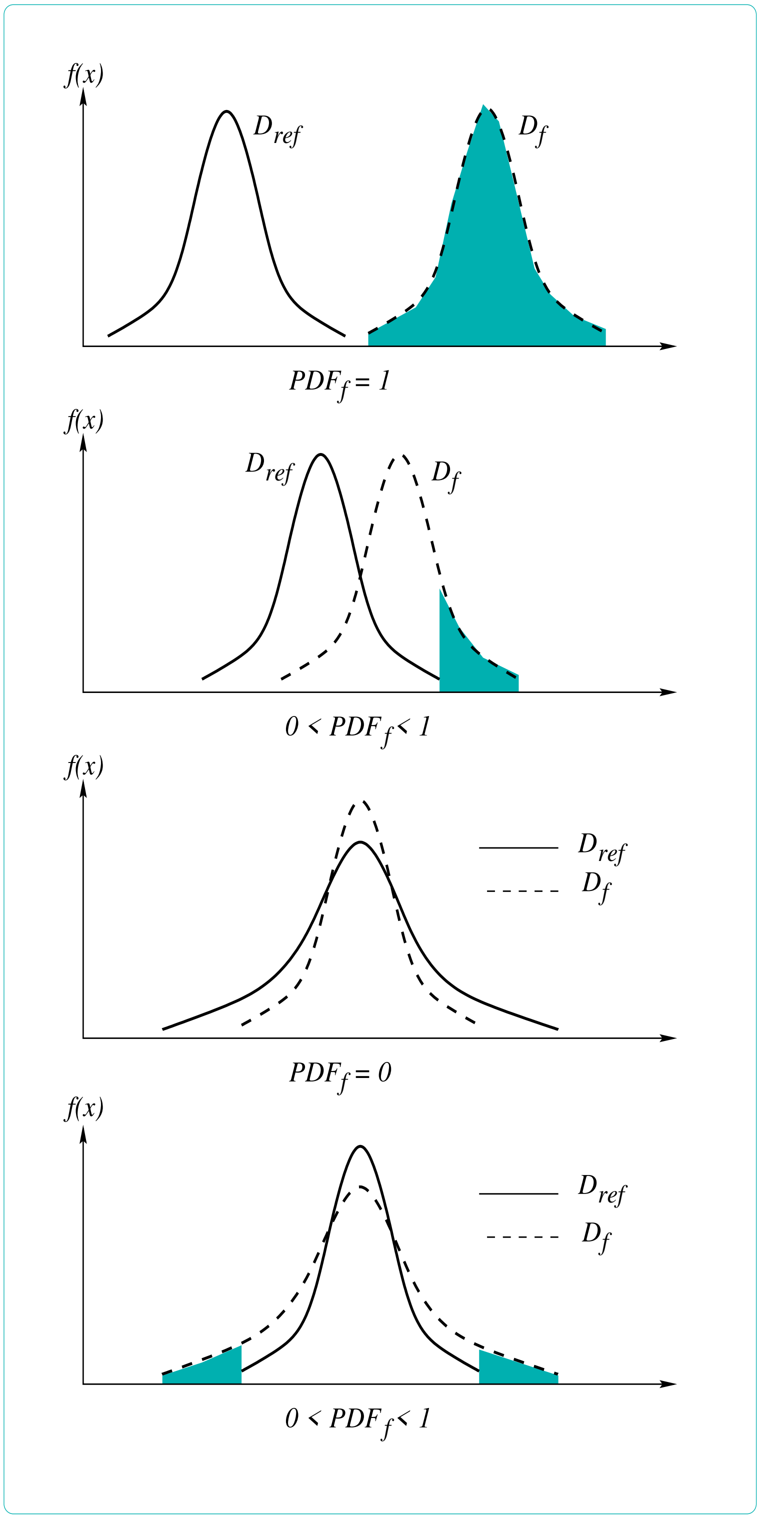

FIG. 4.7: *Calcul de la probabilité de détection de fautes PDF pour différents cas.*

Dans la suite de la thèse nous supposerons que les distributions des valeurs de sortie des circuits suivent une loi normale.

4.4.3 Amélioration de la fonction PDF

Comme la fonction PDF_f représente la probabilité que la faute f soit détectée, alors par définition, elle doit vérifier les conditions suivantes (voir figure 4.7) :

1. Si pour un vecteur de test, la distribution des valeurs de sortie du circuit fautif D_f est identique à la distribution des valeurs de sortie du bon circuit D_{ref}, alors le vecteur de test en question n'active pas la faute, et la faute est complètement non détectable. Ceci se traduit par l'expression suivante :

$$(D_f = D_{ref}) \implies PDF_f = 0 \tag{4.7}$$

2. Si la distribution des valeurs de la sortie du circuit fautif D_f est complètement disjointe de la distribution des valeurs de la sortie du circuit correct D_{ref}, alors la faute doit être complètement détectable. Ceci se traduit par l'expression suivante :

$$(D_f \cap D_{ref}) = 0 \implies PDF_f = 1 \tag{4.8}$$

3. Si toutes les valeurs de la distribution D_f du circuit fautif sont incluses dans la distribution D_{ref} du circuit correct, alors la faute doit être complètement non détectable. Ceci se traduit par l'expression suivante :

$$(D_f \subset D_{ref}) \implies PDF_f = 0 \tag{4.9}$$

Cas $D_f = D_{ref}$

En remplaçant D_f par D_{ref} dans l'expression 4.5 on obtient :

$$PDF_{ref} = 1 - \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_{ref}(x)\, dx \tag{4.10}$$

et par définition du paramètre k, nous savons que la probabilité que les valeurs de la distribution D_{ref} soit à l'intérieur de l'intervalle de confiance $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$ est égale à $(1 - \alpha)$. Ce qui revient à dire que :

$$\int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_{ref}(x)\,dx = 1 - \alpha \tag{4.11}$$

d'où $PDF_{ref} = \alpha$, et donc la fonction PDF ne vérifie pas la condition 4.7, ce qui est normal car nous avons négligé toutes les valeurs de la distribution D_{ref} en dehors de l'intervalle de confiance. Pour remédier à ce problème, nous procédons à la modifications de la fonction PDF définie dans 4.5 en rajoutant la condition suivante :

$$PDF_f = 0 \;\; Si \quad \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \;\geq\; 1 - \alpha \tag{4.12}$$

Cas $D_f \cap D_{ref} = 0$

Comme on travaille toujours avec le risque α, nous dirons que $D_f \cap D_{ref} = 0$ si $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}] \cap [\mu_f - kS_f, \mu_f + kS_f] = 0$. Comme $\int_{\mu_f-kS_f}^{\mu_f+kS_f} D_f(x)\,dx = 1 - \alpha$ (par définition du paramètre k), alors on peut déduire que :

$$\int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \leq \alpha \tag{4.13}$$

car l'intégrale de D_f dans l'intervalle $[-\infty, \mu_f - kS_f] \cup [\mu_f + kS_f, +\infty]$ est égale à α et que cet intervalle contient l'intervalle $[\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$ (cf $D_f \cap D_{ref} = 0$). Donc, pour que la fonction PDF vérifie la condition 4.8, on ajoute la condition suivante:

$$PDF_f = 1 \;\; Si \quad \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \;\leq\; \alpha \tag{4.14}$$

Cas $D_f \subset D_{ref}$

Puisque nous avons $D_f \subset D_{ref}$ alors on a $[\mu_f - kS_f, \mu_f + kS_f] \subset [\mu_{ref} - kS_{ref}, \mu_{ref} + kS_{ref}]$ et donc :

$$\int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \geq \int_{\mu_f-kS_f}^{\mu_f+kS_f} D_f(x)\,dx \tag{4.15}$$

et comme d'après la définition du paramètre k, nous avons :

$$\int_{\mu_f-kS_f}^{\mu_f+kS_f} D_f(x)\,dx = 1-\alpha \tag{4.16}$$

alors on a :

$$\int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \geq 1-\alpha \tag{4.17}$$

et donc en utilisant l'équation 4.12, on obtient $PDF_f = 0$ ce qui montre que la condition 4.9 est vérifiée.

En résumé, pour que les trois conditions (4.7, 4.8 et 4.9) soientt vérifiées, la définition 4.5 de la fonction PDF_f devient :

$$PDF_f = 0 \;\; Si \quad \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \geq 1-\alpha$$

$$PDF_f = 1 \;\; Si \quad \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x)\,dx \leq \alpha$$

$$Sinon \qquad PDF_f = 1 - \int_{\mu_{ref}-kS_{ref}}^{\mu_{ref}+kS_{ref}} D_f(x) \tag{4.18}$$

4.5 Simulation de fautes avec prise en compte des tolérances

Dans la section précédente, nous avons présenté la fonction de probabilité de détection de fautes PDF qui permet de calculer avec précision à quel degré une faute est détectable par un vecteur de test, en tenant compte des tolérances sur les paramètres du circuit. L'inconvénient majeur de cette fonction est qu'elle nécessite plusieurs simulations pour chacun des circuits fautifs, en vue de déterminer la distribution D_f du circuit fautif utilisée dans le calcul de PDF_f. Les simulations analogiques sont en général très coûteuses en temps de simulation, et effectuer plusieurs simulations pour tous les circuits fautifs engendrerait un temps de simulation prohibitif.

Le but de notre approche est de réduire au maximum le nombre de simulations pour chacun des circuits fautifs. En effet, le calcul de la fonction PDF peut s'avérer inutile dans le cas où les deux distributions D_{ref} du circuit correct et D_f du circuit contenant la faute f sont très éloignées ; dans ce cas de figure, on peut tout de suite affirmer que $PDF_f = 1$ et que la faute est complètement détectable. Donc, pour beaucoup de fautes, notamment la majorité des fautes catastrophiques, il n'est pas indispensable d'effectuer des simulations *Monte Carlo* pour calculer la probabilité de détection de la faute PDF_f. Il reste donc à déterminer à partir de quel moment on peut dire que les deux distributions D_{ref} et D_f sont assez distantes l'une de l'autre pour affirmer que $PDF_f = 1$.

4.5.1 Notre approche

Soit μ_{ref} et σ_{ref} la moyenne et l'écart-type de la distribution D_{ref} du bon circuit, et μ_f et σ_f la moyenne et l'écart-type de la distribution D_f du circuit fautif contenant la faute f, on peut dire que la distribution D_f est suffisamment éloignée de la distribution D_{ref} si et seulement si $D_{ref} \cap D_f = 0$. Soit k le paramètre correspondant au risque α que nous souhaitons avoir, comme D_{ref} et D_f sont définies par les deux intervalles de confiance $[\mu_{ref} - k\sigma_{ref}, \mu_{ref} + k\sigma_{ref}]$ et $[\mu_f - k\sigma_f, \mu_f + k\sigma_f]$ respectivement, alors, nous avons :

$$(D_f \cap D_{ref}) = 0 \iff ([\mu_{ref} - k\sigma_{ref}, \mu_{ref} + k\sigma_{ref}] \cap [\mu_f - k\sigma_f, \mu_f + k\sigma_f]) = 0 \tag{4.19}$$

et d'après 4.8 nous avons :

$$(D_f \cap D_{ref}) = 0 \implies PDF_f = 1$$

d'où :

$$([\mu_{ref} - k\sigma_{ref}, \mu_{ref} + k\sigma_{ref}] \cap [\mu_f - k\sigma_f, \mu_f + k\sigma_f] = 0) \implies PDF_f = 1 \qquad (4.20)$$

et on a :

$$|\mu_{ref} - \mu_f| \geq k.(\sigma_{ref} + \sigma_f) \implies ([\mu_{ref} - k\sigma_{ref}, \mu_{ref} + k\sigma_{ref}] \cap [\mu_f - k\sigma_f, \mu_f + k\sigma_f]) = 0 \qquad (4.21)$$

Pour éviter d'avoir à effectuer des simulations *Monte Carlo*, on doit faire une supposition sur l'écart-type σ_f du circuit fautif, cette supposition ne sera pas démontrée, mais sera vérifiée par simulation sur les circuits de validation. Au départ, on suppose qu'il y a un endomorphisme entre les écarts- types du circuit correct et des circuits fautifs ($\sigma_f = \alpha\ \sigma_{ref}$ avec $\alpha = \frac{V_f}{V_{ref}}$), V_{ref} et V_f sont les valeurs simulées du bon circuit et du circuit fautif. En général, lorsque la valeur simulée V_{ref} du circuit fautif est très différente de la valeur V_f, la faute est complètement détectable, et donc la précision sur la valeur de σ_f n'a pas beaucoup d'importance. Par contre la précision de la valeur de σ_f est très importante lorsque les deux valeurs V_f et V_{ref} sont très proches, dans ce cas là on a $(V_f \approx V_{ref})$, et donc $(\alpha = \frac{V_f}{V_{ref}} \approx 1)$ et $(\sigma_f \approx \sigma_{ref})$.

Donc, d'après 4.20 et 4.21 et en supposant que $\sigma_f = \sigma_{ref}$ on obtient :

$$\boxed{|\mu_{ref} - \mu_f| \geq 2k\sigma_{ref} \implies PDF_f = 1} \qquad (4.22)$$

Le problème de l'équation 4.22 est qu'elle utilise les moyennes et les écart-types théoriques des deux distributions D_{ref} et D_f. Si maintenant on considère $\overline{V_f}$ et S_f la moyenne et l'écart-type estimés des n valeurs simulées du circuit contenant la faute f, d'après le paragraphe 3.1.6 sur les intervalles de confiance, si on suppose que la distribution D_f suit une loi normale et pour un échantillon de petite taille ($n \leq 30$), l'intervalle de confiance de la moyenne μ_f est défini par $[\overline{V_f} - |t|\frac{S_f}{\sqrt{n}}, \overline{V_f} + |t|\frac{S_f}{\sqrt{n}}]$ où t est la valeur maximale de la fonction t_{n-1} de *Student* d'ordre $(n-1)$ correspondant au risque α. Le tableau 4.2 récapitule les valeurs du paramètre t de *Student* pour différentes valeurs du

Le risque α	La probabilité correspondante	Les valeurs $\|t\|$ de la fonction t_{n-1} de *Student*			
		$(n=5)$	$(n=10)$	$(n=20)$	$(n=30)$
0.20	80%	1.533	1.383	1.328	1.310
0.10	90%	2.132	1.833	1.729	1.699
0.05	95%	2.776	2.262	2.093	2.045
0.01	99%	4.604	3.250	2.861	2.756

TAB. 4.2: *Les valeurs du paramètres $|t|$ de la fonction t_{n-1} de Student pour différentes valeurs du risque α et de la taille n de l'échantillon.*

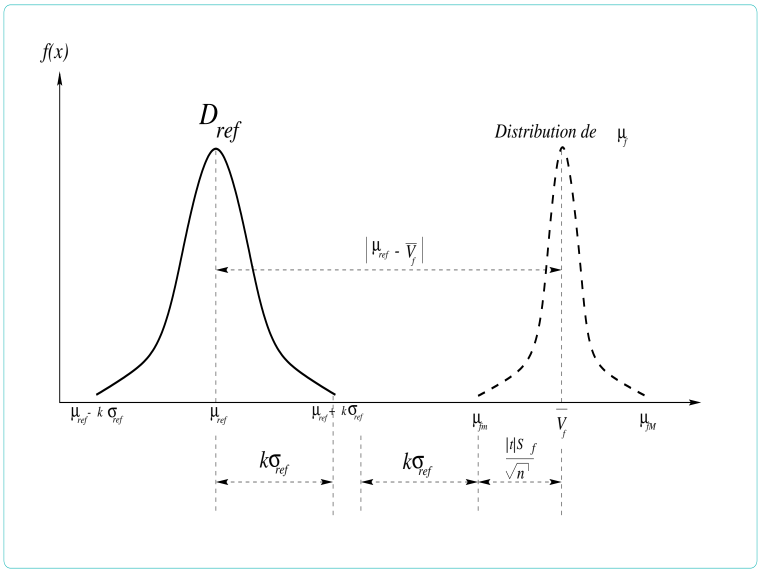

FIG. 4.8: *Distribution de la moyenne théorique μ_f du circuit fautif autour de la moyenne estimée $\overline{V_f}$ en suivant une loi de Student d'ordre (n-1).*

risque α et de la taille n de l'échantillon.

Comme $[\overline{V_f} - |t|\frac{S_f}{\sqrt{n}}, \overline{V_f} + |t|\frac{S_f}{\sqrt{n}}]$ est l'intervalle de confiance de la moyenne μ_f au risque α, alors on a :

$$\boxed{|\mu_f - \overline{V_f}| \leq |t|\frac{S_f}{\sqrt{n}}} \tag{4.23}$$

L'équation 4.23 montre que pour un risque α donné, la moyenne théorique μ_f de la distribution D_f du circuit fautif contenant la faute f se trouve à l'intérieur de l'intervalle $[\mu_{fm}, \mu_{fM}]$ (voir figure 4.8) où $\mu_{fm} = \overline{V_f} - |t|\frac{S_f}{\sqrt{n}}$ représente la valeur minimum que peut prendre μ_f et $\mu_{fM} = \overline{V_f} + |t|\frac{S_f}{\sqrt{n}}]$ représente la valeur maximum que peut prendre μ_f.

En combinant l'expression 4.22 et l'équation 4.23 on obtient l'expression suivante :

$$\boxed{|\mu_{ref} - \overline{V}_f| \geq 2k\sigma_{ref} + |t|\frac{S_f}{\sqrt{n}} \implies PDF_f = 1} \tag{4.24}$$

Cette expression traduit le fait que si la distance entre μ_ref et V_f est supérieure à $(2k\sigma_{ref} + |t|\frac{S_f}{\sqrt{n}})$ alors on peut tout de suite dire que la faute est détectable $PDF_f = 1$ (voir figure 4.9). Cette expression est valable pour tout $n \geq 2$, car pour $n = 1$ (une seule simulation) le paramètre t de *Student* ainsi que l'écart-type S_f ne sont pas définis. Dans la section suivante nous donnerons une expression équivalente pour le cas ou on a une seule simulation $(n = 1)$.

L'avantage de l'expression 4.24 est qu'elle permet de calculer la probabilité de détection de fautes dans certains cas sans avoir à effectuer plusieurs simulations (à partir de $n = 2$), ce qui permettra de diminuer le nombre de simulations globales à effectuer pour déterminer avec précision si une faute est complètement détectable ou non. L'exemple de la figure 4.9 montre la courbe de la distribution D_f du circuit fautif dans les deux cas extrêmes, c'est-à-dire dans le cas où $(\mu_f = \mu_{fm} = \overline{V_f} - |t|\frac{S_f}{\sqrt{n}})$ et dans le cas où $(\mu_f = \mu_{fM} = \overline{V_f} + |t|\frac{S_f}{\sqrt{n}}])$. Comme dans cet exemple la moyenne estimée $\overline{V}_f$ du circuit fautif est supérieure à la moyenne μ_{ref} du circuit correct, le cas le plus critique où l'on risque d'avoir $PDF_f \neq 1$ est le cas $(\mu_f = \mu_{fm})$. C'est pourquoi nous allons prendre ce cas-là comme exemple d'application de la formule 4.24, et si on a $PDF_f = 0$ alors on peut déduire facilement que $PDF_f = 1$ pour tout $\mu_f \in [\mu_{fm}, \mu_{fM}]$.

Application : Cas $\mu_f = \mu_{fm}$

Le but de cette application est de montrer que pour $\mu_f = \mu_{fm}$ et en supposant que $|\mu_{ref} - \overline{V}_f| \geq 2k\sigma_{ref} + |t|\frac{S_f}{\sqrt{n}}$ on a bien $PDF_f = 0$. Comme on a $\mu_f = \overline{V_f} - |t|\frac{S_f}{\sqrt{n}}$ alors

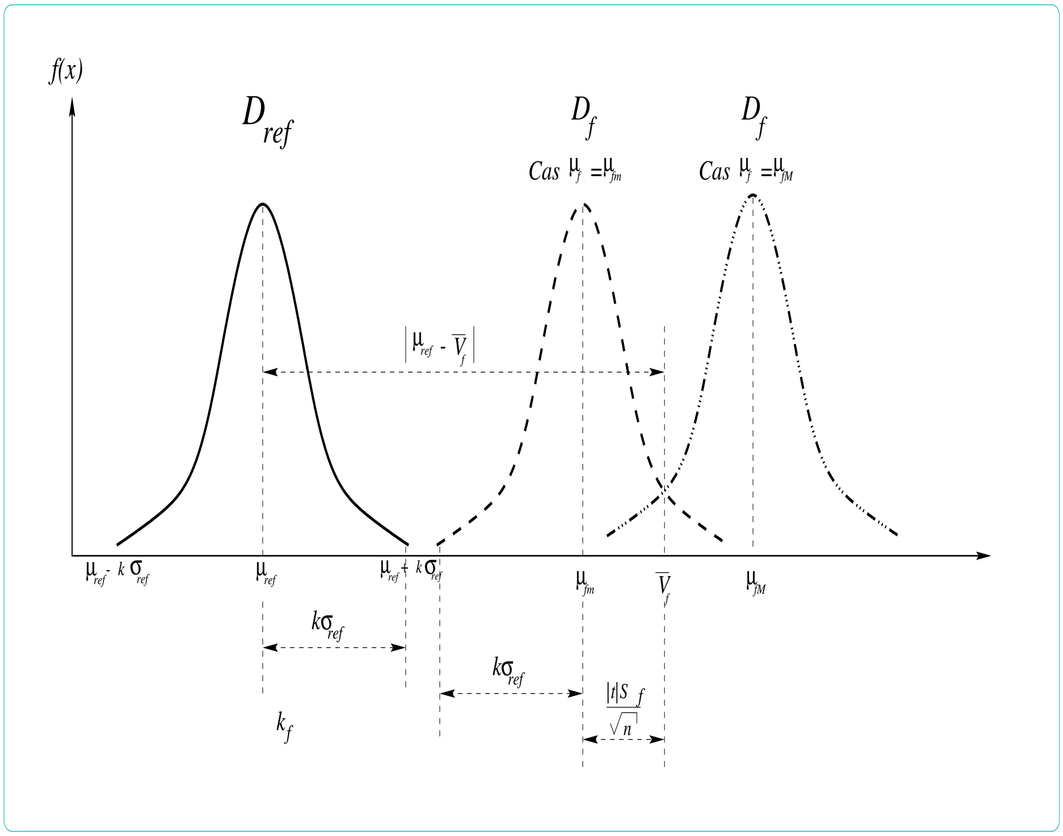

FIG. 4.9: *Calcul de PDF_f dans les deux cas : $\mu_f = \mu_{fm}$ et $\mu_f = \mu_{fM}$.*

on déduit $\overline{V_f} = \mu_f + |t|\frac{S_f}{\sqrt{n}})$ et que :

$$|\mu_{ref} - \overline{V}_f| = |\mu_{ref} - \mu_f - |t|\frac{S_f}{\sqrt{n}}| = |(\mu_f - \mu_{ref}) + |t|\frac{S_f}{\sqrt{n}}|$$

et comme dans notre cas les deux expressions $(\mu_f - \mu_{ref})$ et $|t|\frac{S_f}{\sqrt{n}}$ sont positives, alors on a :

$$|\mu_{ref} - \overline{V}_f| \geq 2k\sigma_{ref} + |t|\frac{S_f}{\sqrt{n}} \implies (\mu_f - \mu_{ref}) + |t|\frac{S_f}{\sqrt{n}} \geq 2k\sigma_{ref} + |t|\frac{S_f}{\sqrt{n}}$$

ce qui au final donne :

$$|\mu_f - \mu_{ref}| \geq 2k\sigma_{ref}$$

et d'après l'équation 4.22 on obtient :

$$PDF_f = 1 \qquad (CQFD)$$

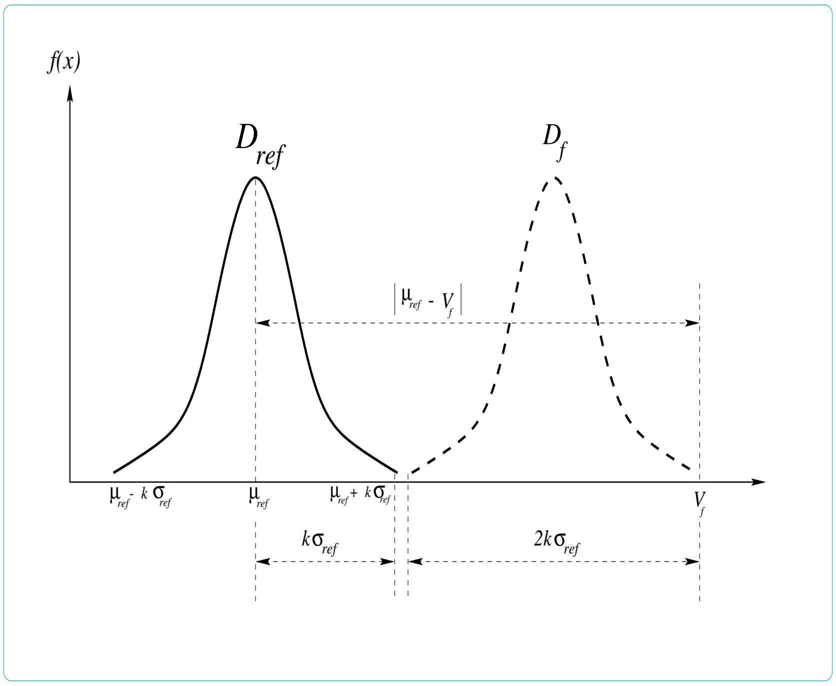

FIG. 4.10: *Calcul de PDF_f dans le cas ou $|\mu_{ref} - V_f| \geq 3k\sigma_{ref}$.*

ceci montre que la faute est complètement détectable dans le cas $\mu_f = \mu_{fm}$ et donc par conséquent, elle est complètement détectable pour tout $\mu_f \in [\mu_{fm}, \mu_{fM}]$.

