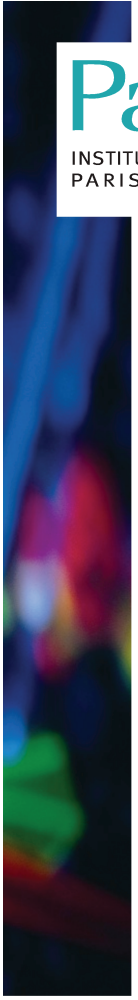


ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2015-ENST-0015

EDITE - ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

TELECOM ParisTech

Spécialité « Électronique et communications »

présentée et soutenue publiquement par

Sébastien SARRAZIN

le 31 Mars 2015

Fiabilisation et test des processeurs dans un contexte embarqué

Directeur de thèse : **Lirida NAVINER**
Co-Directeur de thèse : **Valentin GHERMAN**

Jury

M. Jacques-Oliver KLEIN, Professeur, Université Paris-Sud/IEF	Président du jury
M. Patrick GIRARD, Directeur de Recherches, CNRS/LIRMM	Rapporteur
M. Raoul VELAZCO, Directeur de Recherches, CNRS/TIMA	Rapporteur
M. Laurent BOUDOU, Ingénieur, ST Microelectronics	Examinateur
M. Samuel EVAIN, Chercheur, LFIC, CEA-LIST	Examinateur
Mme. Lirida NAVINER, Professeur, COMELEC, Telecom ParisTech	Directrice de thèse
M. Valentin GHERMAN, Chercheur, LFIC, CEA-LIST	Co-Directeur de thèse

TELECOM ParisTech
école de l'Institut Mines-Télécom - membre de ParisTech
46 rue Barrault 75013 Paris - (+33) 1 45 81 77 77 - www.telecom-paristech.fr

T H È S E

"La pensée vole et les mots vont à pied.
Voilà tout le drame de l'écrivain."
Julien GREEN

Remerciements

Au terme de cette thèse, c'est avec attention que je tiens à remercier tous ceux qui, de près ou de loin, ont contribué à la réalisation de ce projet et m'ont permis de le mener à bien.

Je tiens tout d'abord à adresser mes remerciements au Professeur Lirida Naviner pour avoir accepté de diriger mon travail de thèse et m'avoir permis de le réaliser dans les meilleures conditions. Je souhaite particulièrement la remercier quant à la liberté d'action qu'elle m'a offerte à chacune des étapes de cette aventure et j'espère avoir été à la hauteur de ses espérances.

Je tiens ensuite à remercier Monsieur Valentin Gherman pour son encadrement. Il a su m'accompagner, me guider et me former tout au long de ces trois années et je lui en suis humblement reconnaissant.

Je tiens également à remercier les membres de mon jury qui ont accepté d'évaluer mon travail. Je remercie particulièrement Messieurs Patrick Girard Directeur de Recherches, CNRS/LIRMM et Raoul Velazco Directeur de Recherches, CNRS/TIMA qui ont accepté de relire ce manuscrit. Je remercie aussi tout particulièrement Monsieur Jacques-Olivier Klein de présider mon jury. Je remercie enfin Messieurs Samuel Evain, chercheur au CEA-LIST LFIC et Laurent Boudou, Ingénieur recherche à ST-Microélectronique d'avoir bien voulu faire partie de mon jury de soutenance en tant qu'examinateurs.

Mes travaux de thèse ayant pris place au sein du laboratoire LFSE du CEA-LIST, nouvellement LFIC, je tiens évidement à en remercier tous les membres pour leur accueil toujours chaleureux et la bonne ambiance de travail dans laquelle ils m'ont permis d'évoluer. Mes remerciements vont particulièrement à Madame Wafa Benhassen et a Messieurs Raphael Karabassis et Jaume Benoit pour leur aide, leurs encouragements et leurs conseils toujours avisés.

Je tiens également à remercier Messieurs Fabien Lucet, Jean-Paul Mazelier et Barthelemy Steck de m'avoir convaincu de tenter l'aventure de la thèse.

Enfin, mes plus profonds remerciements vont évidement à ma famille, mais aussi à Yoann, PiBa, Marie, Alex, Fanny, Bernard, Yann et tout particulièrement à Angélique, pour m'avoir soutenu, encouragé et supporté tout aussi bien dans les meilleurs moments de la thèse que dans les pires. Cette aventure n'aurait pas été possible sans eux.

Fiabilisation et test des processeurs dans un contexte embarqué

Avec l'évolution technologique des semi-conducteurs, l'impact des variations dynamiques, notamment dues aux modifications du milieu de mission, représente un défi considérable vis-à-vis de la performance, de la fiabilité ainsi que de l'efficacité énergétique des circuits numériques embarqués. L'ajout de marges temporelles (*slack*) pour tenir compte de l'impact de ces variations sur la latence n'est plus une solution technologiquement viable. Une modalité pour réduire ces marges est d'utiliser des techniques de *monitoring* en-ligne couplées à des solutions d'adaptation dynamique de la fréquence et de la tension d'alimentation. Néanmoins, cela nécessite l'emploi d'infrastructures de surveillance en continu, ce qui soulève des questions quant à leur conception, leur déploiement, leur test, ainsi que leur impact sur les circuits.

L'objectif de ce travail de thèse est de proposer des réponses à ces questions. Une des approches utilisées est de chercher des synergies entre les infrastructures de surveillance continue et celles de test qui restent indispensables afin d'assurer la qualité des circuits en sortie de fonderie.

La première contribution de ce travail de thèse est une solution permettant de réduire l'impact conjoint des deux infrastructures sur la latence des circuits, tout en assurant la testabilité des moniteurs proposés. Cette solution peut également se décliner de manière à concevoir des structures de test avec un plus faible impact sur la latence même en l'absence de monitoring en ligne.

Comme l'infrastructure de monitoring engendre un surcoût en surface et en performance, le déploiement de moniteurs sur l'ensemble d'un circuit n'est pas une solution économiquement viable. Une seconde contribution de ce travail est la proposition d'une méthode d'évaluation des localités propices à recevoir un moniteur, se basant sur la probabilité d'activation, afin de permettre de minimiser l'accroissement en surface tout en minimisant l'impact sur la qualité de la surveillance.

TABLES DES MATIÈRES

INTRODUCTION GÉNÉRALE

Avec l'évolution des technologies sub-microniques, les circuits très intégrés (*VLSI* pour *Very Large Scale Integration*) ont vu, comme cela avait été établi par Gordon E. MOORE dès 1965, leur nombre de transistors implantés par cm^2 continuer à doubler tous les deux ans ("Moore's Law"), tout en observant une réduction continue de leur surface de l'ordre de 30% [Boh09]. Cette augmentation de la densité d'intégration a permis un accroissement général des performances ainsi qu'une réduction des coûts de production. En parallèle à ces évolutions, la voie de la diversification ("More than Moore") définit une tendance consistant à intégrer de plus en plus de fonctionnalités différentes au sein d'une même puce, indépendamment du degré de miniaturisation. L'utilisation de systèmes sur puce (*SoC* pour *System on Chips*) permet de réduire le temps de développement en diminuant radicalement le nombre d'interconnexions entre les puces. Toutefois, en même temps que ces améliorations, on voit émerger de nouvelles problématiques liées notamment à la variabilité des processus de production, à leur test, et aux besoins toujours plus présents en termes de fiabilité et d'efficacité énergétique.

En raison de l'enfouissement et de la complexification des SoC, leur test peut devenir extrêmement difficile. Pour cette raison, les techniques de conception en vue du test (*CVT* ou *DFT* pour *Design For Test*) sont devenues indispensables pour assurer que les circuits manufacturés, et cela même en présence de variations dans le processus de production, soient conformes aux spécifications et puissent être livrés aux clients. Ces techniques consistent à déployer certains efforts, tant matériels que méthodologiques, lors de la conception d'un circuit, dans le but d'en rendre le test économiquement et technologiquement viable. Toutefois, les techniques de *DFT* peuvent néanmoins présenter un impact négatif non négligeable sur le circuit en termes de surface, de consommation, ou de performance.

Avec la réduction de la taille des transistors, une autre source de dégradations en terme de performance et de fiabilité provient du vieillissement des circuits, ainsi que des variations de l'environnement de mission dans lequel ils évoluent. Afin de tenir compte de ces potentielles dégradations, historiquement, une approche conservatrice était d'ajouter des marges temporelles de sécurité. Aujourd'hui, pour tenir compte des incertitudes, il peut être nécessaire d'intégrer des marges temporelles allant jusqu'à 20% de la période idéale du signal d'horloge [BDM02], ce qui présente un impact trop important sur la consommation et la performance du circuit. Une diminution des marges temporelles de sécurité peut être réalisée par un suivi dynamique des caractéristiques propres au circuit afin de garantir son bon fonctionnement. Néanmoins, la plupart des architectures de test de production ne sont pas adaptées pour assurer dynamiquement ce type de suivi.

C'est dans ce contexte que le monitoring en ligne, et plus particulièrement celui des marges temporelles, peut trouver son intérêt. Il peut permettre de diminuer dynamiquement et drastiquement les marges temporelles, tout en assurant la fiabilité du circuit. D'un autre point de vue, la détection d'une diminution des marges temporelles peut être l'indication d'une dégradation ou d'un vieillissement du circuit. Le monitoring des marges temporelles peut donc être employé dans deux contextes complémentaires : l'adaptation dynamique des performances des circuits et leur fiabilisation. Toutefois, comme pour le *DFT*, la mise en place de solutions de monitoring peut présenter certaines contraintes en termes de consommation, performance, surface ou test.

De nos jours, il apparaît intéressant de disposer de moyens mutualisés, à la fois de test de production et de monitoring en fonctionnement, fiables et avec un impact réduit, tant au niveau de la latence, de la surface, de la consommation, que sur le flot de mise en œuvre afin d'être industriellement viables. C'est tout cela qui va justifier cette étude. Elle va consister en le développement d'une architecture de test pouvant inclure du monitoring des marges temporelles et permettant de réduire l'impact du test sur la latence d'un circuit, tout en assurant la testabilité à la fois du circuit et de l'infrastructure de monitoring si celle-ci est implémentée. En ce qui concerne les contraintes en termes de surface et de consommation pour le monitoring, cette étude visera également à développer des méthodologies afin d'optimiser le déploiement des moniteurs au sein des *VLSI*.

Ainsi, on va dans un premier temps développer une bascule de *scan* à faible impact sur la latence, ayant une testabilité équivalente aux standards industriels et n'impactant pas le flot de conception pour rester compatible avec les standards industriel et les outils existants. Pour cela, on montrera comment extraire le multiplexeur de test du chemin de données (*data-path*) d'une bascule sachant que c'est son temps de traversée qui prédomine dans l'impact sur la latence. Ensuite, en faisant appel à un second élément séquentiel implémentant du scan, mais placé en dehors du *data-path* (élément *shadow*), on montrera qu'il est possible de préserver la testabilité du circuit en permettant un transfert asynchrone vers la bascule fonctionnelle des données du test. En s'appuyant sur cette nouvelle bascule *scan*, on montrera qu'il est possible de réaliser un monitoring testable des marges temporelles pour un surcoût limité. Pour implémenter ce monitoring, on fera appel aux techniques de double échantillonnage et de comparaison de signaux retardés, comme développées dans [EKD+03, Nic99, ZSK05]. En combinant les deux implémentations (avec et sans monitoring) de la bascule, on verra par la suite qu'il est possible à la fois de réduire l'impact de l'architecture de test sur la latence du circuit, mais aussi d'absorber l'impact du monitoring sur la latence. Enfin, afin de limiter l'impact du monitoring sur un circuit tout en préservant la qualité du monitoring, on cherchera à optimiser la répartition spatiale et temporelle des moniteurs. Dans cette optique, on développera deux métriques permettant de guider le choix des lieux d'implantation possible des moniteurs. Pour cela, on évaluera par simulation l'activité de chaque moniteur dans le circuit en appliquant un grand nombre d'entrées aléatoires ou un work-load adapté au circuit, afin d'extraire des informations permettant de classifier et d'évaluer les moniteurs.

Ce manuscrit présente l'ensemble des travaux effectués et est organisé de la manière suivante : Le chapitre I rappelle les concepts liés au test et au monitoring des circuits numériques. Les chapitres II et III présentent respectivement comment réutiliser l'architecture de test pour implémenter un monitoring des marges temporelles, testable à faible impact sur le flot, puis comment implémenter à partir de cette structure, une architecture de test à faible impact sur la latence sans monitoring. Le chapitre IV présente une approche de combinaison des deux architectures précédentes afin de limiter l'impact du monitoring sur la latence. Enfin, le chapitre V, décrit l'étude des méthodes pour le choix des lieux d'implantation de moniteurs. La Conclusion et les perspectives seront données à la suite du chapitre V.

Chapitre 1

ÉTAT DE L'ART

Un défi majeur dans la réalisation de circuits électroniques très intégrés est de réussir à obtenir des millions ou des milliards de transistors, tous fonctionnels, pour des milliards de cycles consécutifs de commutation. La réduction d'échelle des technologies CMOS a augmenté l'impact de la variabilité des processus de production. Les transistors deviennent si petits que les moindres écarts dans les processus photo-lithographiques, ou les fluctuations dans le nombre d'atomes dopants ont des effets majeurs sur leur performance et leur consommation. En dehors de ces variations liées au processus de production (P), le fonctionnement des circuits CMOS peut être fortement affecté par des modifications environnementales comme les variations de la tension d'alimentation (V) et de la température (T).

Ainsi, les variations provenant du processus de production sont condisérées comme statiques car elles pourront être identifiées et quantifiées en sortie du processus de fabrication, entre autres, par les étapes de test et de caractérisation des circuits. A l'inverse, les deux autres types de variations sont considérées comme dynamiques car elles seront dépendantes de l'évolution du milieu dans lequel le circuit va réaliser sa mission. Une approche conservative pour tenir compte de ces variations consiste à rajouter des marges temporelles dans la période avec laquelle les signaux logiques sont évalués. La taille de ces marges temporelles peut être limitée en effectuant du monitoring afin d'assurer que les variations n'impactent pas le bon fonctionnement des circuits.

L'objectif de ce premier chapitre est donc de situer le contexte de l'étude. Après un rapide rappel sur les notions de défaut, de faute et d'erreur, on présentera ensuite le modèle des fautes de collage, qui sera utilisé pour toute la

suite de l'étude. Ensuite, les aspects liés au test de production des circuits numériques seront présentés, ainsi que différentes méthodes et outils existants en vue d'améliorer la testabilité des circuits. Enfin, la dernière partie de ce chapitre sera consacrée au monitoring des marges temporelles dans les circuits numériques.

I Défaut, Faute, Erreur et Défaillance

I.1 Définitions et Notations générales

Quelle que soit la complexité d'un circuit, celui-ci doit accomplir la mission conformément à son cahier des charges. Toutefois, des différences entre sa spécification et les résultats qu'il produit réellement peuvent apparaître.

Ainsi, l'apparition d'une différence au sein d'un système électronique entre le matériel mis en œuvre et la conception prévue est appelée un *défaut* [BA00, ALRL04]. Dans les circuits électroniques, on trouve différents types de défauts qui peuvent apparaître pendant la phase de production du circuit ou pendant sa phase de mission. Parmi eux on peut entre autres citer les défauts de [HM81] :

- Processus de production : rupture d'oxyde, impureté, contamination ...
- Matériau : impureté, fissure ...
- Vieillissement : électromigration, claquage de diélectrique ...
- Boîtier : étanchéité, dégradation des contacts ...

La compréhension et le traitement de ces défauts apparaissant dans les circuits peuvent être difficiles, car ils peuvent notamment nécessiter l'utilisation de modèles physiques complexes. C'est pour cela qu'il devient utile de proposer des modèles au niveau électrique ou logique décrivant l'impact des défauts sur le fonctionnement des circuits. Par exemple, en fonction de sa localisation dans le circuit, un défaut peut se modéliser par une "*simple*" modification au niveau logique du circuit. Cette modification au niveau logique ou électrique de la structure ou du comportement d'un circuit correspond à la représentation du défaut à un degré d'abstraction supérieur et est appelée une *faute* [BA00, ALRL04].

Si le circuit présente des défauts, sous certaines conditions particulières de stimulation, on peut s'attendre à ce qu'il fournisse des résultats faux, ce qui se

traduira par l'apparition de valeurs erronées captées dans les points mémoire ou à la sortie d'un circuit. L'état d'un élément mémoire différent de l'état attendu est appelé une *erreur* [BA00, ALRL04].

Si une erreur vient à se propager à l'extérieur du circuit, elle sera susceptible d'induire un dysfonctionnement, c'est-à-dire un écart inacceptable pour l'utilisateur entre le service spécifié et le service actuellement délivré par le circuit. Ce cas de figure est appelé une *défaillance* [LEV14, ALRL04].

I.2 Fautes de collage unique (*Single Stuck Fault ou SSF*)

On peut faire une distinction, entre les défauts qui affectent l'exactitude logique d'un circuit et ceux qui affectent sa vitesse de fonctionnement. Dans le cadre de ce travail, nous ne considèrerons que les défauts affectant l'exactitude logique d'un circuit, et nous nous limiterons au modèle de fautes de collage unique [ABF90, JG03, ABB+04, Nav10].

Le modèle de fautes de collage unique, fut le premier à être utilisé, et cela depuis l'avènement des circuits logiques discrets en technologie bipolaire. Ce modèle s'applique à un seul nœud d'un circuit en lui associant une valeur logique constante, égale à la valeur logique *0* ou *1*, et cela quels que soient les stimuli appliqués à ce nœud. Un signal court-circuité avec une ligne d'alimentation qui présentera la valeur logique *1* (*stuck-at-1* ou *s@1*), un signal relié à la masse qui sera donc un *0* logique permanent (*stuck-at-0* ou *s@0*), ou une connexion mal réalisée et laissée flottante et qui pourra apparaître comme un niveau logique *1* ou *0* indépendamment des stimuli appliqués, sont des exemples de défauts dont le comportement peut être modélisé par des fautes de collage. Un des intérêts majeurs de cette modélisation est que l'on transforme ainsi des problèmes du domaine de la physique, qui peuvent être difficiles à traiter, en des problèmes du domaine de la logique booléenne beaucoup plus aisés à appréhender [ABF90, Pat98].

L'exemple de la Figure 1.1 illustre ce concept dans le cas d'une simple porte logique *ou* à deux entrées. La sortie d'une telle porte qui ne comporte pas de fautes devra prendre la valeur logique *0* si on applique sur les deux entrées la valeur logique *0*. Si l'on considère maintenant le cas présenté en Figure 1.1 où un défaut (poussière, contamination, mauvaise maîtrise du processus de photolithographie) conduit à la présence d'un court-circuit entre une des entrées

avec le bus d'alimentation, dans ce cas la réponse de la porte devient erronée, puisque quelles que soient les valeurs appliquées en entrée, la sortie prendra toujours l'état logique *1*.

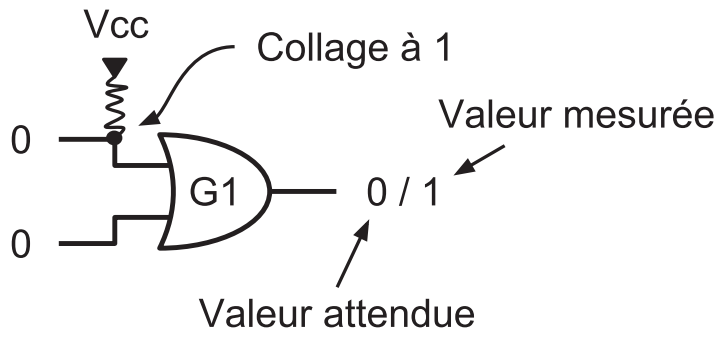


FIGURE 1.1 – Exemple de fautes de collage.

Malgré le fait que de récentes études [MT00] tendent à prouver son inadéquation avec les problématiques du monde actuel des semi-conducteurs, le modèle de fautes de collage unique reste encore aujourd'hui très largement employé. Une des raisons de son succés est qu'il présente une indépendance complète vis-à-vis de la technologie utilisée, et qu'il permet de réemployer les résultats de simulations fonctionnelles pour la génération des stimuli utilisés pour la génération de vecteurs de test. De plus, de par son haut niveau d'abstraction et son indépendance vis-à-vis de la technologie, il permet la modélisation de nombreux types de défauts physiques [ABF90, Pat98, AAA87]. Enfin, le nombre des 'lieux' possibles pour les fautes de collage unique croît linéairement avec la taille du circuit et le nombre de fautes à considérer réellement peut être réduit par différentes techniques d'analyse et de simplification [ABF90, BA00, JG03, ABB+04, Nav10].

II Test

Réaliser le test d'un circuit signifie détecter si celui-ci fonctionne conformément à ses spécifications. Dans le cas du test de production, il est indispensable de pouvoir distinguer les circuits "bons", pouvant être livrés au client, des circuits "mauvais", par exemple à cause d'une variation dans le processus de production, qu'il faudra diagnostiquer et éliminer afin de corriger et améliorer le processus [EBSLM97, BDM02, HAP+05, BKN+03]. Ainsi, pour réaliser le test de production d'un circuit, le processus consiste en l'application d'une séquence

de valeurs en entrées et l'observation de l'évolution des sorties, dans le but d'activer une défaillance particulière.

Le test des circuits intégrés peut être envisagé selon deux approches : le test fonctionnel et le test structurel. Le test fonctionnel vérifie si le circuit effectue bien ce pour quoi il a été conçu, alors que le test structurel vise à s'assurer que le circuit a été convenablement manufacturé à partir des briques de base (*i.e.* portes logiques) comme cela est spécifié dans la *netlist* [Nav10]. Par exemple, on vérifiera que toutes les portes logiques prévues soient bien présentes, qu'elles donnent les réponses attendues selon leurs tables de vérités, et qu'elles soient interconnectées convenablement. L'idée derrière ce type de test est que si la spécification de la netlist était correcte, et que le test structurel a confirmé l'assemblage correct des éléments de base du circuit, alors celui-ci devrait présenter un fonctionnement correct.

II.1 Génération des vecteurs de test

Afin de mettre en évidence un défaut au sein d'un circuit, il est nécessaire de lui appliquer des stimuli en entrée. Cette génération peut être réalisée à différents niveaux d'abstraction et de nombreuses techniques existent. Dans le domaine du test, la notion d'entrées / sorties primaires est particulièrement importante car elles représentent les points d'accès au circuit pour le test. Ainsi, une entrée primaire (ou *PI* pour *Primary Input*) est une entrée pouvant être contrôlée directement par l'environnement de test. De même, une sortie primaire (ou *PO* pour *Primary Output*) est une sortie dont la mesure peut être réalisée directement dans l'environnement de test. L'ensemble des valeurs appliquées à un instant particulier aux *PI*s est appelé un *vecteur d'entrées de test*, alors que l'ensemble des valeurs mesurées sur les *PO*s s'appelle, dans le cas d'un circuit sans erreurs, la *réponse*. La combinaison de l'application d'un vecteur d'entrées de test et de la mesure de la réponse est appelée un *vecteur de test*. Après une brève présentation des techniques de test exhaustif et aléatoire, nous allons nous intéresser plus particulièrement au test structurel déterministe des circuits. Nous noterons $k_{/x}$, le fait qu'un nœud k soit collé à la valeur x ($x \in \{0,1\}$), et nous ne considérerons dans un premiers temps que le test des fautes de collage unique, de circuits logiques combinatoires. C'est-à-dire des circuits pour lesquels à tout instant l'état de leurs *PO*s peut être entièrement déterminé à partir de la seule connaissance de l'état de leurs *PI*s.

II.1.i Test exhaustif

Le test exhaustif, est une technique de test qui consiste à appliquer au circuit sous test (ou *DUT* pour *Device Under Test*) toutes les combinaisons de *PI*s possibles. Si cette approche peut apparaître tout d'abord comme la plus évidente à déployer, elle présente tout de même deux problèmes majeurs.

— Les fautes qui nécessitent plusieurs vecteurs de test pour être testées ne sont pas directement prises en compte.
— La longueur des séquences de test devient rapidement prohibitive.