4.5.2 Calcul de PDF pour $(n = 1)$

L'expression 4.24 n'est utilisable qu'à partir d'un nombre $(n \geq 2)$ de simulations, pour une seule simulation $(n = 1)$, nous allons mettre en œuvre une autre expression équivalente. Soit V_f la valeur simulée du circuit fautif contenant la faute f, D_f sa distribution, μ_f et σ_f la moyenne et l'écart-type théoriques de D_f. Pour un risque α donné, $[\mu_f - k\sigma_f, \mu_f + k\sigma_f]$ est l'intervalle de confiance de la valeur V_f où k correspond au paramètre donné dans le tableau 4.1, et donc on a :

$$|\mu_f - V_f| \leq k\sigma_f \tag{4.25}$$

en combinant cette inégalité avec l'expression 4.22 et en supposant toujours que $\sigma_f = \sigma_{ref}$ on obtient :

$$\boxed{|\mu_{ref} - V_f| \geq 3k\sigma_{ref} \implies PDF_f = 1} \tag{4.26}$$

Cette expression traduit le fait que si la distance entre la moyenne μ_{ref} de la distribution du bon circuit et la valeur simulée V_f du circuit fautif est plus grande que $3k\sigma_{ref}$, alors la faute f est complètement détectable. Ce résultat est montré sur la figure 4.10, on remarque que même si la valeur V_f donnée par le simulateur et la valeur pire cas (la valeur pire cas est la valeur de la distribution D_f la plus loin de μ_{ref}), si la condition $|\mu_{ref} - V_f| \geq 3k\sigma_{ref}$ alors les deux distributions D_{ref} et D_f sont disjointes et donc la faute f est complètement détectable ($PDF_f = 1$). L'expression 4.26 est équivalente à l'expression 4.24 dans le cas d'une seule simulation ($n = 1$).

4.5.3 Algorithme de simulation

L'algorithme de simulation de fautes que nous proposons est basé sur les deux expressions 4.24 et 4.26 ainsi que la formule de calcul de probabilité de détection de fautes PDF définie dans 4.18. L'organigramme de notre algorithme est présenté dans la figure 4.11.

On remarque que la différence de notre approche par rapport à la méthode *Monte Carlo* présentée dans le paragraphe 4.3 réside dans l'ajout de deux tests correspondant aux conditions des expressions 4.24 et 4.26. Ces deux conditions appelées conditions d'arrêt ou encore "*Early Stop Condition*", permettent (quand elles sont vérifiées) d'arrêter plus tôt les simulations et donc de réduire le temps de simulation tout en ayant la même précision que la simulation *Monte Carlo*. Le paramètre *Nmax* représente le nombre de simulations nécessaires pour l'estimation de la distribution D_f du circuit fautif utilisée dans la formule 4.18 pour le calcul de PDF_f, il correspond au nombre de simulations à effectuer dans le cas de la simulation *Monte Carlo* et il est fixé par l'utilisateur en fonction de la précision qu'il souhaite avoir.

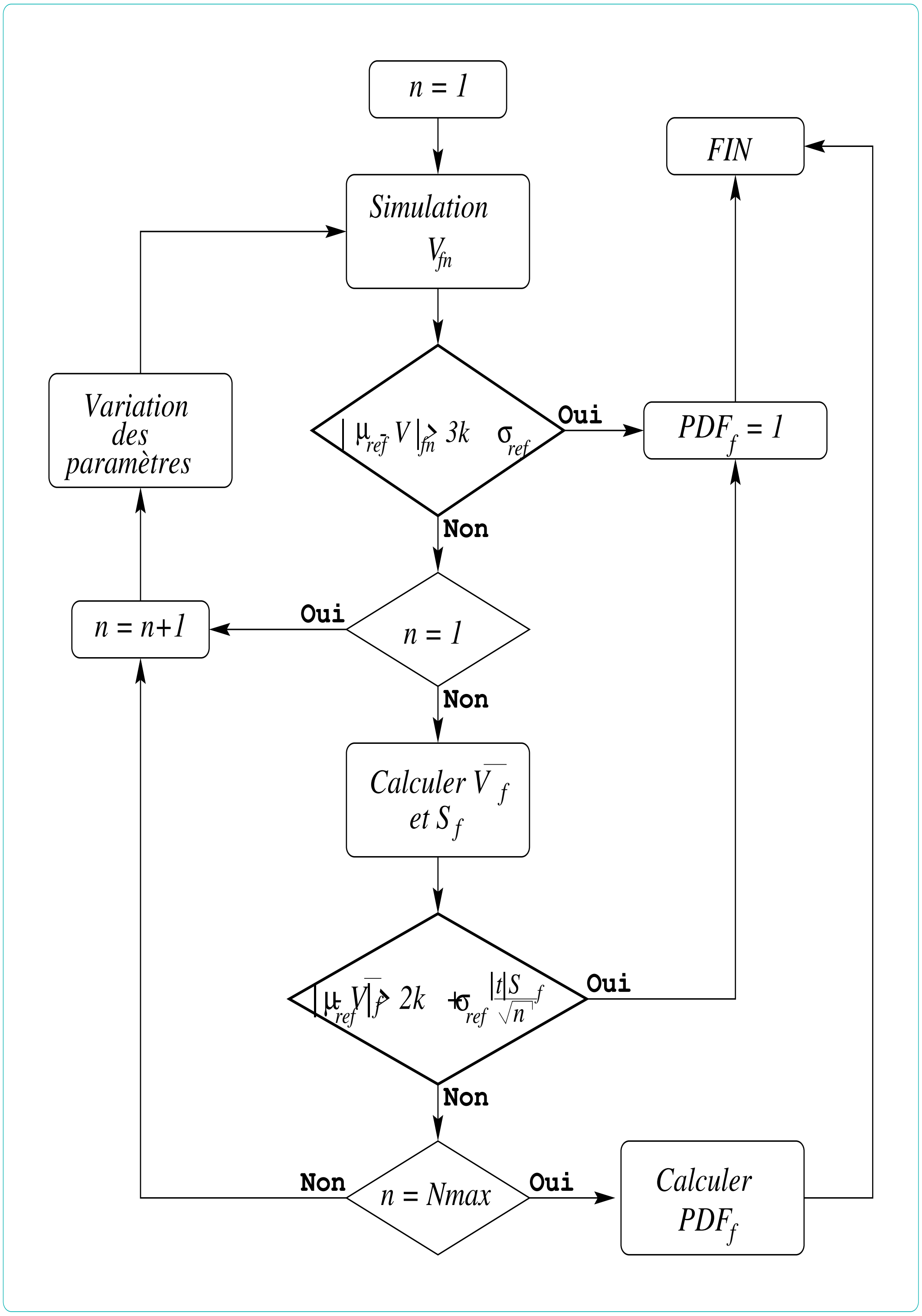

FIG. 4.11: *Organigramme de la simulation de fautes avec prise en compte des tolérances en utilisant les conditions d'arrêt.*

L'algorithme de simulation avec prise en compte des tolérances que nous proposons est le suivant :

// variables locales

entier : Nsim *// pour compter les simulations*
entier : Nmax *// pour le nombre maximum de simulations*

// algorithme

pour *chaque test* $T_i,\ i \in [1..n]$ **faire**
    évaluer μ_{ref} *et* σ_{ref} *pour le circuit correct*
    pour *chaque faute* $F_j,\ j \in [1..m]$ **faire**
        Nsim = 1 ;
        $PDF_{i,j}$ *= 0 ;* *// au début on marque la faute* F_j *non détectable*
        tant que *(* $(PDF_{i,j} == 0)$ *et* $(Nsim \leq Nmax)$ *)* **faire**
            effectuer une variation des paramètres du circuit et une simulation ;
            affecter la valeur $V_{i,j}$ *du circuit fautif ;*
            si $(|\mu_{ref} - V_{i,j}| > 3k\sigma_{ref})$ **alros**
                $PDF_{i,j}$ *= 1 ;* *// la faute est complètement détectable*
            sinon
                sauvegarder $V_{i,j}$ *;* *// à utiliser pour évaluer* $\overline{V}_{i,j}$ *et* $S_{i,j}$
                Si $(NSim \geq 2)$ **alors** *// si le nombre de simulations est* ≥ 2
                    calculer la moyenne $\overline{V}_{i,j}$ *et l'écart-type* $S_{i,j}$ *estimés ;*
                    Si $(|\mu_{ref} - V_{i,j}| > 2k\sigma_{ref} + |t|\frac{S_{i,j}}{\sqrt{n}})$ **alors**
                        $PDF_{i,j}$ *= 1 ;* *// la faute est complètement détectable*
                    Sinon
                        Nsim = Nsim + 1 *// passer à la simulation suivante*
                    fin si
                fin si
            fin si
        fin tant que
        si *(*$PDF_{i,j}$ *= 0)* **alors** *// si faute encore non détectable*
            $PDF_{i,j} = 1 - \int_{\mu_{ref}-2S_{ref}}^{\mu_{ref}+2S_{ref}} \frac{1}{\sqrt{2\pi}S_f} \exp\left(-\frac{(x-\mu_f)^2}{2S_f^2}\right) dx$ *;*
        fin si

(28) **fin pour**
(29) **fin pour**

Les notations utilisées dans cet algorithme correspondent à celles introduites dans le paragraphe 4.1. Les variables *Nsim* et *Nmax* représentent respectivement le compteur de simulations et la valeur maximum de simulations nécessaires pour le calcul de la probabilité de détection de fautes PDF_f par la formule définie dans 4.6. Pour chaque simulation indexée par le compteur *NSim*, $V_{i,j}$ correspond à la valeur simulée de la sortie du circuit fautif contenant la faute F_j avec le vecteur de test T_i, $\overline{V}i, j$ et $S_{i,j}$ correspondent à la moyenne et l'écart-type estimés des *NSim* valeurs $V_{i,j}$ données par les *NSim* simulations déjà effectuées. Cet algorithme est une amélioration de l'algorithme de simulation de fautes basé sur les simulations *Monte Carlo* dans le but de réduire le nombre de simulations et donc le temps de simulation, il est basé sur l'utilisation des expressions 4.26 et 4.24 et la formule 4.6 élaborées dans les sections précédentes.

Les étapes importantes de l'algorithme correspondent aux lignes *(11), (17) et (26)*. Le test de la ligne *(11)* correspond à la condition de détectabilité d'une faute définie dans l'expression 4.24 et celui de la ligne *17* correspond à la condition définie dans l'expression 4.26. Dans le cas où une de ces deux conditions est vérifiée, alors la faute en question est complètement détectable et les simulations sont arrêtées. Sinon, on continue les simulations jusqu'à atteindre les *NMax* simulations et on calcule la probabilité de détection de fautes PDF par la formule définie dans 4.6. Dans tout l'algorithme, nous avons supposé que les distributions des valeurs de sortie du bon circuit et des circuits fautifs suivent une loi normale.

4.5.4 Taux de couverture

Le taux de couverture permet d'exprimer la qualité d'un jeu de vecteurs de test, plus le taux de couverture est grand et plus le jeu de vecteurs de test est efficace. Pour les circuits numériques, le taux de couverture d'un ensemble de vecteurs de test est égal au rapport du nombre de fautes détectées par le nombre de fautes globales ; en effet pour chaque faute, nous avons seulement deux cas de figures : soit la faute est détectable soit la faute est indétectable.

Pour les circuits analogiques, en plus des fautes complètement détectables ($PDF =$

$1)$ et des fautes complètement indétectables ($PDF = 0$), nous disposons aussi des fautes partiellement détectables ($0 < PDF < 1$). C'est pourquoi nous allons utiliser la fonction de probabilité de détection de fautes PDF pour calculer le taux de couverture des jeux de vecteurs de test. Pour le calcul du taux de couverture, nous allons distinguer deux cas de figure :

1. Cas où le circuit analogique a une seule spécification à vérifier en sortie.

2. Cas où le circuit en question a plusieurs spécifications à vérifier en sortie.

Cas d'une seule spécification à vérifier

Pour un circuit analogique à une seule spécification à vérifier en sortie du circuit, et pour un ensemble $F = \{F_1, F_2, \cdots F_m\}$ de m fautes (voir section 4.1), le taux de couverture d'un vecteur de test T_i est noté TC_i et donné par la formule suivante :

$$\boxed{TC_i = \frac{\sum_{j=1}^{m} PDF_{i,j}}{m}} \tag{4.27}$$

Le taux de couverture TC_i est égal au rapport de la somme des probabilités de détection de fautes par le nombre de fautes global. On remarque que dans le cas où toutes les fautes sont soit complètement détectables ($PDF_{i,j} = 1$) soit complètement indétectables ($PDF_{i,j} = 0$), TC_i devient égal au rapport du nombre de fautes détectées par le nombre de fautes globales ce qui correspond à la définition du taux de couverture pour les circuits numériques.

Si on considère maintenant un ensemble $T = \{T_1, T_2, \cdots T_n\}$ de n vecteurs de test, et un ensemble $F = \{F_1, F_2, \cdots F_m\}$ de m fautes, alors le taux de couverture du jeu de vecteurs de test T est noté TC et donné par la formule suivante :

$$\boxed{TC = \frac{\sum_{j=1}^{m} (\underbrace{max}_{T_i \in T}\{PDF_{i,j}\})}{m}} \tag{4.28}$$

Pour une faute donnée F_j, le terme $max\{PDF_{i,j}\}$ représente la meilleure probabilité de détection de fautes possible pour cette faute F_j en considérant tous les vecteurs de test $T_i \in T$. Par exemple, si pour une faute F_j, il existe un vecteur de test $T_i \in T$ tel que $(PDF_{i,j} = 1)$ alors $(max\{PDF_{i,j}\} = 1)$ et la faute F_j est complètement détectable par l'ensemble de vecteurs de test T. Et si pour une autre faute F_k, quelque soit le vecteur de test $T_i \in T$ on a $(PDF_{i,k} = 0)$ alors $(max\{PDF_{i,j}\} = 0)$ est la faute F_k est complètement indétectable par l'ensemble des vecteurs de test T. Le taux de couverture TC exprime l'efficacité de l'ensemble T des vecteurs de test ; plus TC est grand et plus l'ensemble T détecte de fautes. Dans le chapitre suivant, nous proposerons une méthode pour réduire le coût du test de production en diminuant le nombre de vecteurs de test dans l'ensemble T tout en gardant le même taux de couverture TC.

Cas de plusieurs spécifications à vérifier

Pour un circuit analogique donné, on notera $SP = \{s_1, s_2, \cdots s_r, \cdots s_p\}$ l'ensemble des spécifications des sorties à vérifier pour un vecteur de test T_i donné, par exemple pour un test *DC* d'un amplificateur opérationnel, les spécifications à vérifier en sortie de l'amplificateur peuvent être : gain DC, tension d'offset, ...etc. Pour chaque spécification de sortie s_r, on notera $PDF_{i,j}^{s_r}$ la fonction de probabilité de détection de la faute F_j par le vecteur de test T_i en considérant la spécification s_r.

Si pour une spécification donnée s_r, la probabilité de détection de faute $PDF_{i,j}^{s_r}$ est égale à 1, alors la faute F_j est complètement détectable par le vecteur de test T_i, ce qui se traduit par :

$$Si \ \exists \ s_r \in \{s_1, s_2, \cdots s_p\} \ \ tel \ que \ \ PDF_{i,j}^{s_r} = 1 \ \ alors \ \ PDF_{i,j} = 1 \tag{4.29}$$

Par contre, si pour toutes les spécifications $s_r \in \{s_1, s_2, \cdots s_p\}$ du circuit, la probabilité de détections de fautes $PDF_{i,j}^{s_r}$ est égale à zéro, alors la faute F_j est complètement non détectable par le vecteur de test T_i, ce qui se traduit par :

$$Si \ \forall \ s_r \in \{s_1, s_2, \cdots s_p\} \ \ PDF_{i,j}^{s_r} = 0 \ \ alors \ \ PDF_{i,j} = 0 \tag{4.30}$$

Pour les autres cas, la faute F_j est partiellement détectable par le vecteur de test T_i, et la probabilité de détection de la faute F_j par le vecteur de test T_i est donnée par :

$$\boxed{PDF_{i,j} = \underbrace{max}_{s_r \in \{s_1, \cdots s_p\}} \{PDF_{i,j}^{s_r}\}} \tag{4.31}$$

Le terme $max\{PDF_{i,j}^{s_r}\}$ permet de sélectionner la meilleure probabilité de détection de fautes en choisissant la meilleure spécification à tester qui donne la plus grande probabilité. En utilisant cette formule avec la définition du taux de couverture TC_i d'un seul vecteur de test, on obtient la nouvelle définition de TC_i pour un circuit à plusieurs spécifications :

$$\boxed{TC_i = \frac{\sum_{j=1}^{m} \underbrace{max}_{s_r \in \{s_1, \cdots s_p\}} \{PDF_{i,j}^{s_r}\}}{m}} \tag{4.32}$$

De même, pour la définition du taux de couverture TC d'un ensemble T de vecteurs de test pour un circuit à plusieurs spécifications, en combinant la formule 4.31 avec la définition 4.28 on obtient :

$$\boxed{TC = \frac{\sum_{j=1}^{m} (\underbrace{max}_{T_i \in T}\{ \underbrace{max}_{s_r \in \{s_1, \cdots s_p\}} \{PDF_{i,j}^{s_r}\}\})}{m}} \tag{4.33}$$

4.6 Résultats

Pour valider notre approche, nous avons utilisé trois circuits : un amplificateur opérationnel à deux étages, un intégrateur en mode courant et un filtre passe bas à capacités commutées d'ordre 5. Pour les trois circuits, nous avons comparé en termes d'efficacité (taux de couverture) et de vitesse de simulation de fautes les trois méthodes qui sont : la méthode de simulation de fautes unique, la simulation de fautes Monte Carlo et l'approche Monte Carlo amélioré présenté dans la section 4.5. La méthode de simulation de fautes unique consiste à effectuer une seule simulation pour le bon circuit et pour chaque circuit fautif, ensuite on compare les spécifications de sortie des circuits fautifs

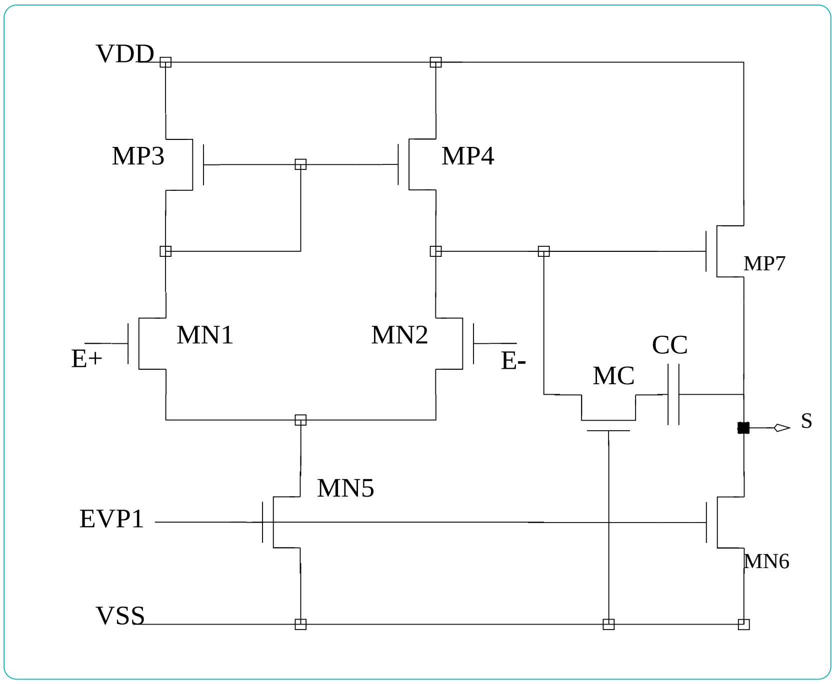

FIG. 4.12: *Amplificateur opérationnel à 2 étages.*

avec celles du bon circuit, et si pour une faute donnée, il y a une spécification dont la valeur V_f est en dehors de l'intervalle $[V_{ref} - 5\%, V_{ref} + 5\%]$, alors la faute est détectable (V_{ref} est la valeur de la spécification de sortie du bon circuit). Pour notre approche, nous avons fixé le risque α à la valeur 0.05 et donc le paramètre k à la valeur 2.

4.6.1 Amplificateur opérationnel

Le premier exemple choisi est l'amplificateur opérationnel à deux étages de la figure 4.12, conçu pour une technologie 0.35μ. Nous avons supposé que tous les paramètres de conception de l'amplificateur (W et L des transistors et les valeurs des capacités) sont distribués selon une loi normale autour de leur valeur nominale avec un écart maximum de $\pm 5\%$.