Considérons l'exemple d'un circuit combinatoire à n entrées, tel que présenté dans la Figure 1.2 (a). Pour tester ce circuit de façon exhaustive, une séquence de 2^n entrées doit être appliquée et mesurée pour exercer l'ensemble des cas possibles.

Si ce circuit est maintenant converti en un circuit séquentiel comme en 1.2 (b), avec l'ajout de m registres ou bascules, on s'aperçoit que l'état interne du circuit devient dépendant à la fois des *PI*s, mais également de son état précèdent. Ici, certaines lignes qui forment les entrées du bloc combinatoire ne sont plus directement contrôlables et seront appelées des entrées *PPI* pour *Pseudo-Primary Input*. De même, certaines des lignes de sorties du bloc combinatoire, qui ne sont plus directement observables, seront appelées des sorties *PPO* pour *Pseudo-Primary Output*.

Ainsi, un minimum de 2^{n+m} vecteurs de test doivent être appliqués en entrée pour tester l'ensemble des états du circuit ce qui peut mener à des temps de test irréalistes.

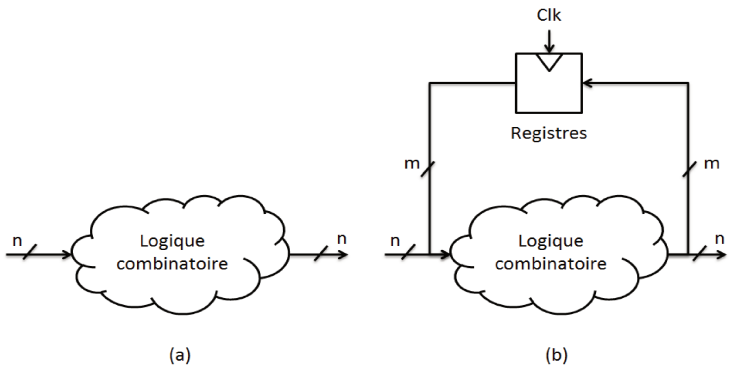


FIGURE 1.2 – Accroissement de la difficulté du test des circuits séquentiels vis-à-vis des circuits combinatoires.

Dans [WP83], il est montré qu'un circuit avec n = 25 entrées et m = 50 registres nécessitera 2^{75} vecteurs de test. Si l'on considère 1µs pour l'application de chaque vecteur de test, la durée minimale de test atteignable serait donc supérieure à un milliard d'années.

II.1.ii Test aléatoire

Le concept associé au test aléatoire des circuits, est d'appliquer un grand nombre de vecteurs d'entrée, générés de façon aléatoire ou pseudo-aléatoire, et de vérifier les réponses obtenues en les comparant à celles d'un circuit sain (*golden-circuit*) ou celles issues d'une simulation logique. L'intérêt de ce genre de technique réside dans l'économie en termes de temps de calcul, et d'espace de stockage nécessaire pour les vecteurs de test. Néanmoins, l'inconvénient est que l'on ne peut pas garantir le test de toutes les fautes pouvant apparaître dans le circuit. De même, la détermination de la longueur de test nécessaire pour atteindre une certaine qualité de test représente une des difficultés majeures de cette technique.

II.1.iii Test déterministe structurel

Afin de générer de manière déterministe un vecteur de test pour une certaine faute, trois étapes sont indispensables :

1. Sensibilisation de la faute.
2. Propagation de l'effet de la faute.
3. Justification des entrées pour le test de la faute considérée.

La sensibilisation d'une faute de collage à x d'un nœud k, consiste à forcer la valeur complémentaire de x ($\overline{x}$) en k afin que la faute puisse avoir un impact sur le fonctionnement du circuit.

La propagation de l'effet de la faute consiste à assurer un chemin logique entre le nœud k et une *PO* du circuit. Afin de s'assurer de la bonne observation de l'effet de la faute, il est indispensable pour toutes les portes faisant partie du chemin logique identifié, de positionner sur leurs entrées non directement incluses sur le chemin, des valeurs définies comme "non prioritaires". On entend par valeurs "non prioritaires", des valeurs qui n'imposent pas une valeur spécifique en sortie de la porte, par exemple une valeur *1* logique pour une porte

et, ou un *0* logique pour une porte *ou*. Cela est indispensable afin de n'observer que l'effet de la faute, et non celui d'un autre signal logique.

Enfin, la dernière étape, appelée "justification", vise à trouver les valeurs à appliquer sur les *PI*s afin d'assurer la mise en oeuvre des deux étapes précédentes. Ce processus peut toutefois s'avérer complexe, car il présente des contradictions dans l'assignation des valeurs des différentes lignes du circuit.

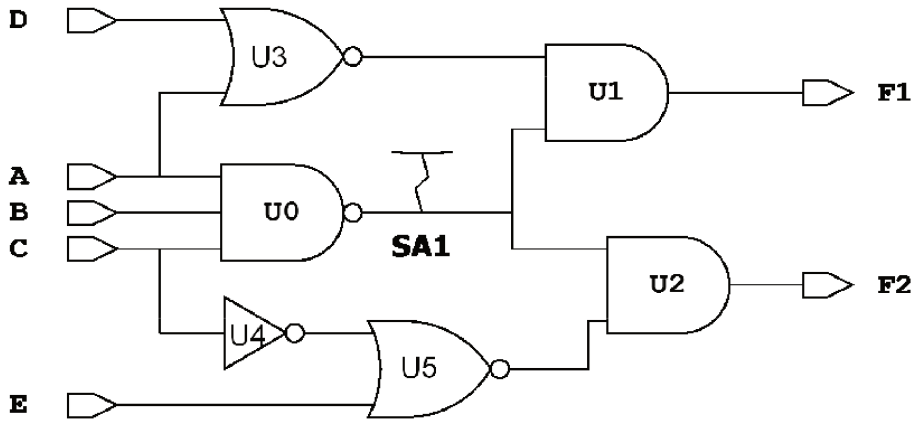


FIGURE 1.3 – Exemple de test déterministe de fautes de collage.

Afin d'illustrer ce processus, prenons l'exemple de la Figure 1.3. Si l'on veut tester la faute *s@1* en sortie de *U0* il faut donc :

1. Sensibiliser la sortie de la porte *U0* à *0* afin de tester le faute *s@1*.

Comme *U0* est une porte *non-et* à trois entrées, cela imposte donc *A=B=C=1* pour obtenir *0* en sortie de *U0*.

2. Propagation de la valeur vers une *PO* du circuit soit *F1* ou *F2*.

Pour pouvoir détecter la faute en sortie d'*U0*, il faut appliquer des valeurs non prioritaires sur les autres entrées de *U1* ou *U2*. Or, comme la sensibilisation impose *A=1*, on s'aperçoit que quelle que soit la valeur appliquée sur *D*, *U3* donnera toujours *0* en sortie, ce qui bloquera *U1* à *0*. Ainsi, le chemin passant par *U1* n'est pas exploitable pour le test de cette faute et il faudra obligatoirement passer par *U2*.

3. Justification des valeurs d'entrées pour le test de la faute.

Afin d'avoir un chemin passant par *U2*, cela implique d'avoir *1* en sortie de *U5*, donc *00* en entrée de *U5*. Dans le but d'avoir *00* aux entrées de *U5* il faut *C=1* et *E=0* ce qui est bien compatible avec l'étape de sensibilisation.

Ainsi, pour tester la faute *s@1* en sortie de *U0* on appliquera le vecteur *111D0* où ici *D* représente la valeur "*don't care*" c'est-à-dire que quelle que soit la valeur appliquée en *D* le test sera effectif.

Pour être complet, une dernière information reste à fournir avec le vecteur de test. C'est la valeur que l'on doit mesurer sur les *PO*s si la faute n'est pas présente. Ainsi, nous devrions mesurer *X0* en *F1 F2*. Ici *F1* prend la valeur *X* inconnue, car il n'est pas utile de la connaître. En effet, tant que *F2=0*, on s'assure de l'absence de la faute considérée.

Quoi qu'il en soit, il peut arriver que certaines fautes ne soient pas testables car il n'existe aucune combinaison de vecteurs en entrée permettant leur justification. À l'inverse, plusieurs fautes peuvent être considérées équivalentes en fonction de la structure du circuit ou si elles sont détectées par les mêmes vecteurs de test.

Aujourd'hui un important travail de recherche a été accompli dans le domaine du calcul de vecteurs de test, ce qui a notamment conduit au développement de nombreux outils de génération automatique de vecteurs de test. Le plus grand nombre de ces travaux portent sur le test de circuits combinatoires en considérant des fautes de collage unique [BA00, ABF90, ABB+04, JG03].

En ce qui concerne les circuits séquentiels, le test d'une faute nécessite souvent la génération de plusieurs vecteurs à appliquer successivement, ce qui rend la tâche beaucoup plus ardue que pour le test de circuits combinatoires. Pour le comprendre, on peut représenter tout circuit séquentiel sous la forme d'une machine d'états finis composée de deux parties comme présenté sur la Figure 1.4 :

— Une partie de logique combinatoire $\mathscr{L}$.

— Des éléments mémoires $\mathscr{R}$ assurant la transition entre les différents états.

C'est au niveau du contrôle et de l'observation des valeurs des entrées/-sorties pseudo-primaires (*PPI*s/*PPO*s) que se situe la difficulté liée au test des circuits séquentiels.

Ainsi, l'observation et le contrôle de l'état des registres dépendent des états précédents ainsi que des *PI*s du circuit. De ce fait, si on note *y(t)* l'état courant du circuit, on peut définir $y(t+1) = F(y(t), PI(t))$, avec, $y(t)$ et $y(t+1)$ qui re-

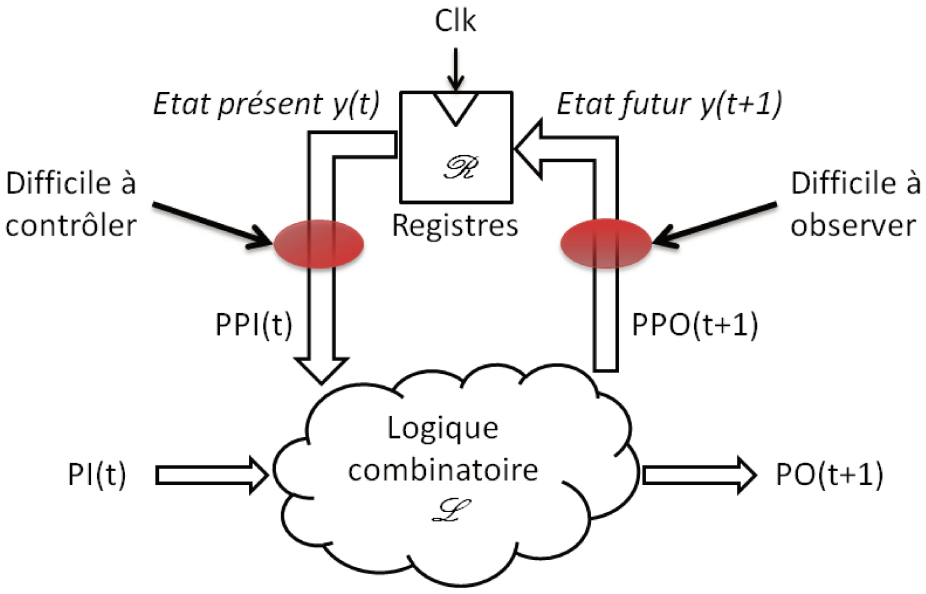


FIGURE 1.4 – Représentation sous forme de machine d'états d'un circuit séquentiel.

présentent respectivement les états présents et futurs des registres aux instants t et $t+1$, et $PI(t)$ qui correspond aux valeurs appliquées aux *PPI*s à l'instant t.

C'est à cause de cette dépendance temporelle entre les différents états que le test des circuits séquentiels est compliqué et peut nécessiter d'appliquer une séquence de plusieurs vecteurs d'entrée sur les *PI*s, à la fois pour positionner les bonnes valeurs sur les *PPI*s, mais aussi pour permettre la transmission de la réponse aux *PPO*s [CA89, Via96, Dar97].

De nos jours, la génération automatique de vecteurs de test se décompose généralement en trois étapes.

1. Une phase de génération aléatoire.
2. Une étape de génération déterministe pour traiter les fautes restantes.
3. Une phase de compactage de la séquence de test afin d'en réduire la longueur.

II.2 Évaluation d'une séquence de test

Un aspect important du test réside dans l'évaluation à proprement parler d'une séquence de test. Cela se réfère à déterminer l'efficacité ou la qualité de la séquence en question, cette détermination se réalisant généralement dans le contexte d'un modèle de faute particulier.

II.2.i Métriques liées au test

En règle générale, deux métriques sont employées pour exprimer la qualité d'une séquence de test. On peut tout d'abord l'exprimer sous la forme d'un taux de couverture de fautes, qui correspond au rapport entre le nombre de fautes détectées par la séquence de test, vis-à-vis du nombre total de fautes pouvant exister dans le circuit. Il est à noter toutefois, que le taux de couverture de fautes n'est représentant que pour une certaine hypothèse de fautes, même s'il a été montré qu'une bonne couverture de fautes pour les fautes de collage s'accompagne souvent d'un bon taux de couverture pour les défauts réels [ABB+04].

La qualité d'une séquence de test peut également être exprimée sous la forme de couverture de test. Cette couverture, très similaire à la couverture de fautes, ne tiendra compte pour sa part que de l'ensemble des fautes détectables pour l'évaluation. Ainsi, la couverture de fautes sera souvent moins bonne que celle de test puisque dans cette dernière, les fautes non testables ne sont pas prises en compte.

Enfin, une autre métrique importante est la longueur de la séquence de test, c'est-à-dir, le nombre de vecteurs de test nécessaires. En effet, comme les équipements industriels de test sont souvent loués à la µs et que leur capacité de stockage est limitée, on cherchera à réduire au maximum le nombre de vecteurs nécessaires.

II.2.ii Simulation de fautes

La simulation de fautes permet de déterminer le nombre de fautes qui peuvent être testées par une séquence de test. Le concept de base consiste en l'exécution de nombreuses simulations en parallèle du même circuit présentant une faute différente pour chaque instance de simulation, ainsi qu'une simulation qui représente le circuit de référence, appelé "*golden-circuit*" et qui servira de point de

comparaison. La simulation de fautes détecte un défaut chaque fois que la réponse fournie par la simulation est une valeur non-*X* et différente de la réponse fournie par la simulation du *golden-circuit*. La simulation de fautes permet donc de déterminer toutes les fautes détectées par un vecteur de test. Cela permet, entre autres, de savoir combien de fois chaque faute sera testée avec le lot de vecteurs fournis en entrée ainsi que d'établir les taux de couverture de fautes et de test de la séquence.

Toutefois, compte tenu du nombre souvent important de fautes à considérer, de la taille des séquences de test, et de la complexité des circuits à simuler, l'approche de réaliser une simulation logique pour chaque faute n'est souvent pas envisageable. C'est ici que se trouve la problématique de la simulation de fautes, qui va viser à mettre en œuvre des solutions pour traiter en parallèle l'impact de chacune des fautes afin de ne simuler qu'une seule fois chaque vecteur.

Ainsi, plusieurs techniques de simulation de fautes ont vu le jour [BA00, ABF90, ABB+04, JG03]. Parmi elles, on peut noter l'existence de la simulation *série*, très lente, car elle reprend le concept évoqué ci-dessus.

Une autre approche de simulation est l'approche dite *parallèle*. Elle vise à tirer parti du traitement parallèle de l'information par mots de la part des ordinateurs afin d'accélérer la simulation en simulant $(k-1)$ circuits fautifs en parallèle sur un ordinateur capable de traiter des mots de k bits.

Une troisième technique est la simulation *déductive*. Elle permet de simuler un nombre de fautes infini car on ne simule ici que le circuit sain et on en déduit le comportement des circuits présentant un défaut.

La quatrième technique est la simulation de fautes *concurrente* et est la méthode la plus générale. Le principe est ici de ne simuler que les parties de circuits fautifs ayant un comportement différent du circuit sain.

La simulation de fautes peut aussi être utilisée au sein même du processus de génération de la séquence de vecteurs de test. En effet, le simulateur peut être utilisé de façon à optimiser la séquence de test en rajoutant des vecteurs ou à l'inverse en supprimant les vecteurs redondants. Il en est de même pour la liste des fautes, qui correspond à l'ensemble de fautes qui doivent être testées. Ainsi, si un vecteur généré permet de couvrir le test de plusieurs fautes, elles pourront être retirées de la liste de fautes tel que montré dans la Figure 1.5.

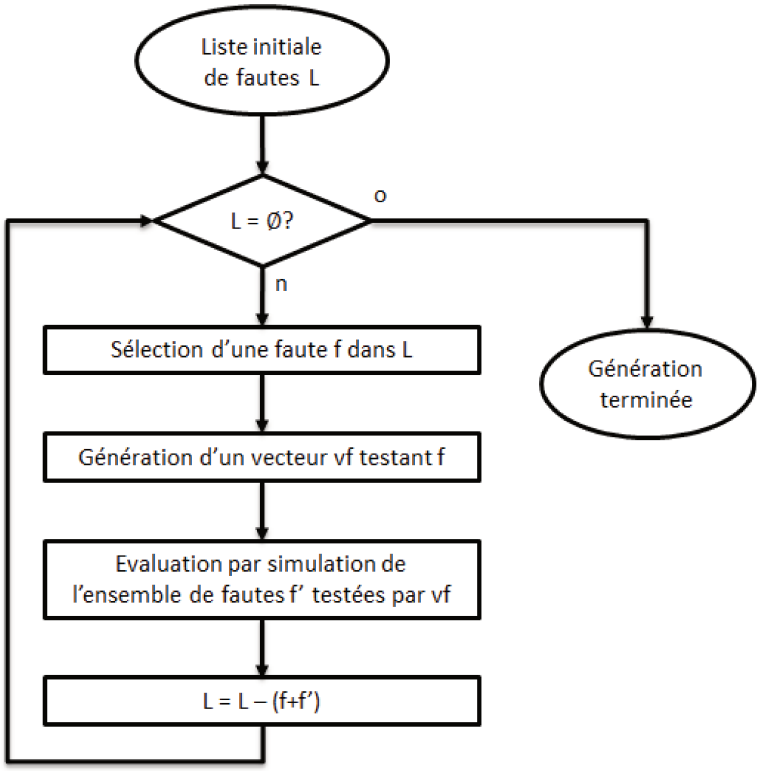


FIGURE 1.5 – Combinaison de la simulation de fautes et de la génération de vecteurs de test.

II.3 Testabilité, notions d'observabilité et de contrôlabilité

Comme nous l'avons vu, pour qu'un circuit puisse être testé de façon simple et fiable, il est indispensable que les lieux potentiels d'apparition de fautes soient à la fois :

— Contrôlables, c'est-à-dire que par le biais de stimuli en entrée, il soit possible d'y appliquer des valeurs particulières.

— Observables, c'est-à-dire que les valeurs prises par ces nœuds doivent également pouvoir être transmises vers "l'extérieur", notamment par exemple à l'équipement de test automatique (*ATE*).

Ainsi, la testabilité d'un système représente le niveau de difficulté vis-à-vis de ces deux aspects.

L'observabilité d'un nœud spécifique au sein d'un circuit est utilisée pour quantifier la facilité avec laquelle la valeur logique prise par ce nœud peut être mesurée au niveau des sorties du circuit [WH, ABF90, BA00, LEV14]. Cette notion est importante afin de s'assurer que le circuit opère convenablement. Toutefois, compte tenu du nombre limité de sorties disponibles, et de la complexité

des circuits, une des missions du concepteur consiste à faire en sorte que cette observation soit facilitée. Idéalement, un nœud présentant une bonne observabilité doit être mesurable directement, ou avec un nombre modéré d'indirections (quelques cycles d'horloge) sur les sorties du circuit.

La contrôlabilité d'un nœud interne d'un circuit dénote pour sa part la facilité de positionner arbitrairement une valeur logique spécifique sur ce nœud [WH, ABF90, BA00, LEV14]. Ainsi, un nœud disposant d'une bonne contrôlabilité sera directement contrôlable par les entrées du circuit. À l'inverse, un nœud présentant une contrôlabilité limitée pourra nécessiter plusieurs centaines de cycles d'horloge avant de prendre la valeur souhaitée. Ainsi, tout comme pour l'observabilité, un bon concepteur de circuit doit également faire en sorte que la contrôlabilité des nœuds du circuit soit facilitée. Cela peut en partie être réalisé par l'utilisation de techniques de conception spécialisées en vue de faciliter le test, telles que par exemple utiliser un signal commun pour la remise à zéro de tous les éléments séquentiels du circuit, ce qui est un premier pas vers une meilleure contrôlabilité.

III Conception en vue du test (*CVT* / *DFT*)

Avec les contraintes du monde industriel actuel, le test de production est une préoccupation croissante dans presque tous les domaines et pour toutes les applications. Pour cette raison, il se doit d'être à la fois, fiable, rapide, et le moins coûteux possible. En effet, en détectant les circuits potentiellement défaillants le plus tôt possible, les coûts liés aux corrections peuvent rester relativement bas [ABB+04, WH]. Ainsi, cela signifie que la mise en place de méthodes de test doit être prise en compte dès les premiers instants, et pour toutes les phases du développement, de la production et du déploiement du produit. Comme nous l'avons vu, la clé pour atteindre une bonne testabilité réside dans une bonne contrôlabilité et une bonne observabilité. En effet, cela permet de réduire le coût du test en permettant un haut niveau de couverture de fautes avec un nombre limité de vecteurs de test. Ainsi, la *Conception en Vue du Test* (*CVT*) ou *Design For Test* (*DFT*) vise à améliorer ces deux aspects d'observabilité et de contrôlabilité d'un circuit pour en faciliter le test. Parmi les techniques de *DFT*, deux grandes tendances se distinguent, celle dite *ad-hoc*, spécifique à un circuit, et celle dite "structurée", plus généraliste.

III.1 *DFT Ad-hoc*

La *DFT ad-hoc* consiste en de "bonnes" pratiques de conception. Un exemple de solution *ad-hoc* est l'insertion de points de contrôle et/ou observation pour les signaux présentant une faible contrôlabilité et/ou observabilité [ABF90,WH, BA00,ABB+04]. D'autres exemples dérivent de l'application de règles de conception comme, par exemple :

— Éviter les rebouclages logiques asynchrones.
— Éviter l'utilisation de portes logiques avec un *Fan-in* important.
— Rendre les éléments séquentiels initialisables.

Ces règles visent principalement à limiter les situations à "risques" qui mèneraient à une mauvaise contrôlabilité et une mauvaise observabilité des signaux, ou nécessiteraient le développement de séquences de test longues pour une couverture de fautes limitée. Dans le cas de circuits complexes, la sélection des points de contrôle et observation peut être laborieuse, c'est pourquoi ces solutions sont donc plutôt appliquées localement et de manière limitée en modifiant le circuit là où des faiblesses vis-à-vis de la testabilité auront été identifiées [WA73, HF74].

Avec l'augmentation de la taille et de la complexité des systèmes numériques, l'application de règles de conception est devenue systématique. Ainsi, de nouvelles formes de *DFT* ont vu le jour et ont largement et rapidement gagné en popularité. Elles sont connues sous le nom de *DFT* "structurée" [Eic91, ABF90, BA00, ABB+04]. Dans ces types d'approches, des signaux et des portes logiques supplémentaires sont ajoutés au circuit afin de faciliter son test.

III.2 *Standard-scan*

Le *standard-scan* est la technique de *DFT* "structurée" la plus populaire car elle dispose d'un grand potentiel d'amélioration en termes de contrôlabilité et d'observabilité. Le principe de cette technique est de modifier les éléments séquentiels d'un circuit, afin de mettre en oeuvre un mode de fonctionnement supplémentaire appelé *mode scan*, dans lequel tous les éléments séquentiels sont chaînés et fonctionnent comme un ou plusieurs registres à décalage appelés *chaîne de scan*. Ainsi, chacun des éléments de la chaîne peut opérer comme une *PI* ou une *PO* durant la phase de test, améliorant ainsi grandement l'observabilité et

la contrôlabilité des nœuds internes du circuit (*PPI*s/*PPO*s) [ABF90, ABB+04, JG03, WH, BA00, Nav10].

Plusieurs solutions techniques existent pour modifier les éléments séquentiels. La Figure 1.6 présente un exemple de transformation d'une bascule standarde en une bascule implémentant du *scan*. Cette transformation consiste à insérer un multiplexeur en entrée de la bascule pour permettre les deux modes de fonctionnement. Ainsi, les entrées du multiplexeur sont :

— Le signal initialement connecté à l'entrée *D* de la bascule d'origine, provenant de la partie fonctionnelle du circuit.
— Le signal *scan_in* provenant d'une bascule *scan* précédente ou d'une *PI*.
— Le signal de test *scan_enable*, global à toute la chaîne et qui permet la sélection entre le signal *D* et le signal *scan_in*.

La sortie *Q* de la bascule est également utilisée comme sortie de *scan* et sera reliée à la fois à la partie fonctionnelle du circuit comme initialement mais également à l'entrée *scan_in* de la bascule suivante dans la chaîne, ou directement à une *PO* si la bascule est la dernière de la chaîne.

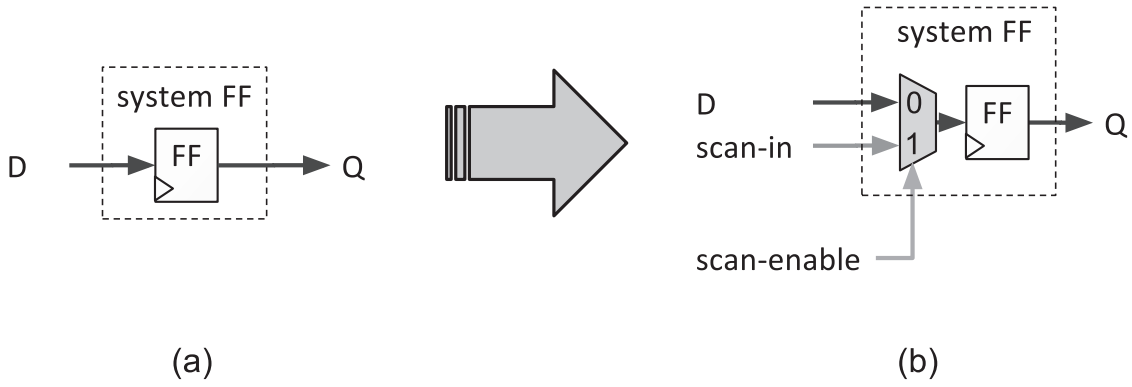


FIGURE 1.6 – Transformation bascule en bascule *standard-scan*.

Les éléments séquentiels ainsi modifiés (aussi appelés bascules de *scan* ou *scan-cells*) deviennent équivalents pour les outils de génération automatique de séquences de test (*ATPG*) à des entrées/sorties pseudo-primaires donc *scan-contrôlables* ou *scan-observables*. Pour qu'un élément séquentiel du circuit soit considéré comme *scan-contrôlable*, il faut qu'il soit possible de positionner depuis l'extérieur une valeur logique spécifique sur sa sortie *Q* par le biais de décalages en série sur la chaîne de *scan* qui lui est associée. De même, pour

que cet élément soit considéré *scan-observable*, il faut que la valeur logique prise par son entrée *D* puisse être mesurée sur une *PO* du circuit, une fois de plus, par le biais de décalages en série sur la chaîne de *scan* associée. Cette approche permet, comme montré sur la Figure 1.7, de "diviser" un circuit séquentiel en plusieurs sous-circuits purement combinatoires, et donc plus simples à tester pour les outils de test.

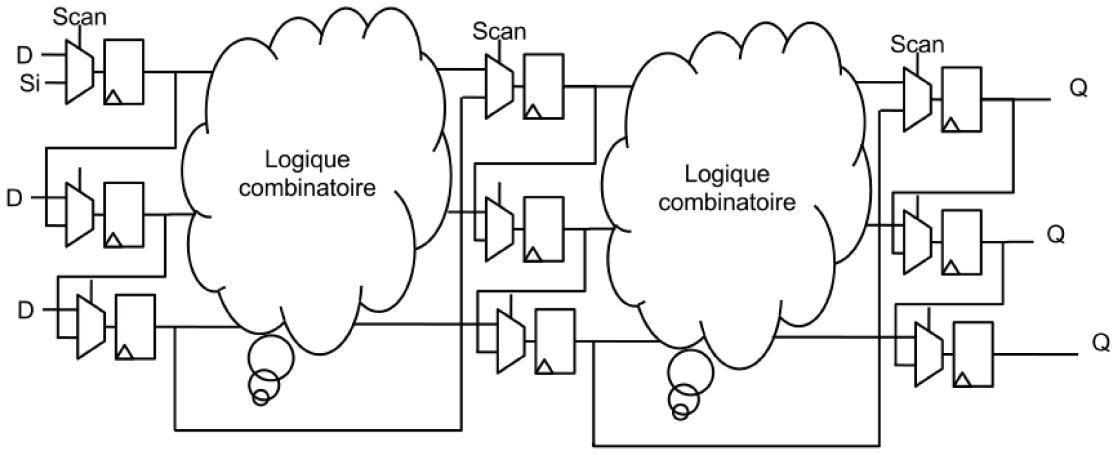


FIGURE 1.7 – Chainage des éléments séquentiels
pour réaliser une chaîne de *scan*.

Dans une approche *standard-scan*, les vecteurs de test sont appliqués au circuit par le biais des chaînes de *scan*. Une procédure de test peut être réalisée de la façon suivante :

1. Placer les chaînes en mode *scan* en pilotant le signal *scan_enable*.
2. Faire transiter les stimuli de test en série à travers les chaînes de *scan*.
3. Placer les bascules *scan* en mode *capture* grâce au signal *scan_enable*.
4. Appliquer les stimuli de test sur les *PI*s du circuit.
5. Mesurer les *PO*s du circuit et comparer avec la réponse attendue d'un circuit sain. Cette étape est appelée *mesure parallèle*.
6. Appliquer des cycles d'horloge afin de capturer la réponse des blocs combinatoires dans les chaînes de *scan*. Cette étape est appelée *capture parallèle*.
7. Placer les chaînes en mode *scan* en pilotant le signal *scan_enable*.
8. Faire transiter les valeurs internes capturées par les chaînes de *scan* vers les sorties de *scan* (*scan_out*) et comparer avec les valeurs attendues d'un circuit sain.

Grâce à l'approche *standard-scan*, on voit qu'il est aisé d'observer et de contrôler l'ensemble des éléments présents dans la chaîne de *scan*, et donc de la logique qui se trouve entre ces éléments, ce qui améliorera et facilitera grandement le test du circuit. Cette approche présente tout de même des impacts non négligeables. En effet, l'utilisation du *standard-scan* affecte la surface et la consommation, à cause de l'ajout d'un multiplexeur à l'entrée de chaque bascule et à cause de l'ajout des signaux de test. Plus important encore, cette technique présente un impact en terme de latence sur le circuit à cause du temps de traversée du multiplexeur qui reste toujours présent sur le chemin de données [JG03,CG11].

Une solution visant à diminuer ces impacts est l'approche *partial-scan*, qui consiste à laisser certaines bascules sans *scan* [JG03,ABF90]. Les deux difficultés de cette approche résident dans le choix des éléments séquentiels à transformer pour répondre à la fois aux critères de surface et de performance, ainsi que dans la préservation d'une couverture de fautes ou de test suffisante. Les bascules *scan* doivent être sélectionnées de façon à minimiser le coût du *scan* tout en préservant ses bénéfices.

Ainsi :

— Pour minimiser l'impact en surface et en performance, le nombre de bascules *scan* retenues devra être le plus faible possible et il faudra éviter de sélectionner des bascules qui se trouvent sur les chemins critiques, qui sont les chemins pour lesquels la latence est la plus importante.
— La sélection des bascules doit permettre une amélioration de la contrôlabilité et de l'observabilité du circuit pour répondre aux critères de testabilité [CA90] [GB90].

III.3 *Shadow-Scan*

Une autre approche permettant de réduire localement l'impact du scan sur la latence d'un circuit est l'approche *shadow-scan* [ABF90,Nav10,Sin12]. Cette technique consiste à remplacer la bascule *standard-scan* (fig. 1.6 (b)) par une bascule du type de celle présentée en Figure 1.8.

Comme on peut le voir, cette bascule est constituée d'un multiplexeur en sortie et de deux bascules :

— Une bascule fonctionnelle sans *scan* (*system FF*) ;
— Une bascule supplémentaire avec du *scan* (*shadow FF*), placée en parallèle du chemin de données et utilisée uniquement pour le test de production.

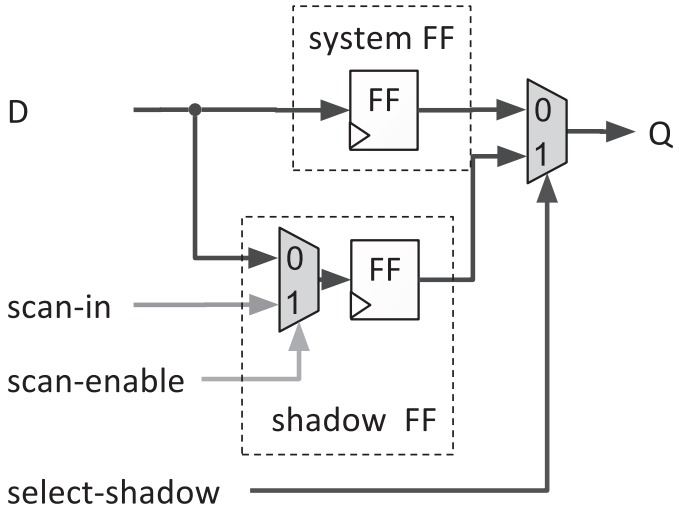


FIGURE 1.8 – *Shadow-scan* [Sin12].

Le multiplexeur de sortie permet de choisir si ce sont les données issues des bascules *shadow* ou fonctionnelles qui seront appliquées à la logique située en aval.

La diminution de l'impact du *scan* sur la latence est permise par l'absence du multiplexeur de *scan* en entrée de la bascule fonctionnelle, ce qui lui procure un temps de *setup*[1] réduit par rapport à une bascule *standard-scan*. Toutefois, l'impact du *scan* sur la latence du circuit est transféré de l'entrée vers la sortie de la bascule fonctionnelle. En effet, cela est dû au fait que le multiplexeur placé en sortie de la bascule *shadow-scan* rallonge le temps *Clk-to-Q*[2] de la bascule fonctionnelle. Enfin, il faut garder à l'esprit que l'impact en surface et en consommation de l'utilisation de bascules *shadow-scan* est très important car cette solution revient à doubler le nombre de bascules placées au bout des chemins critiques. De plus, dans la solution présentée en Figure 1.8, il est nécessaire pendant la phase de *scan* de piloter les deux signaux de test *scan-enable* et *select-shadow*. Pendant le premier cylce d'horloge en mode *scan*, il faut forcer *select-shadow* à la valeur logique *0* afin de faire transiter la donnée précedemment capturée.

1. Intervalle de temps précédant le front actif du signal d'horloge pendant lequel le signal en entrée de la bascule doit être stable afin d'être proprement échantillonné.

2. Intervalle de temps entre le front actif du signal d'horloge et la mise à jour du signal en sortie de la bascule.

IV Monitoring en ligne des marges temporelles

Avec l'évolution des technologies submicroniques, les variations dynamiques au sein des circuits créent le besoin de nouvelles approches pour complémenter le test de production [Nic97, Nic98, NZ99, Bor05]. Historiquement géré par l'ajout de marges temporelles en considérant le pire cas du processus de fabrication et de vieillissement avec ses conditions de température et de d'alimentation les plus défavorables (*worst-case PVT scenario*), les approches conservatrices ne sont plus en phase avec les attentes en termes d'évolution des performances et de la consommation [BDM02, BKN+03].

Une approche pour pallier l'impact des variations dynamiques sur la latence des circuits, est d'utiliser du test en ligne [KML+06, SGH+07, ABB+04]. Cette approche consiste à tester le système en place, de façon périodique ou à l'occurrence de certains événements, le but étant de détecter le plus rapidement possible l'apparition de dégradations, par exemple dues au vieillissement, et d'anticiper l'apparition des erreurs. Si cette approche peut avoir un faible coût matériel, un de ses inconvénients est le fait de devoir arrêter la mission du circuit ciblé et de passer dans un mode test. Par conséquent, la résolution temporelle reste limitée et le test peut être effectué dans des conditions non représentatives. C'est pourquoi d'autres approches dites "concurrentes" ont été proposées afin d'assurer le monitoring en parallèle avec l'exécution de la mission du circuit ciblé.

Le monitoring en ligne est une approche concurrente où l'on ne va pas tester le circuit, mais plutôt "suivre" de façon continue l'évolution de certains de ses paramètres caractéristiques tels que sa tension d'alimentation, sa température ou sa latence. L'idée est ici que certains des paramètres physiques d'un circuit peuvent donner une image de la confiance que l'on peut placer dans celui-ci et peuvent permettre de prévoir les futures défaillances. De plus, en ayant une vue dynamique des caractéristiques du système, l'utilisation du monitoring peut également permettre la mise en place de stratégies agressives de gestion de la consommation ou de la performance des systèmes tout en évitant l'apparition d'erreurs. Ce sont notamment ces problématiques qui deviennent des facteurs prédominants dans l'émergence des applications mobiles. L'intérêt du monitoring en ligne est donc d'être capable de collecter et d'analyser dynamiquement des données propres à l'état du circuit pendant le fonctionnement afin de le

garantir et l'améliorer.

La latence d'un circuit synchrone, c'est à dire un circuit cadencé par au moins un signal d'horloge, peut être surveillée au travers du monitoring en ligne des marges temporelles (ou *slacks*) à l'intérieur des cycles d'horloge. Une première approche pour ce type de monitoring consiste à déployer des moniteurs en dehors de la partie fonctionnelle. Un exemple de solution est de répliquer et monitorer seulement certains chemins logiques déclarés comme critiques lors de la synthèse et de baser l'évaluation de l'état global du circuit sur ceux-ci [TKN+02]. Une autre solution est de déployer des structures uniquement dédiées au monitoring. Toutefois, avec ce type d'approche, la corrélation entre l'information du moniteur et l'état réel du circuit reste limitée [Sam03, NR06], et le test propre des moniteurs ou leur calibration peuvent être difficile à réaliser.

Une autre approche vise à monitorer directement les marges temporelles de certains signaux dans le circuit ciblé comme, par exemple, les signaux échantillonnés par certains événements séquentiels placés au bout de chemins critiques. Une violation des marges temporelles (ou *slack-time violation*) à l'entrée d'un élément séquentiel peut signifier qu'une erreur risque d'affecter le circuit car un des chemins logiques a vieilli de façon assez significative pour que son temps de traversée soit allongé au point de réduire la marge sous un certain seuil. Avec cette approche, si un monitoring avec une bonne granularité spatiale et temporelle peut être assuré, l'impact en surface et en performance doit être étudié attentivement tout comme l'impact sur le test du circuit. En fonction du seuil choisi, l'élément séquentiel pourra toujours capturer une donnée valide, le monitoring se contentant de donner une alerte indiquant que des dégradations commencent à devenir significatives et que des palliatifs doivent être déployés. Cela peut également être utilisé pour des aspects de performance et de consommation, comme présenté dans la Figure 1.9.

Dans la Figure 1.9, trois cas d'évolutions de la marge temporelle (partie grise) sont considérés :

(a) Cas de fonctionnement nominal.

(b) Cas de dégradation involontaire, le temps de traversée de la logique a augmenté à cause d'une dégradation (ex : vieillissement). La marge temporelle initiale est réduite par la gauche.

(c) Cas de dégradation intentionnelle pour améliorer la performance, en

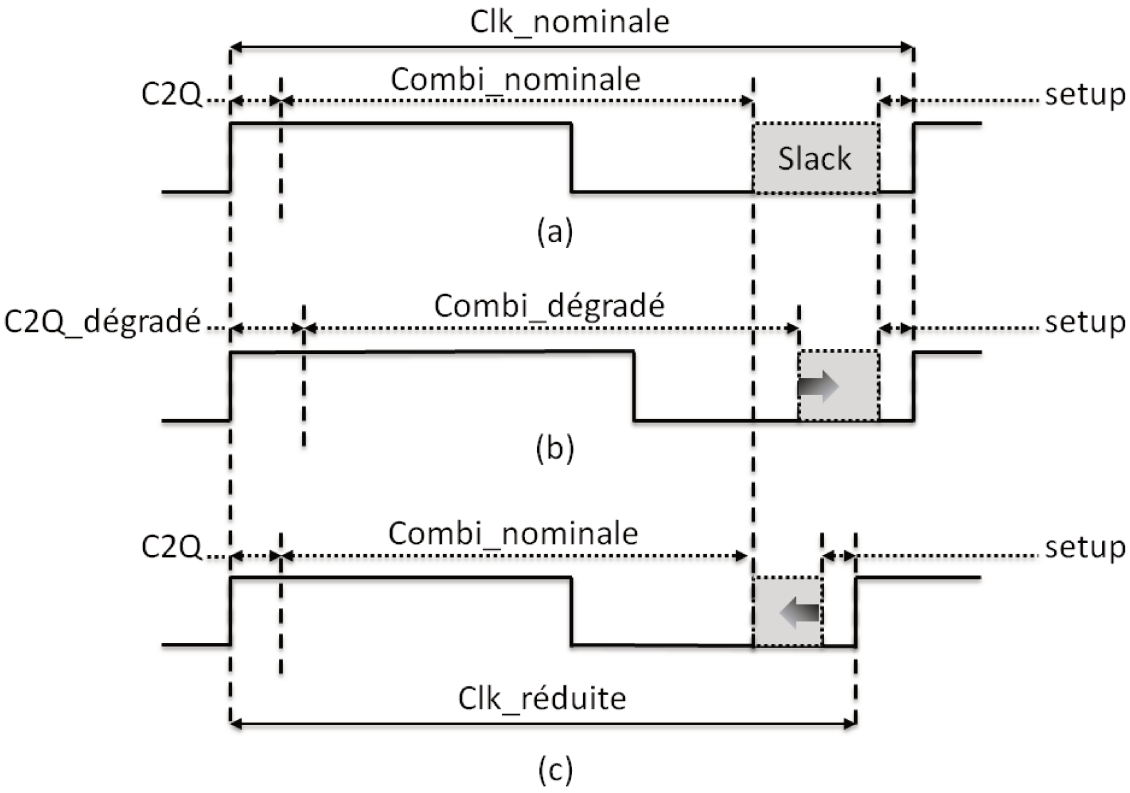


FIGURE 1.9 – Exemples d'évolutions des marges temporelles.

réduisant la période d'horloge. La marge temporelle est réduite par la droite.

Ainsi, si on s'assure à chaque instant que les marges temporelles restent supérieures à un certain seuil, il est possible de s'assurer que le circuit est toujours fiable et on peut dynamiquement adapter certains paramètres tels que la période d'horloge, ou la tension d'alimentation, afin de modifier ses performances jusqu'à atteindre la limite critique de fonctionnement [Keh93].

On peut distinguer deux approches de monitoring direct des marges temporelles, l'approche par détection de transition et l'approche par double échantillonnage.

IV.1 Monitoring direct des marges temporelles par détection de transitions

Cette approche consiste à surveiller et indiquer si une transition se produit au niveau du signal d'entrée d'un élément séquentiel dans la période du temps qui correspond à la marge temporelle de ce signal [Nic99, APZM07, BKL+08, BTK+09, RBB+11].

Une implémentation de cette technique peut être celle présentée en Figure 1.10 (a). Comme on le voit, le moniteur est composé de deux blocs : le bloc *Sensor (b.1)* qui fonctionne comme un détecteur de stabilité et sera implanté au plus proche de la bascule à monitorer et le bloc *Time Window Generator (b.2)* qui générera les signaux définissant la fenêtre de détection, et qui sera placé sur l'arbre d'horloge propre à la bascule considérée.

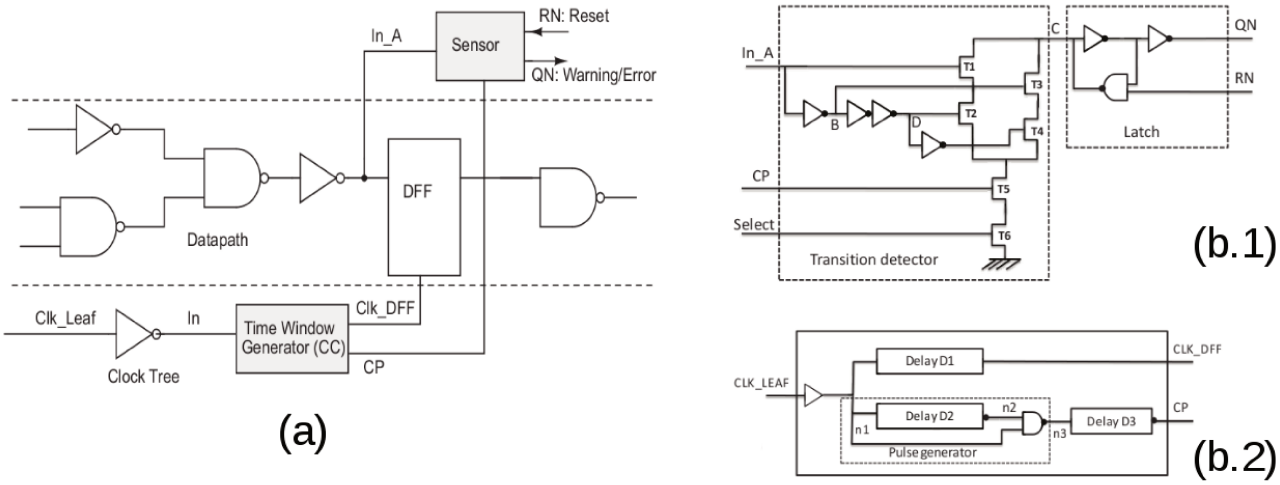


FIGURE 1.10 – Exemple de solution de détecteur de transitions (a) et détails d'implémentation du *Sensor* (b.1) et du *Time Window Generator* (b.2) [RBB+11].

Le détail d'implémentation du bloc *Sensor* est présenté en Figure 1.10 (b.1). Il est constitué de deux sous-blocs : le détecteur de transition et un *latch* servant à capturer le signal d'alerte. Quand une transition se produit sur l'entrée du détecteur *In_A*, elle est propagée au travers de chaînes d'inverseurs jusqu'aux groupes de transistors *T1-T4*. Ainsi, en fonction du front de la transition, soit le groupe *T1/T2*, soit le groupe *T3/T4* deviennent passants pendant un court instant. Si cela ce produit pendant que les signaux *CP* et *Select* sont actifs et cela

pendant assez longtemps pour permettre au nœud *C* de se décharger alors, une transition sera détectée et le signal *QN : Warnign/Error* prendra la valeur logique *0* jusqu'à ce que le moniteur soit réinitialisé grâce au signal *RN*.

La génération et le placement temporel de la fenêtre de détection est un point crucial dans ce type de solution. La Figure 1.10 (b.2) donne une vision basique de la structure permettant cela. Elle est constituée d'un générateur d'impulsion (*D2*) dont la largeur détermine la taille de la fenêtre de détection, et deux autres éléments de délais servant à placer la fenêtre de détection vis-à-vis du signal d'horloge.

Cette technique, bien que présentant un surcoût matériel qui peut être limité, a certains inconvénients notamment en termes de test. En effet, le test du détecteur peut être extrêmement difficile à réaliser puisqu'il nécessite de mettre en œuvre en entrée de la bascule des transitions temporellement compatibles avec la fenêtre de détection qui représente également un élément difficile à tester. De plus, la mise en œuvre des différents générateurs de fenêtre de détection peut présenter un fort impact sur la puissance dynamique à cause de leur commutation à chaque cycle d'horloge.