Pour les vecteurs de test, nous avons choisi un test AC avec l'ensemble des 7 fréquences suivantes $\{2Hz, 20Hz, 200Hz, 2kHz, 20kHz, 200kHz, 2MHz\}$ en considérant le gain et la phase comme étant les deux spécifications de sortie à vérifier. Pour l'ensemble des fautes à détecter, nous avons distingué les deux cas suivants: fautes para-

fautes	composant	description
1	MN1	faute paramétrique, $W = +30\%$
2	MN1	faute paramétrique, $L = +30\%$
3	MN2	faute paramétrique, $W = +30\%$
4	MN2	faute paramétrique, $L = +30\%$
5	MP3	faute paramétrique, $W = +30\%$
6	MP3	faute paramétrique, $L = +30\%$
7	MP4	faute paramétrique, $W = +30\%$
8	MP4	faute paramétrique, $L = +30\%$
9	MN5	faute paramétrique, $W = +30\%$
10	MN5	faute paramétrique, $L = +30\%$
11	MP6	faute paramétrique, $W = +30\%$
12	MP6	faute paramétrique, $L = +30\%$
13	MN7	faute paramétrique, $W = +30\%$
14	MN7	faute paramétrique, $L = +30\%$
15	CC	faute paramétrique, $CC = +30\%$

TAB. 4.3: *Description des fautes utilisées pour les matrices de $PDFi, j$ données dans les tableaux 4.4 et 4.5*

métriques et fautes catastrophiques.

Fautes paramétriques

L'amplificateur opérationnel est composé de 7 transistors et d'une capacité, et comme chaque transistor a deux paramètres (W et L), l'amplificateur possède 15 paramètres. Nous avons considéré 10 déviations par paramètre de 10% à 100% avec un pas de 10%, ce qui donne 150 fautes paramétriques. Comme il est très difficile de donner les résultats de simulation de fautes des 150 fautes paramétriques, un exemple des 15 fautes paramétriques données dans le tableau 4.3 correspondant au déviations de 30% de tous les paramètres de l'*AOP* est utilisé pour l'illustration des résultats de simulation de fautes et de calcul des probabilités de détection de fautes $PDF_{i,j}$.

Les deux tableaux 4.4 et 4.5 présentent les matrices de probabilités de détection de fautes obtenues avec la méthode de simulation de fautes présentée dans ce chapitre.

N^o faute $\{F_i\}$	$\{T_i\}$: fréquences de test (Hz)							$max\{PDF_{i,j}\}$ pour $T_i \in T$
	2E0	2E1	2E2	2E3	2E4	2E5	2E6	
1	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
8	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
9	0.00	0.00	0.00	0.58	0.52	0.51	0.51	0.58
10	0.00	0.00	0.00	0.27	0.43	0.44	0.44	0.44
11	0.00	0.00	0.00	0.72	1.00	1.00	1.00	1.00
12	0.00	0.00	0.00	0.55	0.63	0.63	0.64	0.64
13	0.00	0.00	0.00	0.76	0.72	0.71	0.71	0.76
14	0.00	0.00	0.01	0.45	0.57	0.57	0.57	0.57
15	0.00	0.00	0.00	0.00	0.02	0.03	0.03	0.03
TC_i	53%	53%	53%	75%	79%	79%	79%	80%

TAB. 4.4: *Matrice des probabilités de détection de fautes $PDF_{i,j}$ en considérant le gain de l'aop*

Dans le premier tableau, nous avons les résultats obtenus en considérant le gain de l'*AOP*, alors que le deuxième tableau correspond à la simulation de fautes en vérifiant la phase de l'*AOP*. Les lignes du tableau (à l'exception de la dernière ligne) représentent les fautes F_j et les colonnes (à l'exception de la dernière colonne) représentent les fréquences de test, sachant que chaque fréquence de test correspond à un vecteur de test T_i. La dernière colonne donne pour chaque faute F_j la plus grande probabilité de détection de fautes $PDF_{i,j}$. Cette probabilité est utilisée dans la formule 4.28 pour calculer le taux de couverture global de l'ensemble des vecteurs de test. La dernière ligne donne les taux de couverture TC_i calculés par la formule 4.27 pour chaque fréquence, ces taux de couverture seront utilisés dans le chapitre suivant pour optimiser l'ensemble des vecteurs de test.

La dernière case des tableaux (dernière ligne et dernière colonne) donne le taux de couverture global de l'ensemble des vecteurs de test. On remarque que pour l'ensemble

N^o faute $\{F_i\}$	$\{T_i\}$: fréquences de test (Hz)							$max\{PDF_{i,j}\}$ pour $T_i \in T$
	2E0	2E1	2E2	2E3	2E4	2E5	2E6	
1	1.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00
2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
4	1.00	1.00	0.69	1.00	0.00	1.00	1.00	1.00
5	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
6	1.00	1.00	0.60	1.00	0.00	1.00	1.00	1.00
7	1.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00
8	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
9	1.00	0.96	0.06	0.00	0.40	0.02	0.00	1.00
10	0.07	0.22	0.00	0.00	0.02	0.07	0.15	0.22
11	1.00	1.00	0.00	0.00	0.00	0.00	0.00	1.00
12	1.00	0.96	0.00	0.00	0.14	0.10	0.02	1.00
13	1.00	1.00	0.64	0.00	0.39	0.83	1.00	1.00
14	0.57	0.69	0.00	0.00	0.08	0.00	0.00	0.69
15	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.02
FC_i	84%	85%	53%	53%	33%	60%	60%	86%

TAB. 4.5: *Matrice des probabilités de détection de fautes $PDF_{i,j}$ en considérant la phase de l'aop*

des 7 fréquences et les 15 fautes paramétriques choisies, nous avons un meilleur taux de couverture $TC = 86\%$ en considérant la mesure de la phase, que le taux de couverture $TC = 80\%$ en considérant la mesure du gain. De plus, on remarque qu'on a un ensemble de 9 fautes $\{1, 2, 3, 4, 5, 6, 7, 8, 11\}$ qui sont complètement détectables ($PDF_{i,j} = 1$) par la vérification du gain, alors que pour la vérification de la phase, on a un ensemble de 12 fautes $\{1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12\}$ qui sont complètement détectables ($PDF_{i,j} = 1$). En conclusion, on peut dire que pour cet amplificateur, il est plus intéressant de considérer la mesure de la phase du circuit que la mesure du gain pour détecter les fautes paramétriques .

Dans les tableaux 4.6 et 4.7, nous avons donné la comparaison des résultats de simulation des trois méthodes : simulation de fautes unique, simulation de fautes *Monte Carlo* et la simulation de fautes *M.C.* améliorée, pour les 150 fautes paramétriques. Le tableau 4.6 donne les taux de couverture des trois méthodes et le tableau 4.7 donne les

Mesure en sortie	Taux de couverture global : Amplificateur		
	simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)
phase	100%	82.6%	82.6%
gain	100%	80.4%	80.4%

TAB. 4.6: *Comparaison des taux de couverture obtenus par les différentes méthodes pour les* 150 *fautes paramétriques en considérant le gain de l'AOP.*

temps CPU (s) : Amplificateur			gain en temps de simulation
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)	
94	959	475	49%

TAB. 4.7: *Comparaison des temps CPU pour la simulation des* 150 *fautes paramétriques.*

temps *CPU* des trois méthodes, nécessaires pour achever la simulation de fautes des 150 fautes. On remarque qu'en terme de taux de couverture, l'approche *M.C.* améliorée donne les mêmes résultat ($TC = 82.6\%$ pour la phase et $TC = 80.4\%$ pour le gain) que la simulation de fautes *Monte Carlo*, ce qui prouve que cette approche est aussi précise que la méthode *Monte Carlo*, alors que la simulation de faute unique donne un taux de couverture de 100% pour la mesure de la phase et du gain, ce qui correspond à une erreur de 17.4% pour la phase et 19.6% pour le gain sur le taux de couverture par rapport à la méthode *Monte Carlo*.

En terme de temps de simulation, on remarque que la simulation de faute unique est la plus rapide (ce qui est normal car on n'effectue qu'une seule simulation par circuit fautif), mais on a vu que cette méthode n'est pas efficace en terme de taux de couverture. En revanche, en comparant à celle de *Monte Carlo* , on constate que l'approche *M.C.* améliorée est presque deux fois plus rapide que la simulation de fautes *Monte Carlo* (gain = 49% en temps de simulation de fautes).

Fautes catastrophiques

Pour les fautes catastrophiques, nous avons considéré les 4 fautes les plus probables par transistor définies dans le paragraphe 3.2.4 et qui sont :

– Court-circuit grille source *(GSS) " Gate Source Short "*

Mesure en sortie	Taux de couverture global : Amplificateur		
	simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)
phase	93.3%	93.3%	93.3%
gain	93.3%	90.0%	90.0%

TAB. 4.8: *Comparaison des taux de couverture obtenu par les différentes méthodes pour les* 30 *fautes catastrophiques.*

temps CPU (s) : Amplificateur			gain en temps de simulation
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)	
22	209	69	67%

TAB. 4.9: *Comparaison des temps CPU pour la simulation des* 30 *fautes catastrophiques.*

– Court-circuit grille drain *(GDS) " Gate Drain Short "*

– Circuit ouvert sur le drain *(DOP) " Drain Open "*

– Circuit ouvert sur la source *(SOP) " Source Open "*

Pour la capacité, nous avons considéré deux fautes catastrophiques, le court-circuit et le circuit ouvert, ce qui donne 30 fautes catastrophiques au total pour les 7 transistors et la capacité de l'amplificateur de la figure 4.12. Les résultats de simulation des fautes catastrophiques des trois méthodes, en terme de taux de couverture et de temps de simulation sont respectivement donnés dans les tableaux 4.8 et 4.9.

On remarque que pour le taux de couverture l'approche *Monte Carlo* améliorée donne les même résultats que la méthode *Monte Carlo*. On remarque aussi que la méthode de simulation de fautes unique donne un taux de couverture correct pour la mesure de la phase, et un taux de couverture de 93.3% au lieu de 90.0% (soit une erreur de 3%) pour la mesure du gain. Pour cet exemple là, l'erreur engendrée par la méthode de simulation de fautes unique est minime, ceci est dû au fait que la plupart des fautes catastrophiques de l'amplificateur sont complètement détectables ($PDF = 1$), et dans ce cas de figure l'approche de simulation de fautes unique donne des résultats corrects. En terme de temps de simulation, la méthode *Monte Carlo* améliorée est 3 fois plus rapide que l'approche *Monte Carlo*.

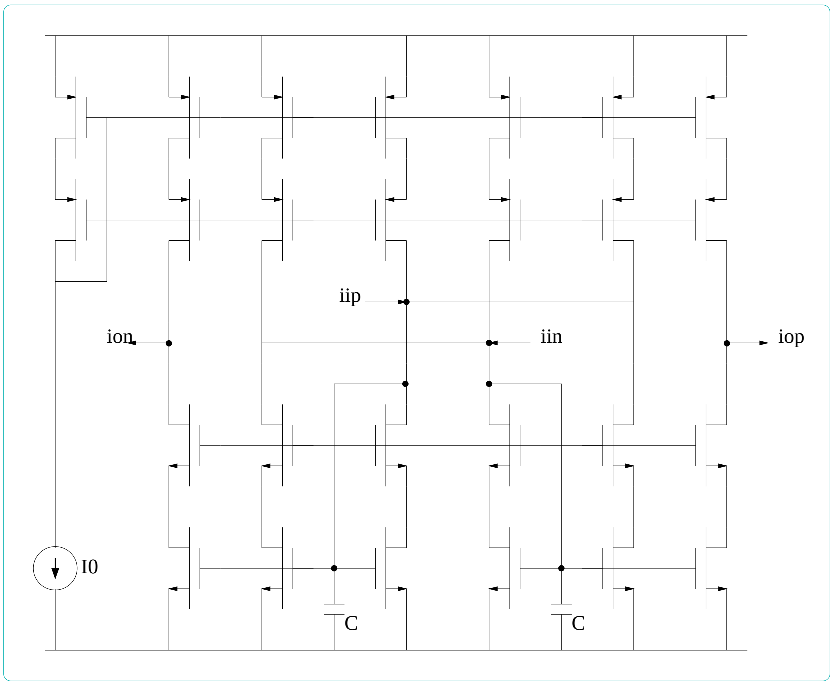

FIG. 4.13: *Intégrateur en mode courant.*

4.6.2 Intégrateur en mode courant

Le deuxième circuit de validation que nous avons choisi est l'intégrateur en mode courant de la figure 4.13, conçu pour une technologie 0.6μ. Nous avons supposé que tous les paramètres de conception de l'amplificateur (W et L des transistors et les valeurs des capacités) sont distribués selon une loi normale autour de leur valeur nominale avec un écart maximum de $\pm 5\%$.

Pour les tests, nous avons choisi en entrée du circuit l'ensemble des 10 fréquences suivantes :

$$\{0.1kHz, 0.3kHz, 1.0kHz, 3.0kHz, 10kHz, 30kHz, 0.1MHz, 0.3MHz, 1MHz, 3MHz\}$$

et en sortie nous avons considéré la mesure du gain en courant ($\frac{i_{op}-i_{on}}{i_{ip}-i_{in}}$), avec :

- i_{ip} l'entrée positive et i_{in} l'entrée négative
- i_{op} la sortie positive et i_{on} la sortie négative

Pour l'ensemble des fautes à détecter, nous avons aussi distingué les deux cas suivants: fautes paramétriques et fautes catastrophiques.

Taux de couverture global : Intégrateur		
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)
84.3%	69.6%	70.4%

TAB. 4.10: *Comparaison des taux de couverture obtenus par les différentes méthodes pour les* 540 *fautes paramétriques de l'intégrateur.*

temps CPU (s) : Intégrateur			gain en temps de simulation
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)	
632	6325	3426	46%

TAB. 4.11: *Comparaison des temps CPU pour la simulation des* 540 *fautes paramétrique de l'intégrateur.*

Fautes paramétriques

Comme pour l'amplificateur opérationnel, nous avons considéré 10 déviations par paramètre de 10% à 100% avec un pas de 10%, chaque déviation représente une faute paramétrique. L'intégrateur possède 2 capacités et 26 transistors, on a un paramètre par capacité (valeur de la capacité) et deux paramètres par transistor (W et L). Ce qui au final donne 54 paramètres et 540 fautes paramétriques.

Les résultats de simulation des fautes paramétriques des trois méthodes, en terme de taux de couverture et de temps de simulation sont respectivement donnés dans les tableaux 4.10 et 4.11. D'après le tableaux 4.10, on remarque que pour le taux de couverture, l'approche *Monte Carlo* améliorée donne des résultats très proches de ceux obtenus avec la méthode *Monte Carlo* (70.4% contre 69.6%). Cette légère différence est due au fait qu'on a supposé que les écarts-types des valeurs simulées des circuits fautifs et du circuit correct sont égaux, ce qui n'est pas toujours vrai. En ce qui concerne les temps de simulation, le tableau 4.11 montre que la méthode *Monte Carlo* a un temps de simulation 1.75 fois plus grand que l'approche *Monte Carlo* améliorée.

Taux de couverture global : Intégrateur		
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)
90.2%	85.5%	85.5%

TAB. 4.12: *Comparaison des taux de couverture obtenus par les différentes méthodes pour les* 54 *fautes catastrophiques de l'intégrateur.*

temps CPU (s) : Intégrateur			gain en temps de simulation
simulation unique	Monte Carlo (10 simulations)	M.C. améliorée ($NMax = 10$ simulations)	
63	632	170	73%

TAB. 4.13: *Comparaison des temps CPU pour la simulation des* 54 *fautes catastrophiques de l'intégrateur.*

Fautes catastrophiques

Pour les fautes catastrophiques de l'intégrateur, nous n'avons considéré que les fautes de court-circuit, d'où pour les transistors, nous avons simuler les deux court-circuits les plus probables qui sont :

- Court-circuit grille source *(GSS) " Gate Source Short "*
- Court-circuit grille drain *(GDS) " Gate Drain Short "*

Pour les 2 capacités et 26 transistors de l'intégrateur de la figure 4.13, nous avons 54 fautes catastrophiques.

Les résultats de simulation des fautes catastrophiques des trois méthodes, en terme de taux de couverture et de temps de simulation sont respectivement donnés dans les tableaux 4.12 et 4.13. D'après les deux tableaux, on remarque que l'approche *Monte Carlo* améliorée donne les même résultats en terme de taux de couverture que la méthode *Monte Carlo* pour un temps de simulation divisé par 3.72. On remarque aussi que le gain en temps de simulation pour les fautes catastrophiques est plus important que pour les fautes paramétriques, ceci est dû au fait que les fautes catastrophiques sont plus faciles à détecter que les fautes paramétriques.

4.6.3 Filtre passe bas à capacités commutées

Le filtre passe bas à capacités commutées d'ordre 5 de la figure 4.14 est utilisé comme troisième circuit de validation avec le gain de la fonction de transfert comme spécification de sortie. Etant donné les interrupteurs du circuit en question, il est impossible d'effectuer une analyse *AC* pour la simulation de la fonction de transfert du filtre. Nous avons donc effectué une analyse transitoire avec une fonction " *pulse* " représentant la fonction de *Dirac* en entrée du circuit. La transformation de *Fourrier* du signal de sortie permet d'extraire la fonction de transfert du filtre dans le domaine des fréquences. Comme dans cet exemple nous souhaitons détecter les fautes sur les capacités du filtre, et pour accélérer les simulations analogiques, nous avons représenté les amplificateurs et les interrupteurs du circuit par leur macro-modèles équivalents.

De même que pour le premier exemple, nous considérerons que pour le bon filtre et les filtres fautifs, les valeurs de toutes les capacités sont distribuées selon une loi normale (loi de Gauss) avec $\pm 5\%$ de tolérance. L'ensemble des vecteurs de test utilisés est donné par l'ensemble des fréquences suivantes :

$$T = \{0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5, 10\}(kHz)$$

Comme les fautes paramétriques sont les plus difficiles à détecter, nous n'avons considéré dans cet exemple-là que les fautes paramétriques. Le circuit contient 17 capacités et nous avons considéré 2 fautes paramétriques par capacité correspondant aux déviations de -50% et $+100\%$.

Pour illustrer les déviations de la fonction de transfert en sortie du filtre, dues aux variations du processus de fabrication, les courbes des deux figures 4.15 et 4.16 montrent les réponses respectives des 30 simulations *Monte Carlo* pour le circuit correct et le circuit fautif contenant la faute (-50% de $C054$). On peut voir pour les deux courbes que les plages de variations de la fonction de transfert dépendent de la fréquence ; pour les fréquences extrêmes (hautes et basses), les plages de variations de la sortie sont petites, alors que pour les fréquences moyennes, les plages de variations sont plus grandes. Ces résultats sont illustrés dans le tableau 4.14 qui donne les moyennes μ_{ref} et les écart-types σ_{ref} du gain de la fonction de transfert du filtre correct ; ces résultats sont calculés à partir de 100 simulations *Monte Carlo*. Les histogrammes (pour différentes fréquences) du gain de la fonction de transfert du bon filtre et du filtre fautif sont donnés respectivement dans les figures 4.17 et 4.18.

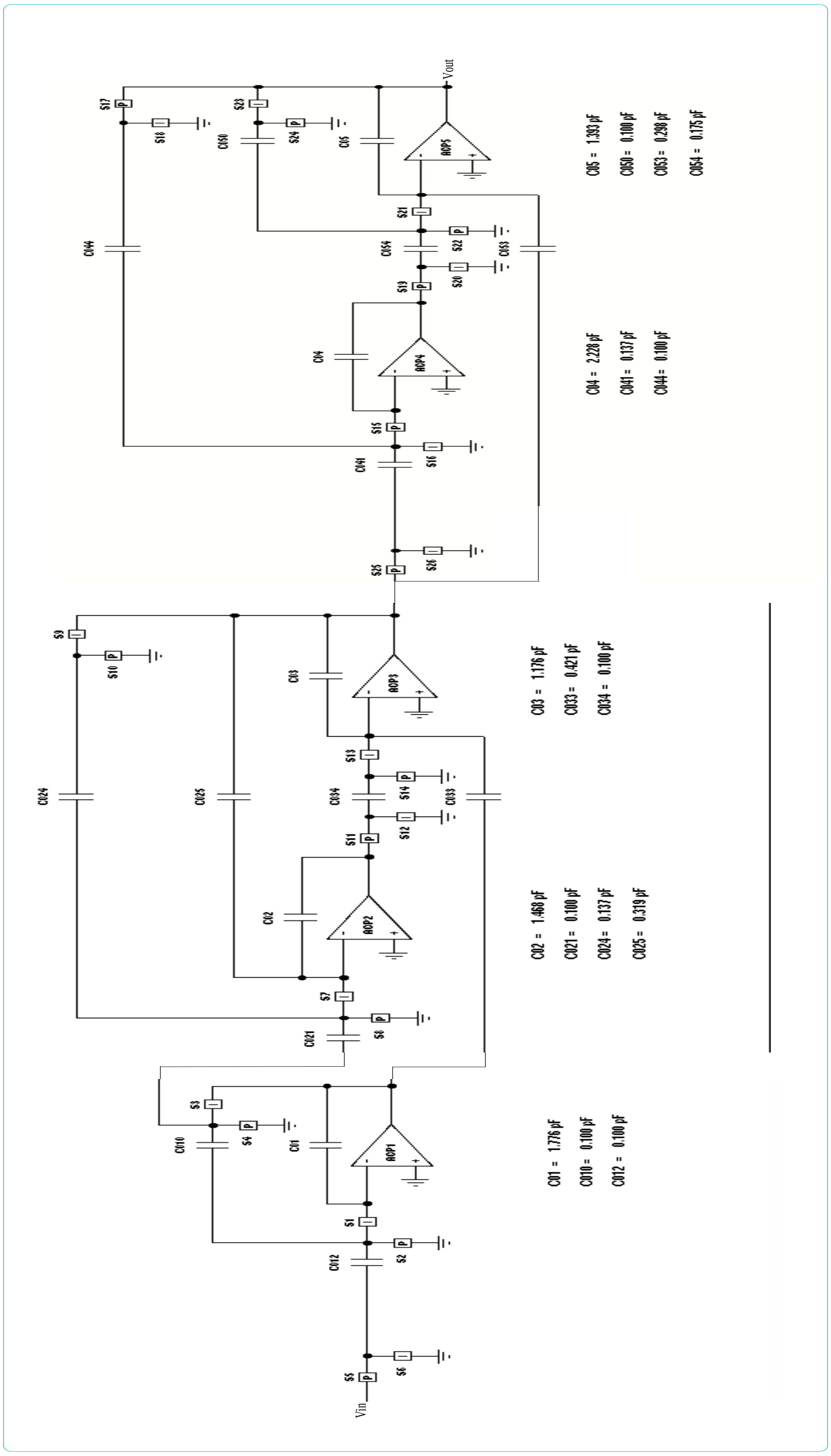

FIG. 4.14: *Filtre passe bas à capacités commutées d'ordre 5.*

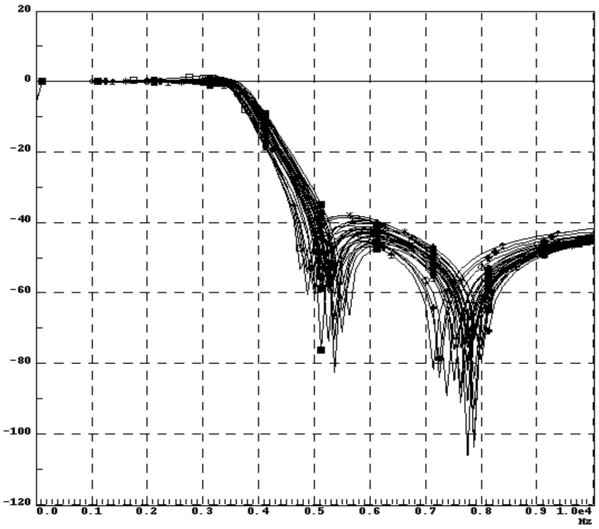

FIG. 4.15: *Fonction de transfert simulée du filtre correct en dB.*

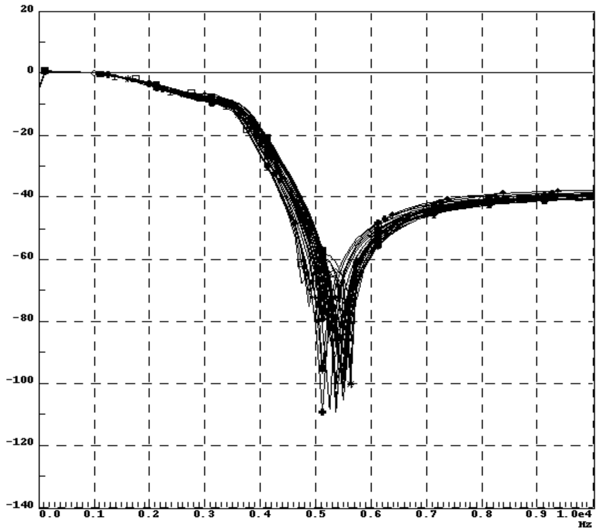

FIG. 4.16: *Fonction de transfert simulée du filtre fautif contenant la faute (−50% de C054) en dB.*