IV.2 Monitoring direct des marges temporelles par double échantillonnage

Une autre approche consiste à dupliquer le signal surveillé et à l'échantillonner avec des marges temporelles différentes [Nic99,EKD+03,APZM07,BTK+09]. En dehors de la bascule fonctionnelle qui doit échantillonner le signal surveillé, cette approche nécessite une bascule supplémentaire qui sera appelée bascule "*shadow*". La différence de marges temporelles entre les deux bascules peut être mise en œuvre soit par le biais du signal d'horloge, soit sur les chemins de propagation du signal surveillé. Une structure possible de la deuxième solution est représentée en Figure 1.11. La différence entre les marges temporelles des deux bascules est ici réalisée sur le chemin du signal surveillé grâce à un élément de délai (δ). Pour réaliser cet élément de délai, une solution peut être d'utiliser une simple chaîne d'inverseurs, ou de choisir un élément shadow avec un temps de *setup* plus long. Dans le cas où le signal *D* commence à violer la marge temporelle imposée par l'élément de délai δ, la bascule *shadow* va capturer une valeur erronée, différente de la bascule fonctionnelle dont la marge temporelle n'est

pas encore affectée. Cela aura pour conséquence d'activer le signal *warning* en sortie de la porte *ou-exclusif* utilisée pour la comparaison.

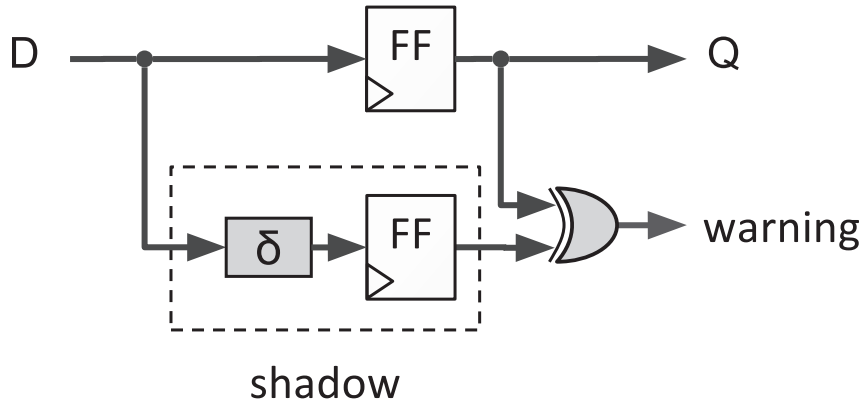


FIGURE 1.11 – Exemple d'implémentation d'un moniteur fonctionnant par double échantillonnage [Nic99].

Cette d'approche, intéressante d'un point de vue monitoring, présente toutefois un inconvénient en terme d'impact sur la latence si le test du moniteur doit être mis en œuvre. En effet, pour rendre plus facilement testable le moniteur, l'utilisation d'une approche *standard-scan* à la fois sur la bascule *shadow* et sur la bascule fonctionnelle peut faire en sorte que l'accumulation de latence de l'élément de délai et du multiplexeur de *scan* et dont il faudra tenir compte lors de l'évaluation de la latence du chemin fonctionnel pour éviter les alertes intempestives, soit trop importante.

Chapitre 2

SHADOW-SCAN À FAIBLE LATENCE ET AVEC MONITORING EN LIGNE DES MARGES TEMPORELLES

L'approche *standard-scan* est une approche communément employée pour traiter efficacement les problématiques liées au test pendant le processus de production. Néanmoins, elle présente un impact sur la latence car les bascules du circuit doivent être remplacées par des bascules *scan* qui sont plus lentes. De plus, elle ne prend pas en compte l'impact des variations du milieu de mission ni le vieillissement du circuit.

Historiquement, ces phénomènes étaient résolus par l'augmentation des marges, mais cette solution conservatrice devient trop chère pour être mise en oeuvre. Une solution pour limiter l'augmentation des marges et plus particulièrement des marges temporelles est le déploiement de solutions de monitoring en ligne.

Le test de production restant indispensable, il est alors intéressant de chercher à mutualiser les structures de test de production et de monitoring en ligne afin de limiter leurs impacts, tant en termes de surface, de consommation, de complexité, mais aussi de performance.

I Principe général de la solution proposée

Dans ce travail, nous considérons un contexte avec du monitoring et l'objectif est de tirer profit de l'approche *shadow-scan* proposée dans [Sin12] pour limiter l'impact du test sur la latence des chemins, tout en la combinant avec les approches "*Razor*" décrites dans [EKD+03] et [ABFM07] pour les aspects de monitoring en ligne des marges temporelles.

Nous mettrons en œuvre une architecture *shadow-scan* au sein de laquelle les transferts des données de test entre bascule *shadow* et bascule fonctionnelle ne se feront plus par l'intermédiaire d'un multiplexeur placé en aval de la bascule comme dans [Sin12], mais seront réalisés de manières asynchrone entre la bascule *shadow* pourvue de *scan-design* et la bascule fonctionnelle sans *scan*.

Afin de permettre la prédiction des violations temporelles et prévenir l'apparition d'un état métastable [1] avant le déclenchement d'une alerte, la bascule *shadow* devra présenter une probabilité supérieure de voir sa marge temporelle violée. Une solution pour implémenter cela est de choisir des bascules *shadow* avec un temps de *setup* plus long que celui des bascules fonctionnelles auxquelles elles sont associées. Ce sera normalement le cas dû au fait que les bascules *shadow* implémentent du *scan-design* alors que les bascules fonctionnelles non.

Cette approche sera appliquée à une bascule *scan* offrant une fonctionnalité de *reset* asynchrone et présentera donc l'avantage de par l'absence de multiplexeurs de test sur le chemin de données, d'avoir un impact plus réduit sur la latence, tout en autorisant un monitoring en ligne des marges temporelles et un test à la fois de l'infrastructure de monitoring mais aussi de la logique de mission.

1. État "inconnu" d'un nœud logique dont la valeur peut être interprétée comme un *0* ou un *1* logique, et dont le temps de stabilisation peut être infini. Cet état peut apparaître en sortie d'une bascule *D* si la donnée en entrée n'est pas stable pendant la fenêtre équivalente au temps de *setup*.

II Approche avec contrôlabilité partielle

La Figure 2.1 montre une implémentation d'une solution *shadow-scan* à très faible impact sur la latence avec monitoring des marges temporelles. Dans cette implémentation, l'impact sur la latence est plus faible car il est uniquement dû à la modification de charge appliquée en entrée et en sortie pour la bascule fonctionnelle.

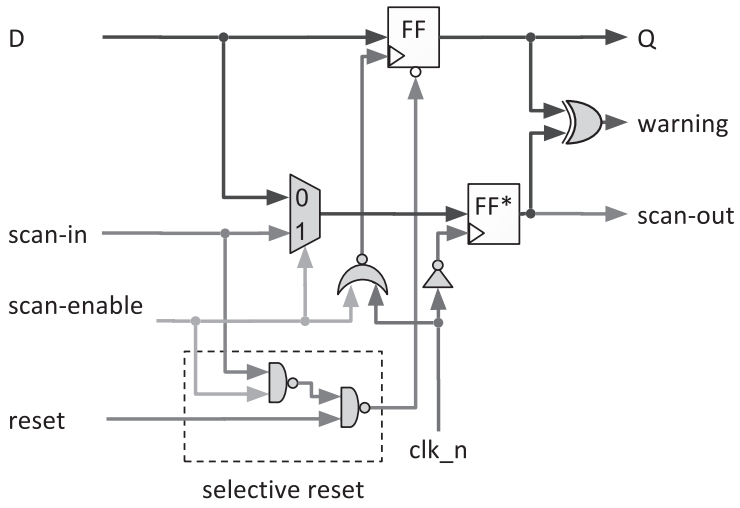


FIGURE 2.1 – Bascule *shadow-scan* à contrôlabilité partielle.

La bascule est constituée d'une bascule *scan shadow* (*FF**) placée en parallèle de la bascule fonctionnelle (*FF*), d'une porte *ou-exclusif*, d'une porte logique *non-ou*, ainsi que d'un bloc logique *selective reset* utilisé pendant la phase de test.

Ici, c'est le multiplexeur de test de la bascule *shadow* qui va servir d'élément de délai pour la double capture du signal *D* avec décalage temporel. La porte logique *ou-exclusif* permet de comparer les données capturées par les deux bascules et ainsi de détecter l'occurrence d'une violation de temps de *setup*.

En mission, ou en mode *capture* pendant le test, le signal *scan-enable* est à *0*, et ainsi les deux bascules capturent la même donnée en l'absence de violation du temps de *setup*. En cas de diminution de la marge temporelle, la bascule *shadow* est supposée être la première à capturer une donnée erronée, différente de la donnée capturée par la bascule fonctionnelle, déclenchant ainsi le signal *warning* en sortie de la porte *ou-exclusif*.

En mode *scan*, la bascule *shadow* autorise l'observation de la logique du circuit sous test, comme dans une approche *standard-scan* classique. Le signal *scan-enable* étant positionné à *1*, le signal d'horloge de la bascule fonctionnelle se retrouve bloqué par la porte *non-ou*, en considérant que la bascule fonctionnelle est active sur front montant. Ainsi, dans cette configuration, la bascule fonctionnelle peut être contrôlée à *0* grâce à l'opération du *selective-reset* qui prendra place au dernier cycle d'horloge en mode *scan*.

Comme présenté sur la Figure 2.1, le *selective-reset* est réalisé grâce à deux portes *non-et*, pilotées par les signaux *scan-enable*, *reset* et *scan-in* en considérant que le signal *reset* des bascules est actif au niveau *1*. Cette structure a pour but de permettre d'initialiser la bascule fonctionnelle en mode *scan* si un *0* doit être chargé dans la bascule fonctionnelle. Ainsi, la valeur *0* passée par *scan-in* se retrouvera bien appliquée en sortie de la bascule fonctionnelle.

Néanmoins, la contrôlabilité à *1* de la bascule fonctionnelle est limitée puisque celle-ci n'est pas directement insérée dans une chaîne de *scan* et ne dispose pas de capacité de *set*. En effet, cette contrôlabilité ne peut être assurée que par l'utilisation de vecteurs de test calculés pour des circuits séquentiels, et avec pour objectif de placer un *1* logique en sortie du bloc combinatoire pilotant l'entrée *D* de la bascule *shadow-scan* en Figure 2.1.

De la même manière, pour des bascules avec *set* asynchrone on peut assurer une contrôlabilité à *1*. Dans le cas des bascules avec *set* et *reset* asynchrones, on peut assurer une contrôlabilité complète (*i.e.* à *0* et *1*). Ce sujet sera développé dans une autre sous-section de ce chapitre.

II.1 Estimation de l'impact sur la latence

Afin d'estimer le gain possible en latence au niveau de la bascule, nous avons comparé les temps de *setup* (T_{setup}) et de *clock-to-Q* (T_{C2Q}) des deux différentes bascules avec et sans capacité de *standard-scan*. C'est ce qui est présenté dans le Tableau 2.1 pour une implémentation de la bascule basée sur une bibliothèque de la technologie 40nm LP de la société TSMC. Ces résultats ont été obtenus en considérant un process *Slow-Slow*, avec une tension d'alimentation de 0.99V, une température de 125°C, et une charge en sortie de bascule C = 0.009pF.

Comme on peut l'observer, bien que les temps de *clock-to-Q* puissent être supérieurs à la version sans *scan*, les temps de *setup* sont plus de trois fois inférieurs pour cette version de la bascule. Ainsi, comme l'on pouvait s'y attendre, la latence plus importante des bascules avec *scan* est due au multiplexeur placé en amont de la bascule et affecte ainsi le temps de *setup*. L'utilisation d'une bascule sans *scan* devrait donc, d'après les résultats du Tableau 2.1, idéalement permettre d'obtenir des gains en latence de l'ordre de 12%.

Des gains en latence supérieurs peuvent être espérés si l'on considère la conception optimisée aux niveaux *layout* et transistor de la bascule.

TABLEAU 2.1 – Évaluation des latences des deux types de bascules D considérées.

bascules D avec *reset* asynchrone pour l'implémentation de la Figure 2.1	front montant [ps]		front descendant [ps]		max
	T_{setup}	T_{C2Q}	T_{setup}	T_{C2Q}	(T_{setup}+T_{C2Q})
sans *scan* (DFCND1BWPLVT) utilisée comme bascule fonctionnelle	26.6	235.3	-2.5	229.5	261.9
avec *scan* (SDFCND1BWPLVT) utilisée comme bascule *shadow*	71.4	215.7	53.5	225.9	297.3
				gain en latence	-11.9%

II.2 Impact sur la couverture de fautes

Afin d'évaluer l'impact de la contrôlabilité partielle sur la testabilité d'un circuit, nous avons considéré les circuits *benchmarks ITC'99* [ITC] pour lesquels nous avons déployé des bascules *shadow-scan* à contrôlabilité partielle sur un certain pourcentage des bascules du circuit. L'outil de génération de vecteurs de test TetraMax de Synopsys a été utilisé pour évaluer la couverture de fautes des différents *benchmarks*.

Les outils d'*ATPG* du marché n'étant pas capables de gérer directement la bascule *shadow-scan*, il a été nécessaire de modéliser celle-ci d'un point de vue "test" afin de réaliser l'évaluation. Le modèle utilisé pour cette expérimentation est celui présenté en Figure 2.2.

Dans ce modèle, deux points d'observabilité sont connectés aux entrées/sorties de chaque bascule afin d'émuler la visibilité de la bascule *shadow* placée dans une chaîne de *scan*. De même, afin d'émuler le *selective-reset*, un point de contrôlabilité à *0* est réalisé grâce à la porte *et* à deux entrées, en combinant la valeur de sortie de la bascule avec la valeur d'une *PI* supplémentaire.

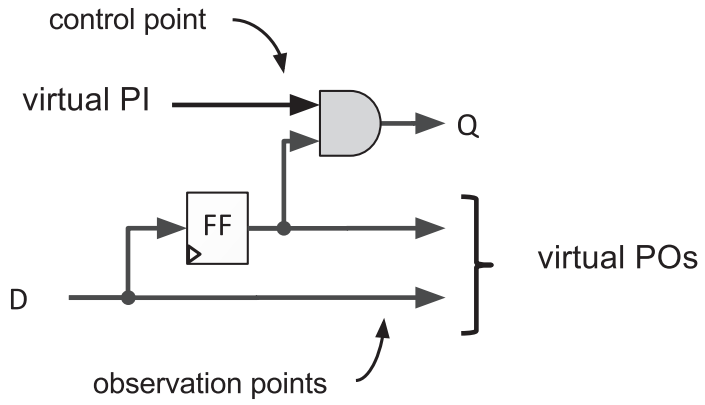


FIGURE 2.2 – Modèle des bascules *shadow-scan* en Figure 2.1 utilisé pour évaluer leur impact sur la testabilité des circuits.

Dans un *DUT*, chaque bascule qui devrait être remplacée par une *shadow-scan* est remplacée par la structure de la Figure 2.2, alors que chaque bascule *standard-scan* est remplacée par une entrée et une sortie primaire afin de permettre le maintien de l'état de la bascule pendant le *scan*. Ensuite, une génération de vecteurs de test séquentiels est réalisée et évaluée à l'aide de TetraMax.

Afin d'évaluer l'impact sur la couverture de fautes, nous avons considéré trois versions de chaque circuit :

(a) Version originale du circuit, où aucune bascules ne dispose de *scan*.
(b) Version mixte avec 10% de *shadow-scan* et 90% de *standard-scan*.
(c) Version 100% *standard-scan*, où toutes les bascules sont *standard-scan*.

Le Tableau 2.2 présente les résultats de couverture de fautes (*FC*) pour les fautes de collage, ainsi que le nombre de vecteurs nécessaires pour obtenir ces résultats.

Comme on peut l'observer, pour la plupart des circuits, la version mixte présente une couverture de fautes bien en deçà de celle obtenue pour une version *standard-scan*, et cela même avec un accroissement du nombre de vecteurs de test. Ainsi pour six des circuits, la couverture de fautes est réduite de 1 à 3 %. Pour le reste des circuits, et à l'exception du circuit *b06* pour lequel aucune modification n'a été observée, la détérioration de la couverture de fautes évolue entre 6 et 9%. Si l'on choisit de déployer cette solution *shadow-scan* non plus sur 10%, mais sur 20% des bascules du circuit, la dégradation moyenne de couverture de fautes sera alors de l'ordre de 10.5%.

TABLEAU 2.2 – Couverture de fautes pour les différentes versions des *benchmarks*.

circuit	version original sans *scan*		version mixte (10% *shadow*)		version *standard-scan*	
	nombre de vecteurs	*FC*	nombre de vecteurs	*FC*	nombre de vecteurs	*FC*
b01	151	85%	40	97% (-3%)	16	100%
b02	58	94%	31	97% (-3%)	12	100%
b03	69	52%	52	93% (-7%)	33	100%
b04	366	83%	86	97% (-1%)	62	98%
b05	15	9%	138	94% (-3%)	97	97%
b06	76	92%	39	99% (0%)	15	99%
b07	7	3%	126	98% (-2%)	79	100%
b08	18	3%	103	94% (-6%)	60	100%
b09	9	4%	74	94% (-6%)	45	100%
b10	118	80%	149	93% (-7%)	52	100%
b11	108	60%	284	94% (-6%)	103	100%
b12	20	12%	235	92% (-8%)	198	100%
b13	59	21%	134	91% (-9%)	60	100%
b14	1631	55%	1290	97% (-2%)	833	99%

II.3 Conclusion

Bien que présentant un réel intérêt en termes d'impact sur la latence, la dégradation de la couverture de fautes engendrée par cette solution est trop importante pour qu'elle soit réellement exploitable. De même, comme on le voit sur la Figure 2.1, dans cette implémentation de la solution, le signal *scan-out* ne s'effectue pas sur la sortie *Q*, mais sur la sortie de la bascule *shadow*. Cela doit donc être pris en compte lors de l'implantation des bascules, et impose de réitérer certaines opérations de raccordement ("*stitching*") des éléments constituant les chaînes de *scan* ce qui allonge et complexifie le travail de *DFT*.

Pour pallier la perte de contrôlabilité, ainsi que pour faciliter l'implantation, une seconde version de l'architecture de la bascule a été recherchée.

III Approche avec contrôlabilité totale

La seconde implémentation d'une solution *shadow-scan* à faible impact sur la latence avec monitoring en ligne des marges temporelles est celle proposée en Figure 2.3.

Comparée à l'implémentation précédente, elle autorise une contrôlabilité totale et la bascule est directement interchangeable avec une bascule *standard-scan* du circuit puisque elle dispose des mêmes interfaces.

III.1 Mise en œuvre et principe de fonctionnement de la bascule *shadow-scan*

Afin d'atteindre la contrôlabilité totale, la bascule fonctionnelle de la Figure 2.1 est remplacée par une bascule avec du *set*/*reset* asynchrone comme illustré en Figure 2.3.

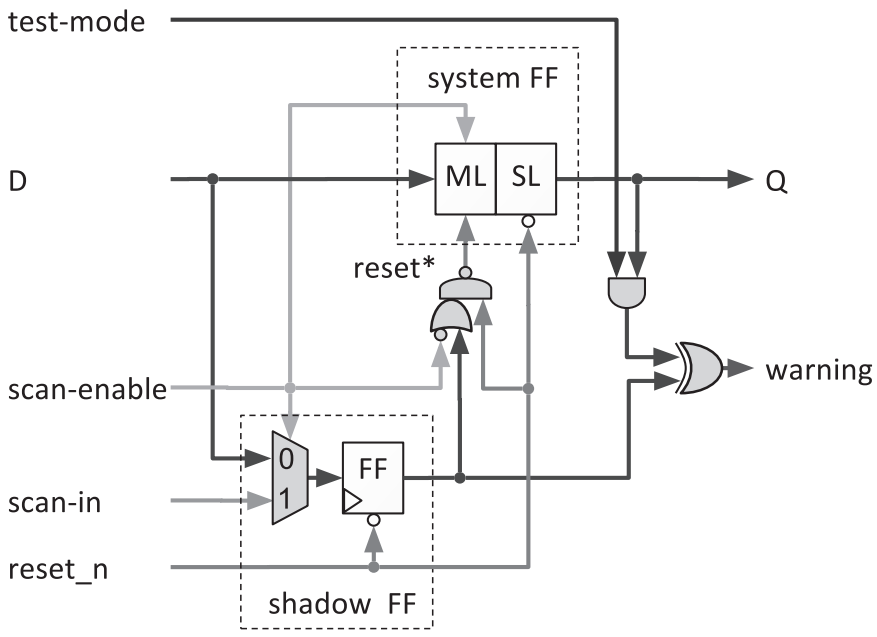


FIGURE 2.3 – Bascule *shadow-scan* à contrôlabilité totale.

Pour piloter le *reset* de la nouvelle bascule fonctionnelle, un groupement de portes logiques (*non-ou-et-inversée*) est utilisé de manière à ce que la valeur scannée dans la bascule *shadow* puisse contrôler le *reset* de la bascule fonctionnelle de façon transparente vis-à-vis du *reset_n* global de la bascule. Enfin, comme

cela sera montré plus loin dans cette section, une porte *et* et un signal de test ont également été ajoutés pour faciliter le test de la porte *ou-exclusif* utilisée pour le monitoring.

Durant la phase de mission, le signal *scan-enable* est maintenu à *0*, alors que le signal *test-mode* est maintenu à *1*. En l'absence de défauts de production ou de violations du temps de *setup*, les bascules fonctionnelle et *shadow* doivent capturer la même valeur.

Néanmoins, et une fois encore, la bascule *shadow* est supposée être la première à être affectée par une violation du temps de *setup*, à cause du délai introduit par le multiplexeur servant au *scan*. En cas de différence entre les données capturées par la bascule fonctionnelle et la bascule *shadow* associée, le signal $warning$ est déclenché et des contre-mesures peuvent être mises en place au niveau système, avant qu'une erreur n'apparaisse et ne se propage dans le reste du circuit.

Durant la phase de test, la bascule *shadow* permet un transfert asynchrone transparent à la bascule fonctionnelle de la valeur transférée sur l'entrée *scan-in*. En mode *scan*, le signal *scan-enable* étant positionné à *1*, le *master-latch* (*ML*) de la bascule fonctionnelle est automatiquement mis à *1* de façon asynchrone. Celui-ci peut également être mis à *0* de façon asynchrone dès qu'un *0* logique est scanné dans la bascule *shadow*.

III.1.i Amélioration de la latence du *master-latch* de la bascule fonctionnelle

La latence du *master-latch* de la bascule fonctionnelle peut être améliorée si la fonctionnalité de *set* est implémentée en dehors du chemin de données comme proposé dans la Figure 2.4. Une approche similaire a été proposée dans [MABF07].

Dans la solution de [MABF07], la porte *non-ou* supérieure, servant à la fonctionnalité de *set* est remplacée par un multiplexeur *scan* dont la sortie est *inversée*, et dont l'entrée *scan-in* est pilotée par le *shadow-latch*. En ce qui concerne le *slave-latch* (*SL*) de la bascule fonctionnelle, celui-ci peut être réalisé sans capacité de *set* tant que cette fonctionnalité n'est pas nécessaire au bon fonctionnement du circuit.

Avec le *master-latch* présenté en Figure 2.4, la sortie *Q* de la bascule proposée en Figure 2.3 peut être affectée par un *glitch*[2] si une valeur *1*, pendant au moins deux cycles successifs d'horloge, est capturée dans la bascule fonctionnelle alors que la logique du système applique une valeur logique *0* en *D*.

Des *glitchs* similaires peuvent également affecter la solution proposée par [MABF07], notamment quand les signaux *scan-in* et *D* présentent des valeurs différentes, le signal *scan-in* étant constant pendant au moins deux cycles successifs d'horloge.