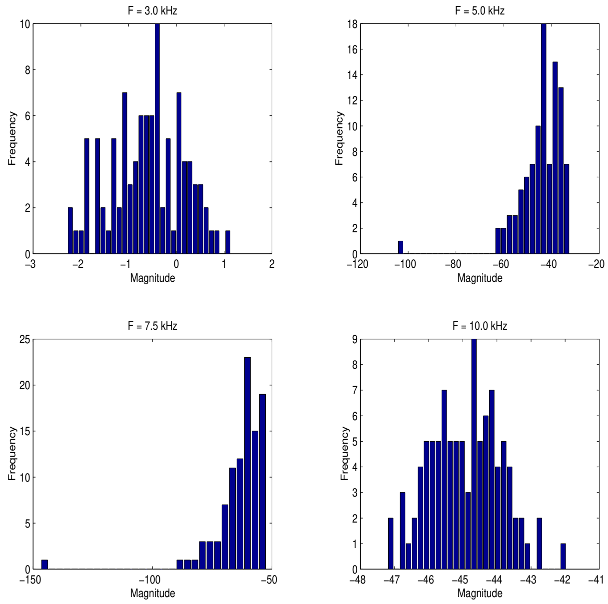

FIG. 4.17: *Histogrammes du gain du filtre correct pour différentes fréquences.*

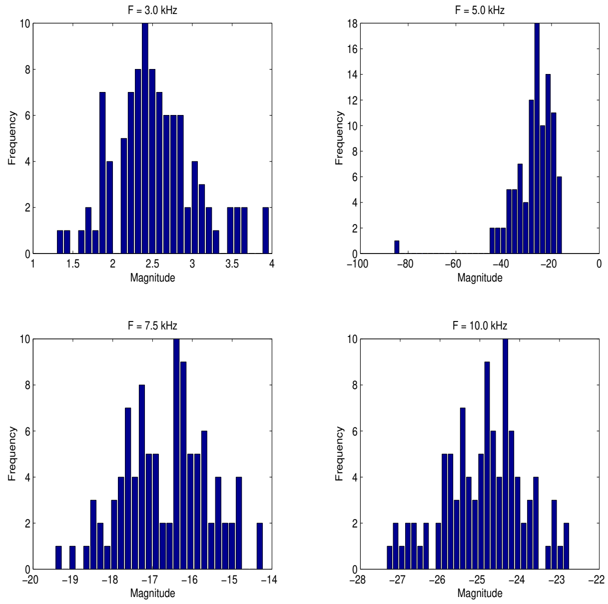

FIG. 4.18: *Histogrammes du gain du circuit fautif contenant la faute (-50% de $C054$) pour différentes fréquences.*

fréquence (kHz)	moyenne $\mu_{ref}(dB)$	écart-type σ_{ref} (dB)
0.5	0.05	0.03
1.5	0.02	0.05
2.5	0.12	0.30
3.5	-0.61	0.78
4.5	-23.77	3.78
5.5	-47.65	7.70
6.5	-45.22	2.19
7.5	-60.49	6.94
8.5	-51.52	2.25
9.5	-45.98	1.11
10.0	-44.70	0.93

TAB. 4.14: *La moyenne et l'écart-type du gain du filtre correct*

D'après le tableau 4.14, pour les fréquences moyennes qui sont dans l'intervalle $[4.5kHz, 8.5kHz]$, l'écart-type est plus grand que pour les fréquences qui sont dans l'intervalle $[0, 3.5kHz] \cup [9kHz, 10kHz]$. Comme on sait (voir section 4.4.1) qu'une faute f est complètement détectable si les deux distributions D_{ref} du bon circuit et D_f du circuit fautif sont complètement disjointes, et que les dispersions des distributions sont données par les écart-types, on peut déduire que pour un vecteur de test donné, plus l'écart-type de la distribution de sortie est petit et plus le vecteur de test détectera de fautes. Ce qui revient à dire que, même en éliminant les fréquences de test pour lesquelles on a un grand écart-type, on aura toujours le même taux de couverture tout en diminuant le temps de simulation. Pour valider ce constat, nous avons comparé notre approche avec l'approche *Monte Carlo* en utilisant les deux ensembles de fréquences : l'ensemble global des fréquences et l'ensemble réduit des fréquences ne contenant que les fréquences de test pour lesquelles l'écart-type σ_{ref} est petit.

A titre d'exemple, le tableau 4.15 représente la matrice des probabilités de détection de fautes $PDF_{i,j}$ pour l'ensemble des cinq fréquences de test ayant un petit écart-type. Pour les fautes, nous avons montré uniquement les résultats de simulation pour les 22 fautes paramétriques qui ne sont pas complètement détectables ($PDF_{i,j} = 1$) par les cinq fréquences de test. On remarque que le taux de couverture global obtenu avec uniquement l'ensemble des cinq fréquences de test est de 100%.

faute $\{F_i\}$	nom cap.	déviation en %	$\{T_i\}$: fréquences de test (kHz)						max $\{PDF_{i,j}\}$
			0.5	1.5	2.5	3.5	9	10	
1	c01	+100%	1.00	1.00	1.00	1.00	0.49	1.00	1.00
2	c02	-50%	1.00	1.00	1.00	1.00	0.47	0.50	1.00
3	c02	+100%	1.00	1.00	1.00	1.00	0.28	1.00	1.00
4	c03	-50%	1.00	1.00	1.00	1.00	1.00	1.00	1.00
5	c04	-50%	0.29	0.07	1.00	1.00	1.00	1.00	1.00
6	c05	+100%	1.00	1.00	0.27	1.00	0.53	1.00	1.00
7	c010	-50%	1.00	1.00	1.00	0.39	0.18	0.27	1.00
8	c010	+100%	1.00	1.00	1.00	1.00	0.03	0.01	1.00
9	c012	-50%	1.00	1.00	1.00	1.00	0.51	1.00	1.00
10	c021	+100%	1.00	1.00	1.00	1.00	0.68	1.00	1.00
11	c024	-50%	1.00	1.00	1.00	1.00	0.04	0.00	1.00
12	c024	+100%	1.00	1.00	1.00	1.00	0.36	1.00	1.00
13	c025	-50%	0.20	0.24	0.28	1.00	0.14	0.17	1.00
14	c025	+100%	0.17	0.37	1.00	1.00	0.13	0.17	1.00
15	c034	-50%	1.00	1.00	1.00	1.00	0.28	1.00	1.00
16	c034	+100%	1.00	1.00	1.00	1.00	0.43	0.46	1.00
17	c044	-50%	1.00	1.00	1.00	0.67	0.08	0.07	1.00
18	c044	+100%	1.00	1.00	1.00	0.43	0.27	0.40	1.00
19	c050	-50%	1.00	1.00	1.00	1.00	0.21	0.30	1.00
20	c050	+100%	1.00	1.00	1.00	1.00	0.03	0.15	1.00
21	c053	-50%	0.51	1.00	1.00	0.18	0.54	1.00	1.00
22	c054	+100%	0.28	0.08	1.00	1.00	1.00	1.00	1.00
TC_i			89.5%	90.4%	95.7%	93.1%	60.8%	75.0%	100%

TAB. 4.15: *Matrice des probabilités de détection de fautes $PDF_{i,j}$ en considérant le gain de la fonction de transfert du filtre.*

ensemble de fréquences	temps CPU (s) : Filtre		gain en temps de simulation
	Monte Carlo (5 simulations)	M.C. améliorée ($NMax = 5$ simulations)	
global	3060	2646	13.5%
réduit	3060	2268	25.8%

TAB. 4.16: *Comparaison des temps CPU pour la simulation des* 34 *fautes paramétriques.*

Le tableau 4.16 donne les comparaisons en terme de temps de simulation de la méthode *Monte Carlo* avec l'approche *M.C.* améliorée pour les deux ensembles de fréquences : global et réduit. On remarque que le temps de simulation de fautes pour la méthode *Monte Carlo* ne dépend pas de l'efficacité des vecteurs de test et est identique pour les deux ensembles, alors que le temps de simulation de la deuxième approche dépend de l'efficacité de l'ensemble de vecteurs de test de départ ; plus on a un jeu de vecteurs de test efficace et plus la simulation de fautes est rapide. En effet, l'algorithme de simulation de fautes pour la méthode *M.C.* améliorée stoppe les simulations dès que l'une des deux conditions énoncées dans les propositions 4.24 et 4.26 est vérifiée, et ces conditions dépendent des écarts-types des distributions du circuit correct et des circuits fautifs.

4.7 Conclusion

Ce chapitre présente une nouvelle approche pour la simulation de fautes pour les circuits analogiques avec prise en compte des tolérances dues aux variations du processus de fabrication. Cette approche est basée sur l'utilisation des probabilités de détection de fautes.

Nous avons présenté les problèmes engendrés par les variations du processus de fabrication sur la simulation de fautes pour les circuits analogiques, ainsi que les différentes approches proposées dans la littérature. Ensuite, nous avons introduit la notion de probabilité de détection de fautes *PDF* permettant de définir avec précision à quel degré une faute est détectable. Comme pour les circuits analogiques les fautes peuvent être partiellement détectables, nous avons défini une nouvelle formule pour le calcul du taux de couverture d'un ensemble de tests basée sur les probabilités de détection de fautes.

En général, le calcul de la probabilité de détection de fautes nécessite des simulations *Monte Carlo* qui sont très coûteuses en temps de simulation. Pour réduire le temps de simulation de fautes, nous avons proposé une amélioration de la méthode *Monte Carlo* pour le calcul de *PDF*. L'algorithme utilisé est basé sur l'application de deux tests permettant de déduire la valeur de *PDF* (dans le cas ou un des deux tests est vérifié) sans avoir à effectuer toutes les simulations *Monte Carlo*.

Pour valider notre approche, nous avons utilisé trois circuits : un amplificateur opérationnel à deux étages, un intégrateur en mode courant et un filtre passe bas à capacités commutées d'ordre 5. Pour l'ensemble des fautes à simuler, nous avons considéré les deux cas de fautes : fautes paramétriques et fautes catastrophiques. En termes d'efficacité (taux de couverture), nous avons montré que l'approche *Monte Carlo* améliorée donne les mêmes résultats que la méthode *Monte carlo*. Par contre, pour la vitesse de simulation de fautes, l'approche *Monte Carlo* améliorée est plus rapide que la méthode *Monte Carlo* et le gain en temps de simulation varie de 25% à 73% (soit 1.37 à 3.75 fois plus rapide). En général, le gain en temps de simulation est plus important dans le cas des fautes catastrophiques que dans le cas des fautes paramétriques, car ces dernières sont plus difficiles à détecter.

Chapitre 5

Optimisation des tests de production pour les circuits analogiques

Pour les circuits analogiques, les vecteurs de test dépendent de la nature du circuit à tester. Il est donc impossible de développer un générateur automatique de vecteurs de test pour tous les types de circuits. Des travaux de recherche sur la génération automatique des tests de production pour les circuits analogiques (*ATPG*) ont été publiés ces dernières années [NCBA93], [HKSZ99] et [ACK99], mais ces ATPGs ne concernent que certains types de circuits en général les circuits linéaires et ne peuvent pas être généralisés pour tous les types de circuits analogiques.

Le coût du test dépend du nombre de vecteurs de test à appliquer et de l'ordre dans lesquels ces vecteurs sont appliqués. En effet, plus on a de vecteurs et plus le temps de test est grand. Comme le test d'un circuit fautif est arrêté dès que la faute est détectée, il est avantageux en terme de temps de test de placer les vecteurs qui détectent le plus de fautes au début du test. L'optimisation automatique de jeux de vecteurs de test pré-existant permet de réduire le nombre de vecteurs dans l'ensemble de test initial en éliminant des tests redondants ou des tests qui ne détectent pas de fautes. Les tests sélectionnés sont ensuite classés selon leur capacité à détecter le plus de fautes, ce qui permet de réduire le coût du test, tout en conservant le même taux de couverture que celui de l'ensemble de départ.

5.1 Evaluation d'un jeu de vecteurs de test

Les tests de production sont utilisés dans la phase de fabrication des circuits en grande série et sont appliqués à des milliers de composants. Les deux caractéristiques importantes d'un jeu de vecteurs destinés au test de production sont :

1. Efficacité du jeu de vecteurs de test
2. Coût du jeu de vecteurs de test

5.1.1 Efficacité d'un ensemble de tests de production

L'efficacité d'un ensemble de tests à détecter les fautes est donnée par le taux de couverture du jeu de vecteurs de test. Pour les circuits numériques, comme les fautes sont soit détectables soit indétectables, le taux de couverture d'un ensemble de vecteurs de test est défini comme le rapport des fautes détectables par le nombre global des fautes. Pour tenir compte des variations du processus de fabrication, nous avons défini un taux de couverture pour les circuits analogiques en utilisant les probabilités de détection de fautes " *PDF* " définies dans le chapitre précédent. Pour l'optimisation des tests de production, nous distinguerons deux définitions du taux de couverture qui sont :

1. taux de couverture global d'un ensemble de test noté TC
2. Taux de couverture relatif d'un seul test noté Q

Taux de couverture global d'un ensemble de test

Pour un ensemble T de n tests et un ensemble F de m fautes, la définition du taux de couverture global d'un ensemble de tests est donné par la formule 4.28 de la section 4.5.4 qui est :

$$TC(T) = \frac{1}{m}\sum_{j=1}^{m}(\underbrace{max}_{T_i \in T}\{PDF_{i,j}\})$$

Le taux de couverture TC représente l'efficacité réelle de l'ensemble des tests choisis.

On définit aussi le taux de couverture des i premiers tests de l'ensemble T par :

$$\boxed{TC(\{T_1 \cdots T_i\}) = \frac{1}{m}\sum_{j=1}^{m} \underbrace{max}_{i\in[1\cdots i]}\{PDF_{i,j}\}} \quad (5.1)$$

Les taux de couverture des ensembles des i premiers tests pour différentes valeurs de i seront utilisés pour le calcul des coûts moyens engendrés par l'application des tests de production.

Taux de couverture relatif d'un seul test

Le taux de couverture relatif d'un ensemble de test traduit l'efficacité d'un seul test par rapport à l'ensemble des tests de production choisis. Pour un ensemble T de n tests et un ensemble F de m fautes, le taux de couverture relatif d'un test T_i par rapport à l'ensemble des tests de l'ensemble T et en considérant les fautes de l'ensemble F est noté $\mathcal{Q}_i$ et défini par la formule suivante :

$$\boxed{\mathcal{Q}_i = \frac{1}{m}\sum_{j=1}^{m}\frac{PDF_{i,j}}{PDF_j}} \quad (5.2)$$

où :

- $PDF_{i,j}$ représente la probabilité que le test T_i détecte la faute F_j
- $PDF_j = Max\{PDF_{i,j}|T_i \in T\}$ représente la meilleure probabilité possible sur la faute F_j en considérant tous les test de l'ensemble T.

De même, si au lieu d'un test T_i, on considère un sous ensemble R de l'ensemble T, le taux de couverture relatif de l'ensemble R par rapport à l'ensemble T en considérant les fautes de l'ensemble F noté $\mathcal{Q}(R)$, est défini par la formule suivante:

$$\boxed{\mathcal{Q}(R) = \frac{1}{m}\sum_{j=1}^{m}\frac{Max\{PDF_{i,j}|T_i \in T'\}}{PDF_j}} \quad (5.3)$$

$$
\begin{array}{cc}
 & \textit{5 tests} \\
\textit{4 fautes} & \begin{bmatrix} 1.0 & 1.0 & 1.0 & 0.5 & 0.9 \\ 1.0 & 0.6 & 0.5 & 0.5 & 0.5 \\ 0.0 & 0.7 & 1.0 & 1.0 & 1.0 \\ 0.0 & 0.5 & 0.4 & 0.6 & 0.7 \end{bmatrix} \\
Qi & \begin{bmatrix} 50\% & 75\% & 77\% & 71\% & 85\% \end{bmatrix} \\
TCi & \begin{bmatrix} 50\% & 70\% & 72\% & 65\% & 77\% \end{bmatrix}
\end{array}
\qquad
\begin{array}{c}
\textit{PDFj} \\
\begin{bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ 0.7 \end{bmatrix} \\
\begin{bmatrix} 92\% \end{bmatrix} \\
TC
\end{array}
$$

FIG. 5.1: *Exemple de calcul des taux de couverture global et relatif.*

Le taux de couverture relatif sera utilisé pour classer les tests entre eux dans le but d'éliminer les tests redondants et ceux qui ne détectent pas de fautes et réduire ainsi le coût du test de production. Il faut noter que le taux de couverture relatif Q_i d'un test T_i est différent du taux de couverture TC_i défini par la formule 4.27 dans la section 4.5.4. En effet, TC_i représente l'efficacité réelle d'un test T_i, c'est à dire le rapport des fautes détectables par les fautes globales alors que Q_i représente l'efficacité relatif du test T_i par rapport à l'ensemble global des tests, il est donc dépendant de l'efficacité des autres tests. Si par exemple, pour un ensemble donné, le taux $Q_i = 100\%$, alors cela ne veut pas dire que le test T_i détecte toutes les fautes de l'ensemble F mais cela veut simplement dire que le test T_i détecte toutes les fautes détectables par l'ensemble T des tests. Par contre, si le taux de couverture $TC_i = 100\%$, alors cela signifie que le test T_i détecte toutes les fautes de l'ensemble F.

Un exemple de calcul des taux de couvertures est donné par la figure 5.1. Nous avons un ensemble de 5 tests et 4 fautes à détecter, la matrice principale représente les probabilités de détection de fautes de chaque faute par chaque test $PDF_{i,j}$. La colonne de droite contient les meilleures probabilités PDF_j de chaque faute F_j, et d'après cette colonne, on remarque que les trois premières fautes sont complètement détectables par l'ensemble des 5 tests ($PDF_1 = PDF_2 = PDF_3 = 1.0$) et que la quatrième faute n'est que partiellement détectable ($PDF_4 = 0.7$) . La première ligne en-dessous de la matrice représente les taux de couverture relatifs $\mathcal{Q}_i$, par exemple le dernier élément de cette ligne :

$$\mathcal{Q}_5 = (\frac{0.9}{1.0} + \frac{0.5}{1.0} + \frac{1.0}{1.0} + \frac{0.7}{0.7}) * \frac{1}{4} = 85\%$$

représente le taux de couverture relatif du test T_5 par rapport à l'ensemble des 5 tests. Le nombre en dessous de la colonne des PDF_j représente le taux de couverture global TC de l'ensemble des 5 tests et est donné par :

$$TC = (1.0 + 1.0 + 1.0 + 0.7) * \frac{1}{4} = 92\%$$

A titre comparatif la deuxième ligne en-dessous de la ligne des $\mathcal{Q}_i$ représente les taux de couverture réels de chaque test qui sont différents des taux de couverture relatifs.

5.1.2 Coût d'un ensemble de tests de production

Les tests de production sont destinés à être appliqués en phase de fabrication de grande série en utilisant des équipements de test " *ATE* " qui sont très coûteux, c'est pourquoi il est très important de limiter le coût d'application d'un ensemble de tests en phase de fabrication. En général, le coût d'un ensemble de tests de production est donné par le temps nécessaire à l'application de tous les tests. Le coût d'un ensemble de test de production ne dépend pas seulement du nombre de tests à effectuer, mais dépend aussi de l'ordonnancement des tests, car durant le test de production, le test d'un circuit défectueux est arrêté dès que l'un des défauts du circuit est détecté. Il est donc plus intéressant d'appliquer en premier les tests les moins coûteux et qui détectent le plus de fautes.

Soit T un ensemble de n tests et F un ensemble de m fautes, on supposera que chaque test T_i a son propre coût qu'on notera $\mathcal{C}_i$. Le coût moyen des tests de l'ensemble T qu'on notera $\mathcal{C}(T)$ est donné par la formule suivante [MSV94] et [HG91]:

$$\boxed{\mathcal{C}(T) = \sum_{i=1}^{n} \mathcal{C}_i(1 - \mathcal{Y}_{i-1})} \tag{5.4}$$

où $(1 - \mathcal{Y}_{i-1})$ représente la probabilité d'effectuer le test T_i sachant que les $(i-1)$ premiers ont déjà été effectués. Comme le test d'un circuit défectueux est arrêté dès qu'un des défauts du circuit est détecté, le test T_i ne sera appliqué que si les $(i-1)$ premiers tests ne détectent aucune faute du circuit. En d'autres termes, le terme $\mathcal{Y}_{i-1}$ représente la probabilité que les $(i-1)$ premiers tests détectent au moins une faute du circuit , en particulier pour le premier test T_1, on a $(1 - \mathcal{Y}_0 = 1)$ car la probabilité de détecter une faute avant d'appliquer le premier test est égale à 0 .

Si chaque faute F_j a une probabilité d'apparition qu'on notera $\mathcal{P}_j$, alors la probabilité que les k premiers tests détectent au moins une faute est égale au rapport de la somme des probabilités de détection PDF de chaque faute pondérée par la probabilité d'apparition $\mathcal{P}_j$ par le nombre de fautes totales. D'où la probabilité $\mathcal{Y}_k$ que les k premiers test détectent au moins une faute est donnée par la formule suivante :

$$\boxed{\mathcal{Y}_k = \tfrac{1}{m} \sum_{j=1}^{m} (\mathcal{P}_j \underbrace{max}_{i \in [1..k]} \{PDF_{i,j}\})} \tag{5.5}$$

et le coût moyen d'application des tests $\mathcal{C}(T)$ de la définition 5.4 devient :

$$\boxed{\mathcal{C}(T) = \sum_{i=1}^{n} \mathcal{C}_i(1 - \frac{1}{m} \sum_{j=1}^{m} (\mathcal{P}_j \underbrace{max}_{i \in [1..k]} \{PDF_{i,j}\}))} \tag{5.6}$$

où $max\{PDF_{i,j} | i \in [1..k]\}$ représente la meilleure probabilité de détecter la faute F_j en considérant les k premiers tests. Les probabilités d'apparition des fautes $\mathcal{P}_j$ sont en général données par les outils d'analyse des fautes par induction *IFA* " *Inductive Fault Analysis* " qui permettent d'extraire les fautes les plus probables d'un circuit à partir de la vue physique du circuit " *layout* " (voir section 2.2.3).