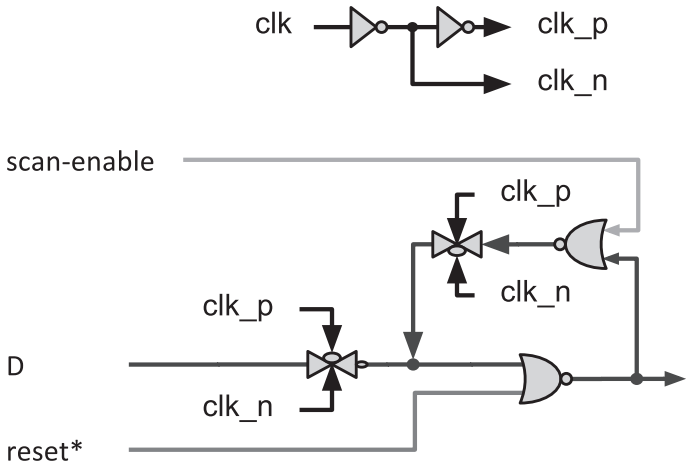


FIGURE 2.4 – Détail du *master-latch* (*ML*) amélioré de la bascule fonctionnelle.

Avec la solution proposée ici, ces *glitchs* peuvent toutefois être éliminés si la capacité de *set* est implémentée directement sur le chemin de données. Pour cela, la porte *non-ou* supérieure de la solution en Figure 2.4 peut être remplacée par un simple *inverseur*, alors que la porte *non-ou* inférieure est remplacée par une porte *et-ou-inversée*, contrôlée à la fois par le signal *reset** et par une version *inversée* du signal *scan-enable*. La mise en œuvre du *reset* directement sur le chemin de données présente elle l'intérêt d'éliminer l'apparition de *glitchs* pendant la phase de *reset*.

En mode *capture*, les *glitchs* en sortie de la bascule *shadow* sont masqués de façon logique par la porte *ou-et-inversée* de la Figure 2.3. De même, les *glitchs* sur le signal *scan-enable* sont filtrés électriquement par la capacité relativement élevée du réseau acheminant ce signal.

2. Transition indésirable qui se produit avant que le signal ne converge à sa valeur prévue.

L'opération de *scan* pouvant être réalisée à une fréquence relativement faible, l'ensemble de la logique autour de la bascule fonctionnelle de la Figure 2.3 peut être mis en œuvre avec des transistors de taille réduite. Cette réduction de taille ne peut toutefois pas concerner les transistors commandant le signal *warning* puisqu'ils doivent assurer une transmission rapide de ce dernier.

III.1.ii Évaluation de l'impact sur la latence au niveau bascule

Comme pour la bascule de la Figure 2.1, une première évaluation de l'évolution de l'impact en latence a été réalisée. L'évaluation a été calculée pour la même technologie (TSMC 40nm LP) et dans les mêmes conditions, c'est-à-dire en considérant un process *Slow-Slow*, avec une tension d'alimentation de 0.99V, une température de 125°C, et une charge en sortie de bascule C = 0.009pF.

Le Tableau 2.3 présente la synthèse des résultats de l'évaluation. L'utilisation d'une bascule *set/reset* sans *scan* devrait idéalement permettre d'obtenir des gains en latence de l'ordre de 6% au niveau bascule comparé à du *standard-scan*.

TABLEAU 2.3 – Évaluation des latences des deux types de bascules considérées.

type de bascules pour l'implémentation la Figure 2.3	front montant [ps]		front descendant [ps]		max
	T_{setup}	T_{C2Q}	T_{setup}	T_{C2Q}	(T_{setup}+T_{C2Q})
set reset asynchrone sans *scan* (DFCSND1BWPLVT) utilisée pour implémenter la bascule fonctionnelle	21.9	225.3	15.9	257.2	261.9
reset asynchrone avec *scan* (SDFCND1BWPLVT) utilisée pour implémenter la bascule *shadow*	71.4	215.7	53.5	225.9	297.3
				gains en latence	-6.1%

III.2 Collecte des signaux *warning* et intérêt du signal *test-mode*

Dans un module ou sous-système avec sa propre gestion de la fréquence d'horloge et tension d'alimentation, les signaux *warning* provenant des différentes bascules avec moniteur de marges temporelles peuvent être combinés à l'aide d'un arbre de *ou* comme montré sur la Figure 2.5, afin d'être regroupés et traités comme un signal unique. Si le nombre de signaux *warning* est conséquent et si l'infrastructure de monitoring doit fonctionner à la même fréquence

d'horloge que la logique de mission, il est alors nécessaire de "*pipeliner*" l'arbre de *ou*.

Dans la solution d'arbre proposée en Figure 2.5, le signal *global-warning* est positionné à *1*, et cela tant qu'un *warning* perdure et que le signal d'acquittement *acknowledge* n'a pas été activé.

Le signal *test-mode* de la Figure 2.3, commun à tous les moniteurs, est utilisé afin de faciliter le test de la faute collage à *0* (*s@0*) affectant l'arbre de *ou*. En l'absence de ce signal, cette faute est très difficile à tester puisque le signal *warning* de chaque bascule est à *0* si il n'y a pas de violation du temps de *setup*.

Quand la valeur *0* logique est appliquée au signal *test-mode*, le signal *warning* de chaque bascule devient alors dépendant de la valeur scannée dans la bascule *shadow* qui lui est associée. Cela facilite grandement le test de l'arbre qui peut alors être réalisé avec une procédure type *walkging-1* [Bar99, Ada02].

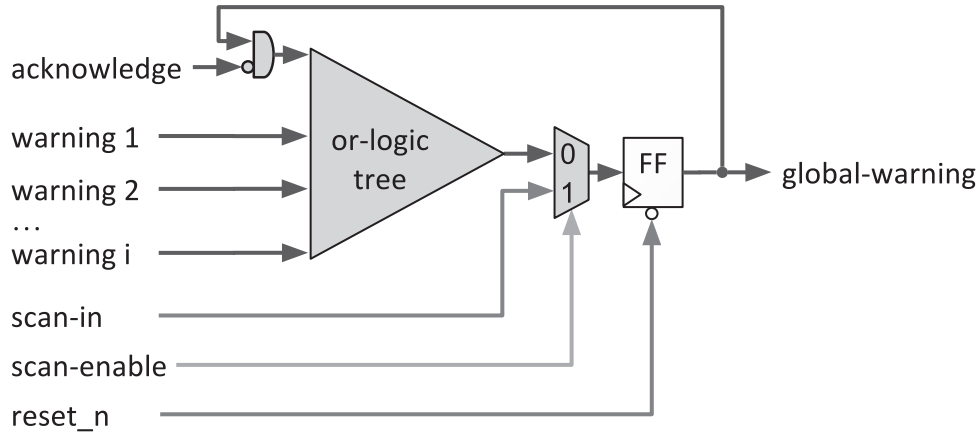


FIGURE 2.5 – Collecte des signaux *warning*.

III.3 Modèle équivalent pour les outils de *DFT* et d'*ATPG*

Afin que la bascule soit exploitable de façon transparente par les outils actuels du marché de la *DFT* et de l'*ATPG*, elle doit être représentée par des modèles intelligibles du point de vu de ces outils. Ces modèles, indépendants du *DUT*, sont décrits de façon à permettre aux outils de câbler les chaînes de *scan*, de calculer les vecteurs de test, ainsi que de calculer les évaluations de la latence, de la surface et de la consommation de la solution. Ces modèles sont pour le moment indispensables car la bascule est implémentée au niveau porte, mais si

celle-ci était directement intégrée dans une bibliothèque de conception *standard-cell*, l'utilisation de modèles externes aux outils ne serait plus nécessaire.

La Figure 2.6 présente le modèle retenu pour l'*ATPG* et le câblage des chaînes de *scan* associé à la bascule de la Figure 2.3. Pour l'évaluation en surface, latence et consommation, l'implémentation utilisée est celle présentée en Figure 2.3.

Comme on peut le voir sur la Figure 2.6, le modèle est uniquement constitué d'une bascule *scan*, de la porte *ou-exclusif* et de la porte *et* servant à en faciliter le test de l'arbre de *ou* en Figure 2.5. On peut se permettre cette modélisation, car la bascule *shadow-scan* fonctionne exactement comme une bascule *standard-scan*. De plus, la plupart des fautes présentes sur les nœuds internes de la bascule en Figure 2.3 sont implicitement testables.

Si l'on considère par exemple les fautes de collage du nœud entre la bascule *shadow* et la porte *ou-et-inversée* en Figure 2.3, ou les fautes de collage sur l'entrée contrôlée par le signal *scan-enable* du *master-latch* de la bascule fonctionnelle, elles peuvent être implicitement testées comme des fautes de la chaîne de *scan*.

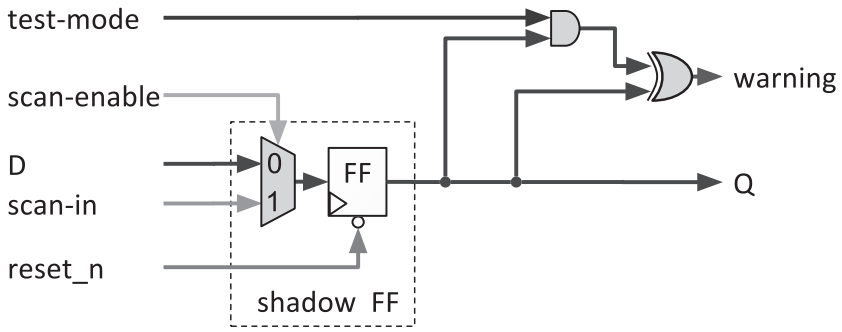


FIGURE 2.6 – Modèle équivalent pour la *DFT* et l'*ATPG*.

III.4 Évaluation de la solution

Afin d'évaluer les performances de la bascule présentée en Figure 2.3, il a été choisi de la comparer à une solution de l'état de l'art proposant à la fois du monitoring des marges temporelles, et du *standard-scan*. Cette comparaison a été faite à la fois en termes de couverture de fautes et d'impact sur la latence.

Dans cette évaluation, nous avons choisi de déployer des moniteurs sur l'ensemble des chemins des différents circuits du *benchmark ITC'99* après les avoir synthétisés à l'aide de l'outils Synopsis DesignCompiler de façon itérative pour obtenir les meilleures performances en termes de latence.

L'insertion de chaînes de *scan* a été réalisée avec l'aide de l'outil DFTCompiler de la société Synopsys, alors que le calcul des vecteurs de test a été réalisé avec l'outil TetraMax de la même société.

La Figure 2.7 présente l'architecture de la bascule de type "*Razor*" testable qui sera utilisée comme référence. Cette bascule est basée sur l'architecture présentée dans le chapitre consacré à l'état de l'art, en section IV.2. Elle a été rendue testable en utilisant une bascule fonctionnelle et une bascule *shadow* implémentant toutes les deux du *standard-scan*. Afin de maintenir les capacités de monitoring par double échantillonnage, un délai supplémentaire (δ) a dû être inséré en amont de la bascule *shadow*. Ici l'implémentation de l'élément de délais δ est réalisée par une chaîne de deux inverseurs.

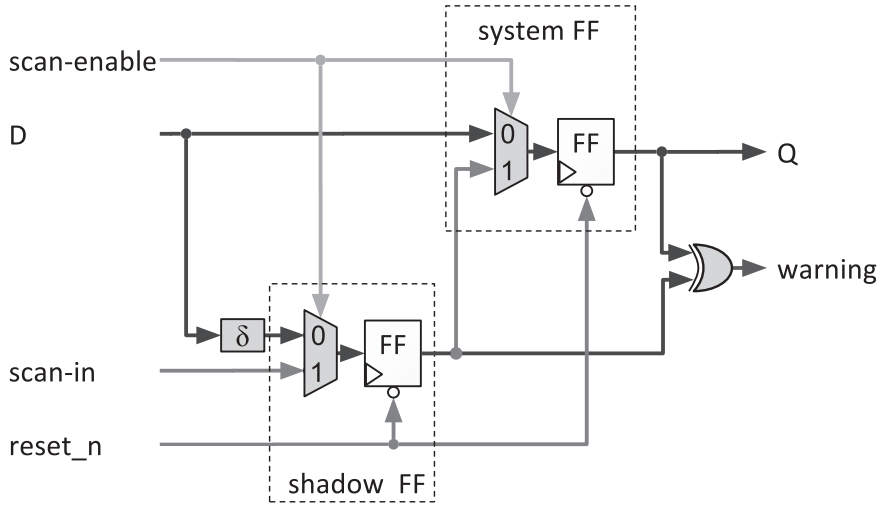


FIGURE 2.7 – Bascule type *Razor* testable.

III.4.i Impact sur la latence et sur la surface au niveau circuit

Le Tableau 2.4 présente l'impact de la solution *shadow-scan* de la Figure 2.3 sur la latence et la surface d'un circuit, vis-à-vis de l'impact de la solution de référence de la Figure 2.7.

Comme on peut l'observer, la solution *shadow-scan* de la Figure 2.3 présente toujours un impact inférieur en terme de latence. Le gain obtenu varie de 1.2% à 8.5%, pour un impact en surface entre 2.6% et 5.3% vis-à-vis de la solution de référence. Ce gain peut simplement s'expliquer par les temps de *setup* réduits de la bascule fonctionnelle utilisée dans la Figure 2.3, comme cela a été présenté dans le Tableau 2.3.

Ces résultats, présentés en Tableau 2.4, correspondent à une implémentation au niveau portes des bascules en Figure 2.3 et 2.7. Des résultats plus précis et réalistes pourraient être obtenus par l'implémentation de bascules optimisées au niveau transistor. En effet, la bibliothèque utilisée pour cette expérimentation ne nous a pas permis d'utiliser une bascule *master-slave* avec du *set* implémenté uniquement sur la partie *master-latch* comme présenté dans la Figure 2.3. De même, certains signaux internes tel que le signal *scan-enable-inversé* ont dû être recréés de manières redondantes, alors que leur génération pourrait être optimisée dans une version *custom-design*.

III.4.ii Impact sur la couverture de fautes

Le Tableau 2.5 rapporte les résultats en termes de couverture de fautes de collage, ainsi que le nombre de vecteurs nécessaire pour atteindre cette couverture de fautes. Comme on le remarque dans la deuxième et la quatrième colonne, des couvertures de fautes similaires sont atteignables par les deux types de bascules *scan* considérées.

On s'aperçoit néanmoins que l'approche *shadow-scan* de la Figure 2.3 nécessite souvent un nombre supplémentaire de vecteurs de test. Cette différence est due à la difficulté d'assigner des valeurs différentes entre les bascules fonctionnelles et *shadow* des bascules *shadow-scan* afin de tester l'arbre de *ou* et la logique de mission d'une façon complètement indépendante, ce qui n'est pas le cas dans l'implémentation de la Figure 2.7 où le test de tous les moniteurs peut-être parrallélisé.

La colonne Δvecteurs du Tableau 2.5 présente le ratio de vecteurs de test supplémentaires requis par l'approche *shadow-scan* vis-à-vis du fan-in du plus large cône logique du premier étage de l'arbre de *ou* de la Figure 2.5 en cas de *pipelining* de l'arbre. Pour la majeure partie des circuits, Δvecteurs est inférieur à 1,

TABLEAU 2.4 – Impact en latence et en surface au niveau circuit.

circuit	nombre de bascules modifiées	impact en latence	impact en surface
b01	5	-2.8%	3.5%
b02	4	-2.7%	4.1%
b03	30	-8.2%	5.2%
b04	108	-3.3%	5.2%
b05	34	-2.4%	3.1%
b06	8	-5%	4.4%
b07	46	-5.4%	3.6%
b08	41	-3.5%	5.3%
b09	28	-5.4%	5.2%
b10	17	-3.8%	3.8%
b11	30	-3.2%	2.6%
b12	119	-3.1%	4.6%
b13	45	-8.5%	5.1%
b14	380	-1.7%	3.6%
b15	451	-2.9%	3.9%
b17	1419	-2.9%	4.1%
b18	3479	-1.2%	4.6%
b19	6959	-2.1%	4.8%
b20	796	-1.7%	3.6%
b21	796	-1.7%	3.5%
b22	1194	-2.4%	3.8%

ce qui montre que le nombre de vecteurs de test supplémentaires requis par la solution *shadow-can* est limité par la taille du plus grand cône logique du premier étage du *pipeline* de l'arbre de *ou*. Cela est dû au fait que des bascules avec du *standard-scan* peuvent être utilisées pour *pipeliner* l'arbre de *ou* de la Figure 2.5 et ainsi faciliter son test.

Dans notre cas d'étude, la taille du cône logique le plus grand du premier étage du *pipeline* de l'arbre de *ou* (*Fan-in*) a été fixée à 1024 entrées, et le même *pipelines* ont été déployés pour les deux approches considérées.

TABLEAU 2.5 – Impact sur la longueur de test et la couverture de fautes pour les fautes de collage.

circuit	*standard-scan*		*shadow-scan* mixte			*Fan-in* max 1er étage arbre de *ou*
	nombre de vecteurs	*FC*	nombre de vecteurs	Δvecteurs	*FC*	
b01	18	98.7%	17	-0.2	99.2%	5
b02	15	98.0%	14	-0.25	98.8%	4
b03	46	99.6%	54	0.27	99.8%	30
b04	122	98.7%	168	0.43	98.7%	108
b05	92	99.3%	116	0.71	99.3%	34
b06	18	99.0%	19	0.13	99.4%	8
b07	119	99.8%	154	0.76	99.9%	46
b08	70	99.7%	90	0.49	99.8%	41
b09	52	99.5%	60	0.29	99.7%	28
b10	52	99.4%	57	0.29	99.6%	17
b11	147	99.9%	166	0.63	99.9%	30
b12	141	99.8%	221	0.67	99.9%	119
b13	70	99.7%	83	0.29	99.8%	45
b14	550	99.9%	882	0.83	99.9%	398
b15	699	99.8%	1077	0.84	99.8%	451
b17	1037	99.9%	1747	0.69	99.9%	1024
b18	1052	99.8%	2227	1.15	99.8%	1024
b19	1045	99.8%	1787	0.72	99.8%	1024
b20	830	99.8%	1218	0.49	99.8%	796
b21	819	99.9%	1275	0.57	99.9%	796
b22	1069	99.9%	1464	0.39	99.9%	1024

III.5 Conclusion

Une solution *shadow-scan* a été proposée de façon à implémenter une architecture de test ayant un faible impact sur la latence et étant compatible avec du *standard-scan*, tout en permettant un monitoring en ligne des marges temporelles. Les latences des circuits *benchmarks ITC'99* ont pu être réduites jusqu'à plus de 8% par l'utilisation de bascules *shadow-scan* comparées à l'utilisation de bascule type *Razor* implémentant du *standard-scan*.

Il a été montré que la solution *shadow-scan* ne présente pas d'impact sur le flot, et peut facilement être déployée pour des outils de *DFT* et d'*ATPG* existants. De même, il a été montré que la solution n'a pas d'impact négatif sur la couverture de fautes et que les outils d'*ATPG* et de *DFT* du commerce sont capables de générer des vecteurs permettant de tester aussi bien la logique de mission que l'architecture de monitoring, pour une surcoût limité vis-à-vis de la longueur de la séquence de test.

Chapitre 3

SHADOW-SCAN À FAIBLE LATENCE SANS MONITORING EN LIGNE DES MARGES TEMPORELLES

Ce chapitre explore l'utilisation de la structure *shadow-scan* originalement développée pour le monitoring en ligne en tant que solution pour le test de production à faible impact sur la latence. Comme dans le chapitre précédent, la solution proposée sera appliquée à une bascule *scan* offrant une fonctionnalité de *reset* asynchrone.

I Implémentation d'une bascule *shadow-scan* à faible impact sur la latence

I.1 Mise en œuvre au niveau portes logiques de la bascule *shadow-scan* sans monitoring

La Figure 3.1 présente une version de la bascule *shadow-scan* de la Figure 2.3 sans capacité de monitoring. Dans cette implémentation, la bascule *shadow* de la Figure 2.3 a été remplacée par un *shadow-latch*. La taille de ce *shadow-latch* peut être drastiquement réduite, puisqu'il n'est utile qu'en mode *scan*, à des fréquences d'horloge relativement faibles.

La bascule fonctionnelle reste constituée d'un *master-latch* (*ML*) avec *set*/*reset* asynchrone et d'un *slave-latch* (*SL*) implémentant uniquement du *reset* asynchrone comme dans la solution de la Figure 2.3. La fonctionnalité de *set*/*reset*

de la bascule fonctionnelle est implémentée de la même manière que pour la solution avec monitoring. La fonctionnalité de *set* du *master-latch* de la bascule fonctionnelle est directement commandée par le signal *scan-enable*. En ce qui concerne le *reset* du *master-latch*, il est commandé, comme dans le cas de la bascule *shadow-scan* avec monitoring par la porte logique *non-ou-et-inversée*. Cette structure logique est utilisée de manière à ce que la valeur scannée dans le *shadow-latch* puisse contrôler le *reset* de la bascule fonctionnelle de façon transparente vis-à-vis du *reset_n* global de la bascule.

En mode *scan*, la bascule *shadow-scan* est équivalente à une bascule *standard-scan*. La donnée de test peut être transférée de façon asynchrone depuis le *shadow-latch* jusqu'au *master-latch* de la bascule fonctionnelle, et cela avant que le *slave-latch* ne devienne transparent car le *shadow-latch* est transparent pendant le même niveau de l'horloge que le *master-latch* de la bascule fonctionnelle.

En mode capture, le signal *scan-enable* prend la valeur logique *0* ce qui élimine l'influence du *shadow-latch* sur le fonctionnement de la bascule fonctionnelle. Par conséquent, la bascule *shadow-scan* se comporte comme une bascule avec *reset* asynchrone.

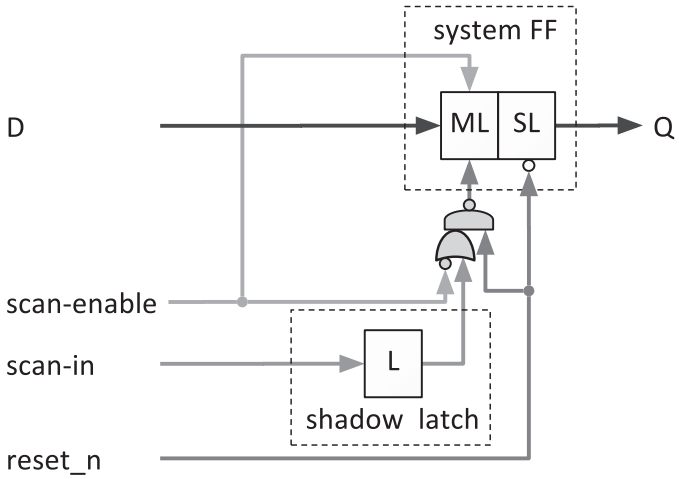


FIGURE 3.1 – Bascule *shadow-scan* pour le test uniquement.

La solution de *shadow-scan* présentée en Figure 3.1 peut être adaptée à un circuit utilisant des *latchs* pour la partie système. Une implémentation possible est celle montrée en Figure 3.2. Le *shadow-latch* original à été remplacé par une bascule *shadow* assurant un fonctionnement équivalent en mode *scan*.

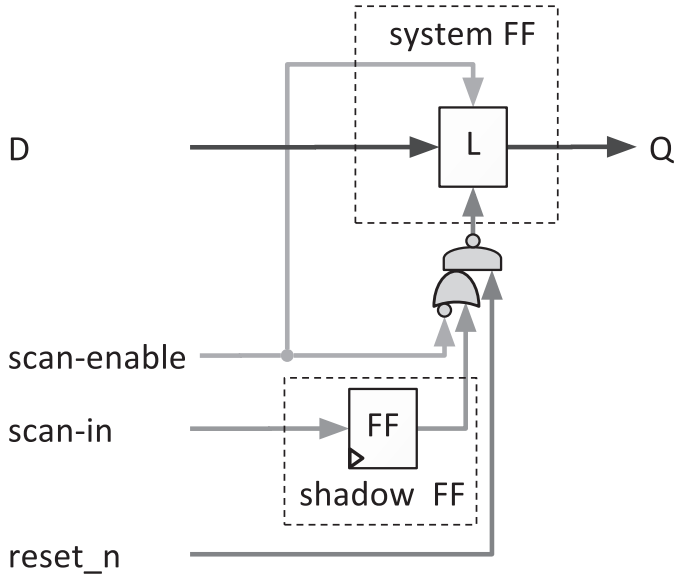


FIGURE 3.2 – Bascule *shadow-scan* utilisant un *latch* système et une bascule *shadow*.