Si maintenant, on suppose que les m fautes sont équiprobables et que la probabilité globale d'apparition d'un circuit défectueux dans la chaîne de fabrication est égale à $\mathcal{P}_0$ (donc $\forall j\ \mathcal{P}_j = \frac{\mathcal{P}_0}{m}$) , alors la probabilité $\mathcal{Y}_k$ de la formule 5.5 devient égale à :

$$\boxed{\mathcal{Y}_k = \tfrac{1}{m^2}\mathcal{P}_0 \sum_{j=1}^{m} \underbrace{max}_{i\in[1..k]}\{PDF_{i,j}\}} \tag{5.7}$$

D'après le définition 5.1, le taux de couverture du sous ensemble $\{T_1, T_2, .., T_k\}$ qu'on notera aussi $TC_{1..k}$ est donné par :

$$TC_{1..k} = TC(\{T_1 \cdots T_k\}) = \frac{1}{m}\sum_{j=1}^{m} \underbrace{max}_{i\in[1..k]}\{PDF_{i,j}\}$$

D'où la formule 5.7 devient:

$$\boxed{\mathcal{Y}_k = 1 - \tfrac{\mathcal{P}_0}{m}\ TC_{1..k}} \tag{5.8}$$

En remplaçant $\mathcal{Y}_k$ dans la définition 5.4, le coût moyen $\mathcal{C}(T)$ d'un ensemble de tests T devient :

$$\boxed{\mathcal{C}(T) = \sum_{i=0}^{n-1} \mathcal{C}_{i+1}(1 - \frac{\mathcal{P}_0}{m}\ TC_{1..i})} \tag{5.9}$$

Si en plus, on suppose que tous les tests ont le même coût intrinsèque $\mathcal{C}_i = \mathcal{C}_0$ (ce qui est le cas si tous les tests nécessitent les mêmes moyens et le même temps d'exécution), alors la formule 5.9 devient :

$$\boxed{\mathcal{C}(T) = \mathcal{C}_0(n - \tfrac{\mathcal{P}_0}{m}\sum_{i=0}^{n-1} TC_{1..i})} \qquad (5.10)$$

avec :

- n = nombre de vecteurs de test à appliquer au circuit
- m = nombre de fautes dans le circuit
- $\mathcal{C}_0$ = coût intrinsèque de chaque test
- $\mathcal{P}_0$ = probabilité d'apparition d'un circuit défectueux dans la chaîne de fabrication
- $TC_{1..i}$ = taux de couverture des i premiers tests avec $TC_{1..0} = 0$

Comme $\mathcal{C}(T)$ dépend des taux de couverture $TC_{1..k}$ des sous ensembles $\{T_1, .., T_k\}$ pour $k \in [1..(n-1)]$, alors on en déduit qu'en plus des nombres de test à appliquer, le coût moyen de test d'un circuit dépend aussi de l'ordre dans lequel les tests sont appliqués.

Pour le cas particulier où l'on n'a que des circuits corrects ($\mathcal{P}_0 = 0$), le coût moyen de test d'un circuit sera égal à $n\mathcal{C}_0$. Ce qui est normal, car dans ce cas-là on applique les n tests pour chaque circuit et donc le coût du test ne dépend plus de l'ordre dans lequel les tests sont appliqués et est égal au coût d'application des n tests.

Pour illustrer cette dépendance entre le coût moyen d'un ensemble de test et l'ordre d'application des tests on prendra un exemple T de 5 tests notés : T_1, T_2, T_3, T_4, T_5, ayant respectivement les taux de couverture intrinsèques suivants : $0.5, 0.25, 0.10, 0.05, 0.05$. Pour faciliter les calculs, on supposera que chaque test T_i détecte uniquement les fautes non détectables par les autres tests. Dans ces conditions, le taux de couverture d'un sous-ensemble de T est égal à la somme des taux de couverture intrinsèques des tests le composant. Si on applique les 5 tests dans l'ordre croissant des taux de couverture intrinsèques, on aura :

$$TC_{1..0} = TC(\{\}) = 0$$
$$TC_{1..1} = TC(\{T_1\}) = 0.5$$
$$TC_{1..2} = TC(\{T_1, T_2\}) = 0.75$$
$$TC_{1..3} = TC(\{T_1, T_2, T_3\}) = 0.85$$
$$TC_{1..4} = TC(\{T_1, T_2, T_3, T_4\}) = 0.90$$
$$TC_{1..5} = TC(\{T_1, T_2, T_3, T_4, T_5\}) = 0.95$$

et on obtient un coût moyen des tests égal à :

$$\mathcal{C}_{1..5} = \mathcal{C}_0(5 - \frac{\mathcal{P}_0}{m}(0 + 0.5 + 0.75 + 0.85 + 0.9) = \mathcal{C}_0(5 - 3\frac{\mathcal{P}_0}{m})$$

alors que si on applique les tests dans l'ordre décroissant des taux de couverture intrinsèque, ou aura :

$$TC_{1..0} = TC(\{\}) = 0$$
$$TC_{1..1} = TC(\{T_5\}) = 0.05$$
$$TC_{1..2} = TC(\{T_5, T_4\}) = 0.1$$
$$TC_{1..3} = TC(\{T_5, T_4, T_3\}) = 0.20$$
$$TC_{1..4} = TC(\{T_5, T_4, T_3, T_2\}) = 0.45$$
$$TC_{1..5} = TC(\{T_5, T_4, T_3, T_2, T_1\}) = 0.95$$

et le coût moyen des tests égal à :

$$\mathcal{C}_{5..1} = \mathcal{C}_0(5 - \frac{\mathcal{P}_0}{m}(0 + 0.05 + 0.1 + 0.20 + 0.45) = \mathcal{C}_0(5 - 0.8\frac{\mathcal{P}_0}{m})$$

Pour application numérique, on prendra $\mathcal{C}_0 = 10\ Francs$, $\mathcal{P}_0 = 0.3$ (30% de circuit défectueux) et $m = 10$ fautes. On supposera aussi qu'on aura 142000 circuits à tester dans lesquels on a 100000 circuits corrects et 42000 circuits défectueux (30% de 142000). Ce qui donne un coût global de test des 142000 circuits de :

$$\mathcal{COUT}_{1..5} = 142000 * 10(5 - 3\frac{0.3}{10}) = 6.97\ \ 10^6\ Francs$$

dans le cas du premier ordonnancement et de :

$$\mathcal{COUT}_{5..1} = 142000 * 10(5 - 0.8\frac{0.3}{10}) = 7.66\ 10^6\ francs$$

dans le cas du deuxième ordonnancement, ce qui fait une différence de $0.69\ 10^6\ Francs$ (10%).

Cet exemple montre l'importance de l'ordre d'applications des tests dans la fonction de coût. Il faut noter que dans cet exemple là, les 10% de gain en coût de test représente le gain dû simplement à l'ordonnancement des tests. L'ordonnancement des tests n'influe que sur le coût de test des 30% de circuits défectueux, car pour les circuits corrects tous les tests sont appliqués. Si on ramène ce gain en coût de test $(0.69\ 10^6\ Francs)$ par rapport au seul coût de test des circuits défectueux $(2.66\ 10^6\ Francs)$, on aura un pourcentage en gain de 26%. Pour la réduction de l'ensemble des tests, le gain en coût de test est en général plus important car la réduction des tests influe beaucoup plus sur le coût de test des circuits corrects (qui représentent plus de 70% du coût global de test) que sur le coût de test des circuits défectueux.

5.2 Optimisation des tests de production

Le but de l'optimisation de l'ensemble des tests de production est de réduire le coût de l'ensemble des tests de départ tout en gardant la même efficacité. On part donc d'un ensemble de départ T ayant un taux de couverture $\mathcal{TC}(T)$ et un coût $\mathcal{C}(T)$ pour arriver à un ensemble optimisé O avec un taux de couverture $\mathcal{TC}(O)$ et un coût $\mathcal{C}(O)$ et vérifiant les deux conditions suivante :

1. $TC(O) = TC(T)$
2. $\mathcal{C}(O) < \mathcal{C}(T)$

Si on privilégie le coût par rapport au taux de couverture, on peut tolérer une certaine perte sur le taux de couverture entre l'ensemble de départ T et l'ensemble optimisé O, dans ce cas là, la première condition devient $(TC(O) \geq \alpha.TC(T))$ avec $(0 < \alpha < 1)$.

D'après la formule 5.9, le coût d'un ensemble de test dépend du nombre des tests et de leur ordonnancement, la réduction du coût d'un ensemble de test de production se fait donc en deux étapes :

1. Réduction du nombre de tests
2. Ordonnancement des tests.

5.2.1 Réduction du nombre de tests

Le problème de trouver un nombre minimum de vecteurs de tests tout en ayant le même taux de couverture est un problème d'optimisation combinatoire " *POC* " $NP - complet$. Pour un ensemble de départ de n tests, la méthode exacte de réduction du nombre de tests conduit à un algorithme de complexité exponentiel : l'énumération des parties d'un ensemble de n éléments est $\mathcal{O}(2^n)$. Jusqu'à ce jour, il n'existe pas encore d'algorithmes polynomiaux pour résoudre les problèmes d'optimisation $NP-complet$ et comme les méthodes exactes de résolution ont une complexité exponentielle sur ces problèmes, seules les méthodes approchées ou heuristiques peuvent résoudre des cas de grande taille. Une méthode approchée ou heuristique pour un problème d'optimisation est un algorithme qui a pour but de trouver une solution réalisable en un temps raisonnable, mais sans garantie d'optimalité.

La méthode approchée que nous avons choisie pour illustrer l'utilisation des probabilités de détections de fautes pour l'optimisation des tests de production est une adaptation de l'heuristique de "*Chvatal*" utilisée en algorithmique pour résoudre les problèmes connus sous le nom de " *recouvrement d'ensemble* ", cet heuristique est déjà utilisé dans [MV89] et [DSGH99] pour l'optimisation des ensembles de tests. L'heuristique utilisée dans cet algorithme est basée sur le fait que les tests ayant un taux de couverture relatif $\mathcal{Q}_i$ élevé ont une grande probabilité d'appartenir à la solution optimale. L'algorithme de réduction des test de production que nous proposons est le suivant :

// définitions
(01) T : l'ensemble des n tests de départ
(02) TC_0 : le taux de couverture de l'ensemble de départ T
(03) R : l'ensemble réduit des tests de production
(04) $TC(R)$: le taux de couverture de l'ensemble réduit R
(05) F : l'ensemble des m fautes à détecter
(06) $PDF_{i,j}$: probabilité de détecter la faute F_j avec le test T_i
(07) $\mathcal{Q}_i$: le taux de couverture relatif du test T_i
(08) α : le paramètre de perte d'efficacité sur le taux de couverture qu'on peut tolérer $0 < \alpha \leq 1$

// algorithme : Réduire$(T, F, PDF_{i,j}, \alpha)$
(09) $R = \emptyset$

tant que $(TC(R) < \alpha.TC_0)$ **faire**
pour *chaque test* $T_i \in T$ **faire**
Recalculer $\mathcal{Q}_i$ *en considérant que les fautes non détectées de l'ensemble* F
fin pour
Choisir le test $(T_k \in T)$ *ayant* $\mathcal{Q}_k$ *maximum*
Ajouter T_k *à l'ensemble* R $(R = R \cup \{T_k\})$
Enlever T_k *de l'ensemble* T $(T = T - \{T_k\})$
pour *chaque faute* $F_j \in F$ **faire**
si $(T_k$ *détecte la faute* $F_j)$ **faire**
Enlever F_j *de l'ensemble* F $(F = F - \{F_j\})$
fin si
fin pour
fin tant que
$T = R$
$R = \emptyset$
tant que $(TC(R) < \alpha.TC_0)$ **faire**
Choisir le dernier test T_i *trouvé parmi les tests restants*
Enlever T_i *de l'ensemble* T $(T = T - \{T_i\})$
Calculer $TC(R \cup \{T_i\})$
si $(TC(R \cup \{T_i\}) > TC(R))$ **faire**
Ajouter T_i *à l'ensemble* R $(R = R \cup \{T_i\})$
fin si
fin tant que

A chaque itération de la première boucle *"tant que"*, on élimine le test sélectionné de l'ensemble T, et les fautes détectées par ce test de l'ensemble F, les deux ensembles T et F changent donc à chaque itération. C'est pourquoi, on est obligé de recalculer à chaque itération les taux de couverture relatifs des tests non encore éliminés en considérant que les fautes non encore détectées. Comme à chaque itération, ce qui nous intéresse est de détecter les fautes restantes, il est logique de recalculer le taux de couverture relatif des tests restants en considérant uniquement les fautes non encore détectées.

La première amélioration apportée par cet algorithme par rapport aux algorithmes présentés dans [MV89] et [DSGH99] est le fait d'utiliser les taux de couverture relatifs $\mathcal{Q}_i$ et non les taux de couverture réels TC_i des tests de production. En effet, un test ayant une probabilité de détection de 0.5 pour une faute F_j dont la probabilité maximum ($PDF_j = 0.5$) est aussi important qu'un autre test ayant une probabilité de détection de 1.0 pour une autre faute F_k ayant une probabilité de détection maximum ($PDF_k = 1.0$). La deuxième amélioration apportée par cet algorithme réside dans la répétition de la boucle *"tant que"* sur le premier ensemble réduit trouvé en commençant par les derniers tests trouvés, ceci permet d'éliminer encore plus des tests redondants. En effet, comme la première boucle n'assure pas un ensemble optimum, la deuxième boucle *"tant que"* permet de réduire encore plus l'ensemble des tests.

Pour illustrer notre algorithme, prenant la matrices $\mathcal{A}$ des $PDF_{i,j}$ suivante :

$$\mathcal{A} = \begin{array}{ccccccc} & T_1 & T_2 & T_3 & T_4 & T_5 & T_6 \\ F_1 & 1.0 & 1.0 & 0.0 & 1.0 & 0.0 & 1.0 \\ F_2 & 1.0 & 1.0 & 0.0 & 1.0 & 0.0 & 1.0 \\ F_3 & 1.0 & 0.0 & 1.0 & 1.0 & 0.0 & 0.0 \\ F_4 & 0.0 & 0.0 & 0.0 & 1.0 & 1.0 & 1.0 \\ F_5 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ F_6 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 \\ F_7 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 \end{array}$$

correspondant à un ensemble T de 6 tests et un ensemble F de 7 fautes. On remarque que les 6 tests permettent de détecter 100% des fautes. En déroulant notre algorithme avec cet exemple, on obtient :

// initialisation
$T = \{T_1, T_2, T_3, T_4, T_5, T_6\}$
$TC_0 = 100\%$
$F = \{F_1, F_2, F_3, F_4, F_5, F_6, F_7\}$
$\alpha = 1.0$

// début de la première boucle tant que
$\mathcal{Q}_1 = 0.43, \quad \mathcal{Q}_2 = 0.43, \quad \mathcal{Q}_3 = 0.43, \quad \mathcal{Q}_4 = 0.57, \quad \mathcal{Q}_5 = 0.29, \quad \mathcal{Q}_6 = 0.43$
meilleur test : $T_i = T_4$
$R = \{T_4\}, \quad TC(R) = 57\%, \quad T = \{T_1, T_2, T_3, T_5, T_6\}, \quad F = \{F_5, F_6, F_7\}$

$\mathcal{Q}_1 = 0.0, \quad \mathcal{Q}_2 = 0.33, \quad \mathcal{Q}_3 = 0.33, \quad \mathcal{Q}_5 = 0.33, \quad \mathcal{Q}_6 = 0.0$

meilleur test : $T_i = T_2$
$R = \{T_4, T_2\}, \;\; TC(R) = 71\%, \;\; T = \{T_1, T_3, T_5, T_6\}, \;\; F = \{F_6, F_7\})$

$\mathcal{Q}_1 = 0.0, \;\; \mathcal{Q}_3 = 0.5, \;\; \mathcal{Q}_5 = 0.5, \;\; \mathcal{Q}_6 = 0.0$
meilleur test : $T_i = T_3$
$R = \{T_4, T_2, T_3\}, \;\; TC(R) = 86\%, \;\; T = \{T_1, T_5, T_6\}, \;\; F = \{F_7\})$

$\mathcal{Q}_1 = 0.0, \;\; \mathcal{Q}_5 = 1.0, \;\; \mathcal{Q}_6 = 0.0$
meilleur test : $T_i = T_5$
$R = \{T_4, T_2, T_3, T_5\}, \;\; TC(R) = 100\%, \;\; F = \emptyset$
// fin de la première boucle tant que

$T = R = \{T_4, T_2, T_3, T_5\}, \;\; R = \emptyset$
$F = \{F_1, F_2, F_3, F_4, F_5, F_6, F_7\}$

// début de la deuxième boucle tant que
dernier test trouvé : $T_i = T_5$
$R = \{T_5\}, \;\; TC(R) = 29\%, \;\; T = \{T_4, T_2, T_3\}, \;\; F = \{F_1, F_2, F_3, F_5, F_6\}$

dernier test trouvé : $T_i = T_3$
$R = \{T_3, T_5\}, \;\; TC(R) = 57\%, \;\; T = \{T_4, T_2\}, \;\; F = \{F_1, F_2, F_5\}$

dernier test trouvé : $T_i = T_2$
$R = \{T_2, T_3, T_5\}, \;\; TC(R) = 100\%, \;\; F = \emptyset$
// fin de la deuxième boucle tant que

Après réduction, on obtient donc l'ensemble réduit $R = \{T_2, T_3, T_5\}$ qui donne le même taux de couverture $TC = 100\%$ que l'ensemble de départ mais avec seulement 3 tests au lieu de 6. D'après cet exemple, on remarque l'importance de la répétition de la boucle *"tant que"* avec l'ordre inverse dans lesquels les tests ont été sélectionnés. On remarque que pour notre exemple, on a réussi à obtenir la meilleure solution, mais en règle générale notre algorithme ne donne pas toujours l'ensemble minimum de tests.

5.2.2 Ordonnancement des tests

Le but de l'ordonnancement des tests de production est de réduire le temps d'application des tests pour les circuits défectueux. En effet, pour les circuits corrects, on est obligé d'appliquer tous les tests choisis pour être sûr qu'ils ne contiennent pas de fautes et donc l'ordonnancement n'est pas nécessaire. Par contre, pour les circuits défectueux, le test du circuit est arrêté dès qu'un des défauts du circuit est détecté. Pour réduire le coût de test, il est donc important d'appliquer en premier :

1. Les tests qui détectent le plus de fautes.
2. Les tests qui ont le moindre coût.
3. Les tests détectant des fautes différentes.

Le problème d'ordonnancement des tests est aussi considéré comme un problème d'optimisation combinatoire " *POC* " $NP - complet$. Si on note n le nombre des tests à ordonnancer, la méthode exacte d'ordonnancement des n tests nécessite l'énumération complète des $n!$ (factoriel n) cas possibles. Cette méthode conduit à un algorithme de complexité en $\mathcal{O}(n!)$ qui devient impossible à résoudre en un temps raisonnable pour $(n > 10)$, et par conséquent, comme pour l'éliminations des tests, le problème d'ordonnancement pour les cas de grande taille ne peut être résolu que par des heuristiques.

Quelques travaux de recherche ont été publiés dans ce domaine aussi bien pour les circuits numériques que pour les circuits analogiques [HG91] et [MSV94], toutes les approches proposées sont basées sur des heuristiques. Dans notre cas, comme notre but est de montrer l'utilisation des probabilités de détection de fautes pour l'optimisation des tests de production, nous avons préféré implémenter un algorithme simple même s'il ne donne pas les meilleures solutions, à un algorithme complexe assurant des meilleurs résultats.

D'après la formule 5.4, on remarque que le coût de test moyen d'un ensemble de n tests appliqués dans l'ordre croissant des indices dépend :

1. des probabilités $\mathcal{Y}_i$ que les i premiers tests détectent au moins une faute, pour i variant de 1 à $(n-1)$. Si les $\mathcal{Y}_i$ sont croissants alors le coût moyen de test $\mathcal{C}(T)$ décroît.
2. des coût intrinsèques $\mathcal{C}_i$ de chaque test T_i. Si les coûts intrinsèques $\mathcal{C}_i$ décroissent, alors le coût moyen de test $\mathcal{C}(T)$ décroît.

D'après ces deux remarques, on déduit que le premier test sera toujours appliqué, que le deuxième test sera appliqué plus souvent que le troisième et ainsi de suite. Il est donc plus avantageux en terme de coût de test d'appliquer en premier :

1. les tests ayant les grandes probabilités de détecter des fautes
2. les tests ayant les faibles coûts intrinsèques $\mathcal{C}_i$.

Pour tenir compte des deux facteurs, nous allons trier les tests suivant la somme $\mathcal{W}_i$ des deux facteurs : probabilité de détecter au moins une faute $\mathcal{Y}_i$ et coût intrinsèque $\mathcal{C}_i$ de chaque test T_i donné par :

$$\boxed{\mathcal{W}_i = \alpha_c \mathcal{C}_i + \alpha_y (1 - \mathcal{Y}_i)} \tag{5.11}$$

Les paramètres α_c et α_y permettent d'exprimer l'importance relative des coûts par rapport à l'efficacité des tests. Le paramètre $\mathcal{W}_i$ nous permettra de classer les tests entre eux, les tests pour lesquels $\mathcal{W}_i$ est petit assurent un meilleurs compromis entre leur coût et leur efficacité à détecter les fautes.

L'algorithme d'ordonnancement que nous avons adopté est basé sur l'ordonnancement des tests selon le paramètre $\mathcal{W}_i$, et sur l'application en priorité des tests assurant un bon compromis entre leur coût et leur efficacité à détecter un maximum de fautes (ayant un $\mathcal{W}_i$ petit). Ce qui au final donne l'algorithme suivant :

// définitions
(01) R : l'ensemble des n tests à ordonnancer
(02) F : l'ensemble des fautes à détecter
(03) O : l'ensemble optimisé (ordonnancé) des tests
(04) $PDF_{i,j}$: probabilité de détecter la faute F_j avec le test T_i
(05) $\mathcal{C}_i$: le coût intrinsèque du test T_i pour tout $i \in [1..n]$
(06) $\mathcal{P}_j$: la probabilité d'apparition de la faute F_j pour tout $j \in [1..m]$

// algorithme : $Ordonnancer(R, F, PDF_{i,j}, \mathcal{C}_i, \mathcal{P}_j))$
(07) **pour** $l = 1$ *à* n **faire**
(08) **pour** *chaque test* $T_i \in T$ **faire**
(09) $m = Card(F)$

$\mathcal{Y}_i = \frac{1}{m}\sum_{j=1}^{m}\mathcal{P}_j PDF_{i,j}$

$\mathcal{W}_i = \alpha_c\mathcal{C}_i + \alpha_y(1-\mathcal{Y}_i)$

fin pour

Choisir le test $T_k \in R$ *ayant* $\mathcal{W}_k$ *minimum*

Ajouter T_k *avec l'ordre* l *à l'ensemble* O $(O = O \cup \{T_k\})$

Enlever T_k *de l'ensemble* R $(R = R - \{T_k\})$

pour *chaque faute* $F_j \in F$ **faire**

si *(*T_k *détecte la faute* F_j*)* **faire**

Enlever F_j *de l'ensemble* F $(F = F - \{F_j\})$

fin si

fin pour

fin pour

Comme pour l'algorithme de réduction, l'ensemble de fautes F change à chaque fois qu'on choisit un test, il est donc important de recalculer les paramètres $\mathcal{W}_i$ des tests restants à chaque itération. On remarque qu'il est plus avantageux de choisir un test qui ne détecte pas beaucoup de fautes mais qui détecte des fautes non détectables par les premiers tests que de choisir un test qui détectent beaucoup de fautes mais dont la plupart sont déjà détectables par les premiers tests. C'est pourquoi, à chaque itération, on recalcule les paramètres $\mathcal{W}_i$ pour les tests restants en ne considérant que les fautes non encore détectées par les premiers tests ordonnancés.

5.2.3 Algorithme d'optimisation des tests de production

L'algorithme final d'optimisation des tests de production utilise les deux algorithmes définis plus haut : l'algorithme de réduction et l'algorithme d'ordonnancement. Ce qui au final donne l'algorithme suivant :

// définitions

T *: l'ensemble des* n *tests de départ*

F *: l'ensemble des* m *fautes à détecter*

$PDF_{i,j}$ *: probabilité de détecter la faute* F_j *avec le test* T_i

R *: l'ensemble réduit des tests de production*

O *: l'ensemble optimisé (réduit et ordonnancé) des tests de production*

(06) α : le paramètre de perte d'efficacité sur le taux de couverture qu'on peut tolèrer $0 < \alpha \leq 1$
(07) $\mathcal{C}_i$: le coût intrinsèque du test T_i
(08) $\mathcal{P}_j$: la probabilité d'apparition de la faute F_j

// algorithme
(09) $R = Réduire(T, F, PDF_{i,j}, \alpha)$
(10) $O = Ordonnancer(R, F, PDF_{i,j}, \{\mathcal{C}_i | T_i \in R\}, \{\mathcal{P}_j | F_j \in F\})$

L'algorithme de réduction des tests " *Réduire()* " prend en entrée l'ensemble des tests T à réduire, l'ensemble des fautes F à détecter, les probabilités de détection de fautes $PDF_{i,j}$ et le paramètre α représentant la perte d'efficacité sur le taux de couverture qu'on tolère avoir entre l'ensemble de départ et l'ensemble réduit. Si on ne tolère aucune perte sur le taux de couverture entre l'ensemble réduit et l'ensemble de départ, on prend $\alpha = 1$, sinon il est inférieur à 1. Si on suppose que les tests ont le même coût intrinsèque et que toutes les fautes ont la même probabilité d'apparition, alors l'algorithme d'ordonnancement " $Ordonnancer()$ " ne prendra en entrée que l'ensemble des tests, l'ensemble des fautes et les probabilités de détection de fautes $PDF_{i,j}$.

Le fait d'appliquer l'algorithme d'ordonnancement sur l'ensemble réduit permet de diminuer considérablement le temps d'exécution du fait que l'ensemble réduit contient moins de test à ordonnancer que l'ensemble de départ. Si on remarque que pour la majorité des circuits, l'ensemble réduit des tests contient en général moins de 10 tests, alors on peut envisager d'appliquer l'algorithme exact d'ordonnancement des tests en énumérant les $n!$ cas possibles.

5.3 Applications et Résultats

Nous avons appliqué l'algorithme d'optimisation sur les trois exemples présentés dans le chapitre précédent. Pour tous les exemples, nous avons supposé que tous les tests ont un même coût intrinsèque et que toutes les fautes ont la même probabilité d'apparition.

5.3.1 Amplificateur opérationnel

Pour l'amplificateur opérationnel de la figure 4.12, nous avons un ensemble de tests de départ composé de 7 fréquences qui sont :

$$T = \{2Hz, 20Hz, 200Hz, 2kHz, 20kHz, 200kHz, 2MHz\}$$

et nous avons considéré deux mesures en sortie de l'amplificateur qui sont : le gain et la phase.

Pour les 150 fautes paramétriques (voir section 4.6.1), nous avons montré qu'on a un meilleur taux de couverture en considérant la mesure de la phase que la mesure du gain (voir tableau 4.6). En utilisant l'algorithme d'optimisation des tests, nous avons trouvé :

- Pour la mesure de la phase, l'ensemble optimisé permettant d'atteindre le taux de couverture de 82.6% se compose de trois tests qui sont :

$$T_1 = 20Hz, T_2 = 2Hz \ \ et \ \ T_3 = 2MHz$$

- Pour la mesure du gain, l'ensemble optimisé permettant d'atteindre le taux de couverture de 80.4% se compose de deux tests qui sont :

$$T_1 = 2MHz \ \ et \ \ T_2 = 2kHz$$

On remarque donc, qu'on a un meilleur taux de couverture mais aussi un coût de test plus élevé en considérant la phase, alors qu'on a un ensemble de test moins coûteux mais également un taux de couverture plus petit en considérant le gain.

Pour les fautes 30 fautes catastrophiques, l'ensemble optimisé permettant d'avoir le taux de couverture de 93.3% pour la mesure de la phase et le taux de couverture de 90.0% pour la mesure du gain se réduit à une seule fréquence :

$$T_1 = 2MHz$$

En terme de temps de simulation de fautes, nous avons constaté dans le chapitre précédent qu'avec la méthode *Monte carlo* améliorée, les fautes catastrophiques sont plus rapides à simuler que les fautes paramétriques. En terme de coût de test, comme les fautes catastrophiques nécessitent moins de tests, on déduit que les fautes catastrophiques sont aussi moins coûteuses à détecter que les fautes paramétriques. En conclusion, on peut dire qu'en général les fautes catastrophiques sont plus faciles à simuler et à détecter que les fautes paramétriques.

5.3.2 Intégrateur en mode courant

Pour l'intégrateur en mode courant de la figure 4.13, nous avons un ensemble de tests de départ composé de 10 fréquences qui sont :

$$\{0.1kHz, 0.3kHz, 1.0kHz, 3.0kHz, 10kHz, 30kHz, 0.1MHz, 0.3MHz, 1MHz, 3MHz\}$$

et en sortie de l'intégrateur, nous avons considéré la mesure du gain différentiel en courant ($\frac{i_{op}-i_{on}}{i_{ip}-i_{in}}$).

Pour les 540 fautes paramétriques, l'ensemble des fréquences de départ peut être réduit à l'ensemble des 3 fréquences suivantes :

$$T_1 = 0.1kHz, T_2 = 0.3MHz \ \ et \ \ T_3 = 3MHz$$

tout en ayant le même taux de couverture de l'ensemble de départ $TC = 70.4\%$. Par contre, pour les 54 fautes catastrophiques, l'ensemble de départ peut être réduit à une seule fréquence

$$T_1 = 0.1kHz$$

qui permet d'atteindre le taux de couverture de 85.5%. De même que pour l'amplificateur opérationnel, on remarque que les fautes catastrophiques sont moins coûteuses à détecter que les fautes paramétriques.

5.3.3 Filtre passe bas à capacités commutées

Pour les 34 fautes paramétriques du filtre passe bas à capacités commutées d'ordre 5 de la figure 4.14, l'ensemble suivant :

$$T = \{0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5, 10\}(kHz)$$

des fréquences de départ peut être optimisé à l'ensemble des deux fréquences suivantes :

$$T_1 = 2.5kHz \ \ et \ \ T_2 = 3.5kHz$$

ces deux seules fréquences de test permettent d'obtenir un taux de couverture de 100%. Si on accepte une perte d'efficacité de l'ensemble optimisé de l'ordre de 5%, en tolèrant donc un taux de couverture minimum de 95%, l'ensemble des fréquences de test de départ peut être réduit à une seule fréquence $T_1 = 2.5kHz$, qui permet d'obtenir un taux de couverture $TC = 95.7\%$.

5.4 Conclusion

Réduire le coût des tests de production tout en ayant une meilleure efficacité est l'une des problématiques posée par le test des circuits analogiques. Les variations des paramètres des circuits analogiques rendent le problème d'optimisation des tests de production plus complexe, car avec les tolérances sur les mesures en sortie des circuits, la comparaison de l'efficacité des tests devient plus difficile.

Dans ce chapitre, nous avons montré l'utilisation des probabilités de détection de fautes pour une optimisation efficace des ensembles de tests. Nous avons montré que le coût d'un ensemble de test dépend à la fois du nombre des tests à appliquer et de l'ordre dans lequel les tests sont appliqués. L'optimisation des tests de production se décompose donc en deux étapes :

1. Réduction des tests.

2. Ordonnancement des tests choisis.

Nous avons vu que ces deux problèmes sont des problèmes d'optimisation combinatoire *NP-complet* et dont les méthodes exactes de résolution nécessitent des algorithmes de complexité exponentielle qui sont impossibles à résoudre en un temps raisonnable lorsqu'on a un grand nombre de tests. Nous avons donc proposé pour chacun des deux problèmes des méthodes approchées "*heuristiques*" basées sur les probabilités de détection de fautes.

L'algorithme d'optimisation final a été utilisé pour réduire les ensembles de fréquences de test pour les trois circuits analogiques de validation : l'amplificateur opérationnel, l'intégrateur en mode commun et le filtre passe bas à capacités commutées. Cet algorithme a permis de réduire considérablement l'ensemble des fréquences de tests pour les trois circuits de validation.

Chapitre 6

Mise en œuvre logicielle

Pour pouvoir valider notre approche de simulation de fautes et d'optimisation des tests de production avec prise en compte des tolérances, nous avons développé un prototype logiciel pour l'optimisation des tests de production avec prise en compte des tolérances basé sur les probabilités de détection de fautes appelé *ATSO* pour " *Automatic Test Set Optimization* " ou optimisation des ensembles de tests de production.

Pour pouvoir réutiliser les descriptions utilisées par les concepteurs de circuits analogiques, l'outil *ATSO* travaille sur des descriptions de type *SPICE* utilisées en général par tous les outils de simulation analogique. Comme le module de simulation de fautes intégré à *ATSO* est basé sur un simulateur électrique, dans notre cas le simulateur électrique *ELDO* de chez *ANACAD*, les options et les caractéristiques de l'outil *ATSO* sont celles du simulateur électrique, en conséquence dans notre cas, se sont celles du simulateur *ELDO*. L'outil *ATSO* permet donc d'analyser des tests en mode continu *DC*, en mode petit signaux *AC* et en mode transistoire.

L'avantage de l'outil *ATSO* est qu'il permet de calculer les taux de couverture avec une bonne précision et aussi d'optimiser les tests préexistants sans se soucier de la fonctionnalité du circuit à tester. Le module d'optimisation des tests de productions de l'outil *ATSO* peut aussi optimiser statistiquement des tests de productions en analysant des résultats de mesure déjà effectués sur un échantillon de circuits fabriqués [HG91] et [MSV94]. Grâce à son module de simulation de fautes avec prise en compte des tolérances, l'outil *ATSO* permet aussi d'étudier avec précision l'efficacité des méthodes de conception en vue du test *DFT* que l'on peut être amené à développer pour des circuits analogiques.

6.1 Architecture globale de l'outil (ATSO)

Le prototype *ATSO* de l'outil d'optimisation des tests de production avec prise en compte des tolérances que nous avons développé prend en entrée :

- La description du circuit avec les tests à optimiser au format du simulateur électrique.
- La liste de fautes à détecter.
- Les modèles de fautes à utiliser au format du simulateur électrique.
- Les variations sur les paramètres du circuit dues au processus de fabrication.

Et en sortie, l'outil *ATSO* génère :

- Le taux de couverture global.
- La liste des fautes avec leur probabilité de détection.
- L'ensemble optimisé des tests de production.

L'architecture globale de l'outil est donnée dans la figure 6.1, l'outil se décompose en trois modules qui sont :

1. Module d'injection de fautes *AFI " Analog Fault Injection "*
2. Module de simulation de faute *AFS " Analog Fault Simulation "*
3. Module d'optimisation des ensembles de tests *TSO " Test Set Optimization "*

ATSO supporte l'utilisation des trois modes standards de simulations électriques : simulation continue *AC*, simulation petit signaux *DC* et simulation transitoire. Les tests à simuler et à optimiser sont donnés dans la description du fichier au format *SPICE* et les spécifications à mesurer en sortie du circuit sont aussi données dans le fichier de description du circuit en utilisant la commande *.PRINT* qui permet d'écrire dans le fichier de sortie les résultats de simulation que l'on souhaite comparer. La commande *.PRINT* permet de spécifier, pour chaque type de simulation, le paramètre à observer et à comparer (tension, courant, gain, ... etc) entre le circuit correct et les circuits fautifs.

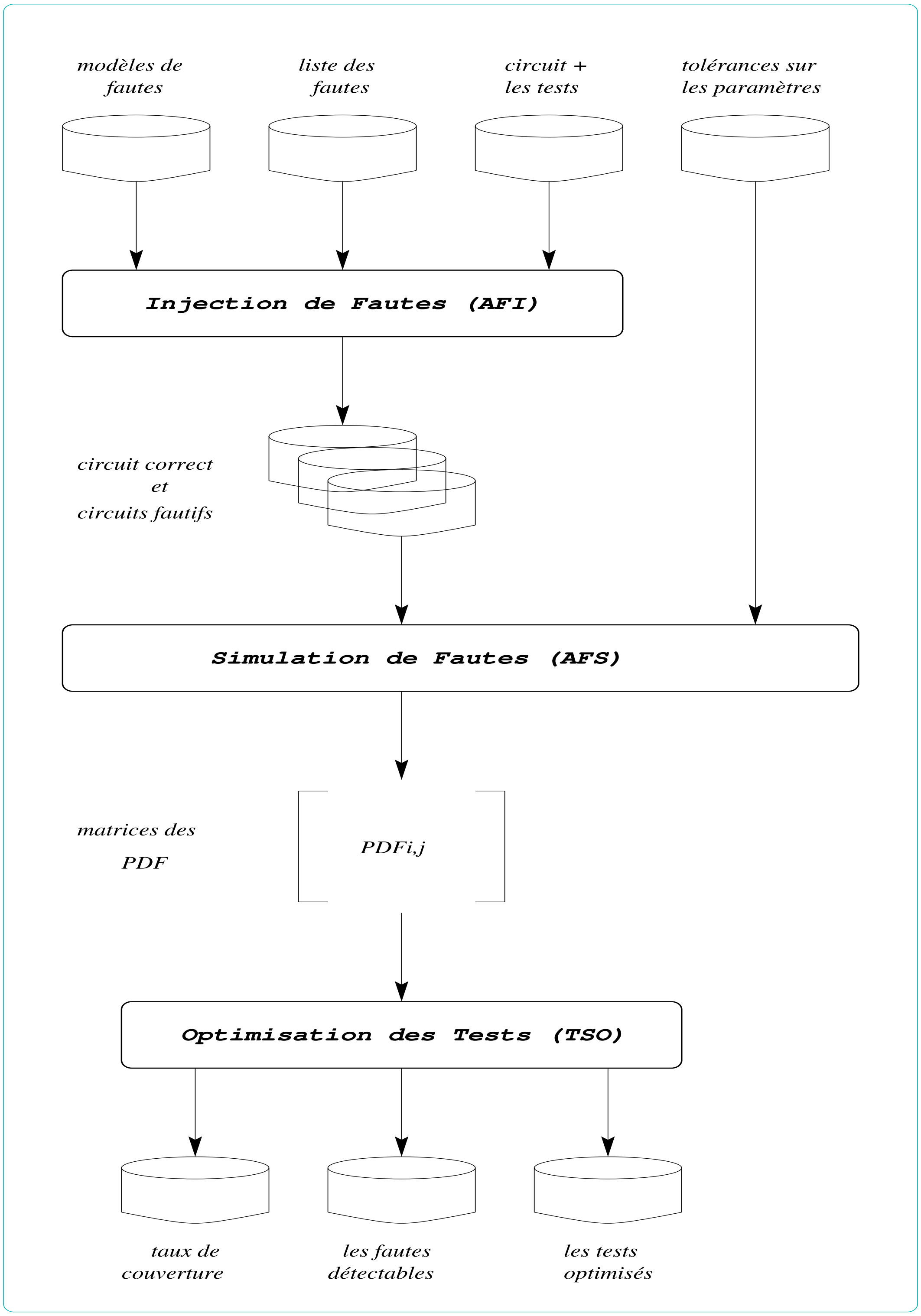

FIG. 6.1: *Architecture globale de l'outil ATSO.*

Un exemple de l'amplificateur opérationnel à deux étages avec les tests et les spécifications à comparer en sortie est donné ci dessous :

```
* FICHIER : aop.cir
* AMPLIFICATEUR OPERATIONNEL A DEUX ETAGES

* MODELS DE TRANSISTORS
.include MODN_MODP
* CIRCUIT
.SUBCKT OTA eplus emoins S evp1
MP1 2 emoins 3 3 MODP L=3.00u  W=5.08u
MP2 S eplus 3 3 MODP W=5.08u L=3.00u
MN3 2 2 vss vss MODN W=2.33u L=2.10u
M4N S 2 vss vss MODN W=2.33u L=2.10u
MP5 3 evp1 vdd vdd MODP W=10.59u L=3.00u
.ENDS
.SUBCKT INV z e s evp1
MN6 s e vss vss MODN W=19.39u L=3.00u
MP7 s evp1 vdd vdd MODP W=51.63u L=3.00u
Rc z s 8489.06
.ENDS
.SUBCKT AOP eplus emoins evp1 evp2 S
xota eplus emoins 1 evp1  OTA
xin  z  1  s   evp2  INV
Cc 1 z 1.47pf
.ENDS
xaop eplus emoins evp1 evp1 S AOP
Vevp1  evp1    0       dc      1.3
Cl      s      0       4.00pf
* OPTIONS
.OPTION eps=1.0E-6  vmin=-2.25  vmax=2.25
* SUPPLY
vdd vdd 0 2.25
vss vss 0 -2.25
VPLUS eplus 0 1.217645E-05
VE emoins 0 dc 0.0 ac 1
* STIMULI
.AC dec 1 0.5 5e+06
.PRINT AC vdb(s)
.PRINT AC vp(s)
.END
```

La ligne " *.AC dec 1 0.5 5e+06* " indique qu'on effectue une analyse *AC* entre les fréquences $0.5Hz$ et $5Mhz$ avec 1 fréquence par décade, ce qui fait un total de 8 fréquences. La commande *.PRINT* a été utilisée pour le gain $vdb(s)$ et la phase $vp(s)$ de la sortie s du circuit, ce qui implique que pour chaque fréquence, on compare le gain et la phase de la sortie s entre le bon circuit et les circuits fautifs.

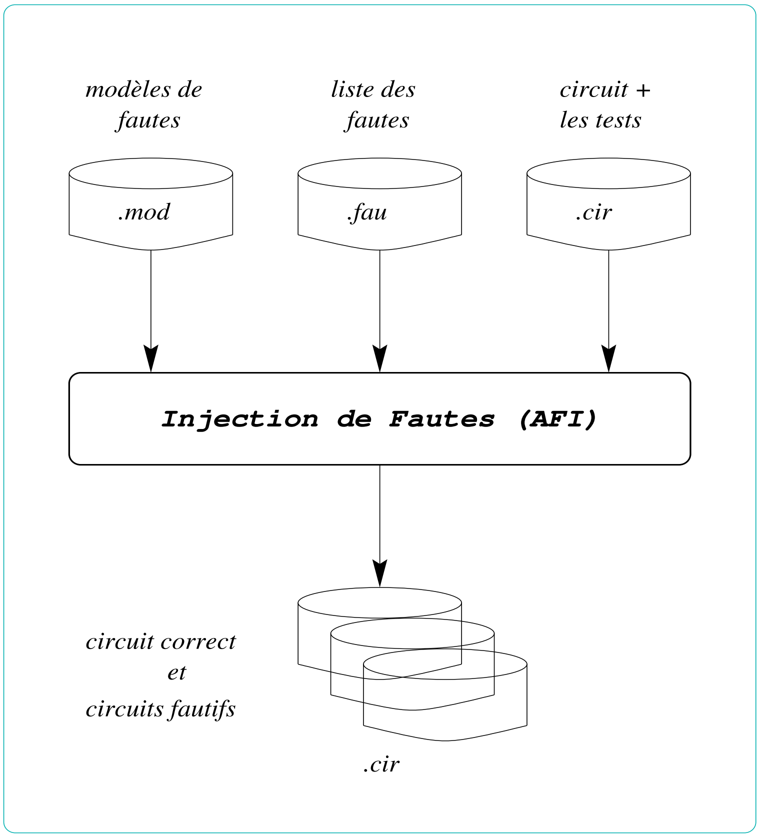

FIG. 6.2: *Architecture générale du module d'injection de fautes AFI.*

6.2 Module d'injection de fautes (AFI)

Le module d'injection de fautes permet de traduire les fautes qu'on souhaite détecter au format du simulateur électrique utilisé, il prend en entrée la description du circuit, la liste des fautes à détecter et le fichier des modèles de fautes et donne en sortie la description des circuits fautifs au format du simulateur. Les fautes à détecter peuvent être données par un expert ou extraites à partir de la vue physique du circuit " *layout* " [DS94]. L'avantage d'avoir un module d'injection de fautes est qu'il permet de séparer les fautes des modèles de fautes, en effet, on peut être amené à améliorer ou à changer les modèles des fautes sans avoir à modifier la liste de toutes les fautes à détecter.

6.2.1 Architecture Générale

L'architecture générale du module d'injection de fautes est donnée dans la figure 6.2. Le module *AFI* prend la description du circuit correct et la modifie en injectant toutes les fautes que l'on souhaite simuler. Il existe deux méthodes pour injecter les fautes :

1. Utilisation d'un seul fichier de simulation pour le circuit correct et tous les circuits fautif en utilisant la commande *.ALTER* qui permet de modifier le circuit et de relancer une nouvelle simulation. Cette approche évite la duplication des descriptions des circuits et donc diminue l'espace disque, mais en contre partie ne permet pas la distribution des simulations sur plusieurs machines.
2. Utilisation d'un fichier de simulation pour le circuit correct et chaque circuit fautif. Cette méthode duplique les descriptions des circuits et donc augmente l'espace disque, mais permet une distribution facile des simulations sur plusieurs machines.

Dans notre cas, et pour des raisons de facilité, nous avons adopté la première approche utilisant un seul fichier de simulation pour le circuit correct et les circuits fautifs avec la commande *.ALTER*.

6.2.2 Modèles de fautes

Le manque de modèles de fautes standards pour les circuits analogiques est une des difficultés du test analogique. Pour *ATSO*, nous avons défini certains modèles de fautes simples pour les fautes catastrophiques et les fautes paramétriques. Ces modèles de fautes sont facilement modifiables en fonction des types de circuits et des processus de fabrication. Le fichier des modèles de fautes à appliquer est défini au format du simulateur et par conséquent pour notre outil, le fichier de modèles de fautes est au format *ELDO*.