I.2 Contraintes pour le test liées à l'utilisation d'un *shadow-latch*

Dans cette section, on s'intéresse aux transitions entre les modes *scan* et *capture*. Si la transition du mode *scan* vers le mode *capture* ne pose aucun problème, la transition inverse peut apporter une corruption de la donnée capturée pendant le dernier cycle en mode *capture*.

En effet, en mode *capture*, l'état du *shadow-latch* et l'état du *master-latch* de la bascule fonctionnelle de la Figure 3.1 ne sont plus corrélés, ce qui peut laisser apparaître un problème pendant la phase de transition entre les modes *capture* et *scan*. Par conséquent, la réponse capturée par la bascule fonctionnelle peut être corrompue par la donnée alors présente dans le *shadow-latch* qui lui est associé. Cela arrive si un *1* logique est positionné sur le signal *scan-enable* alors que le *slave-latch* de la bascule fonctionnelle est toujours transparent.

Afin de pallier ce problème, il est nécessaire de contraindre le signal *scan-enable* ou le signal d'horloge.

Le chronogramme de la Figure 3.3 présente une mise en œuvre "sûre" du pilotage du signal *scan-enable*, et plus particulièrement de la transition entre les modes *capture* et *scan*. Les états en mode *scan* sont signalés par un *S*, alors que

ceux représentant celui de capture sont notés avec un *C*.

La donnée scannée dans le *shadow-latch* pendant le dernier cycle en mode *scan* (*S1*) est transférée au *slave-latch* de la bascule fonctionnelle afin d'être appliquée au circuit sous test. La donnée capturée par le *master-latch* de la bascule fonctionnelle lors du dernier cycle d'horloge en mode *capture* (*C4*) est transférée au *slave-latch* de la bascule fonctionnelle sans être corrompue par l'état inconnu présent dans le *shadow-latch*.

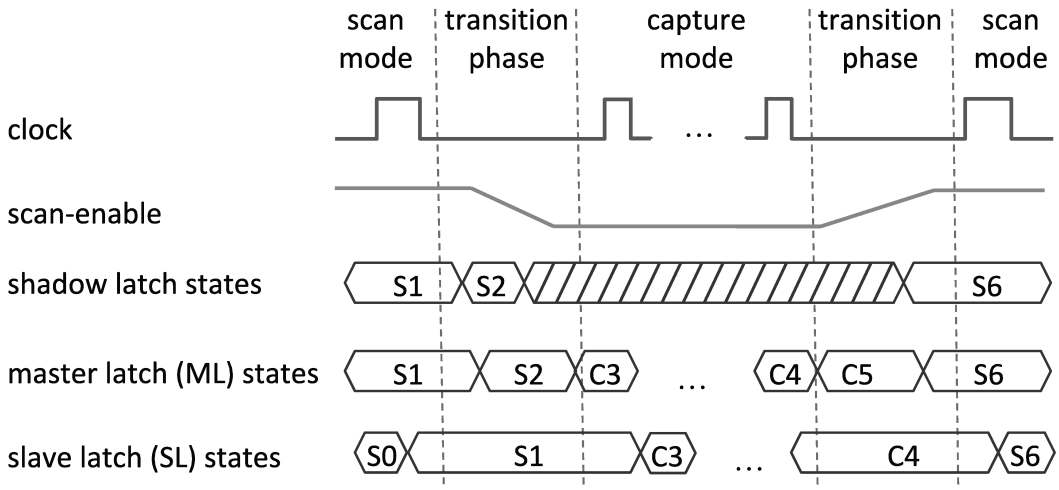


FIGURE 3.3 – Contraintes du signal *scan-enable* pour l'utilisation "sure" d'un *shadow-latch*.

La contrainte à appliquer sur l'horloge, en accord avec la forme du signal *scan-enable* doit également être prise en compte lors de la phase de programmation des unités de génération d'horloges embarquées (*On-Chip Clocking* ou *OCC*). Néanmoins, cette contrainte peut être éliminée, si un multiplexeur de test est placé en amont du *shadow-latch* de la Figure 3.1.

Pendant le test, les problèmes liés aux impulsions de *set/reset* fonctionnels appliqués pendant le mode *capture* peuvent être évités si les conditions présentées dans le chronogramme de la Figure 3.3 sont respectées. Si toutefois des impulsions *set/reset* doivent être appliquées en mode *scan*, il est préférable d'utiliser un *shadow-latch* implémentant les mêmes fonctionnalités de *set/reset* que la bascule fonctionnelle, cela afin de garantir que le circuit fonctionnera bien hors test.

II Déploiement des bascules *shadow-scan* sans monitoring

La bascule proposée dans ce chapitre peut être utilisée directement par les outils d'un flot standard pour la synthèse et l'insertion de chaînes de *scan* si celle-ci est insérée dans une bibliothèque de conception *standard-cell*. En l'absence d'un tel support, il a été nécessaire d'adapter un flot standard afin de permettre une première évaluation du potentiel de la solution *shadow-scan* sans monitoring.

Afin de limiter l'impact de la solution *shadow-scan* sur le circuit, l'insertion des bascules *shadow-scan* est réalisée de façon itérative, en évaluant le gain à chaque itération. Différentes conditions d'arrêt peuvent être choisies telles qu'une certaine contrainte en surface, un objectif en latence à atteindre, ou un nombre maximum d'itérations autorisées. Dans la suite des expérimentations, nous avons choisi de ne pas imposer de contraintes en surface, afin d'obtenir les plus grandes améliorations possibles en termes de latence.

La Figure 3.4 présente un algorigramme du flot d'insertion des bascules *shadow-scan* telles que proposées en Figure 3.1. Ce flot prend en entrée la description VHDL ou VERILOG d'un circuit pour y déployer d'abord une architecture *standard-scan* et ensuite des bascules *shadow-scan* afin d'en améliorer les performances en termes de latence si cela est possible.

On va tout d'abord synthétiser le circuit original à l'aide de l'outil Synopsis DesignCompiler et réaliser l'insertion de chaînes de *scan* à l'aide de l'outil Synopsis DFTCompiler. L'outil Synopsys TetraMax est ensuite utilisé pour déterminer les vecteurs de test et évaluer la couverture de fautes du circuit.

Partant de cette version *standard-scan* du circuit, on va tout d'abord extraire le groupe de bascules situées au bout des chemins critiques présentant le *slack* le plus faible. Cette extraction, réalisée avec DesignCompiler ou PrimeTime, permet de créer la liste de bascules à remplacer car les bascules *shadow-scan* ne permettent d'améliorer que le temps de *setup* et non pas le temps *clock-to-Q*.

Après le remplacement des bascules *standard-scan* listées précédemment avec des bascules *shadow-scan*, les latences des chemins sont réévaluées et une nouvelle liste de bascules à transformer est extraite.

Le processus itératif continue tant que la nouvelle liste de bascules à transformer ne contient aucune bascule *shadow-scan*. Une telle présence indique que plus aucune amélioration n'est possible.

À la fin du processus, la version du circuit avec la meilleure latence est retenue. Cette version peut être composée d'un mélange de bascules *standard-scan* et de bascules *shadow-scan* ou bien être simplement la version *standard-scan* originale dans le cas où aucune des itérations n'a permis d'améliorer le *slack* et donc implicitement la latence du circuit.

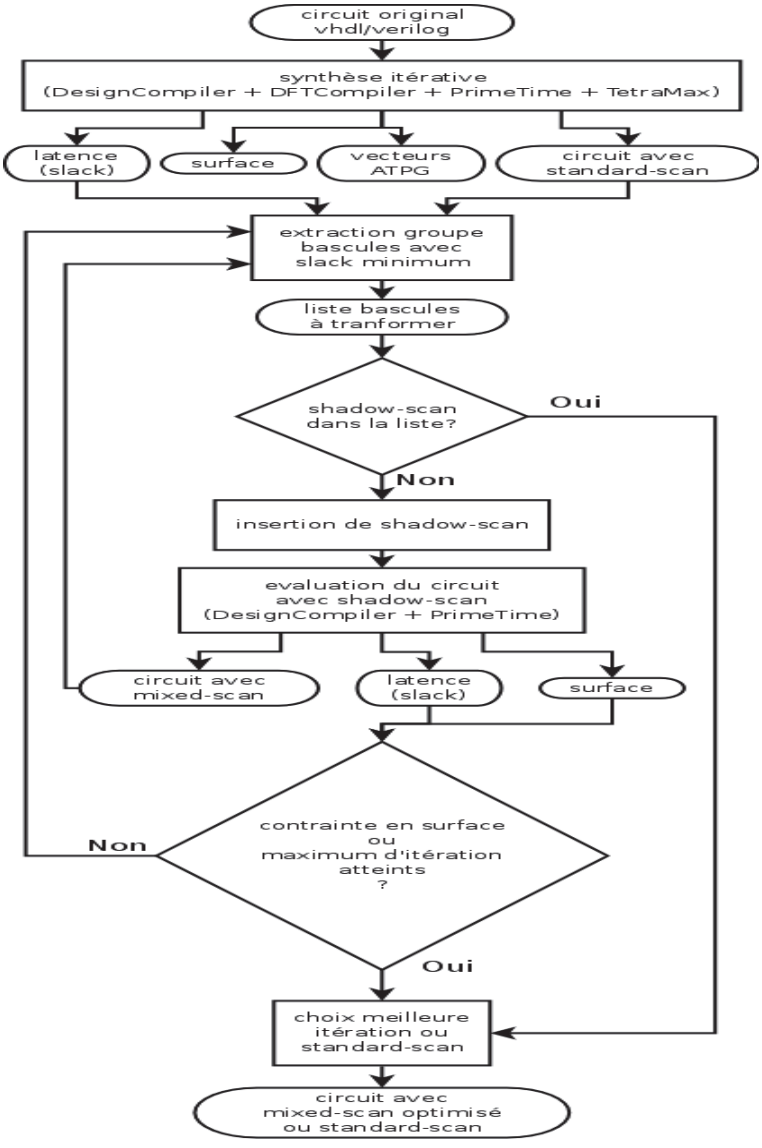


FIGURE 3.4 – Flot de déploiement des bascules *shadow-scan*.

III Évaluation en termes de latence, surface et test de la solution proposée vis-à-vis d'une approche *standard-scan*

Afin d'évaluer le potentiel d'amélioration en latence de la solution avec *shadow-latch* présentée en Figure 3.1, l'algorithme d'insertion présenté en Figure 3.4 a été utilisé sur les circuits *ITC'99* [ITC].

Le Tableau 3.1 présente l'amélioration en latence et l'impact en surface de la mise en œuvre de la solution *shadow-scan* sans monitoring vis-à-vis d'une solution *standard-scan*. La seconde colonne de ce Tableau donne le nombre de bascules transformées dans l'algorithme d'insertion décrit dans la section II. Le nombre important de bascules ayant subi une transformation est dû à la sévérité des contraintes de temps imposées pendant la synthèse des circuits dont l'effet secondaire a été l'augmentation du nombre de chemins critiques.

Comme on le voit dans le Tableau 3.1, l'amélioration maximale en latence peut atteindre jusqu'à 4.6%.

En ce qui concerne l'impact en surface, celui-ci est inférieur à 10% pour les circuits les plus gros. Pour les circuits de taille plus réduite, l'impact en surface est supérieur car la proportion de bascules transformées est plus élevée.

Pour ce qui est de l'impact de la solution en termes de consommation, il devrait être limité. En effet, le *shadow-latch* n'étant utilisé que pendant la phase de test de production, on peut imaginer qu'il soit déconnecté de l'alimentation par exemple par l'utilisation d'un transistor de *power-gating* commandé par le signal de test *scan-enable*, comme illustré en Figure 3.5.

En ce qui concerne le test, la solution *shadow-scan* sans monitoring est équivalente à une solution *standard-scan*. En effet, la bascule *shadow-scan* de la Figure 3.1 dispose d'un brochage et d'un fonctionnement analogue à celui d'une bascule *standard-scan* classique. Ainsi, s'il est possible de piloter le signal *scan-enable* de manière à respecter le chronogramme de la Figure 3.3, il suffit pour les outils de *DFT* et d'*ATPG*, de réutiliser un modèle présentant le comportement d'une bascule *standard-scan* lors du déploiement des chaînes de *scan* et du calcul des

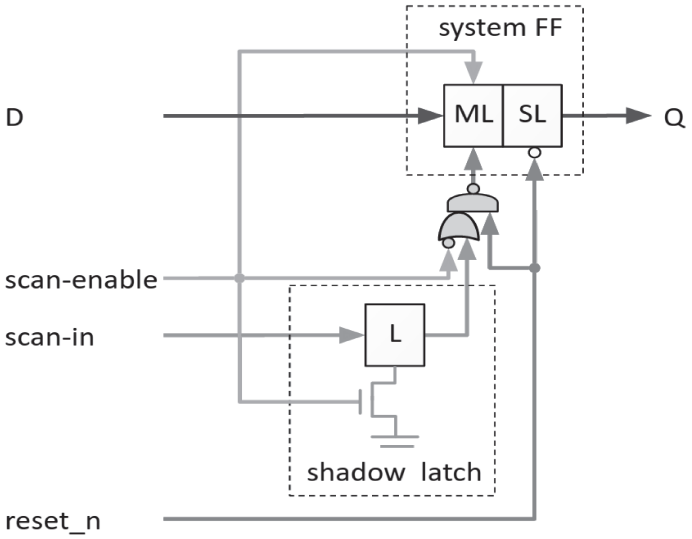


FIGURE 3.5 – Bascule *shadow-scan* avec *power-gating*.

vecteurs de test par l'*ATPG*. Si la bascule *shadow-scan* est directement intégrée à une bibliothèque *standard-cell*, cette modélisation est implicitement assurée.

IV Conclusion

Une déclinaison sans monitoring de l'approche *shadow-scan* a été proposée afin de permettre de réduire l'impact sur la latence de l'infrastructure de test. Il a été montré que sur la grande majorité des circuits étudiés, l'emploi de bascules *shadow-scan* à base de *shadow-latch* permettait une réduction de la latence pouvant atteindre jusqu'à 4.6% par rapport à une implémentation du même circuit avec une approche purement *standard-scan*.

Bien que présentant des améliorations en latences pouvant sembler limitées vis-à-vis de l'important impact en surface, une situation où les *shadow-scan* sans monitoring peuvent être intéressantes à utiliser, est celle de l'amélioration du circuit pendant l'étape de *Back-end*. En effet, il peut arriver qu'à la suite de toutes les étapes de synthèse, seuls quelques chemins présentent une latence trop contrainte, et que la seule solution possible si l'on ne souhaite pas réitérer l'intégralité du processus de synthèse, soit de simplement supprimer le *scan-design* sur ces chemins. C'est notamment dans ce genre de cas que l'utilisation de *shadow-scan* uniquement sur les quelques chemins problématiques peut être intéressante, pour permettre le maintient de la testabilité du circuit, tout en relâchant les contraintes de latence.

TABLEAU 3.1 – Impact en latence et en surface un niveau circuit apportés par la solution *shadow-scan* par rapport à une solution purement *standard-scan*.

circuit	nombre de bascules avec *shadow-latch*	impact en latence	impact en surface
b01	5 (100%)	-1.8%	23.6%
b02	4 (100%)	-1.1%	33.5%
b03	12 (40%)	-3.1%	15.9%
b04	55 (51%)	-1.4%	18.4%
b05	15 (44%)	-4.6%	7.4%
b06	7 (87%)	-3.1%	30.0%
b07	30 (65%)	-1.8%	16.7%
b08	8 (19%)	-1.3%	8.0%
b09	14 (50%)	-3.0%	21.3%
b10	17 (100%)	-2.1%	24.4%
b11	11 (36%)	-0.8%	5.6%
b12	0 (0%)	0%	0%
b13	25 (55%)	-1.1%	18.3%
b14	142 (35%)	-2.8%	7.2%
b15	134 (29%)	-1.1%	6.5%
b17	601 (42%)	-0.9%	9.7%
b18	431 (12%)	-1.8%	3.5%
b19	218 (3%)	-0.6%	0.9%
b20	230 (29%)	-0.9%	5.4%
b21	220 (27%)	-0.8%	5.3%
b22	376 (31%)	-0.9%	6.2%

Chapitre 4

FLOT MIXED-SCAN : COMBINAISON DE STANDARD-SCAN ET SHADOW-SCAN AVEC ET SANS MONITORING EN LIGNE

I Impact du monitoring en ligne

Le déploiement de bascules *shadow-scan* avec monitoring en ligne sur l'ensemble d'un circuit n'est pas toujours possible pour des raisons d'impact en surface et consommation. D'un autre coté, si l'on choisi de déployer des bascules *shadow-scan* avec monitoring uniquement sur les chemins les plus critiques d'un circuit, les résultats en termes d'amélioration de la performance obtenus dans la section III.4.i du chapitre 2 peuvent se retrouver diminués ou même éliminés.

Le Tableau 4.1 présente ces diminutions en comparant une version des circuits *ITC'99* [ITC] avec bascules *shadow-scan* comme illustrée en Figure 2.3, avec une version utilisant des bascules *standard-scan* telles qu'illustrée en Figure 2.7. Ces bascules ont été insérées uniquement sur les chemins les plus critiques en s'assurant que les mêmes chemins soient surveillés dans les deux versions de chaque circuit.

Comme on le voit, pour les circuits *b02*, *b15*, *b17*, *b21*, *b22*, l'amélioration en latence est inférieure à 1%. Par rapport aux valeurs reportées dans le Tableau 2.4, l'amélioration en latence a diminué en moyenne de 1.9%. De plus, pour trois autres circuits (*b09*, *b10*, *b12*), on note même une dégradation de l'ordre de

1% de la latence du fait du déploiement de *shadow-scan*. Cette dégradation est notament due au fait que la solution *shadow-scan* ne permet d'améliorer que le temps de *setup* et qu'elle peut présenter une dégradation du temps de *clock-to-Q*.

TABLEAU 4.1 – Impact en latence et en surface du *shadow-scan* avec monitoring déployé seulement sur les chemins les plus critiques par rapport à une version *standard-scan* avec la même capacité de monitoring.

circuit	nombre de bascules transformées en *shadow-scan* avec monitoring	impact en latence	impact en surface
b01	5	-2.8%	3.5%
b02	3	-0.4%	3.2%
b03	11	-4.0%	2.8%
b04	53	-2.0%	3.2%
b05	14	-3.2%	1.6%
b06	6	-2.6%	3.9%
b07	12	-2.3%	1.6%
b08	9	-4.4%	1.9%
b09	10	+0.8%	2.9%
b10	14	+0.4%	3.4%
b11	10	-1.1%	1.1%
b12	39	+0.3%	2.0%
b13	27	-3.8%	3.3%
b14	117	-2.5%	1.4%
b15	112	-0.9%	1.3%
b17	367	-0.7%	1.4%
b18	174	-1.9%	0.4%
b19	99	-1.4%	0.1%
b20	99	-1.5%	1.1%
b21	199	-0.2%	1.1%
b22	289	-0.4%	1.1%

Une solution pour pallier cet impact sur la latence, si le budget en surface le permet, peut être de remplacer les bascules *standard-scan* sur les chemins critiques restants par des bascules *shadow-scan* sans monitoring pour compenser l'impact du monitoring en ligne.

II Déploiement des bascules shadow-scan avec et sans monitoring

L'idée va être ici de mettre consécutivement en œuvre les deux flots présentés dans le chapitre 2 section III et dans le chapitre 3. Dans un premier temps, on déploiera, grâce au flot du chapitre 2, des bascules *shadow-scan* avec monitoring à la place des bascules *standard-scan* initialement choisies pour le monitoring en-ligne des marges temporelles. A cette étape de l'étude, seulement les bascules au bout des chemins les plus critiques seront choisies. Ensuite, on utilisera le flot présenté au chapitre 3 sur le circuit avec monitoring afin d'améliorer ses performances si cela est possible.

A la suite de ce processus, on obtient donc un circuit *mixte*, composé de bascules *standard-scan*, de bascules *shadow-scan* avec monitoring, et de bascules *shadow-scan* sans monitoring implantées uniquement pour absorber l'impact du monitoring sur la latence. Cette approche peut être représentée par l'organigramme de la Figure 4.1

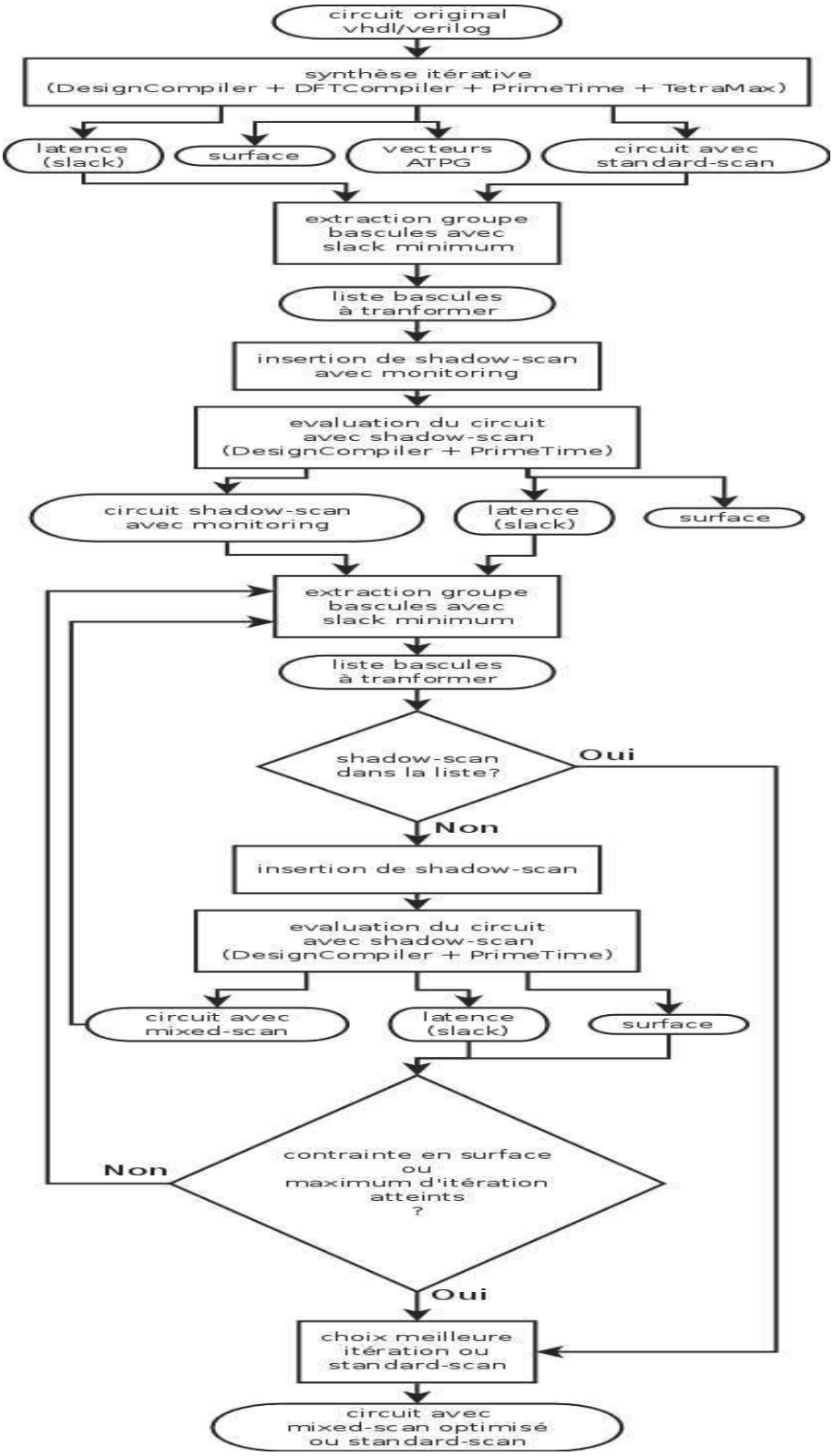


FIGURE 4.1 – Flot de déploiement des bascules *shadow-scan* avec et sans monitoring.