Chaque faute à injecter dans le circuit correct sera considérée comme l'ajout d'un sous circuit modélisant la faute en question, par conséquent, les modèles de fautes sont définis avec la commande *.SUBCKT* du simulateur *ELDO* qui permet de définir les sous circuits. Pour les fautes paramétriques, nous n'avons pas besoin de modèle de fautes puisque les fautes paramétriques consistent à seulement modifier les valeurs des paramètres (Résistance, Capacité, Longueur et Largeur des transistors) du circuit. Pour les

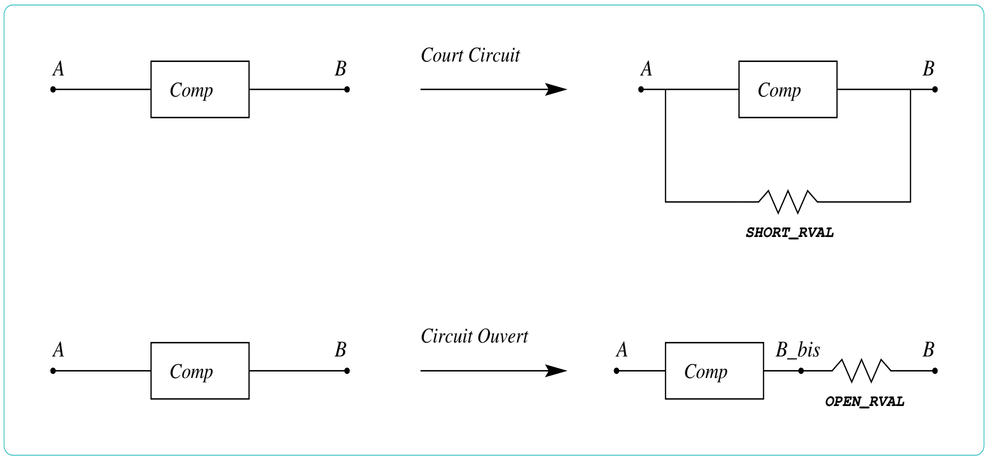

FIG. 6.3: *Injection d'un court circuit et d'un circuit ouvert.*

fautes catastrophiques, nous n'avons défini que deux modèles de fautes correspondant aux circuits ouverts et aux court circuits. Le fichier des modèles que nous avons utilisé est le suivant :

```
* FICHIER : modeles.mod
* DESCRIPTION DES MODELES DE FAUTES

* MODELE DE FAUTE POUR COURT CIRCUIT (SHORT)
.SUBCKT  SHTMOD  node1  node2
R_short  node1  node2  SHORT_RVAL
.ENDS

* MODELE DE FAUTE POUR CIRCUIT OUVERT (OPEN)
.SUBCKT OPNMOD node1 node2
R_open node1 node2 OPEN_RVAL
.ENDS
```

Les paramètres *SHORT_RVAL* et *OPEN_RVAL* correspondent respectivement à la résistance du court circuit et la résistance du circuit ouvert. Ces paramètres seront définis par l'utilisateur de l'outil en fonction des caractéristiques des défauts de fabrication données par le fondeur du circuit. Pour injecter une faute correspondant à un court-circuit sur un composant $COMP$, il suffit de rajouter la ligne correspondant à l'instantiation du sous circuit modélisant la faute en question, en utilisant les deux terminaux A et B du composant $COMP$ à court-circuiter (voir figure 6.3). Par contre, pour les circuits ouverts, il ne suffit pas de rajouter la ligne en instanciant le sous circuit modélisant le circuit ouvert, mais il faut aussi modifier les terminaux du composant $COMP$ contenant le circuit ouvert en ajoutant un nœud supplémentaire (voir figure 6.3).

6.2.3 Liste des fautes

La liste des fautes est contenue dans un fichier portant l'extension $< .fau >$ où chaque ligne correspond à une ou plusieurs fautes. La syntaxe des lignes du fichier $< .fau >$ contenant la liste des fautes est la suivante :

```
TYPE     COMP    {PARAM}    {DEV%}
```

où le champ *TYPE* définit le type de faute à appliquer, *COMP* le composant contenant la faute, *PARAM* le paramètre du composant à modifier dans le cas d'une faute paramétrique et *DEV* la valeur de la déviation en pourcentage pour une faute paramétrique. Le champ *TYPE* peut prendre les valeurs suivantes (voir section 3.2.4) :

– **SHT** : faute catastrophique correspondant à un court circuit sur le composant *COMP*.

– **OPN** : faute catastrophique correspondant à un circuit ouvert sur le composant *COMP*.

– **GSS** : faute catastrophique correspondant au court circuit entre la grille et la source du transistor nommé *COMP*.

– **GDS** : faute catastrophique correspondant au court circuit entre la grille et le drain du transistor nommé *COMP*.

– **DOP** : faute catastrophique correspondant au circuit ouvert sur le drain du transistor nommé *COMP*.

– **SOP** : faute catastrophique correspondant à un circuit ouvert sur la source du transistor nommé *COMP*.

– **PAR** : faute paramétrique correspondant à la déviation du paramètre *PARAM* de *DEV%* du composant *COMP*.

Le nom du composant *COMP* peur être remplacé par *R**, *C** et *M** pour appliquer la faute en question respectivement à toutes les résistances, toutes les capacités et tous les transistors MOS du circuit.

Un exemple de liste de fautes pour l'amplificateur décrit dans la section précédente est donné ci-dessous :

```
* FICHIER : aop.fau
* LISTE DES FAUTES POUR L'AOP
SHT  Rc
PAR  Cc  +50
PAR  MN6  W -30
DOP M*
```

Ce qui correspond à un court circuit sur la résistance *Rc*, une déviation de $+50\%$ sur la capacité *Cc*, une déviation de -30% sur la largeur *W* du transistor *MN6* et tous les circuits ouverts sur le drain de tous les transistors *MOS*.

6.3 Module de simulation de fautes (AFS)

Le but du simulateur de fautes est de déterminer les fautes qui sont détectables par chaque test en tenant compte des variations des paramètres du circuit dues aux fluctuations de l'environnement du processus de fabrication des circuits. Le simulateur de fautes est donc utilisé pour calculer la matrice des probabilités de détection de fautes $PDF_{i,j}$ de chaque faute F_j par chaque test T_i.

Pour réduire le temps de simulation, le module de simulation de fautes *AFS "Analog Fault Simulation"* implémenté utilise l'algorithme de simulation de fautes avec prise en compte des tolérances présenté dans la section 4.5.3. Cet algorithme permet de réduire le nombre de simulations nécessaires au calcul des probabilités de détection de faute PDF, en insérant deux tests d'arrêt permettant de arrêter les simulations et déduire la valeur de PDF sans avoir à effectuer toutes les simulations *Monte Carlo*.

6.3.1 Architecture Générale

L'architecture générale du module de simulation de fautes *AFS* est présentée dans la figure 4.1. Le module *AFS* se décompose en deux étapes :

1. Simulation *Monte Carlo* du circuit correct en effectuant $Nmax$ simulations électriques avec le simulateur *ELDO*. A chaque simulation, nous modifions les valeurs des paramètres du circuit suivant leurs tolérances. A la fin des $Nmax$ simulations, nous calculons la moyenne μ_{ref} et l'écart type σ_{ref} des sorties à mesurer du circuit correct.

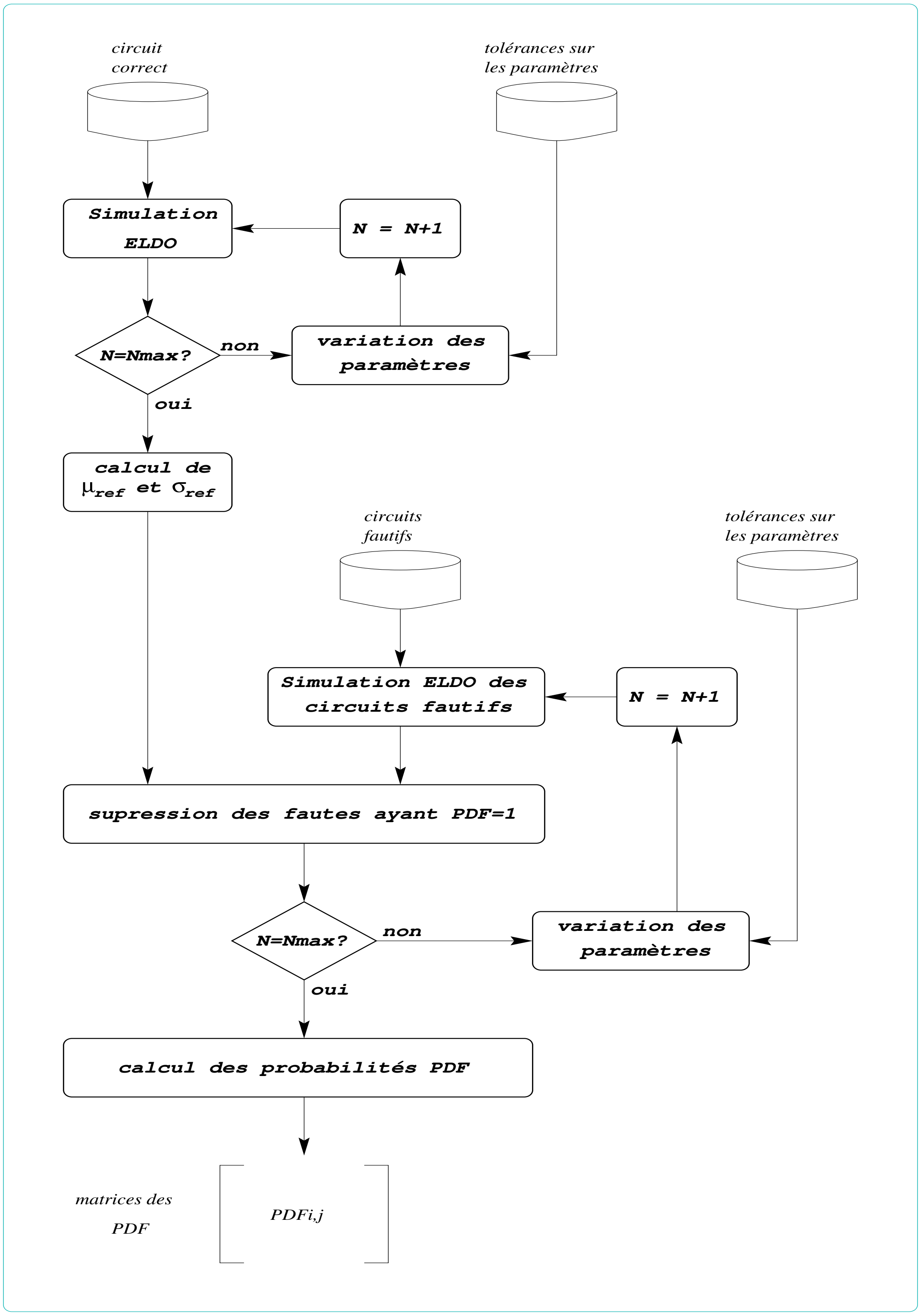

FIG. 6.4: *Architecture générale du module de simulation de fautes AFS.*

2. Simulation des circuits fautifs suivant l'algorithme présenté dans la section 4.5.3. De même que pour le circuit correct, à chaque simulation, on fait varier les valeurs des paramètres du circuit suivant leurs tolérances. Après chaque simulation, on effectue les deux tests correspondant aux conditions 4.24 et 4.26 et on élimine du fichier contenant les circuits fautifs toutes les fautes qui sont déjà détectables (qui ont $PDF = 1$). Ainsi, à chaque nouvelle simulation, le nombre de circuits fautifs à simuler diminue, ce qui diminue le temps de simulation de fautes. On répète ainsi les simulations des circuits fautifs jusqu'à ce que toutes les fautes soient détectables ou bien lorsque les $Nmax$ simulations ont été effectuées. A la fin des simulations, on calcule les probabilités de détection de fautes $PDF_{i,j}$ pour chaque test T_i et chaque faute F_j, la matrice des $PDF_{i,j}$ sera analysée par le module d'optimisation des tests *TSO*.

6.3.2 Variation des paramètres des composants

Les tolérances sur les paramètres traduisent les différentes déviations que peuvent avoir les valeurs des composants du circuit dues aux fluctuations de l'environnement du processus de fabrication des circuits. Les tolérances sont différentes d'une technologie de fabrication à une autre et dépendent aussi du fondeur. Pour l'outil *ATSO* nous avons défini un fichier des tolérances $< .tol >$ très simple qui donne les différents pourcentages de déviation des paramètres, pour des raisons de simplification, on a supposé que les variations des paramètres suivent une loi normale (loi de Gauss). Pour une version finale de l'outil *ATSO*, on pourra utiliser le fichier de technologie donné par le fondeur pour l'analyse *Monte Carlo*. La syntaxe des lignes du fichier $< .tol >$ contenant les tolérances des paramètres est la suivante :

```
PARAM   POUR%
```

où les champs *PARAM* et *POUR* définissent respectivement le paramètre et la déviation tolérée en pourcentage. Le champ *PARAM* peut prendre les valeurs suivantes :

– **R** : pour les tolérances absolues des résistances.

– **RR** : pour les tolérances relatives des résistances.

– **C** : pour les tolérances absolues des valeurs des capacités.

– **CC** : pour les tolérances relatives des valeurs des capacités.

- **L** : pour les tolérances absolues des longueurs des transistors.
- **LL** : pour les tolérances relatives des longueurs des transistors.
- **W** : pour les tolérances absolues des largeurs des transistors.
- **WW** : pour les tolérances relatives des largeurs des transistors.

Les tolérances absolues des paramètres correspondent aux déviations que peuvent avoir les paramètres par rapport à leur valeur nominale. Par contre, les tolérances relatives correspondent aux déviations sur le même paramètre que peuvent avoir des composants appariés entre eux. Par exemple, on sait que les déviations des valeurs des capacités entre elles est de l'ordre de 0.1%, alors que les déviations des valeurs absolues des capacités d'un circuit intégré sont de l'ordre de 5%. En général, les tolérances absolues des paramètres sont plus importantes que les tolérances relatives. Pour l'instant, notre outil *ATSO* ne prend en compte que les tolérances absolues des paramètres des composants.

6.4 Module d'optimisation des ensembles de tests (TSO)

Le module final du prototype *ATSO* est celui d'optimisation des tests de production avec prise en compte des tolérances appelé *TSO* " pour *Test Set Optimization* ". Le module *TSO* permet d'analyser les résultats donnés par le module de simulation de fautes pour déterminer les fautes détectables, le taux de couverture et l'ensemble optimisé des tests de production.

L'architecture générale du module d'optimisation des tests *TSO* est présentée dans la figure 6.5. Le module *TSO* prend en entrée la matrice des probabilités de détection de fautes PDF, les probabilités d'apparition de fautes et les coûts intrinsèques des tests, et donne en sortie le taux de couverture, la probabilité de détection pour chaque faute et l'ensemble optimisé des tests. Le module se décompose en deux étapes :

1. Elimination des tests redondants et de ceux qui ne détectent pas de fautes en utilisant l'algorithme de réduction des tests présenté dans la section 5.2.1.
2. Ordonnancement des tests restants en utilisant l'algorithme d'ordonnancement présenté dans le section 5.2.2.

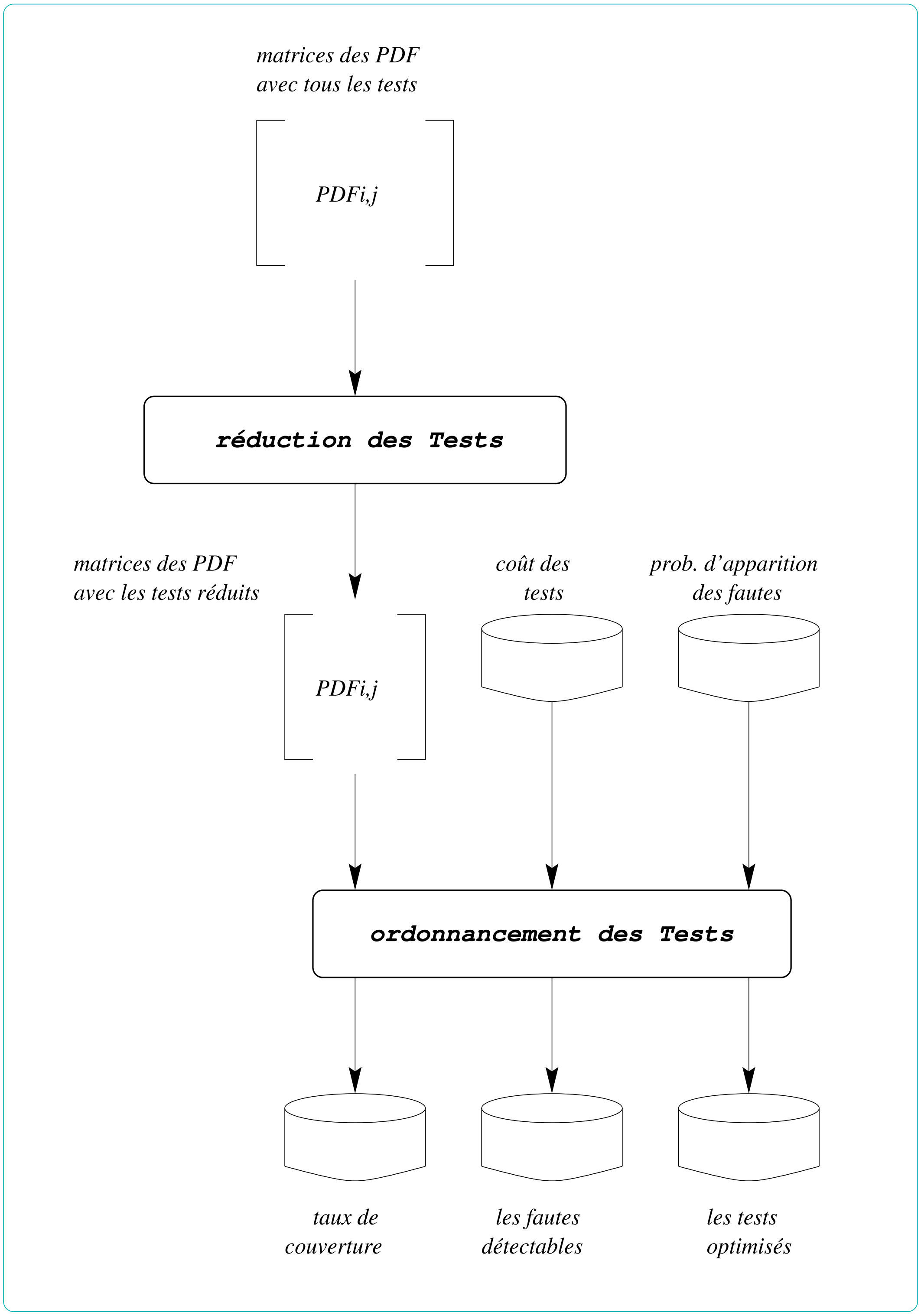

FIG. 6.5: *Architecture générale du module d'optimisation des tests de production TSO.*

6.5 Conclusion

Pour valider notre approche basée sur la notion de probabilité de détection de fautes, nous avons développé un prototype d'un outil d'optimisation des tests de production pour les circuits analogiques avec prise en compte des tolérances, appelé *ATSO*. L'outil est basé sur les différentes méthodes et algorithmes présentés dans la thèse et se décompose en trois modules principaux qui sont :

1. Module d'injection de fautes *AFI*.

2. Module de simulation de fautes avec prise en compte des tolérances et de calcul des probabilités de détection de fautes *AFS*.

3. Module d'analyse des probabilités de détection de fautes pour la réduction et l'ordonnancement des tests de production *TSO*.

Le prototype *ATSO* utilise des fichiers de modèles de fautes et des tolérances sur les paramètres très simples, notre but principal n'est pas de faire un outil de test des circuits analogiques, mais de montrer la faisabilité et l'efficacité de notre approche pour le test des circuits analogiques assujettis à des déviations des paramètres, dues aux variations des paramètres du processus de fabrication.

L'outil *ATSO* a été utilisé avec succès pour la validation de nos algorithmes de simulations de fautes, de réduction et de l'ordonnancement des tests de production pour les trois circuits de validation : l'amplificateur opérationnel, l'intégrateur en mode courant et le filtre passe bas à capacités commutées.

Conclusion

Dans ce travail, nous avons présenté une nouvelle approche pour la simulation de fautes et l'optimisation des tests de production pour les circuits analogiques avec prise en compte des tolérances dues aux variations des paramètres du processus de fabrication.

Avec les progrès accomplis dans le domaine de l'intégration des circuits intégrés et l'évolution croissante du marché des télécommunications, les circuits analogiques et mixtes analogique-numérique sont de plus en plus complexes et difficiles à tester. Pour les circuits numériques, il existe déjà plusieurs outils automatiques universitaires et industriels de test ou d'aide à la conception en vue du test. Malheureusement, il n'existe pas encore d'outils automatiques de test ou d'aide à la conception en vue du test pour les circuits analogiques. La nature complexe des signaux analogiques, la très grande sensibilité au bruit, les tolérances sur les paramètres des circuits et le manque de modèles de fautes efficaces sont les principales raisons qui freinent le développement d'outils de test analogiques.

L'outil de simulation de fautes est indispensable au développement de toute méthodologie de test. En effet, il permet de déterminer si les tests choisis ou si les techniques de conception en vue du test adoptées sont efficaces. Pour les circuits analogiques, une des principales difficultés des simulateurs de fautes est la variation des paramètres des composants due aux fluctuations du processus de fabrication du circuit. Nous avons montré que les déviations des paramètres des circuits, si elles ne sont pas prises en compte, peuvent engendrer des résultats de simulation de fautes complètement faux. Pour résoudre le problème des tolérances, nous avons développé une nouvelle approche de simulation de fautes avec prise en compte des tolérances basée sur les probabilités de détection de fautes *PDF*.

Pour les circuits numériques, une faute peut être soit détectable soit non détectable par un vecteur de test. Par contre, pour les circuits analogiques et à cause des tolérances sur les paramètres, une faute analogique peut être soit complètement détectable, soit complètement non détectable soit partiellement détectable (ou partiellement non détectable) par un vecteur de test. Pour résoudre ce problème de détectabilité d'une faute analogique, nous avons défini une probabilité de détection de faute *PDF* permettant de quantifier par un nombre compris entre 0 et 1 le degré détectabilité d'une faute analogique par un vecteur de test. Ces probabilités de détection de fautes sont calculées par le simulateur de fautes et seront utilisées pour le calcul des taux de couverture et pour l'optimisation des ensembles de tests dans le but de réduire le coût du test.

Pour chaque circuit fautif, le calcul des probabilités *PDF* nécessite plusieurs simulations électriques, ce qui rend le simulateur de fautes coûteux en temps de simulation. Pour réduire ce temps de simulation de fautes, nous avons proposé une amélioration de la méthode *Monte Carlo*, basée sur la déduction de la valeur de *PDF*. La nouvelle approche utilisée effectue plusieurs simulations, mais à chaque simulation, on effectue deux tests d'arrêt permettant dans le cas où l'un des deux tests est vérifié de déduire la valeur de *PDF* et de stopper les simulations électriques.

Une des applications importantes du simulateur de fautes pour les circuits numériques est la génération automatique des vecteurs de test. Pour les circuits analogiques, les vecteurs de test dépendent de la nature du circuit à tester. Il est donc impossible de développer un générateur automatique de vecteurs de test pour tous les types de circuits analogiques. C'est pourquoi, nous nous sommes intéressé au problème de l'optimisation automatique des ensembles de tests pré-existants. Pour résoudre le problème des tolérances dans l'optimisation des tests, nous avons utilisé les probabilités de détection de fautes *PDF* pour développer un algorithme d'optimisation des ensembles de test avec prise en compte des tolérances. L'algorithme proposé se décompose en deux étapes : réduction de l'ensemble des tests et ordonnancement des tests choisis.

Cette thèse présente, à travers l'outil *ATSO* "*Automatic Test Stimuli Optimization*" un prototype pour un outil de simulation de fautes et d'optimisation des tests de production pour les circuits analogiques avec prise en compte des tolérances. Le prototype *ATSO* est basé sur le calcul et l'utilisation des probabilités de détection de fautes *PDF* et est composé du module d'injection de faute *AFI*, du simulateur de fautes *AFS* et du module d'optimisation des ensembles de tests de production *TSO*. L'utilisation de l'outil *ATSO* nous a permis de valider les différentes approches et algorithmes développés tout au long de cette thèse sur trois circuits analogiques différents : un amplificateur opérationnel, un intégrateur en mode courant et un filtre passe-bas à capacités commutées. Les résultats obtenus en terme d'efficacité (taux de couverture) et en terme de temps de simulation de fautes sont très satisfaisants.

Quelques améliorations peuvent être apportées à notre approche pour améliorer encore plus la précision et le temps de simulation de fautes. Pour le calcul de la probabilité de détection de fautes *PDF*, nous avons supposé que la distribution des valeurs simulées du circuit correct suit une loi normale, ce qui n'est pas toujours vrai. Pour améliorer la précision de *PDF*, nous pourrons extraire pour chaque test la distribution des valeurs simulées du circuit correct et l'utiliser pour le calcul des valeurs de *PDF* correspondant. Pour le temps de simulation de fautes, on sait que les simulateurs électriques de type *SPICE* et *ELDO* sont très coûteux en temps de simulation. Pour réduire ce temps de simulation de fautes, il est donc plus intéressant de développer un simulateur électrique propre à la simulation de fautes présentant les deux caractéristiques suivantes :

1. Dans le cas où la valeur simulée du circuit fautif est très loin de la valeur simulée du bon circuit, il n'y a pas d'intérêt à connaître la valeur simulée du circuit fautif avec précision. Le simulateur doit donc pouvoir avoir deux modes : un mode rapide et un mode plus précis et lent. Dans un premier temps, on effectue une simulation rapide pour avoir l'ordre de grandeur de la valeur simulée du circuit fautif et si cette valeur est proche de celle du circuit correct, alors on effectue une deuxième simulation précise. On pourra ainsi réduire considérablement le temps de simulation des circuits fautifs.

2. Pour les simulations *DC* et transitoires, ce sont les itérations *Newton-Raphson* qui consomment le plus de temps. Sachant que le nombre d'itérations d'une simulation dépend de la valeur de départ, on peut considérablement réduire le nombre d'itérations *Newton-Raphson* en choisissant les résultats de simulation du circuit correct comme valeurs de départ des simulations des circuits fautifs. Et pour réduire encore plus le temps de simulation, pour le test *DC*, on peut intégrer les deux tests d'arrêt du simulateur de fautes à l'intérieur de la boucle de *Newton-Raphson*.

Bibliographie

[ABK95] Bechir Ayari, Naim BenHamida, and Bozena Kaminska. Automatic test vector generation for mixed-signal circuits. In *Proc. European Design and Test Conference*, pages 458–463, March 1995.

[ACK96] Abdessatar Abderrahman, Eduard Cerny, and Bozena Kaminska. Optimization-based multifrequency test generation for analog circuits. *Journal of Electronic Testing, Kluwer Academic Publishers*, 9:59–73, Aug./Oct. 1996.

[ACK99] Abdessatar Abderrahman, Eduard Cerny, and Bozena Kaminska. Worst case tolerance analysis and clp-based multifrequency test generation for analog circuits. *IEEE Trans. Computer-Aided Design*, 18(3):332–345, March 1999.

[ADP+90] Edouardo L. Acuna, James P. Dervenis, Andrew J. Pagones, Fred L. Yang, and Resve A. Saleh. Simulation technique for mixed analog/digital circuits. *IEEE Journal of Solid-State Circuits*, 25(2):353–363, April 1990.

[AK97] Karim Arabi and Bozena Kaminska. Oscillatio built-in self test (OBIST) scheme for functional ans structural testing of analog and mixed-signal integrated circuits. In *Proc. International Test Conference*, 1997.

[AZ96] Bert Atzema and Taco Zwemstra. Exploit Analog IFA to Improve Specification Based Tests. In *Proc. European Design and Test Conference*, 542–546 1996.

[Aza96] Florence Azais. *Conception en Vue du Test pour Circuits Integrés Analogiques et Mixtes*. Thèse de Doctorat de l'Université Montpellier II, France, 1996.

[BA82] P. Banerjee and J. A. Abraham. Fault characterization of VLSI MOS circuits. In *IEEE Int. Conf. Circuits and Computers*, pages 564–568, 1982.

[BBL$^+$96] F. Bailleu, Y. Blanchard, P. Loumeau, H. Petir, and J. Porte. *Capacités commutées & Applications*. DUNOD, Paris, 1996.

[BHSMK97] Naim Ben-Hamida, Khaled Saab, David Marche, and Bozena Kaminska. A perturbation based fault modeling and simulation for mixed-signal circuits. In *Proc. IEEE Asian Test Symposium*, pages 182–187, 1997.

[CA95] Pascal Caunegre and Claude Abraham. Achieving Simulation-Based Test Program Verification and Fault Simulation Capabilities for Mixed-Signal Systems. In *Proc. European Design and Test Conference*, March 1995.

[CA96] Pascal Caunegre and Claude Abraham. Fault simulation for mixed-signal systems. *Journal of Electronic Testing, Kluwer Academic Publishers*, 8(2), April 1996.

[CC99] S. Cherubal and A. Chatterjee. A Parametric Fault Diagnosis for Analog Systems Using Functional Mapping. In *Proc. Design And Test in Europe*, 1999.

[Deb94] Philippe Debaud. *TAUMMI: Un Systeme de Génération de vecteurs, de Simulation de Fautes et de Compilation en vue du Test Utilisant une Méthode Localement Exhaustive*. Thèse de Doctorat de l'Université de Paris VI, 1994.

[DLSVV94] Alper Demir, Edward Liu, Alberto L. Sangiovanni-Vincentelli, and Iasson Vassiliou. Behaviour simulation technique for phase/delay-locked systems. In *Proc. IEEE Custom Integrated Circuits Conference*, pages 453–456, May 1994.

[DR79] P. Duhamel and J. C. Rault. Automatic Test Generation Techniques for Analog Circuits and Systems: A Review. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 26(7):411–440, July 1979.

[DR99] Benoit Dufour and Gordon W. Roberts. On-Chip Analog Signl Generation for Mixed-Signal Built-In Self-Test. *IEEE Journal of Solid-State Circuits*, 34(3):318–330, March 1999.

[DS94] Giri Devarayanadurg and Mani Soma. Analytical Fault Modeling and Static Test Generation for Analog ICs. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 44–47, 1994.

[DSGH99] Giri Devarayanadurg, Mani Soma, Prashant Goteti, and D. Huynh. Test set selevtion for structural faults in analog ic's. *IEEE Trans. Computer-Aided Design*, 18(7):1026–1039, July 1999.

[FA97] A. Frisch and T. Almy. HABIST: Histogram-based analog built in self test. In *Proc. International Test Conference*, 1997.

[Flo98] Olivier Florent. *Une Méthode de Test des Circuits Intégrés, Basée sur un Découpage Structurel Peu Recouvrant*. Thèse de Doctorat de l'Université Paris VI, France, 1998.

[FMPS99] Giulio Fedi, Stefano Manetti, Maria Cristina Piccirilli, and Janusz Starzyk. Determination of an optimum set of testable components in the fault diagnosis of analog linear circuits. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 46(7):779–787, July 1999.

[GWS99] Georges Gielen, Zhihua Wang, and Willy Sansen. Fault Detection and Imput Stimulus Determination for the Testing of Analog Integrated Circuits Based on Power-Supply Current Monitoring. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 495–498, 1999.

[HG91] S. D. Huss and R. S. Gyurcsik. Optimal ordering of analog integrated circuit tests to minimize test time. In *Proc. Design Automation Conference*, pages 494–499, 1991.

[HK93a] Nain Ben Hamida and Bozena Kaminska. Analog Circuit Testing Based on Sensitivity Computation and New Cirucit Modeling. In *Proc. International Test Conference*, pages 652–661, 1993.

[HK93b] Nain Ben Hamida and Bozena Kaminska. Multiple Fault Analog Circuit Testing by Sensitivity Analysis. *Analog Integrated Circuits and Signal Processing, Kluwer Academic Publishers*, 4:231–243, 1993.

[HKSZ99] Sam D. Huynh, Seongwon Kim, Mani Soma, and Jinyan Zhang. Automatic Analog Test Signl Generation using Multifrequency Analysis. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 46(5):565–576, May 1999.

[HR95] Xavier Haurie and W. Roberts. Arbitrary-Precision Signal Generation for Bandlimited Mixed-Signal Testing. In *Proc. International Test Conference*, pages 78–85, 1995.

[HRBB94] R. J. Harvey, A. M. D. Richardson, E. M. J. G. Bruls, and K. Baker. Analogue Fault Simulation Based on Layout Dependent Fault Models. In *Proc. International Test Conference*, 641–649 1994.

[HRK95] R. J. Harvey, A. M. D. Richardson, and H. G. Kerkhoff. Defect Oriented Test Developement Based on Layout Inductive Fault Analysis. In *Proc. IEEE International Mixed-Signal Testing Workshop*, pages 2–9, 1995.

[HSM+96] Nain Ben Hamida, Khaled Saab, David Marche, Bozena Kaminska, and Guy Quesnel. LIMSoft: Automated Tool for Design and Test Integration of Analog Circuits. In *Proc. International Test Conference*, 1996.

[JG80] M. Vergniault J. Galiay, Y. Crouzet. Physical versus logical fault models mos-lsi circuits impact on their testability. *IEEE Trans Computers*, C-29:527–531, 1980.

[LCSV93] Edward W. Y. Liu, Henry C. Chang, and Alberto L. Sangiovanni-Vincentelli. Analog System Verification ine the Presence of Parasitics using Behavioural Simulation. In *Proc. Design Automation Conference*, pages 159–163, 1993.

[LGA99] Walter M. Lindermeir, Helmut E. Graeb, and Kurt J. Antreich. Analog testing by characteristic observation inference. *IEEE Trans. Computer-Aided Design*, 18(9):1353–1368, September 1999.

[LRM+98] M. Lubaszewski, M. Renovell, S. Mir, F. Azaïs, and Y. Bertrand. A multi-mode stimuli generator for analogue and mixed-signal built-in self-test. In *Proc. IEEE International Mixed-Signal Testing Workshop*, pages 100–106, 1998.

[LS85] S. Liu and K. Singhal. A statistical model for MOSFETS. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 78–80, 1985.

[Mal87] Wojciech Maly. Realistic Fault Modeling for VLSI Testing. In *Proc. Design Automation Conference*, pages 173–180, 1987.

[Max89] S. Max. Fast, accurate, and complete adc testing. In *Proc. International Test Conference*, 1989.

[Mil98] Linda Milor. A Tutorial Iintroduction to Research on Analog and Mixed-Signal Circuit Testing. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 45(10):1389–1407, October 1998.

[MLC96a] S. Mir, M. Lubaszewski, and B. Courtois. Automatic Test Generation for Maximal Diagnosis of Linear Analog Circuits. In *Proc. European Design and Test Conference*, pages 254–258, 1996.

[MLC96b] S. Mir, M. Lubaszewski, and B. Courtois. Fault-Based ATPG for Linear Analog Circuit with Minimal Size Multifrequency Test Sets. *Journal of Electronic Testing, Kluwer Academic Publishers*, 9:43–57, 1996.

[Moh98] Firas Mohamed. *Approche à base de logique floue pour le test et le diagnostic des circuits analogiques*. Thèse de Doctorat de l'Université de Paris VI, 1998.

[MRO+97] S. Mir, A. Rueda, T. Olbrich, E. Peralias, and J.L. Huertas. SWITTEST: Automatic Switch-level Fault Simulation and Test Evaluation of Switched-Capacitor Systems. In *Proc. Design Automation Conference*, pages 281–286, June 1997.

[MRVH98] S. Mir, A. Rueda, D. Vazquez, and J.L. Huertas. Switch-level Fault Coverage and Analysis for Switched-Capacitor Systems. In *Proc. Design And Test in Europe*, pages 810–814, 1998.

[MSD86] W. Maly, A.J. Strojwas, and S.W. Director. VLSI Yield Prediction and Estimation: A unified Framework. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 5:114–130, 1986.

[MSV94] Linda Milor and A. L. Sangiovanni-Vincentelli. Minimizing Production Test Time to Detect Faults in Analog Circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 13(6), 1994.

[MV89] Linda Milor and V. Visvanathan. Detection of Catastrophic Faults in Analog Integrated Circuits. *IEEE Trans. Computer-Aided Design*, 8(2):114–130, February 1989.

[NCA93a] Naveena Nagi, Abhijit Chatterjee, and Jacob A. Abraham. DRAFTS: Discretized Analog Circuit Fault Simulator. In *Proc. Design Automation Conference*, pages 509–514, 1993.

[NCA93b] Naveena Nagi, Abhijit Chatterjee, and Jacob A. Abraham. Fault Simulation of Linear Analog Circuits. *Analog Integrated Circuits and Signal Processing, Kluwer Academic Publishers*, 4:245–260, 1993.

[NCA93c] Naveena Nagi, Abhijit Chatterjee, and Jacob A. Abraham. MIXER: Mixed-Signal Fault Simulator. In *Proc. IEEE International Conference on Computer Design*, pages 568–571, 1993.

[NCA94] Naveena Nagi, Abhijit Chatterjee, and Jacob A. Abraham. A signature analyser for analog and mixed-signal circuits. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 284–287, 1994.

[NCBA93] Naveena Nagi, Abhijit Chatterjee, Ashok Balivada, and Jacob A. Abraham. Fault-Based Automatic Test Generator for Linear Analog Circuits. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 88–91, 1993.

[OGA+97] T. Olbrich, I. A. Grout, Y. Eben Aimine, A.M. Richardson, and J. Contensou. A new quality estimation methodology for mixed-signal and analogue ic's. In *Proc. European Design and Test Conference*, pages 573–580, 1997.

[Ohl91] M. J. Ohletz. Hybrid built-in self-test (HBIST) for mixed analogue/digital integrated circuits. In *Proc. European Design and Test Conference*, pages 307–316, 1991.

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

[PC95] Chwn-Yang Pan and Kwang-Ting Cheng. Pseudo-Random Testing and Signature Analysis for Mixed-Signal Circuits. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 102–107, 1995.

[PC99] Chen-Yang Pan and Kwang-Ting Cheng. Test generation for linear time-invariant analog circuits. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 46(5):554–564, May 1999.

[PCG94] Chwn-Yang Pan, Kwang-Ting Cheng, and Sandeep Gupta. A Comprehensive fault Macromodel For Opamps. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 344–348, 1994.

[PR82] A. Pahwa and R. A. Rohrer. Band-faults: Efficient approximations to fault bands for the simulation before fault diagnosis of linear circuits. *IEEE Trans. on Circuits and Systems–II: Analog and Digital Signal Processing*, 29(2):81–88, Feb 1982.

[PRG+98] J. A. Prieto, A. Rueda, I. A. Grout, E. Peralías, J. L. Huertas, and A.M. Richardson. An approach to realistic fault prediction and layout design for testability in analog circuits. In *Proc. Design And Test in Europe*, pages 905–909, 1998.

[PZ97] Paulo Prinetto and Yervant Zorian. Tutorial b: Systems-on-chip: From design validation to system test. In *Proc. European Design and Test Conference*, March 1997.

[PZCW98] A.J. Perkins, M. Zwolinski, C. D. Chalk, and B. R. Wilkins. Fault Modeling and Simulation Using VHDL-AMS. *Analog Integrated Circuits and Signal Processing, Kluwer Academic Publishers*, 16:53–67, 1998.

[RAB96] M. Renovell, F. Azaïs, and Y. Bertrand. The multi-configuration: A dft technique for analog circuits. In *Proc. IEEE VLSI Test Symposium*, pages 54–59, 1996.

[RAB97] M. Renovell, F. Azaïs, and Y. Bertrand. On-chip analog output response compaction. In *Proc. European Design and Test Conference*, pages 568–572, 1997.

[RAB98] M. Renovell, F. Azaïs, and Y. Bertrand. Optimized implementation of the multi-configuration dft technique for analog circuits. In *Proc. Design And Test in Europe*, pages 815–821, 1998.

[RABB98] M. Renovell, F. Azaïs, J-C. Bodin, and Y. Bertrand. Desig-for-testability for switched-current circuits. In *Proc. IEEE VLSI Test Symposium*, pages 370–375, 1998.

[RCA85] M. Renovell, G. Campon, and D. Auvergne. FSPICE : a tool for fault modeling in MOS circuits. *INTEGRATION, VLSI J.*, 3:245–255, 1985.

[Rob97] Gordon W. Roberts. Improving the testability of mixed-signal integrated circuits. In *Proc. IEEE Custom Integrated Circuits Conference*, 1997.

[RS89] G. Russell and I. L. Sayers. *Advanced Simulation and Test Methodologie for VLSI Design*. Van Nostrand reinhold International, 1989.

[RVMN99] I. Rayane, J. Velasco-Medina, and M. Nicolaidis. A digital BIST for operational amplifiers embedded in mixed-signal circuits. In *Proc. IEEE VLSI Test Symposium*, 1999.

[SA95] Manoj Sachdev and Bert Atzema. Industrial Relevance of Analog IFA : A Fact or Fiction. In *Proc. International Test Conference*, pages 61–70, 1995.

[SFT90] K. Suyama, S.C. Fang, and Y.P. Tsividis. Simulation of mixed switched-capacitor/digital networks with signal driven switches. *IEEE Journal of Solid-State Circuits*, 25(6):1403–1413, December 1990.

[SG97] C.-J. Richard Shi and Nihal J. Godambe. Behavioral Fault Modeling and Simulation of Phase-Locked Loops Using VHDL-A Like Language. In *Proc. IEEE International ASIC Conference and Exhibit*, 1997.

[SK93] M. Slamani and B. Kaminska. T-BIST: A built-in self-test for analog circuits based on parameter translation. In *Proc. IEEE Asian Test Symposium*, pages 172–177, 1993.

[SO93] C. Sebeke and M.J. Ohletz. Analogue Fault Simulation. In *Poster on CAVE Workshop*, May 1993.

[Som90] Mani Soma. A design-for-test methodology for active analog filters. In *Proc. International Test Conference*, pages 183–192, 1990.

[Som91] Mani Soma. An experimental approach to analog fault models. In *Proc. IEEE Custom Integrated Circuits Conference*, 1991.

[Som93] Mani Soma. Fault Coverage of DC Parametric Tests for Embedded Analog Amplifiers. In *Proc. International Test Conference*, pages 566–573, 1993.

[SS91] NJaidip Singh and Resve Saleh. iMACSIM: A Program for multi-level Analog Circuit Simulation. In *Proc. IEEE International Conference on Computer-Aided-Design*, pages 16–19, 1991.

[SSI97] Zwolinski M. Spinks S.J., Chalk C.D. and Bell I.M. Generation and Verification of Tests for Analogue Circuits Subject to Process Parameter Deviations. In *IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems, Paris, France*, October 1997.

[ST98] C.-J. Richard Shi and Michael W. Tian. Automatic Test Generator of Linear Analog Circuits Under Parameter Variations. In *Proc. Asian and South Pacific Design Automation Conference*, pages 501–505, 1998.

[STO95] C. Sebeke, J. P. Teixeira, and M.J Ohletz. Automatic Fault Extraction and Simulation of Layout Realistic Faults for Iintegrated Analogue Circuits. In *Proc. European Design and Test Conference*, pages 464–468, 1995.

[Sun95] S. Sunter. The p1149.4 mixed-signal test bus: Costs and benefits. In *Proc. International Test Conference*, pages 444–450, 1995.

[Syr87] Marek Syrzycki. Modeling of Spot Defects in MOS Transistors. In *Proc. International Test Conference*, pages 148–157, 1987.

[THB93] C. Toumazou, J. B. Hughes, and N. C. Battersby. *SWITCHED-CURRENTS an analogue technique for digital technology*. Peter Peregrinus Ltd. on behalf of the Institution of Electrical Engineers, London, United Kingdom, 1993.

[TS97] Michael W. Tian and C.-J. Richard Shi. Rapid Frequency-Domain Analog Fault Simulation Under Parameter tolerances. In *Proc. Design Automation Conference*, pages 275–280, 1997.

[TS98a] Michael W. Tian and C.-J. Richard Shi. Efficient DC Fault Simulation of Nonlinear Analog Circuits. In *Proc. Design And Test in Europe*, pages 899–904, 1998.

[TS98b] Michael W. Tian and C.-J. Richard Shi. Worst-case analysis of linear analog circuits using sensitivity bands. In *Proc. IEEE International Symposium on Circuits and Systems*, volume 6, pages 110–113, 1998.

[T.W86] T.W.Williams. *Advanced in CAD for VLSI, VLSI Testing*. North Holland. P.O. BOX 1991, 1000BZ AMSTERDAM, 1986.

[VCH+97] R. Voorakaranam, S. Chakrabarti, J. Hou, S; Cherubal, and A; Chatterjee. Hierarchical specification-driven analog fault modeling for efficient fault simulation and diagnosis. In *Proc. International Test Conference*, pages 903–912, 1997.

[VdPG97] Wim Verhaegen, Geert Van der Plas, and Georges Gielen. Automated test pattern generation for analog integrated circuits. In *Proc. IEEE VLSI Test Symposium*, pages 296–301, 1997.

[Vin98] Bapiraju Vinnakota. *Analog and Mixed-Signal Test*. Prentice HALL PTR. Upper Saddle River, NJ 07458, 1998.

[WD87] Hank Walker and Stephen W. Director. VLASIC: A Catastrophic Fault Yield Simulation for Integrated Circuit. *IEEE Trans. Computer-Aided Design*, 5(4):541–556, October 1987.

[YZ99] Z.R. Yang and M. Zwolinski. Fast, Robust DC and Transient Fault Simulation for Nonlinear Analogue Circuits. In *Proc. Design And Test in Europe*, pages 244–248, 1999.

[ZBC97] M. Zwolinski, A.D. Brown, and C.D Chalk. Concurrent Analog Fault Simulation. In *Proc. IEEE International Mixed-Signal Testing Workshop*, 1997.

[ZCW96] M. Zwolinski, C. Chalk, and B.R. Wilkins. Analog Fault Modeling and Simulation for Supply Current Monitoring. In *Proc. European Design and Test Conference*, pages 547–552, 1996.

Liste des publications

[1] M. Firas, A. Khouas, and A. Derieux. L'optimisation des vecteurs de test analogique à base d'une approche floue. In *encontre Francophone sur la Logique Floue et ses Applications*, pages 107–112, 1997.

[2] A. Khouas and A. Derieux. Optimisation des tests de production pour les circuits analogiques avec prise en compte des tolérances. In *Colloque CAO de Circuits Intégrés et Systèmes*, Mai 1999.

[3] A. Khouas and A. Derieux. Methodology for Fast and Accurate Analog production Test Optimisation. In *Proc. IEEE International Mixed-Signal Testing Workshop*, pages 215–219, June 1999.

[4] A. Khouas and A. Derieux. Optimization of production tests for analog circuits under parameter variations. In *Mixed Design of Integrated Circuits and Systems*, June 1999.

[5] A. Khouas and A. Derieux. Speed-up of high accurate analog test stimulus optimization. In *Proc. International Test Conference*, pages 230–236, September 1999.

[6] A. Khouas, M. Dessouky, and A. Derieux. Optimized statistical analog fault simulation. In *Proc. IEEE Asian Test Symposium*, pages 227–232, October 1999.

[7] A. Khouas and A. Derieux. Fault Simulation for Analog Circuits under Parameter Variations. In *Journal of Electronic Testing, Kluwer Academic Publishers*, volume 16, pages 269–278, June 2000.

[8] A. Khouas and A. Derieux. Analog fault detection based on statistical analysis. In *Proc. IEEE International Mixed-Signal Testing Workshop*, June 2000.