III Évaluation de la solution *shadow-scan mixte* avec et sans monitoring

Le Tableau 4.2 présente l'impact de l'insertion des bascules avec *shadow-scan* avec et sans monitoring sur la latence et la surface d'un circuit, vis-à-vis de la même solution de référence utilisée pour les résultats reportés dans le Tableau 4.1.

Comme on peut l'observer, la solution *mixte* présente toujours un impact inférieur en termes de latence. Le gain obtenu atteint jusqu'à 5.6%, pour un impact en surface inférieur à 8.2% vis-à-vis de la solution de référence. Par rapport aux résultats reportés dans le Tableau 4.1, la latence des circuits a été améliorée de 1.5% en moyenne pour une augmentation moyenne de la surface de 1.3%. Ces résultats correspondent à une implémentation au niveau portes des bascules *shadow*. Une optimisation au niveau transistor des bascules pourrait permettre des améliorations plus importantes.

En termes d'impact de longueur de test et de couverture des fautes de collage, le fait de déployer des bascules *shadow-scan* sans monitoring ne présente pas d'impact. En effet, comme les bascules *shadow-scan* sans monitoring sont équivalentes à des bascules *standard-scan*, les outils d'*ATPG* et de *DFT* sont capables de les gérer sans surcoût. Ici, c'est uniquement le test de l'arbre de *ou*, comme cela était le cas dans la section III.4.i du chapitre 2 qui peut augmenter la longueur de la séquence de test.

La colonne Δvecteurs du Tableau 4.3 présente le ratio de vecteurs de test supplémentaires requis par l'approche *shadow-scan*. Le nombre de vecteurs supplémentaires est théoriquement borné par le *Fan-in* du plus large cône logique dans le premier étage de *pipeline* dans le cas où cet arbre est *pipeliné*. Pour la majeure partie des circuits étudiés, Δvecteurs est inférieur à 1. Cela est dû au fait que des bascules avec du *standard-scan* peuvent être utilisées pour *pipeliner* l'arbre de *ou* de la Figure 2.5 et ainsi faciliter son test. Dans notre cas d'étude, la taille du cône logique le plus grand du premier étage du *pipeline* de l'arbre de *ou* a été fixée à 1024 entrées, et le même arbre de *ou* avec la même structure de *pipeline* a été déployé dans les versions *shadow-scan* et purement *standard-scan* de chaque circuit.

TABLEAU 4.2 – Impact en latence et en surface au niveau circuit après déploiement des bascules *shadow-scan* sans monitoring.

circuit	nombre de bascules *shadow-scan* sans monitoring	impact en latence	impact en surface
b01	0	-2.8%	3.5%
b02	1	-2.1%	8.2%
b03	0	-4.0%	2.8%
b04	2	-3.5%	3.7%
b05	0	-3.2%	1.6%
b06	1	-4.3%	6.4%
b07	8	-5.6%	5.7%
b08	0	-4.4%	1.9%
b09	3	-4.1%	6.2%
b10	2	-3.6%	5.3%
b11	1	-2.3%	1.6%
b12	18	-2.8%	5.9%
b13	0	-3.8%	3.3%
b14	0	-2.5%	1.4%
b15	9	-2.8%	1.6%
b17	177	-2.8%	3.9%
b18	0	-1.9%	0.4%
b19	0	-1.4%	0.1%
b20	32	-2.7%	1.8%
b21	26	-2.1%	1.6%
b22	79	-2.5%	2.3%

TABLEAU 4.3 – Impact sur la longueur de test et la couverture de fautes de collage.

circuit	*standard-scan*		*shadow-scan mixte*			*Fan-in* max 1er étage arbre de *ou*
	nombre de vecteurs	*FC*	nombre de vecteurs	Δvecteurs	*FC*	
b01	20	98.8%	20	0.00	99.3%	5
b02	16	97.9%	15	-0.33	98.8%	3
b03	28	99.6%	30	0.18	99.8%	11
b04	71	98.8%	105	0.64	98.8%	53
b05	88	99.2%	87	-0.07	99.3%	14
b06	18	99.0%	17	-0.17	99.4%	6
b07	77	99.9%	84	0.58	100.0%	12
b08	46	99.9%	47	0.11	100.0%	9
b09	31	99.8%	32	0.10	100.0%	10
b10	48	99.8%	56	0.57	100.0%	14
b11	151	99.9%	148	-0.30	100.0%	10
b12	121	99.9%	145	0.62	100.0%	39
b13	60	99.8%	75	0.56	99.8%	27
b14	478	99.9%	490	0.10	99.9%	117
b15	659	99.8%	669	0.09	99.8%	112
b17	707	99.8%	1023	0.86	99.8%	367
b18	659	99.7%	738	0.45	99.7%	174
b19	665	99.8%	662	-0.03	99.8%	99
b20	496	99.9%	522	0.13	99.9%	199
b21	460	99.8%	503	0.23	99.8%	190
b22	460	99.9%	598	0.48	99.9%	289

IV Conclusion

La combinaison des solutions *shadow-scan* avec et sans monitoring a été proposée afin d'implémenter des circuits présentant une latence améliorée et un support du monitoring en ligne des marges temporelles.

En cas de déploiement des bascules *shadow-scan* avec monitoring seulement, et cela uniquement sur les chemins les plus critiques des circuits *ITC'99*, nous avons vu que le gain en latence par rapport à une solution de l'état de l'art peut être négatif ou très limité (de l'ordre de 1.9%).

En déployant également des bascules *shadow-scan* sans monitoring, il a été montré qu'il est possible d'améliorer la latence des versions avec *shadow-scan* de 1.5% en moyenne sans affecter la qualité du monitoring en-ligne et pour un impact en surface moyen de 1.3%.

Chapitre 5

RÉDUCTION DU NOMBRE DE MONITEURS AVEC MAÎTRISE DE L'IMPACT SUR LA QUALITÉ DU MONITORING

I Problématique liée à l'utilisation du monitoring dans les VLSI

Les méthodes présentées dans les chapitres précédents permettent de réduire l'impact de ce monitoring du *slack* sur la vitesse des circuits. Le but de ce chapitre est de proposer des méthodes de réduction de l'impact du monitoring sur la surface et, implicitement, la consommation du circuit ou du système. L'approche envisagée sera de réduire le nombre de moniteurs tout en maîtrisant l'impact sur la qualité du monitoring.

Une approche naturelle vis-à-vis du déploiement du monitoring est de placer des moniteurs sur l'ensemble des bascules du circuit qui se trouvent sur des chemins critiques ou susceptibles de le devenir à cause du vieillissement ou d'autres phénomènes comme la variabilité induite par le processus de production du circuit. Toutefois, pour des circuits avec un nombre important de chemins critiques, cette approche risque d'aboutir à un surcoût surfacique trop important pour des améliorations, notamment en termes de latence, plutôt limitées.

Une solution pour surmonter ce problème a été proposée dans [LCAG13]. Elle consiste à ne déployer des moniteurs qu'au bout de la partie commune des chemins critiques. Afin d'éviter l'utilisation de signaux d'horloge déphasés, un chemin de retardement doit toutefois être ajouté entre la partie commune d'un chemin à surveiller et le moniteur associé. Pour autant, l'insertion de tels chemins non fonctionnels augmente le nombre de chemins critiques du circuit tout en laissant les chemins fonctionnels critiques non monitorés.

Un problème également associé au déploiement de moniteurs uniquement sur les chemins les plus critiques d'un circuit, est qu'il n'est pas certain que ceux-ci fournissent une information fiable quant à l'état global du circuit. En effet, il peut se produire que, bien qu'étant temporellement critiques, les chemins sélectionnés pour le monitoring ne présentent que trop peu d'activité vis-à-vis du reste du circuit. Ainsi, il est possible qu'avant même que les moniteurs ne soient activés par des transitions propagées au long des chemins critiques et que des alertes ne soient générées quant à la dégradation du circuit, des zones initialement moins critiques présentent déjà des signes avancés de dégradation ou ont déjà produit des erreurs.

Une autre situation risquant de se présenter peut être celle dans laquelle la première transition sur un chemin critique apparaissant après une longue période d'inactivité du chemin génère directement une erreur et cela sans alerte préalable.

Une solution pour essayer de traiter ce problème peut être d'évaluer directement pendant la conception, le potentiel de chaque bascule à améliorer la qualité du monitoring. En l'occurrence, une approche pour mettre en œuvre cela, peut être d'évaluer la probabilité qu'une transition soit propagée le long d'un chemin critique. En effet, une telle transition sera susceptible d'activer un moniteur associé à une bascule et ainsi déclencher une alerte.

Dans la suite de ce chapitre, une méthode visant à évaluer la probabilité d'activation des moniteurs de *slack* sera tout d'abord présentée. Ensuite, deux métriques permettant d'évaluer la qualité du monitoring d'un sous-ensemble de bascules seront proposées et présentées. Elles permettront d'obtenir une estimation à la fois du nombre moyen de moniteurs susceptibles d'être activés à chaque cycle d'horloge, ainsi que du ratio de cycles d'horloge pour lesquels

au moins un moniteur du lot est susceptible d'être activé. Ces deux métriques permettront d'estimer à la fois la couverture spatiale et temporelle du monitoring pour un circuit et un ensemble de moniteurs donnés. Enfin, il sera montré que l'utilisation de moniteurs de *slack* avec fenêtre de détection adaptée au *slack* du cône logique observé permettra d'améliorer considérablement la qualité du monitoring.

II Évaluation de la probabilité d'activation des moniteurs

Afin d'estimer la probabilité d'activation des moniteurs dans un circuit, la solution proposée consiste à s'appuyer sur l'utilisation de simulations *back-annotées* jouées pour différents niveaux de dégradations du *slack* des chemins critiques. Ces niveaux de dégradations sont choisis de telle manière qu'une transition propagée sur un chemin fonctionnel critique soit susceptible de générer une alerte. L'idée est ici de compter le nombre d'alertes qui apparaissent au sein du circuit au cours des différentes simulations afin d'estimer la probabilité d'activation des moniteurs.

Pour réaliser une estimation fiable, des moniteurs doivent être déployés sur l'ensemble du circuit afin d'assurer l'estimation de la couverture spatiale. De même, un grand nombre de vecteurs de test doit être appliqué en entrée du circuit afin d'assurer l'estimation de la couverture temporelle.

Bien que devant être représentatifs du type de donnée traitée pendant la mission du circuit (*work-load* réel), les vecteurs à appliquer en entrée peuvent toutefois être aléatoires si l'on ignore au moment de la simulation le type d'application supposée fonctionner sur le circuit (ex : pour un processeur générique ou un contrôleur mémoire). *Acontrario*, on pourra faire appel à des données issues de la simulation fonctionnelle du circuit si elles représentent la charge réelle que celui-ci rencontrera pendant sa mission.

En ayant instrumenté l'ensemble du circuit et en comptant l'occurrence des alertes pour différentes valeurs du *slack*, il devient alors possible de trier les bascules en fonction du taux d'activation du moniteur associé. Il est ainsi possible d'évaluer le potentiel de certains sous-groupes de bascules vis-à-vis de la

qualité du monitoring, notamment en étudiant la distribution des moniteurs activés dans le temps et dans l'espace. On obtient alors la couverture spatiale et temporelle du monitoring offerte par ces sous-groupes.

La méthode proposée ici se décompose en trois grandes étapes. Tout d'abord le circuit considéré est synthétisé par exemple à l'aide de *Design Compiler* et des moniteurs sont insérés sur l'ensemble des bascules du circuit. Suite à cela, les différentes valeurs d'horloge nécessaires aux simulations back-annotées des différents états de dégradation du *slack* sont extraites à l'aide d'un outil d'analyse temporelle statique (*STA* pour *Static Timing Analysis*) tel que l'outil *PrimeTime* de Synopsys. Cette étape va consister à rechercher les fréquences d'horloge pour lesquelles le ou les chemins les plus critiques du circuit présentent la dégradation du *slack* recherchée. Ensuite, des simulations *back-annotées* des différentes fréquences d'horloge sont réalisées. Ces simulations peuvent utiliser des entrées issues de vérifications fonctionnelles si celles-ci existent, ou des valeurs aléatoires si le *work-load* n'est pas encore connu. Enfin, pour chaque simulation et pour chaque moniteur le nombre de cycles d'horloge pour lesquels une alerte a été émise en sortie d'un moniteur est compté.

Nous avons basé nos évaluations sur le circuit *b22* du benchmark ITC'99. Le type de moniteurs déployés sur le circuit dans notre cas d'étude, sera un moniteur de marges temporelles avec prédiction d'erreur, tel que celui présenté en Figure 1.11. Ce type de moniteur présente l'intérêt de ne pas imposer l'emploi de mécanismes de recouvrement d'erreur, ni d'imposer de contraintes en terme de latence minimale vis-à-vis des chemins. L'élément de retard δ offre une fenêtre de détection afin de détecter les transitions tardives du signal D.

En ce qui concerne le choix vis-à-vis des simulations, le *slack* est choisi en considérant le chemin le plus critique qui arrive à la bascule shadow (notée shadow FF sur la Figure 1.11) d'un des moniteurs insérés. Pour des raisons de temps de simulation, celles-ci ont été limitées à deux valeurs de dégradation du *slack*.

Ainsi, nous avons choisi de simuler respectivement des valeurs de *slack* réduite de $-\sigma$ et $-\frac{\sigma}{2}$, avec σ représentant la déviation standard relative de la latence des chemins critiques dans un circuit due aux variations de type intra-die [EBSLM97, BDM02]. Ici, σ à été fixé à 4% de la latence du chemin de plus

critique. Le fait d'avoir choisi une fenêtre de détection plus grande que σ garantit ainsi un *slack* positif pour tous les chemins arrivant à de bascules système et, implicitement, la génération des alertes pour toute transition transmise le long d'un chemin avec *slack* négatif vers la bascule shadow d'un moniteur. Le nombre de valeurs de *slack* simulées a été ici limité à deux pour des raisons de temps de simulation.

Concernant les vecteurs de test appliqués lors de la simulation, comme le *work-load* spécifique du circuit *b22* était inconnu, des entrées aléatoires ont été utilisées. Afin que l'évaluation soit significative, tout en limitant le temps de simulation, 10000 vecteurs ont été appliqués à chacune des deux valeurs de dégradations du *slack* lors de la simulation *back-annotée*.

III Métriques d'évaluation de la qualité du monitoring

Pour évaluer la qualité du monitoring d'un groupe de bascules vis-à-vis du reste du circuit, deux métriques ont étés développées. Ces deux métriques se basent sur le taux moyen simulé de moniteurs activés par cycle d'horloge (*SANAM* pour *Simulated Average Number of Activated Monitors*) du groupement de bascules évalué et est défini comme le nombre moyen de moniteurs activés par cycle d'horloge de la simulation.

Ainsi, si l'on considère que les latences des chemins critiques sont normalement distribuées sur l'ensemble du circuit avec une déviation standard relative de σ vis-à-vis de la latence maximale rapportée par la STA, il devient possible de calculer la première métrique représentant la limite basse du nombre moyen de moniteurs du groupe activés par cycle d'horloge (ou *ENAMC* pour *expected number of activated monitors per clock cycle*). Cela permet ainsi d'estimer la couverture spatiale du groupe de moniteurs dans le circuit.

$$\mathrm{ENAMC}_{-\frac{\sigma}{2}} = \frac{1}{2}\mathrm{SANAM}_{-\frac{\sigma}{2}} \tag{5.1}$$

$$\mathrm{ENAMC}_{-\sigma} = \frac{1+\mathrm{erf}\left(\frac{1}{2\sqrt{2}}\right)}{2}\mathrm{SANAM}_{-\frac{\sigma}{2}} + \frac{1}{2}\left(\mathrm{SANAM}_{-\sigma} - \mathrm{SANAM}_{-\frac{\sigma}{2}}\right) \tag{5.2}$$

Ici erf() représente la fonction d'erreur mathématique. Le premier facteur du premier terme de ENAMC$_{-\sigma}$ est introduit afin de tenir compte du fait que la probabilité d'activation des moniteurs qui ont été activés pour une dégradation du *slack* de $-\frac{\sigma}{2}$ est plus grande que la probabilité d'activation de ceux qui ont été activés seulement pour une dégradation du *slack* de $-\sigma$. Ce facteur correspond au résultat de l'intégration de la distribution normale entre une valeur obtenue en soustrayant $-\frac{\sigma}{2}$ de la valeur moyenne et $+\infty$. Le coefficient $\frac{1}{2}$ des autres termes correspond au résultat de l'intégration de la même distribution entre la valeur moyenne et $+\infty$.

Afin d'estimer la couverture temporelle d'un groupe de bascules avec moniteur, on peut utiliser une métrique qui représente le taux attendu de cycles d'horloge pour lesquels au moins un moniteur du groupe de bascules considérées est activé (ou *ERCMA* pour *expected ratio of clock cycles with at least one monitor activated*). Ainsi, toujours dans l'hypothèse d'une distribution gaussienne de la latence des chemins, pour chaque cycle d'horloge il est alors possible grâce à cette métrique d'estimer la limite minimale de la probabilité qu'au moins un des moniteurs du groupe soit actif durant chaque cycle d'horloge considéré.

$$\text{ERCMA}_{-\frac{\sigma}{2}}(\text{cycle d'horloge}) = 1 - \frac{1}{2^n} \tag{5.3}$$

$$\text{ERCMA}_{-\sigma}(\text{cycle d'horloge}) = 1 - \frac{1}{2^{(m-n)}} \left(\frac{1 - \text{erf}\left(\frac{1}{2\sqrt{2}}\right)}{2} \right)^n \tag{5.4}$$

Le second terme de chaque équation représente la limite supérieure de la probabilité qu'aucun moniteur ne soit activé. Les paramètres m et n représentent quant à eux les nombres de moniteurs actifs pendant le cycle d'horloge considéré, pour les différentes valeurs de dégradations du *slack*. Respectivement, n et m correspondent à une dégradation du *slack* de $-\frac{\sigma}{2}$ et de $-\sigma$. Comme précédemment, le deuxième facteur du deuxième terme de ERCMA$_{-\sigma}$ est introduit afin de tenir compte du fait que la probabilité d'activation des moniteurs qui ont été activés pour une dégradation du *slack* de $-\frac{\sigma}{2}$ est plus grande que la probabilité d'activation de ceux qui ont été activés seulement pour une dégradation du *slack* de $-\sigma$. Une limite basse de ERCMA peut être calculée en moyennant les résultats des équations précédentes sur l'ensemble des cycles d'horloge de la simulation.

Le Tableau 5.1 présente les résultats des différentes métriques pour le circuit *b22* pour les deux niveaux de dégradations du *slack*. Comme on peut le voir, seul 1% des moniteurs présente de l'activité pour une dégradation du *slack* équivalente à $-\frac{\sigma}{2}$, alors que pour une dégradation plus importante, ce pourcentage atteint 4.45%. Ce sont les bascules présentes dans ces 4.45% que nous allons retenir pour notre étude. On peut estimer le nombre moyen de moniteurs actifs du groupe, par cycle d'horloge, en fonction de la dégradation du *slack*. Dans cette étude, dans le cas d'une dégradation du *slack* de l'ordre de $-\sigma$ en moyenne 3.62 moniteurs du groupe seront actifs. À l'inverse, dans le cas d'une dégradation du *slack* de l'ordre de $-\frac{\sigma}{2}$ en moyenne moins d'un moniteur est actif par cycle d'horloge. De même, après la simulation, il est possible d'estimer que seuls 2% des cycles d'horloge dans le cas d'une dégradation du *slack* équivalente à $-\frac{\sigma}{2}$ présentent de l'activité sur les moniteurs. Ce ratio atteint 33% pour une dégradation équivalente à $-\sigma$.

TABLEAU 5.1 – Évolution des métriques en fonction de la dégradation du *Slack* avec σ fixé à 4% de la latence du chemin le plus critique.

dégradation du slack	moniteurs activés pendant la simulation	SANAM	ENAMC	cycles d'horloge avec moniteurs activés	ERMAC
$-\frac{\sigma}{2}$	1.0%	0.05	0.03	2%	0.01
$-\sigma$	4.45%	3.62	1.82	33%	0.28

IV Moniteurs à fenêtre de détection adaptée

Une approche visant à améliorer le taux d'activation des moniteurs peut consister à adapter la fenêtre de détection de chaque moniteur à la latence du cône logique qu'il surveille. Ainsi, une plus faible dégradation déclenchera une alerte, même pour des parties du circuit avec des chemins moins critiques du point de vue de leur latence.

Avant l'insertion des moniteurs, les bascules sont classées par rapport à leur *slack*, c'est à dire par rapport à la marge temporelle du chemin avec la plus grande latence dans le cône logique arrivant à cette bascule. Bien évidement, il s'agit ici de la marge temporelle obtenue dans le cas de la période minimale du signal d'horloge pour laquelle le circuit fonctionne toujours sans faute en simulation. Ensuite, les bascules sont regroupées par valeurs de *slack* similaires. Enfin, des moniteurs sont insérés auprès des bascules de chaque groupe. Chaque

groupe disposera de moniteurs avec la même fenêtre de détection mais qui sera différente de la fenêtre de détection des moniteurs associés à des bascules dans des groupes différents.

À titre d'exemple, cinq plages de *slack* ont étés considérées :

a) Cas du $slack < \frac{\sigma}{2}$: la fenêtre de détection n'a pas été étendue.

b) Cas du $\frac{\sigma}{2} < slack < \sigma$: la fenêtre de détection est étendue de $\frac{\sigma}{2}$.
la fenêtre de détection est étendue en ajoutant des inverseurs supplémentaires aux éléments de délai en Figure 1.11.

c) Cas du $\sigma < slack < \frac{3\sigma}{2}$: la fenêtre de détection est étendue de σ.

d) Cas du $\frac{3\sigma}{2} < slack < \sigma$: la fenêtre de détection est étendue de $\frac{\sigma}{2}$.

e) Cas du $slack > \sigma$: la fenêtre de détection n'a pas été étendue.
Ces moniteurs ne sont jamais activés pendant les simulations effectuées avec de dégradations du *slack* de $-\frac{\sigma}{2}$ et $-\sigma$.

En modifiant ainsi les moniteurs, il deviendra alors possible de détecter plus tôt l'apparition d'une dégradation du *slack*. Cela est dû au fait que l'on augmente la probabilité d'activation des moniteurs, en réduisant l'amplitude de la dégradation nécessaire au déclenchement d'une alerte.

Comme présenté dans le Tableau 5.2, si les simulations sont rejouées et les calculs des métriques ENAMC et ERMAC de nouveau effectués, pour le même lot de bascules que précédemment, on note une augmentation significative des métriques.

TABLEAU 5.2 – Évolution des métriques en fonctions de la dégradation du *Slack* dans le cas de l'utilisation de moniteurs avec fenêtre de détection adaptée avec σ fixé à 4% de la latence du chemin le plus critique.

dégradation du slack	% de moniteurs activés pendant la simulation	SANAM	ENAMC	% de cycles d'horloge avec moniteurs activés	ERMAC
$-\frac{\sigma}{2}$	2.43%	0.37	0.18	23%	0.15
$-\sigma$	4.45%	7.30	3.72	41%	0.40

L'adaptation de la fenêtre de détection au *slack* de la bascule associée et, implicitement, au *slack* du cône surveillé permet d'obtenir un meilleur ratio de bascules avec un moniteur activé. De même, cela permet d'avoir un plus grand nombre de cycles d'horloge avec au moins un moniteur du groupe activé,

comme rapporté dans la cinquième colonne du Tableau 5.2. Par conséquent, la qualité du monitoring sera améliorée.

En l'occurrence, par rapport aux valeurs rapportées dans le Tableau 5.1, la métrique ENAMC est augmentée respectivement d'un facteur 6 pour une dégradation du *slack* de $-\frac{\sigma}{2}$ et d'un facteur 2 lors d'une dégradation de $-\sigma$. Pour sa part, la métrique ERMCA est améliorée d'un facteur 15 dans le cas d'une dégradation du *slack* de $-\frac{\sigma}{2}$ et d'un facteur 1.4 pour une dégradation de $-\sigma$. Pour un monitoring utilisant des fenêtres de détection fixes, et pour une dégradation du *slack* de $-\frac{\sigma}{2}$, on obtient un ERMCA égal à 0.01 ce qui correspond au fait qu'au moins 1% des cycles d'horloge soient susceptibles d'avoir au moins un moniteur actif. Pour un monitoring utilisant des fenêtres de détection adaptées, et sous les mêmes conditions, un ERMCA de 0.15 équivaut à une probabilité bien plus importante. En effet, un ERMCA de 0.15 équivaut au fait qu'au moins 15% des cycles d'horloge soient susceptibles d'avoir au moins un moniteur du groupement actif.

V Conclusion

Une méthode pour évaluer un groupe de bascules avec moniteurs de *slack* vis-à-vis de la qualité du monitoring d'un circuit a été proposée. L'évaluation de la qualité du monitoring est basée sur la probabilité d'activation des chemins critiques et implicitement des moniteurs qui pourraient les surveiller. Pour cela, une simulation *back-annotée* du circuit est réalisée. Celle-ci peut mettre en œuvre soit des entrées aléatoires, soit un *work-load* spécifique.

En se basant sur cette probabilité, deux métriques ont spécifiquement été développées afin de permettre d'évaluer la qualité de monitoring assuré par un groupe de bascules avec moniteurs de *slack*. Une des métriques permet d'avoir une estimation du nombre moyen de moniteurs actifs à chaque instant. L'autre métrique vise à donner une information quant au ratio de cycles d'horloge pour lesquels au moins un moniteur est activé.

En se basant sur ces métriques, il a été montré l'intérêt de l'adaptation de la fenêtre de détection des moniteurs vis-à-vis du *slack* des chemins arrivant à la bascule associée. Adapter le moniteur devrait permettre de détecter de plus faibles variations, et ainsi de déclencher des alertes plus rapidement.

CONCLUSION

Avec l'essor des technologies à base de semi-conducteurs, les variations dynamiques en partie liées aux modifications du milieu de mission représentent un défi majeur vis-à-vis de la fiabilité, de l'efficacité énergétique et de la performance des circuits numériques embarqués. Le simple ajout de marges temporelles afin de tenir compte de l'impact de ces variations sur la latence des chemins n'est plus une solution économiquement viable, du fait des conséquences de ces marges sur les performances des circuits. Un moyen pour limiter cet impact est d'utiliser des techniques de surveillance en continu des marges temporelles, aussi appelée monitoring en-ligne de *slack*, couplées à des solutions d'adaptation dynamique de la fréquence et de la tension d'alimentation. L'emploi d'infrastructures de monitoring, amène toutefois de nouvelles questions quant à leur mise en œuvre, leur test, ainsi que leur impact dans les circuits.

Contributions

Dans le cadre de cette thèse, une solution a été proposée dans le but d'implémenter une architecture de test ayant un faible impact sur la latence des circuits, tout en autorisant un monitoring en ligne des marges temporelles et en étant compatible avec les approches *standard-scan* actuelles du monde industriel. Grâce à cette solution utilisant des bascules avec *shadow-scan*, les latences des circuits *benchmarks ITC'99* ont pu être réduites jusqu'à 8% comparées à l'utilisation d'une solution de l'état de l'art à base de bascules de type *Razor* implémentant du *standard-scan*.

Il a été montré que la solution *shadow-scan* proposée présente un impact minimal sur le flot, et peut facilement être déployée au sein d'outils de *DFT* et d'*ATPG* pré-existants. De même, il a été montré que la solution ne présente pas d'impact sur la couverture de fautes et que les outils d'*ATPG* et de *DFT* sont capables de générer des vecteurs permettant de tester la logique de mission mais

aussi l'architecture de monitoring et cela au prix d'un surcoût en termes de longueur de la séquence de test qui reste limité.

Une déclinaison sans monitoring de l'approche *shadow-scan* a aussi été proposée dans le but de permettre uniquement la réduction de l'impact sur la latence de l'infrastructure de test. Grâce à l'emploi de bascules *scan* avec *shadow-latch*, il a été possible d'achever une réduction de la latence pouvant atteindre jusqu'à 4.6% par rapport à une implémentation avec une approche purement *standard-scan*.

Afin d'implémenter des circuits proposant une amélioration de la latence ainsi qu'un support du monitoring en ligne des marges temporelles, une combinaison des deux solutions *shadow-scan* a été proposée. Dans le cas d'un déploiement des bascules *shadow-scan* avec monitoring seulement sur les chemins les plus critiques, il a été montré que l'amélioration de la latence est réduite et même une dégradation peut apparaître par rapport à une solution de l'état de l'art. En déployant des bascules *shadow-scan* avec monitoring uniquement sur les chemins les plus critiques du circuit et en déployant également des bascules *shadow-scan* sans monitoring sur le reste des chemins critiques du circuit, il a été montré qu'il est possible de diminuer de façon plus significative la latence des circuits avec monitoring. Cette optimisation a été permise sans affecter la qualité du monitoring en ligne et pour un impact en surface limité.

Enfin, une méthode a été proposée pour évaluer la qualité du monitoring d'un groupe de bascules avec moniteurs de *slack*. Elle se base sur la probabilité d'activation des chemins critiques ainsi que des moniteurs qui pourraient les surveiller. Afin d'évaluer la probabilité d'activation, la méthode proposée utilise une simulation *back-annotée* du circuit. La simulation peut être réalisée soit en utilisant des entrées aléatoires, soit un *work-load* spécifique au circuit si celui-ci est connue.

En utilisant cette probabilité, deux métriques ont été développées afin de permettre l'évaluation de la qualité de monitoring d'un groupe de bascules avec monitoring de *slack* en présence de variations. Une des métriques permet l'estimation du nombre moyen de moniteurs activés à chaque instant dans le circuit, alors que l'autre métrique donne une information quant au pourcentage de cycles d'horloge pour lesquels au moins un moniteur est activé.

En combinant les deux métriques, il a également été montré que l'utilisation de moniteurs de *slack* dont la fenêtre de détection est corrélée au *slack* minimum du cône logique qu'il surveille permet d'améliorer leur probabilité d'activation. Adapter la fenêtre de détection du moniteur pourrait permettre de détecter plus rapidement les variations, et ainsi de déclencher des alertes plus tôt.

Perspectives

L'exploitation des solutions proposées nécessite la continuation de ces travaux. Tout d'abord, la création et la caractérisation au niveau transistor et *layout* des bascules avec *shadow-scan* permettraient leur insertion dans une bibliothèque de synthèse. Cette intégration devrait permettre de réduire drastiquement l'impact de la solution sur le flot du circuit du fait que l'intégralité du flot d'insertion pourrait être directement gérée par les outils de synthèse dans le cas de bascules avec *shadow-latch*.

Une autre perspective vise à améliorer le flot d'évaluation et d'insertion de moniteurs de *slack* dans les circuits. Cela pourrait passer par l'intégration du calcul de la probabilité d'activation des chemins critiques directement au sein des outils de synthèse ou d'évaluation des caractéristiques d'un circuit comme sa latence ou sa consommation. En procédant ainsi, l'étape de simulation *back-annotée* ne serait alors plus nécessaire. En effet, les outils de synthèse pourraient être capables d'évaluer directement pendant ou après la synthèse d'un circuit la capacité de chaque bascule à améliorer le monitoring. Le choix d'associer un moniteur à une bascule en particulier pourrait alors être facilité et optimisé.

BIBLIOGRAPHIE

[AAA87] S.A Al-Arian and D.P. Agrawal. Physical failures and fault models of cmos circuits. *Circuits and Systems, IEEE Transactions on*, 34(3) :269–279, Mar 1987.

[ABB+04] Florence Azaïs, Serge Bernard, Yves Bertrand, Marie-Lise Flottes, Patrick Girard, C Landrault, Laurent Latorre, Serge Pravossoudovitch, Michel Renovell, Bruno Rouzeyre, et al. *Test de circuits et de systèmes intégrés*. 2004.

[ABF90] Miron Abramovici, Melvin A Breuer, and Arthur D Friedman. *Digital systems testing and testable design*, volume 2. Computer science press New York, 1990.

[ABFM07] Todd Michael Austin, David Theodore Blaauw, Krisztian Flautner, and Trevor Nigel Mudge. Data retention latch provision within integrated circuits, December 18 2007. US Patent 7,310,755.

[Ada02] R.D. Adams. *High Performance Memory Testing : Design Principles, Fault Modeling and Self-Test*. Frontiers in Electronic Testing. Springer, 2002.

[ALRL04] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *Dependable and Secure Computing, IEEE Transactions on*, 1(1) :11–33, 2004.

[APZM07] M. Agarwal, B.C. Paul, Ming Zhang, and S Mitra. Circuit failure prediction and its application to transistor aging. In *VLSI Test Symposium, 2007. 25th IEEE*, pages 277–286, May 2007.

[BA00] Michael Bushnell and Vishwani Agrawal. *Essentials of electronic testing for digital, memory, and mixed-signal VLSI circuits*, volume 17. Springer, 2000.

[Bar99] M. Barr. *Programming Embedded Systems in C and C++*. O'Reilly Series. O'Reilly, 1999.

[BDM02] K.A. Bowman, S.G. Duvall, and J.D. Meindl. Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. *Solid-State Circuits, IEEE Journal of*, 37(2) :183–190, Feb 2002.

[BKL+08] D. Blaauw, S. Kalaiselvan, K. Lai, Wei-Hsiang Ma, S. Pant, C. Tokunaga, S. Das, and D. Bull. Razor ii : In situ error detection and correction for pvt and ser tolerance. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, pages 400–622, Feb 2008.

[BKN+03] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A Keshavarzi, and V. De. Parameter variations and impact on circuits and microarchitecture. In *Design Automation Conference, 2003. Proceedings*, pages 338–342, June 2003.

[Boh09] Mark Bohr. The new era of scaling in an soc world. In *Solid-State Circuits Conference-Digest of Technical Papers, 2009. ISSCC 2009. IEEE International*, pages 23–28. IEEE, 2009.

[Bor05] S. Borkar. Designing reliable systems from unreliable components : the challenges of transistor variability and degradation. *Micro, IEEE*, 25(6) :10–16, Nov 2005.

[BTK+09] K.A. Bowman, J.W. Tschanz, Nam Sung Kim, J.C. Lee, C.B. Wilkerson, S.L. Lu, T. Karnik, and V.K. De. Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. *Solid-State Circuits, IEEE Journal of*, 44(1) :49–63, Jan 2009.

[CA89] K. T Cheng and V.D. Agrawal. An economical scan design for sequential logic test generation. In *Fault-Tolerant Computing, 1989. FTCS-19. Digest of Papers., Nineteenth International Symposium on*, pages 28–35, June 1989.

[CA90] Kwang-Ting Cheng and V.D. Agrawal. A partial scan method for sequential circuits with feedback. *Computers, IEEE Transactions on*, 39(4) :544–548, Apr 1990.

[CG11] S. Churiwala and S. Garg. *Principles of VLSI RTL Design : A Practical Guide*. SpringerLink : Bücher. Springer, 2011.

[Dar97] Alain Dargelas. *Approche multi-stratégique pour la génération déterministe de vecteurs de test de circuits séquentiels synchrones*. PhD thesis, 1997. Thèse de doctorat dirigée par Bertrand, Yves Électronique, optronique et systèmes Montpellier 2 1997.

[EBSLM97] M. Eisele, J. Berthold, D. Schmitt-Landsiedel, and R. Mahnkopf. The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 5(4) :360–368, Dec 1997.

[Eic91] E.B. Eichelberger. *Structured logic testing*. Prentice Hall series in computer engineering. Prentice Hall, 1991.

[EKD+03] D. Ernst, Nam Sung Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. Razor : a low-power pipeline based on circuit-level timing speculation. In *Microarchitecture, 2003. MICRO-36. Proceedings. 36th Annual IEEE/ACM International Symposium on*, pages 7–18, Dec 2003.

[GB90] R. Gupta and M.A Breuer. The ballast methodology for structured partial scan design. *Computers, IEEE Transactions on*, 39(4) :538–544, Apr 1990.

[HAP+05] M. Horowitz, E. Alon, D. Patil, S. Naffziger, Rajesh Kumar, and K. Bernstein. Scaling, power, and the future of cmos. In *Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International*, pages 7 pp.–15, Dec 2005.

[HF74] John P Hayes and Arthur D Friedman. Test point placement to simplify fault detection. *Computers, IEEE Transactions on*, 100(7) :727–735, 1974.

[HM81] Michael John Howes and David Vernon Morgan. Reliability and degradation : semiconductor devices and circuits. *Chichester, Sussex, England and New York, Wiley-Interscience, 1981. 454 p*, 1, 1981.

[ITC] ITC99. Itc'99 benchmarks. CAD Group, Politecnico di Torino, `http://www.cad.polito.it/downloads/tools/itc99.html`.

[JG03] Niraj K Jha and Sandeep Gupta. *Testing of digital systems*. Cambridge University Press, 2003.

[Keh93] T. Kehl. Hardware self-tuning and circuit performance monitoring. In *Computer Design : VLSI in Computers and Processors, 1993. ICCD '93. Proceedings., 1993 IEEE International Conference on*, pages 188–192, Oct 1993.

[KML+06] N. Kranitis, A Merentitis, N. Laoutaris, G. Theodorou, A Paschalis, D. Gizopoulos, and C. Halatsis. Optimal periodic testing of intermittent faults in embedded pipelined processor applications. In

Design, Automation and Test in Europe, 2006. DATE '06. Proceedings, volume 1, pages 1–6, March 2006.

[LCAG13] Liangzhen Lai, Vikas Chandra, Robert Aitken, and Puneet Gupta. Slackprobe : A low overhead in situ on-line timing slack monitoring methodology. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 282–287, March 2013.

[LEV14] Régis LEVEUGLE. Test des circuits intégrés numériques notions de base, génération de vecteurs. *Techniques de l'ingénieur Architecture et tests des circuits numériques*, base documentaire : TIB276DUO.(ref. article : e2460), 2014. fre.

[MABF07] T.N. Mudge, T.M. Austin, D.T. Blaauw, and K. Flautner. Data retention latch provision within integrated circuits, December 18 2007. US Patent 7,310,755.

[MT00] E.J. McCluskey and Chao-Wen Tseng. Stuck-fault tests vs. actual defects. In *Test Conference, 2000. Proceedings. International*, pages 336 –342, 2000.

[Nav10] Z. Navabi. *Digital System Test and Testable Design : Using HDL Models and Architectures*. SpringerLink : Bücher. Springer, 2010.

[Nic97] M. Nicolaidis. On-line testing for VLSI. In *Test Conference, 1997. Proceedings., International*, pages 1042–, Nov 1997.

[Nic98] M. Nicolaidis. Design for soft-error robustness to rescue deep submicron scaling. In *Test Conference, 1998. Proceedings., International*, pages 1140–, Oct 1998.

[Nic99] M. Nicolaidis. Time redundancy based soft-error tolerance to rescue nanometer technologies. In *VLSI Test Symposium, 1999. Proceedings. 17th IEEE*, pages 86–94, 1999.

[NR06] M. Nourani and Arun Radhakrishnan. Testing on-die process variation in nanometer vlsi. *Design Test of Computers, IEEE*, 23(6) :438–451, June 2006.

[NZ99] M. Nicolaidis and Y. Zorian. Scaling deeper to submicron : on-line testing to the rescue. In *Design, Automation and Test in Europe Conference and Exhibition 1999. Proceedings*, pages 432–, March 1999.

[Pat98] J.H. Patel. Stuck-at fault : a fault model for the next millennium. In *Test Conference, 1998. Proceedings., International*, pages 1166–, Oct 1998.

[RBB+11] Bettina Rebaud, Marc Belleville, Edith Beigné, Christian Bernard, Michel Robert, Philippe Maurine, and Nadine Azemard. Timing slack monitoring under process and environmental variations : Application to a dsp performance optimization. *Microelectronics Journal*, 42(5) :718–732, 2011.

[Sam03] S.B. Samaan. Parameter variation probing technique, March 18 2003. US Patent 6,535,013.

[SGH+07] Jared C. Smolens, Brian T. Gold, James C. Hoe, Babak Falsafi, and Ken Mai. Detecting emerging wearout faults. In *In Proceedings of the IEEE Workshop on Silicon Errors in Logic - System Effects*, 2007.

[Sin12] Ozgur Sinanoglu. Eliminating performance penalty of scan. In *VLSI Design (VLSID), 2012 25th International Conference on*, pages 346–351. IEEE, 2012.

[TKN+02] J. Tschanz, James Kao, S. Narendra, R. Nair, D. Antoniadis, A Chandrakasan, and Vivek De. Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. In *Solid-State Circuits Conference, 2002. Digest of Technical Papers. ISSCC. 2002 IEEE International*, volume 2, pages 344–539, Feb 2002.

[Via96] Hélène Viallon. *Contribution au test intégré : optimisation des générateurs de vecteurs de test matériels et leur adaptation à la détection de fautes complexes*. PhD thesis, 1996. Thèse de doctorat dirigée par LANDRAULT, C. Électronique, optronique et systèmes Montpellier 2 1996.

[WA73] Michael John Yates Williams and James B Angell. Enhancing testability of large-scale integrated circuits via test points and additional logic. *IEEE Transactions on Computers*, 22(1) :46–60, 1973.

[WH] N.H.E. Weste and D.M. Harris. *CMOS VLSI Design : A Circuits and Systems Perspective*. Pearson Education India.

[WP83] T.W. Williams and K.P. Parker. Design for testability - a survey. *Proceedings of the IEEE*, 71(1) :98–112, Jan 1983.

[ZSK05] Ming Zhang, Quan Shi, and Kee Sup Kim. Robust system design with built-in soft-error resilience. 2005.

LISTE DES FIGURES

LISTE DES TABLEAUX

PUBLICATIONS

Articles de conférences

1. S. SARRAZIN, S. EVAIN, I. MIRO-PANADES, L. ALVES DE BARROS NAVINER, V. GHERMAN "*Flip-Flop Selection for In-Situ Slack-Time Monitoring based on the Activation Probability of Timing-Critical Paths.*" IEEE INTERNATIONAL ON-LINE TEST SYMPOSIUM (IOLTS), 2014, pp. 160-163.
2. S. SARRAZIN, S. EVAIN, I. MIRO-PANADES, A. VALENTIAN, S. PAJANIRADJA, L. ALVES DE BARROS NAVINER, V. GHERMAN "*Shadow-scan design with low latency overhead and in-situ slack-time monitoring.*" IEEE EUROPEAN TEST SYMPOSIUM (ETS), 2014, pp. 1-6.
3. S. SARRAZIN, S. EVAIN, L. ALVES DE BARROS NAVINER, Y. BONHOMME, V. GHERMAN "*Scan design with shadow flip-flops for low performance overhead and concurrent delay fault detection.*" IEEE DESIGN AND TEST IN EUROPE (DATE), 2013, pp. 1077–1082.

WorkShop

1. S. SARRAZIN, S. EVAIN, L. ALVES DE BARROS NAVINER, V. GHERMAN "*Shadow-scan design with reduced latency overhead*" GDR-SOCSIP, 2014.

Brevet

1. S. SARRAZIN, S. EVAIN, V. GHERMAN BREVET FRANCE N°1450725 "*Dispositif d'élément séquentiel scan*", Janvier 2014.

Fiabilisation et test des processeurs dans un contexte embarqué

Sébastien SARRAZIN

RÉSUMÉ :

La réduction des marges temporelles dans les circuits synchrones est une manière d'améliorer leur performance. En cas de vieillissement, de fluctuations de la tension d'alimentation ou de la température du milieu de mission, des réductions sévères des marges temporelles peuvent néanmoins avoir un impact négatif sur la fiabilité des circuits. La réduction des marges temporelles sans dégradation de la fiabilité peut être réalisée à l'aide des informations d'un contrôle en ligne de ces marges. Cette thèse porte sur l'étude du suivi en ligne des marges temporelles des circuits intégrés synchrones. La première contribution de ce travail consiste en une nouvelle solution *shadow-scan* bien adaptée au suivi en ligne des marges temporelles, permettant une mise en œuvre de bascules *scan* plus rapides et pouvant être gérées de façon transparente par les outils de conception du commerce.

Une approche naturelle de mise en œuvre du suivi des marges temporelles est le déploiement de moniteurs sur tous les chemins critiques ou susceptibles de le devenir à cause du vieillissement ou des variations dues au processus de production. Dans des circuits très contraints en termes de temps de propagation, avec un grand nombre de chemins critiques, cette approche peut conduire à un surcoût en surface trop important pour un gain en performance limité. Afin de pouvoir réduire le nombre de moniteurs avec un impact limité sur la qualité du monitoring, la seconde contribution de ce travail est la proposition d'une nouvelle méthode d'évaluation de la qualité du monitoring d'un groupe de bascules. Cette méthode est basée sur l'estimation de la probabilité d'activation des moniteurs. Deux métriques sont proposées pour quantifier la qualité du suivi des marges temporelles. La première sert à estimer la couverture temporelle d'un groupe de bascules, alors que la deuxième permet d'évaluer la couverture spatiale.

Enfin, en se basant sur ces deux métriques, la dernière contribution de ce travail est la démonstration du fait qu'il est possible avec un surcoût limité d'améliorer significativement la qualité du monitoring d'un circuit si les moniteurs sont adaptés aux cônes logiques qu'ils surveillent.

MOTS-CLEFS :

scan design ; shadow-scan ; impact en latence ; monitoring en ligne ; monitoring de marges temporelles ; variations dynamiques ; probabilité d'activation.

ABSTRACT :

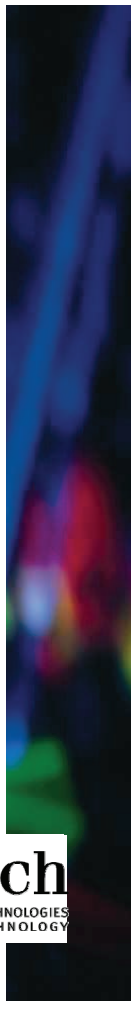
Slack-time reduction is a way to improve the performance of synchronous sequential circuits. In the presence of circuit wear-out, supply voltage fluctuations and temperature variations, aggressive slack-time reduction can be achieved based on adaptive voltage and frequency scaling with feedback from in-situ slack-time monitoring. The first contribution of this work consist of a new shadow-scan solution which facilitates the implementation of faster scan Flip-Flops (FFs), enables in-situ slack-time monitoring and can be transparently handled by commercial tools for automated scan stitching and automated test pattern generation.

A natural approach is to place in-situ slack-time monitors close to all sequential elements with incoming timing-critical paths or susceptible to become timing-critical due to wear-out or manufacturing variability. In latency-constrained circuits with large ratios of timing-critical paths, this methodology may result in large area overheads and minor power improvements. The second contribution of this work is an evaluation methodology of the monitoring quality delivered by a set of FFs. This methodology estimates monitor activation probabilities based on which two evaluation metrics are provided. On one hand, the expected ratio of clock cycles with at least one monitor activated can be used to estimate the temporal coverage of the in-situ slack-time monitoring scheme. On the other hand, the expected number of activated monitors per clock cycle can be used to evaluate the spatial coverage of the monitoring scheme.

Finally, based on these metrics, it is shown that the monitoring quality can be significantly improved if the size of the detection window of each in-situ slack-time monitor is correlated to the slack-time of the monitored timing-critical paths.

KEY-WORDS :

scan design ; shadow-scan ; latency overhead ; on-line monitoring ; in-situ slack-time monitoring ; dynamic variations ; monitor activation probability.



TELECOM ParisTech

EDiTE

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY