TECHNIQUES
DE L'INGÉNIEUR

Réf. : **E2461 V1**

Date de publication :
10 août 2002

Test des circuits intégrés numériques - Conception orientée testabilité

Cet article est issu de : **Électronique - Photonique | Électronique**

par **Régis LEVEUGLE**

Pour toute question :
Service Relation clientèle
Techniques de l'Ingénieur
Immeuble Pleyad 1
39, boulevard Ornano
93288 Saint-Denis Cedex

Par mail :
infos.clients@teching.com
Par téléphone :
00 33 (0)1 53 35 20 20

Test des circuits intégrés numériques

Conception orientée testabilité

par **Régis LEVEUGLE**
Ingénieur de l'École nationale supérieure d'électronique et de radioélectricité de Grenoble (ENSERG)
Professeur à l'Institut national polytechnique de Grenoble (INPG)
Laboratoire des techniques de l'informatique et de la microélectronique pour l'architecture d'ordinateurs (TIMA)

Dans la première partie intitulée « Test des circuits intégrés numériques – Notions de base. Génération de vecteurs » [E 2 460], les principaux concepts du domaine ont été introduits. Cette deuxième partie présente plus en détail différentes techniques pouvant être mises en œuvre, pendant la conception d'un circuit, pour faciliter son test en fin de fabrication ou dans l'équipement. Quelques techniques de base employées pour réaliser un test pendant l'exécution de l'application sont également introduites.

Cet article constitue la deuxième partie d'un ensemble consacré aux tests des circuits intégrés numériques :

— *Test des circuits intégrés numériques – Notions de base. Génération de vecteurs ;*
— *Test des circuits intégrés numériques – Conception orientée testabilité* [E 2 461] *;*
— *Test des circuits intégrés numériques – Pour en savoir plus* [Doc. E 2 462].

Nous rappelons au lecteur qu'un glossaire des termes utilisés dans l'article est présenté dans la première partie de l'article ([E 2 460], encadré 1).

1. Techniques de conception pour augmenter la testabilité d'un circuit

Ce paragraphe présente les techniques de base permettant d'améliorer la testabilité d'un circuit pour faciliter en particulier son test hors ligne. En fonction des contraintes de conception d'un circuit donné, ces différentes approches peuvent être combinées pour donner un nombre considérable de variantes, détaillées dans la littérature (cf. *Pour en savoir plus* [Doc. E 2 462]. Nous nous limiterons ici à une introduction rapide permettant de comprendre ce que recouvrent les termes les plus employés.

1.1 Types d'approches

Les techniques de conception orientées vers le test sont habituellement regroupées en deux classes : les **approches structurées et les approches « *ad hoc* »**. Elles peuvent par ailleurs viser le support du test externe ou la réalisation d'un autotest.

Une **approche** est dite **structurée** lorsqu'un travail préliminaire de segmentation a été réalisé sur le circuit, conduisant à placer selon une stratégie précise l'ensemble des cellules à vocation de test.

À l'inverse, une **approche « *ad hoc* »** place des éléments de test au fur et à mesure de la détection de problèmes d'accès ou d'observabilité. Une approche « *ad hoc* » tend donc en général à ajouter plus d'éléments que ceux réellement nécessaires pour atteindre un niveau de testabilité donné. De plus, les éléments étant ajoutés sans stratégie d'ensemble, le nombre de signaux de commande à gérer pendant le test est généralement élevé et la gestion des phases de test peut devenir très complexe.

Il est donc recommandé, en général, d'utiliser une approche structurée.

1.2 Éléments de base pour le support du test externe

Ce paragraphe résume les principaux éléments de base utilisés par les différentes approches.

Dans une approche « *ad hoc* », il est aussi possible d'ajouter à l'intérieur du circuit des plots (carrés dessinés sur le niveau métallique supérieur) similaires à ceux placés dans la couronne du circuit pour les entrées/sorties primaires. Ces plots peuvent être utilisés pour descendre des pointes de test dans le cœur du circuit pendant une phase de déverminage ou, plus classiquement, pour lire le niveau logique de signaux internes à l'aide d'un microscope électronique (*e-beam pads*). Dans les deux cas, ces plots occupent une surface considérable et leur utilisation est donc limitée à des circuits prototypes suffisamment délicats à concevoir pour qu'une seconde version soit prévue avant le passage en production de série (version dans laquelle les plots de test seront supprimés). À l'inverse, les autres éléments ajoutés pour le test (portes logiques, multiplexeurs, bascules, ...) restent utilisés après la phase de déverminage.

1.2.1 Portes élémentaires et arbres de portes

Le premier ensemble d'éléments utilisable pour augmenter la testabilité est constitué de toutes les portes logiques élémentaires, en particulier ET, OU et XOR (ou XNOR) .

Une **porte OU (respectivement ET)** à deux entrées, insérée entre deux éléments logiques quelconques d'un circuit, permet en effet de forcer facilement, à partir d'une entrée externe, un niveau logique 1 (respectivement 0) sur le nœud interne choisi. Ce type de modification peut cependant entraîner un nombre global de signaux de commande élevé, chaque porte logique insérée étant *a priori* commandée indépendamment de toutes les autres.

Les **portes XOR ou XNOR** sont davantage utilisées pour augmenter l'observabilité des nœuds internes. Les différents nœuds à observer sont alors placés en entrée d'un arbre constitué de telles portes, la sortie de l'arbre correspondant à une sortie primaire utilisée pendant la phase de test.

Ces types de modifications sont classiquement employés dans une approche « *ad hoc* ». Cependant, elles peuvent aussi être proposées en complément d'un outil de génération de vecteurs de test. L'outil identifie alors l'ensemble minimum de nœuds à modifier pour obtenir le taux de couverture souhaité et génère des vecteurs de test permettant de minimiser le nombre de signaux de commande. Ce type d'approche avait été proposé par exemple dans les outils de la société SunRise.

1.2.2 Test parallèle

L'insertion de multiplexeurs permet facilement de rendre certains signaux, voire un ou plusieurs blocs du circuit, directement contrôlables et/ou observables par les entrées et/ou sorties primaires du circuit. La commande de plusieurs multiplexeurs par un même signal permet d'amener en parallèle les différents bits d'entrée d'un vecteur de test ou d'observer en parallèle un ensemble de nœuds internes.

Ce type d'insertion peut être utilisé aussi bien dans une approche « *ad hoc* » que dans une approche structurée. Pour une telle insertion, le nombre de signaux de commande indépendants et la complexité de leur gestion pendant le test croissent en fonction de la complexité du circuit et du nombre de bancs de multiplexeurs insérés.

1.2.3 Test sériel

Par opposition au test parallèle, il est possible de faire progresser de proche en proche les différents bits d'un vecteur de test, que ce soit pour entrer un vecteur ou sortir la valeur présente sur des nœuds internes. Ceci peut être réalisé à l'aide d'un registre à décalage. Deux approches complémentaires peuvent être employées pour créer ce registre : la modification de points de mémorisation utilisés par les fonctions du circuit, ou l'ajout de points de mémorisation utilisés uniquement pour le test.

La figure **1** illustre la modification d'une bascule D, utilisée dans le circuit en mode fonctionnel normal, pour augmenter à la fois la contrôlabilité et l'observabilité du nœud pendant les phases de test. Le multiplexeur ajouté permet de placer cette bascule dans un registre à décalage et donc de la charger depuis l'extérieur du circuit par l'intermédiaire de l'entrée série (augmentation de la contrôlabilité en sortie de la bascule) ou de lire sa valeur sur une sortie primaire par l'intermédiaire de la sortie série (augmentation de l'observabilité en entrée de la bascule). Il faut noter que cette modification a un impact sur le chemin critique du circuit, si la bascule modifiée se trouve sur ce chemin, soit à cause du temps de traversée du multiplexeur sur l'entrée, soit à cause de l'augmentation de charge sur la sortie. Le même type de modification est envisageable sur un verrou, en rajoutant éventuellement un second verrou pour créer une paire maître-esclave pouvant être connectée dans le registre à décalage.

La figure **2** illustre l'ajout d'un point de mémorisation utilisé uniquement pour le test. Dans les deux cas représentés, le même type de matériel est ajouté, à savoir un point de mémorisation et un multiplexeur (ou une cellule parfois appelée *T-cell*). Le point de mémorisation devant être connecté dans un registre à décalage, il correspond soit à une bascule, soit à deux verrous montés en maître-

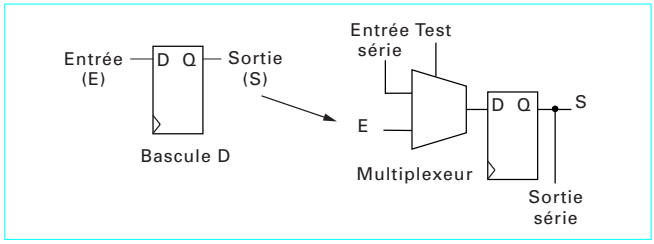

Figure 1 – Principe de modification d'un point mémoire fonctionnel pour le test sériel

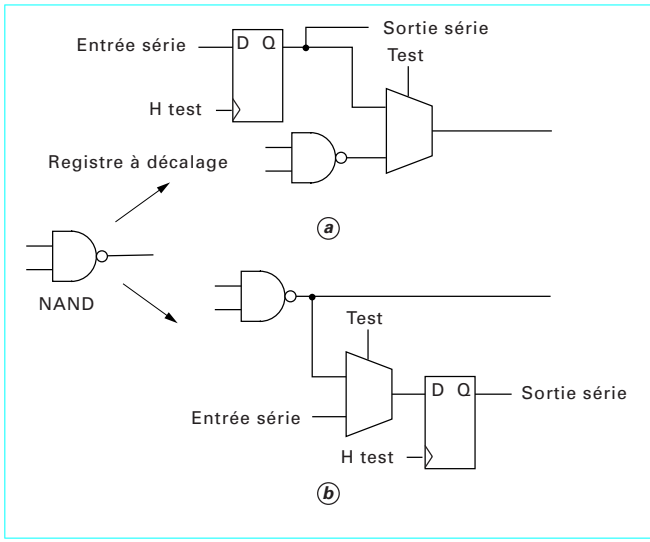

Figure 2 – Principes d'ajout d'un point de mémorisation pour le test sériel, dans le but d'augmenter la contrôlabilité (*a*) ou l'observabilité (*b*) d'un nœud interne

esclave. La connexion de la bascule en entrée du multiplexeur permet d'augmenter la contrôlabilité en sortie de l'élément logique fonctionnel ; la connexion de la bascule en sortie du multiplexeur permet d'augmenter l'observabilité en sortie de l'élément logique fonctionnel. L'impact sur le chemin critique est généralement plus faible dans le second cas, puisqu'il n'est dû qu'à la modification de la charge sur le nœud observé au lieu d'une traversée de multiplexeur.

L'avantage principal d'un test sériel par rapport à un test parallèle est que le nombre de signaux externes à ajouter pour le test peut être très faible et indépendant de la complexité du circuit. En effet, si tous les éléments de mémorisation modifiés ou ajoutés sont placés dans un même registre à décalage, le test n'utilise qu'une entrée série, une sortie série, une horloge pour le chargement des bascules et un signal de commande permettant de sélectionner le chargement parallèle ou le décalage du registre. En revanche, le nombre de cycles d'horloge nécessaire pour l'application d'un vecteur de test croît avec le nombre d'étages du registre à décalage et peut donc devenir très élevé. Plusieurs registres peuvent alors être employés pour obtenir un compromis satisfaisant entre le temps de test et le nombre de signaux de commande à gérer.

1.3 Contrôlabilité ou observabilité

Dans le cas de la figure **1**, la modification entraîne à la fois une augmentation de contrôlabilité et d'observabilité. En revanche, dans le cas de la figure **2**, la modification choisie n'améliore que l'une des deux caractéristiques ; un choix est alors à faire.

Un registre créé pour augmenter l'observabilité d'un certain nombre de nœuds internes selon le principe illustré en figure **2*b*** peut être employé indépendamment du fonctionnement du circuit s'il est commandé par des signaux utilisés uniquement en mode de test et connectés à des broches du circuit. Ce type de registre peut donc être employé pour échantillonner des nœuds internes pendant le fonctionnement normal du circuit dans un équipement, par exemple dans une optique de réalisation d'un test en ligne.

Dans la plupart des cas, ce choix est indifférent, guidé par l'impact acceptable sur le chemin critique ou dicté par le problème de testabilité induit par la structure globale du circuit considéré. Pour certaines applications ayant des contraintes fortes en sûreté de fonctionnement, comme par exemple l'avionique ou le domaine spatial, il est cependant important de préférer systématiquement les modifications orientées vers l'observabilité. En effet, ce type de modification est moins intrusif et permet d'éviter d'induire des modes de défaillance supplémentaires dans le circuit.

Exemple : dans le cas présenté figure **2**, supposons que le vieillissement du circuit conduise à un dysfonctionnement du multiplexeur ajouté pour le test (pour simplifier, imaginons un collage de la sortie de cet élément).

Le montage permettant d'augmenter l'observabilité se trouve en dehors du chemin fonctionnel ; le dysfonctionnement du multiplexeur n'aura aucune influence sur la fonction réalisée par le circuit et donc sur l'application. Lors d'un test de maintenance suivant l'occurrence de la panne, le circuit sera identifié comme défectueux et remplacé.

Dans le cas du montage améliorant la contrôlabilité en mode de test, le dysfonctionnement du multiplexeur peut avoir un impact immédiat sur l'application ; ce dysfonctionnement potentiel est donc à prendre en compte lors de l'analyse des modes de défaillance critiques, pour l'évaluation de la fiabilité et/ou de la sécurité-innocuité de l'équipement. Le premier cas est donc préférable.

1.4 Techniques structurées de test parallèle

L'insertion de multiplexeurs, présentée en paragraphe 1.2.2, peut être guidée par un partitionnement préalable du circuit identifiant des blocs à tester indépendamment. Des multiplexeurs sont alors insérés sur les différentes entrées-sorties des blocs ne correspondant pas à des entrées-sorties primaires, de façon à les connecter en mode test sur des accès plus contrôlables ou observables. Ce principe est illustré figure **3**.

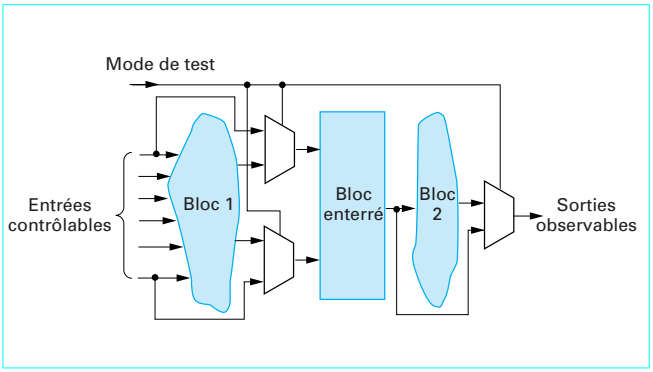

Figure 3 – Principe d'insertion de multiplexeurs selon une approche structurée

Dans une approche structurée, le nombre de signaux de commande différents se déduit du nombre de blocs identifiés lors du partitionnement et leur gestion est relativement simple puisque chaque signal prend une valeur déterminée en fonction du bloc en cours de test. Le nombre d'entrées-sorties primaires rajoutées au circuit peut être limité en réutilisant des broches d'entrée ou de sortie existantes, comme illustré en figure **3**.

L'ajout de multiplexeurs sur le chemin critique pouvant être inacceptable, et la surface additionnelle pouvant être strictement limitée, il est possible de limiter les modifications à certains blocs et/ou chemins dans le circuit (compromis entre performances, testabilité et surface).

1.5 Techniques structurées de test sériel

Lorsque le nombre de broches et/ou le nombre d'éléments logiques à ajouter doit être limité strictement, une technique structurée de test parallèle est souvent difficile à appliquer. Il est alors possible de se tourner vers une technique de test sériel, au prix d'un allongement notable du temps d'application des vecteurs de test.

1.5.1 Utilisation de verrous

La technique la plus célèbre dans le cas de circuits utilisant des verrous et des horloges biphasées est nommée *Level Sensitive Scan Design* (LSSD). Cette technique est essentiellement fondée sur l'utilisation de cellules verrou particulières et nécessite la génération d'horloges séparées pour réaliser les transferts entre verrous.

Deux méthodes différentes sont utilisées, selon le type de fonction à réaliser en dehors du mode de test :

— si la fonction nominale est un verrou, la modification revient à ajouter un verrou supplémentaire pour réaliser l'étage de base du registre à décalage (*Single Latch Design*) ;

— si la fonction nominale dans le circuit est une bascule maître-esclave réalisée avec deux verrous (*Double Latch Design*), le registre à décalage est réalisé en ajoutant seulement les connexions série. Cette technique étant protégée par un brevet IBM, elle reste peu utilisée par rapport à l'approche *scanpath*.

1.5.2 Utilisation de bascules : le « *scanpath* »

Le type de modification structurée (voire systématique) le plus répandu est dénommé insertion de *scanpath*, ou simplement insertion de *scan* et utilise le principe de modification présenté en figure **1**.

La modification peut être systématique sur toutes les bascules du circuit ; il s'agit alors d'un *scan* complet, ou *full scan*. Elle peut aussi être limitée à certains éléments de mémorisation, déterminés après un partitionnement du circuit ou en fonction de certains critères ; il s'agit alors d'un *scan* partiel, ou *partial scan*.

■ Le principe d'insertion d'un ***scan* complet** est illustré en figure **4** sur le modèle de Huffman d'un circuit. Tout circuit séquentiel peut être décomposé en un bloc de logique combinatoire et en un ensemble d'éléments de mémorisation (ici, des bascules).

L'insertion du *scan* revient à modifier toutes les bascules pour les connecter en registre à décalage. En mode de test, toutes les entrées de bascules deviennent donc l'équivalent de sorties primaires observables et toutes les sorties de bascules deviennent l'équivalent d'entrées primaires contrôlables.

Les bascules sont testées directement par le décalage de vecteurs dans le registre ; il ne reste donc à générer des vecteurs de test que pour la logique combinatoire, ce que les outils savent faire de façon efficace, selon une approche structurelle, pour les modèles de fautes les plus courants. Le décalage et le chargement du registre en mode de test peuvent être commandés par l'intermédiaire de l'horloge système (en particulier dans le cas des circuits n'ayant qu'une horloge) ou par une horloge de test. Dans tous les cas, les signaux de validation de chargement doivent être gérés explicitement pendant le test.

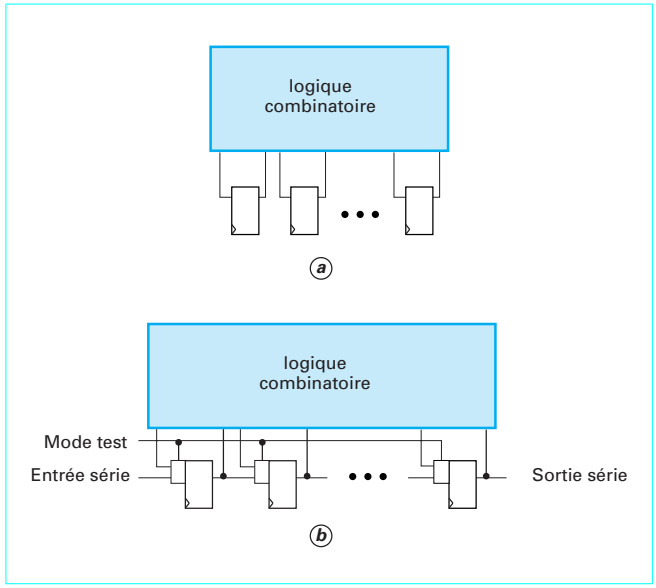

Figure 4 – Principe d'implantation d'un *scanpath* complet : circuit initial (*a*) et circuit modifié (*b*)

Le *scan* complet permet donc d'éviter une phase de partitionnement et d'analyse de testabilité minutieuse, tout en simplifiant au maximum la génération de vecteurs de test ayant un taux de couverture très élevé. En revanche, il entraîne obligatoirement un allongement du chemin critique, et le surcoût en surface n'est pas négligeable. De plus, le registre à décalage peut être très long dans le cas d'un circuit complexe. Ces limitations conduisent à différentes adaptations : découpage du registre à décalage en plusieurs registres pouvant être chargés simultanément et/ou *scan* partiel.

■ L'insertion d'un ***scan* partiel** permet par exemple d'éviter l'allongement du chemin critique ou de ramener à moindre coût le test d'une structure séquentielle à fort rebouclage à celui d'une structure séquentielle à faible rebouclage. En revanche, le circuit reste séquentiel en mode de test et les vecteurs de test ne peuvent donc pas être obtenus avec un simple outil d'ATPG combinatoire.

■ Lorsque l'insertion de *scan* est réalisée de façon hiérarchique, bloc par bloc, les entrées-sorties primaires d'un bloc ne sont pas forcément connectées sur des bascules ou des entrées-sorties primaires du circuit. Ceci peut compliquer nettement l'obtention des vecteurs de test pour le circuit complet, après assemblage des blocs. Il peut alors être utile d'ajouter certains éléments de mémorisation selon le principe illustré par la figure **2**.

1.5.3 Précautions d'implantation pour un test sériel

L'implantation d'un registre à décalage pour propager les vecteurs de test à l'intérieur d'un circuit peut nécessiter un certain nombre de précautions, en fonction de la structure interne du circuit.

La figure **5** illustre un cas typique, correspondant à l'implantation d'un *scanpath* dans un bloc générant des commandes de portes trois états connectées sur un bus. En mode fonctionnel, les signaux de commande sont générés de façon à assurer que deux portes trois

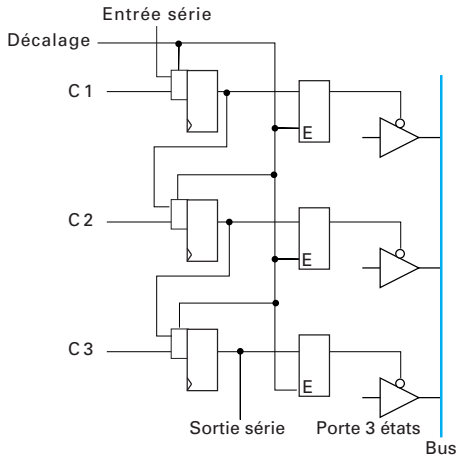

Figure 5 – Principe de modification pour le test série *scanpath* dans le cas d'un risque de contention de bus

états ne sont jamais validées simultanément. Il n'est plus possible d'assurer cette condition pendant le décalage d'un vecteur de test à l'intérieur du circuit, si les sorties du registre *scanpath* restent reliées directement sur les entrées de commande des portes trois états. En conséquence, il est nécessaire d'ajouter des verrous, permettant de maintenir une configuration valide en entrée des portes trois états, pendant le décalage des vecteurs de test. En dehors des phases de décalage, ces verrous sont transparents ; ils n'influent pas sur la fonctionnalité du circuit, mais modifient naturellement les temps de propagation (en plus de l'impact sur le coût d'implantation du *scanpath*).

Notons que l'activité des portes à l'intérieur du circuit est très élevée pendant les phases de décalage, les entrées des blocs combinatoires étant modifiées quasi aléatoirement, en fonction des vecteurs de test à propager sur la chaîne. Ceci peut conduire à une forte surconsommation pendant le test, induisant un stress inacceptable et réduisant la fiabilité du circuit. L'emploi de verrous en sortie du registre à décalage, selon le principe illustré par la figure **5**, permet aussi d'éliminer ce problème en maintenant une configuration stable des entrées de la logique combinatoire pendant les décalages. L'ajout d'un ensemble de verrous permet donc d'établir un compromis entre la consommation des blocs pendant le test, le coût en surface du *scanpath* et l'impact sur le chemin critique.

1.6 Autotests

Les méthodes présentées dans les paragraphes précédents (§ 1.2, § 1.4, § 1.5) ont pour objectif de faciliter le test appliqué depuis l'extérieur du circuit. Une autre approche consiste à implanter l'ensemble des dispositifs de test directement à l'intérieur du circuit, qui est alors capable de réaliser un autotest (cf. [E 2 460], § 1.2.3). L'avantage majeur de cette approche, outre la simplification des équipements de test externes, est la possibilité de réaliser un test à fréquence nominale, même pour des circuits très rapides.

On présente dans ce paragraphe les concepts de base pour l'implantation d'un bloc d'autotest (ou BIST).

1.6.1 Éléments de base

La plupart des blocs de BIST sont construits à partir d'un nombre réduit d'éléments de base. Les principaux sont les registres à décalage avec rebouclage linéaire, disposant d'une entrée série (LFSR : *Linear Feedback Shift Register*) ou de plusieurs entrées parallèles (MISR : *Multiple Input Shift Register*). Dans certains cas, ces éléments sont remplacés par des automates cellulaires (CA : *Cellular Array*). Ces automates ne seront pas détaillés ici ; on se reportera aux références [1] et [2] pour plus d'informations.

■ Différentes **structures** existent pour les LFSR et MISR, toutes basées sur un principe de division polynomiale.

● Sans rentrer dans les détails, un **LFSR** est un registre à décalage complété par un certain nombre de portes OU Exclusif permettant de réaliser l'entrée de l'information externe et des rebouclages internes. Le nombre et la position des portes OU Exclusif dépendent du polynôme diviseur choisi. Le degré de ce polynôme diviseur correspond au nombre de bascules dans le registre.

Pour un même polynôme, différentes structures équivalentes peuvent être utilisées, les deux plus communes étant la **structure « standard »** et la **structure « modulaire »**, illustrées en figure **6** :

— dans le cas de la structure « standard », tous les rebouclages internes sont ramenés sur l'entrée du LFSR ; les portes OU Exclusif sont donc placées en dehors du registre à décalage, ce qui facilite l'implantation de celui-ci. En revanche, la fréquence de fonctionnement maximum de cette structure est limitée par le chemin de rebouclage externe (qui peut de plus correspondre à une interconnexion assez longue si le nombre de bascules est élevé) ;

— la structure « modulaire » permet d'atteindre des fréquences de fonctionnement plus élevées. En revanche, le registre à décalage doit être interrompu en certains points pour insérer les portes OU Exclusif réalisant les rebouclages internes.

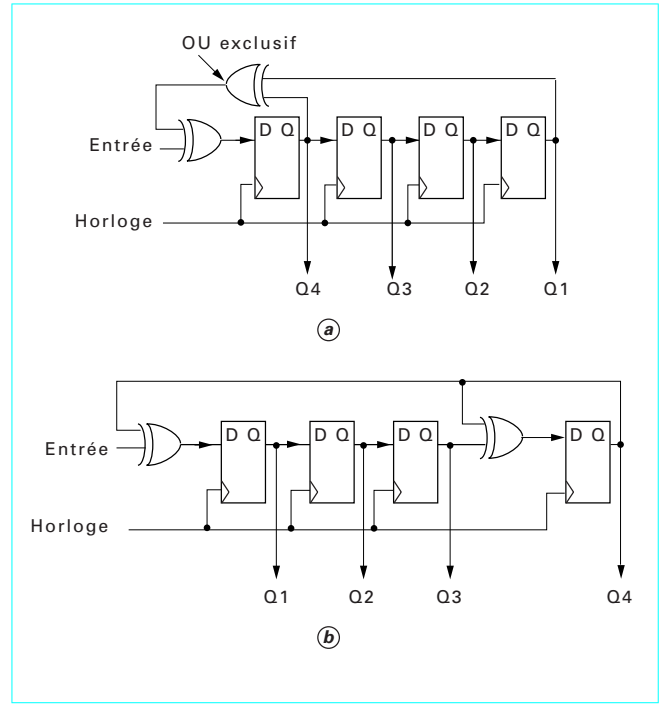

Figure 6 – Exemples d'implémentation d'un LFSR : structure « standard » (*a*) et structure « modulaire » (*b*) pour le polynôme diviseur $G(x) = x^4 + x^3 + 1$

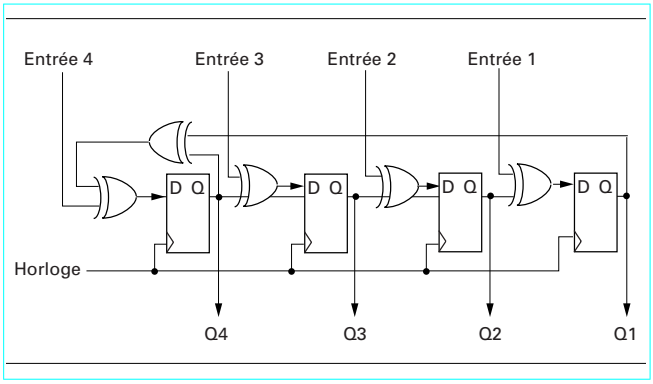

Figure 7 – Implémentation d'un MISR en structure « standard » pour le polynôme diviseur $G(x) = x^4 + x^3 + 1$

● Un MISR est implanté simplement en rajoutant à la structure du LFSR un ensemble de portes OU Exclusif permettant d'entrer simultanément un bit d'information externe au niveau de plusieurs bascules du registre. Le plus souvent, le nombre de bits du registre correspond au nombre de bits d'information à lire en parallèle. Ceci est illustré par la figure **7**.

Un LFSR ou un MISR est classiquement employé pour réaliser un test compact (cf. [E 2 460], § 1.2.6). Le LFSR réalise une compaction temporelle, tandis que le MISR réalise une compaction à la fois spatiale et temporelle. Dans les deux cas, l'objectif est de limiter au strict minimum la probabilité d'obtenir une signature juste malgré des erreurs dans les informations compactées (probabilité de masquage). Des études ont montré qu'il faut pour cela utiliser un polynôme diviseur premier primitif ; la probabilité de masquage s'établit alors en valeur asymptotique (c'est-à-dire pour une séquence suffisamment longue) à 2^{-k}, où k est le nombre de bascules dans le registre (ou encore le degré du polynôme choisi).

La figure **8** illustre l'implantation d'un **LFSR « autonome »**. Ce type de structure est dérivé de la structure du LFSR en supprimant l'entrée d'information externe. La séquence de vecteurs obtenue sur les sorties des bascules en faisant évoluer le signal d'horloge est alors exclusivement déterminée par le choix du polynôme diviseur. Ce principe peut être employé pour générer une séquence pseudo-aléatoire. La longueur de la séquence générée et la valeur des vecteurs dépendent du polynôme diviseur choisi et de la valeur initiale dans le registre. La séquence maximale pour un LFSR comportant k bascules est de longueur $2^k - 1$; elle est obtenue pour un polynôme diviseur premier primitif et une valeur initiale différente de zéro. Pour les structures de base d'un LFSR autonome, la valeur zéro est toujours un état puits, quel que soit le polynôme diviseur choisi. Si nécessaire, il est cependant possible de modifier légèrement la structure d'un LFSR autonome pour insérer la valeur zéro dans la séquence de vecteurs générée.

Les différentes fonctions LFSR, LFSR autonome, MISR et registre parallèle (ou à décalage) peuvent dans certains cas être combinées sur un même registre afin de limiter les coûts d'implantation ; un tel registre à usages multiples est fréquemment appelé BILBO (*Built-In Logic Block Observer*).

1.6.2 Autotest pseudo-aléatoire

Un autotest pseudo-aléatoire simple est illustré par la figure **9**. Il utilise un LFSR autonome comme générateur de vecteurs et un MISR pour compacter les résultats de test en une signature. Le polynôme diviseur du LFSR autonome et la valeur initiale placée dans ce registre sont déterminés pour maximiser le taux de couverture et/ou minimiser la longueur de la séquence de test à appliquer.

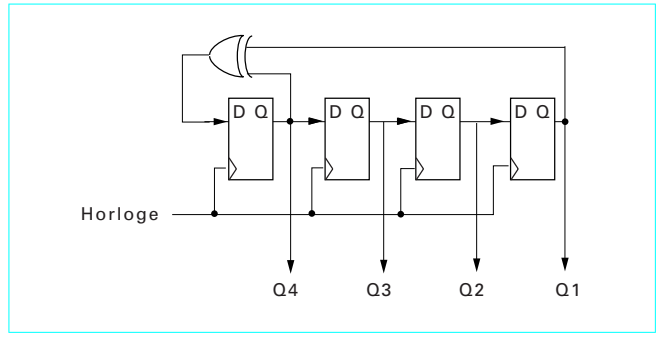

Figure 8 – Générateur pseudo-aléatoire réalisé avec un LFSR « autonome » de polynôme diviseur $G(x) = x^4 + x^3 + 1$, en structure « standard »

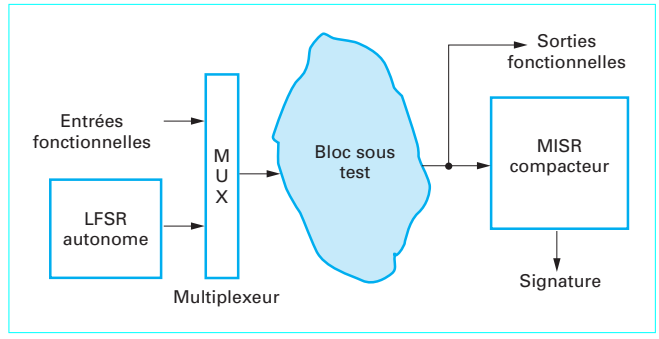

Figure 9 – Principe d'implémentation d'un autotest pseudo-aléatoire

Dans certains cas, il est possible de réduire la longueur de la séquence de test en réinitialisant le registre une ou plusieurs fois pendant l'application des vecteurs. Ceci permet, au prix d'une augmentation de complexité de la logique d'autotest, d'éviter l'application de sous-séquences longues ne permettant pas la détection de nouvelles fautes. Les sous-séquences inefficaces (pour un modèle de fautes donné) sont déterminées à l'aide de simulations de fautes. Cette approche à « semences multiples » permet de réduire le temps de test et l'énergie consommée pendant le test. Ce dernier point est par exemple intéressant pour augmenter l'autonomie d'un équipement alimenté sur batterie, si l'autotest est effectué à chaque mise sous tension.

De nombreuses variantes existent, basées sur ces principes. Elles visent essentiellement le test de logique aléatoire (blocs de cellules standards) et en particulier le test de blocs combinatoires. Ce type d'autotest peut aussi être adapté au cas d'une implantation avec un *scanpath*, en utilisant une sortie d'un LFSR autonome pour charger la chaîne de *scanpath* et un LFSR pour compacter la sortie série de cette chaîne.

1.6.3 Autotest avec séquence déterministe

Des algorithmes ont été développés, permettant de faire générer une séquence de vecteurs prédéterminée par un LFSR autonome (ou un automate cellulaire autonome) dont la structure est un peu modifiée. Il s'agit de choisir de façon efficace un (ou plusieurs) polynômes diviseurs et une (ou plusieurs) valeurs initiales afin de ne générer dans la séquence que les vecteurs de test souhaités. En général, ces algorithmes s'appliquent au cas de séquences non ordonnées, donc se limitent au test de logique combinatoire. L'intérêt est de pouvoir combiner certains avantages du test pseudo-aléatoire (cf. § 1.6.2) (faible coût d'implantation et test à fréquence nominale) avec l'avantage du test déterministe (longueur de séquence minimale pour un taux de couverture donné).

Bien sûr, il est aussi possible d'implanter un autotest déterministe en utilisant d'autres types de générateurs de vecteurs ; c'est généralement le cas pour le test de macrocellules.

1.6.4 Autotests pour macrocellules

Les autotests se sont tout d'abord généralisés dans l'industrie pour les macrocellules et en particulier les mémoires RAM (*random access memory*) ou ROM (*read only memory*). Pour ces macrocellules, l'approche pseudo-aléatoire n'est pas satisfaisante. Par ailleurs, des séquences longues et précises doivent souvent être générées pour détecter des fautes complexes, comme par exemple des couplages. L'approche mentionnée dans la section précédente n'est donc pas non plus adaptée. Les blocs de BIST correspondent en fait pour ces cellules à une **implantation matérielle des algorithmes** utilisés par ailleurs pour la génération de séquences de test externe.

Dans le **cas d'une mémoire ROM**, le test consiste à s'assurer que les valeurs programmées aux différentes adresses peuvent être lues correctement. Il s'agit donc de lire chaque mot et de le comparer à la valeur attendue. Un exemple d'implantation est illustré par la figure **10**. Les dispositifs de test comportent un bloc de commande gérant l'exécution du test, un bloc de génération de vecteurs pouvant être positionné sur les entrées d'adresse par l'intermédiaire de multiplexeurs et un bloc d'analyse des résultats. À ceci peut s'ajouter un ensemble de multiplexeurs permettant de forcer pendant le test une valeur neutre sur les sorties de données de la mémoire.

L'ordre de lecture des mots dans la mémoire n'ayant pas d'importance, le coût matériel du BIST peut être réduit en utilisant et en incluant la valeur 0 dans la séquence pour la génération des adresses un LFSR autonome modifié, avec un polynôme diviseur premier primitif. Cet élément est moins coûteux qu'un compteur binaire mais permet d'accéder successivement à toutes les adresses en commandant simplement son horloge. Les données obtenues en sortie de la ROM peuvent être lues à l'extérieur par l'intermédiaire d'un *scanpath*, ou être compactées par un MISR. Dans ce dernier cas, la signature générée peut elle-même être lue à l'extérieur ou être comparée en interne avec la signature correcte attendue.

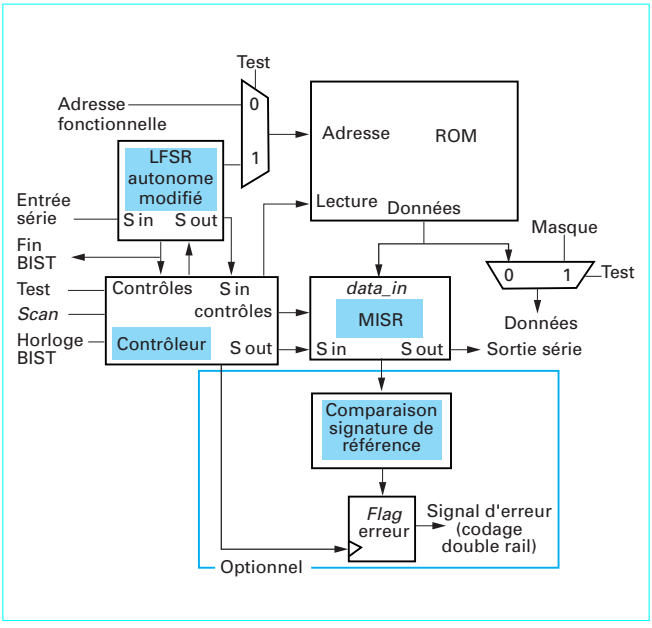

Figure 10 – Exemple d'implémentation d'un autotest déterministe pour une ROM [3]

Dans le **cas d'une mémoire RAM**, le principe de base est similaire mais le test est plus complexe.

- Le bloc de génération de vecteurs doit permettre d'accéder dans un ordre précis aux mots de la mémoire ; il comporte en général un compteur/décompteur binaire. Il doit de plus permettre d'écrire dans la mémoire des données précises à des instants précis. Ces mêmes données sont ensuite souvent comparées avec les valeurs obtenues lors de la relecture de la mémoire au lieu d'utiliser un MISR sur les sorties.
- Le bloc d'analyse des résultats est donc aussi plus complexe que dans le cas d'une ROM. De nombreuses variantes existent, en fonction des caractéristiques de la mémoire à tester (SRAM – *static random access memory* – ou DRAM – *dynamic random access memory* –, un ou plusieurs ports, bus de données multiplexés ou non, architecture interne, etc. ...) et en fonction des modèles de fautes considérés.

De tels autotests spécifiques peuvent aussi être développés par exemple pour des éléments de calcul arithmétique comme les multiplieurs combinatoires.

1.6.5 Précautions d'implantation et d'utilisation pour un autotest

Une première précaution à prendre lors de l'implantation d'un autotest est de s'assurer que le déroulement du test ne peut pas être faussé par une faute dans les éléments logiques du BIST ! Ces éléments doivent donc être conçus de façon à s'autotester, ou tout au moins de façon à garantir qu'une faute dans ces éléments ne peut pas conduire à masquer le mauvais fonctionnement du circuit.

Exemple : supposons que la logique de test effectue l'ensemble des comparaisons requises et envoie juste vers l'extérieur un signal d'erreur. Ce signal ne doit alors pas être codé sur un seul bit ; un collage sur cette sortie du circuit suffirait dans ce cas à valider le test, indépendamment de son résultat réel. Pour éviter cela, le signal d'erreur transmis vers l'extérieur doit être codé de façon redondante. Le cas le plus classique est un codage double rail (sortie de deux bits complémentaires).

Une autre précaution concerne l'utilisation des autotests, et plus particulièrement la définition du plan de test. Il a déjà été mentionné pour le *scanpath* que le test pouvait entraîner une surconsommation très importante du circuit, nuisible pour sa fiabilité. Si plusieurs blocs du circuit sont équipés d'un autotest et si tous ces autotests sont exécutés en parallèle, il peut de même y avoir pendant le test une activité des blocs logiques très supérieure à ce qui est attendu lors du fonctionnement normal. La surconsommation résultante peut entraîner un échauffement important et des problèmes sur les lignes d'alimentation, dus au phénomène d'électromigration. Le plan de test doit donc être déterminé de façon cohérente avec la conception physique du circuit (dimensionnement des alimentations et étude thermique). Ceci peut conduire à allonger le temps de test, en exécutant certains autotests l'un après l'autre, afin de respecter les limites acceptables de consommation.

1.7 Critères généraux de partitionnement et de conception

Afin de guider le concepteur, il est possible de citer quelques **critères généraux** à prendre en compte **pour partitionner** un circuit en vue du test :

— isolation des blocs fonctionnels indépendants (macrocellules, blocs arithmétiques complexes, IP...) ;

— isolation des blocs de logique séquentielle (compteurs, machines à états finis, ... difficiles à traverser) ;

— coupure des boucles de rétroaction dans les blocs séquentiels à fort rebouclage (test en boucle ouverte dans le cas idéal).

Indépendamment du partitionnement, destiné à l'application de méthodes structurées adaptées à chaque type de bloc, quelques **critères généraux de conception** permettent d'éviter par avance un certain nombre de problèmes de test :

— contrôle direct ou facile des signaux de commande clés pendant le test (horloges divisées ou validées par de la logique, signaux de validation de chargement des éléments de mémorisation, signaux de commande des portes trois états, signaux d'initialisation) ;

— contrôle direct en mode de test des sorties de monostables (et utilisation uniquement de monostables synchrones) ;

— suppression soigneuse des redondances logiques (entrées de portes connectées directement à la masse ou à l'alimentation, logique redondante due à la connexion de blocs conçus séparément…) ;

— segmentation pendant le test des fonctions longues à tester (par exemple, compteur avec un grand nombre de bits…) ;

— insertion d'éléments de BIST sur les blocs enterrés, notamment ceux nécessitant un grand nombre de vecteurs de test (en particulier les mémoires) ;

— limitation de l'emploi des logiques dynamique ou à précharge.

Certains de ces critères peuvent conduire à ajouter immédiatement de la logique destinée au test, sans attendre une analyse globale de la testabilité du circuit (à l'exception des cas où il est prévu une approche systématique de conception pour le test comme par exemple le *full scan*).

Exemple : supposons qu'un bloc B du circuit soit commandé par une horloge temps réel, activée une fois par heure et générée à partir d'une horloge externe à 10 MHz. Chaque vecteur de test du bloc B nécessitant un front d'horloge actif sur ce bloc nécessiterait… plusieurs dizaines de milliards de fronts d'horloge appliqués par l'ATE ! Ceci est évidemment inacceptable, et un dispositif de multiplexage peut être prévu dès la phase initiale de conception pour permettre un contrôle direct depuis l'extérieur de l'horloge du bloc B. Naturellement, il est alors indispensable de prendre en compte cette modification lors du dimensionnement des lignes d'alimentation du bloc B, ce bloc voyant pendant le test sa consommation augmenter de façon phénoménale.

2. Norme IEEE 1149.1 « *boundary scan* »

Différence majeure avec ce qui précède (§ 5), les dispositifs de la norme IEEE 1149.1 ne sont pas destinés en priorité au test du circuit lui-même, mais au support de la testabilité des cartes et des équipements. Cette utilisation hiérarchique dans un produit ne sera pas détaillée ici ; on présentera les concepts de base de l'approche et les principaux éléments à intégrer dans un circuit pour qu'il soit compatible avec cette norme. Pour plus de détails, la référence reste le document officiel *IEEE standard test access port and boundary-scan architecture* (cf. Pour en savoir plus [Doc. E 2 462]).

La norme votée en 1990 à l'issue des travaux du groupe IEEE 1149.1 a fait suite aux propositions d'un premier groupe de travail nommé « Joint test action group » (JTAG), ce qui explique qu'il soit assez souvent fait référence, de façon abusive, à la « norme JTAG ».

Ces groupes de travail ont été constitués pour faire face à la complexité croissante du test des cartes électroniques, liée à l'augmentation de densité résultant de l'évolution des techniques d'interconnexion et d'encapsulation (nombre de couches élevé, montage en surface, circuits hybrides et MCM – *multi chip module* –, etc.). Cette évolution ne permettait plus un test efficace par des moyens traditionnels (de type « planche à clou »), d'où l'idée d'adapter aux cartes et aux équipements les méthodes mises en œuvre pour le test des circuits intégrés.

L'autre objectif, en normalisant cette approche, était de réduire les coûts en rendant possible le test unifié d'un système comportant des cartes et des composants d'origines diverses. Une approche de test sériel a été retenue afin de minimiser le nombre des broches utilisées pour le test.

L'architecture « *boundary scan* » permet de tester sans contact les différents circuits logiques présents sur une carte (ou dans un circuit hybride) et surtout les interconnexions entre ces circuits. Les dispositifs de test intégrés dans les différents circuits (« *scanpath* » ou BIST) peuvent être commandés par l'intermédiaire de l'interface *boundary scan*, ce qui permet également :

— de faciliter le test et le diagnostic de la partie numérique d'un système ;

— de réduire la complexité des testeurs de cartes ;

— de réutiliser les tests fonctionnels ou structurels et les dispositifs de test intégré des circuits pour le test *in situ*.

Pour atteindre ces objectifs, des éléments doivent être intégrés dans les circuits utilisés dans l'équipement (ou tout au moins dans la majorité d'entre eux), afin de permettre un test sériel. Ces éléments à ajouter dans chaque circuit incluent en particulier quatre ou cinq broches externes [TDI, TDO, TCK, TMS, et éventuellement TRST (cf. § 2.3.1)], des registres (*boundary scan*, *bypass*, instruction,…) et une machine à états finis (*TAP controller*) gérant l'exécution d'instructions de test. Le jeu d'instructions et les différents éléments implantés dans un circuit donné peuvent être décrits dans un langage spécifique, nommé BSDL (*Boundary Scan Description Language*), dans un but de documentation et/ou de spécification. L'utilisation de ce langage ne sera pas détaillée ici.

2.1 Principe du test « *boundary scan* »

L'implantation du « *boundary scan* » sur une carte est illustrée par la figure **11**. Le principe de base consiste à créer un registre à décalage passant par tous les circuits de façon à pouvoir contrôler et observer toutes les entrées-sorties de tous les composants. L'entrée série, la sortie série et les signaux de contrôle permettant d'effectuer le test forment le « *Test Access Port* » (TAP), ajouté aux entrées-sorties primaires de chaque composant à chaque niveau de la hiérarchie (circuit, circuit hybride, carte…).

L'application de ce principe varie en pratique d'une carte à l'autre, en fonction du type des composants employés et des contraintes de conception.

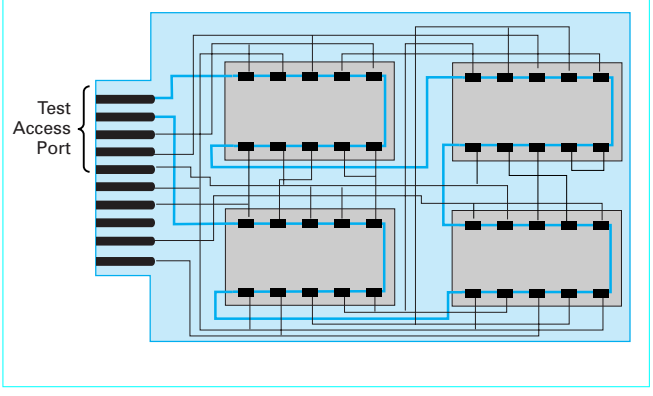

Figure 11 – Principe de création d'une chaîne *boundary scan* sur une carte

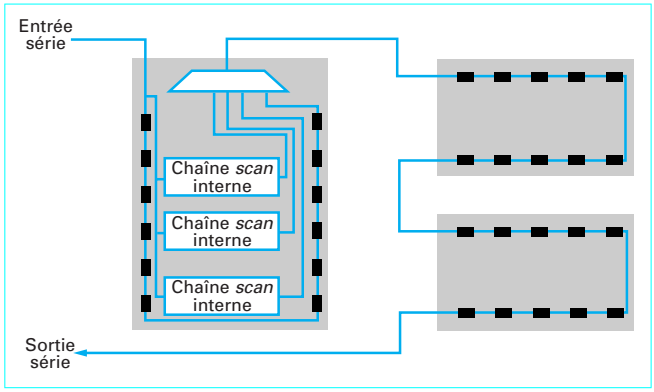

Figure 12 – Principe d'insertion dans une chaîne « *boundary scan* » d'une chaîne de test série interne à un circuit

● Tout d'abord, il est fréquent que certains circuits ne soient pas compatibles avec la norme « *boundary scan* » (en particulier les circuits SSI). Dans ce cas, seuls certains circuits sont interconnectés dans le chemin de test et les autres circuits (et leurs interconnexions) sont testés de façon indirecte. Il s'agit alors d'établir un compromis acceptable entre l'efficacité du test et les éventuels dispositifs ajoutés sur la carte pour améliorer l'accès aux nœuds qui ne sont pas connectés dans la chaîne série.

● Un autre type de variante concerne le nombre de chaînes séries. En effet, pour une carte complexe, une chaîne « *boundary scan* » unique peut comporter un très grand nombre d'étages, ce qui induit des temps élevés de chargement des vecteurs de test. Pour réduire le temps d'application du test, il est possible, comme dans le cas des *scanpath* à l'intérieur d'un circuit intégré, de diviser la chaîne *boundary scan* globale en plusieurs chaînes connectées en parallèle. Dans ce cas, seules certaines sous-chaînes sont à modifier lors de l'application d'un vecteur de test donné.

● Enfin, il est possible d'inclure d'autres chaînes séries dans la chaîne *boundary scan*. En particulier, il est possible de multiplexer le registre de périphérie (cf. § 2.3.2) d'un circuit avec des registres de *scanpath* internes, comme illustré en figure **12** ; ceci permet dans certains cas d'accéder au cœur des circuits pendant le test de la carte ou de l'équipement.

■ L'architecture *boundary scan* peut donc permettre, en fonction de l'implantation exacte réalisée, de vérifier le bon fonctionnement de chaque élément sur la carte, de confirmer que tous les composants sont correctement positionnés et interconnectés et de s'assurer que tous les circuits "dialoguent" correctement. Elle permet même, grâce à des codes mémorisés dans les circuits, de s'assurer de la bonne version des composants implantés (fabricant, numéro de référence, version du logiciel embarqué, etc.). Elle nécessite toutefois la présence d'un certain nombre de dispositifs intégrés dans la majorité des circuits.

2.2 Instructions de commande

Hormis le contrôle assuré par l'intermédiaire des signaux du TAP, les différents types de tests possibles au niveau de la carte (test des circuits, test des interconnexions, etc.) sont sélectionnés en chargeant une instruction de test dans chaque circuit.

Le jeu d'instructions disponible dans un circuit « *boundary scan* » est variable. Trois instructions sont obligatoires dans tout circuit compatible avec la norme : EXTEST, BYPASS et SAMPLE/PRELOAD. Des instructions optionnelles sont aussi définies dans le document IEEE 1149.1, certaines d'entre elles ayant été ajoutées lors de la révision de la norme initiale. Des instructions propriétaires peuvent aussi être prévues par le concepteur d'un circuit afin d'accéder aux diverses ressources de test implantées dans le circuit.

Le jeu d'instructions est composé d'instructions publiques et d'instructions privées :

— les instructions publiques doivent être documentées (si elles sont implantées) pour pouvoir être employées par tout utilisateur du circuit ;

— les instructions privées peuvent être documentées ou être réservées à l'usage du fabricant du circuit.

Les codes binaires des instructions sont laissés au choix du concepteur, à l'exception des codes des instructions EXTEST et BYPASS, fixés par la norme comme étant les codes où tous les bits sont à 0 (EXTEST) ou à 1 (BYPASS).

Les instructions principales, présentes dans la version initiale de la norme, sont les suivantes :

— **EXTEST** (publique, obligatoire) : test externe. Cette instruction configure les dispositifs de test de façon à tester les éléments de la carte extérieurs aux circuits *boundary scan* (interconnexions ou circuits non compatibles avec la norme *boundary scan*) ;

— **BYPASS** (publique, obligatoire) : réduction de la longueur de la chaîne série. Lorsque l'on veut accéder à un ou plusieurs circuits bien particuliers lors du test d'une carte, il est intéressant d'éviter de traverser les autres circuits afin de réduire le temps d'application du test. Il faut pour cela « court-circuiter » la chaîne au niveau des circuits auxquels on ne s'intéresse pas (cf. figure **15**). Ceci est contrôlé par l'instruction BYPASS ;

— **SAMPLE/PRELOAD** (publique, obligatoire) : échantillonnage ou préchargement. Lorsque cette instruction est chargée, les dispositifs de test sont transparents et le circuit se comporte donc normalement sur la carte. Cependant, le registre de périphérie *boundary scan* peut être chargé en série ou en parallèle :

- le chargement série permet de préparer une phase de test en préchargeant un vecteur,
- le chargement parallèle permet d'échantillonner les données transitant à un instant précis sur la carte, ce qui autorise la vérification des interactions dynamiques entre les circuits ;

— **INTEST** (privée, optionnelle) : test interne. Cette instruction configure les dispositifs de test de façon à tester le circuit lui-même. Plusieurs instructions de ce type peuvent être définies par le concepteur du circuit afin de sélectionner différents dispositifs internes lors du test ;

— **RUNBIST** (privée, optionnelle) : activation des dispositifs de BIST. Cette instruction sélectionne l'activation des dispositifs de BIST intégrés dans le circuit. Plusieurs instructions de ce type peuvent être définies par le concepteur du circuit afin de sélectionner différents dispositifs de BIST ;

— **IDCODE** (publique, optionnelle) : lecture du code d'identification fabricant. Un registre d'identification peut être intégré dans le circuit. Ce registre contient un code sur 32 bits permettant d'identifier le composant et son fabricant. L'instruction IDCODE permet de lire ce code d'identification par l'intermédiaire de la chaîne série *boundary scan*. Cette instruction devient obligatoire lorsque le registre d'identification existe dans le circuit ;

— **USERCODE** (publique, optionnelle) : lecture du code d'identification utilisateur. Dans le cas d'un composant programmable, le code d'identification du fabricant doit être complété par un code identifiant la version de la programmation du composant. L'instruction USERCODE permet de lire ce second code d'identification. Cette instruction devient obligatoire lorsque le registre d'identification existe dans un circuit programmable.

2.3 Éléments à intégrer

2.3.1 Vue d'ensemble des éléments à intégrer

La figure **13** donne une vue d'ensemble de l'architecture matérielle au niveau circuit.

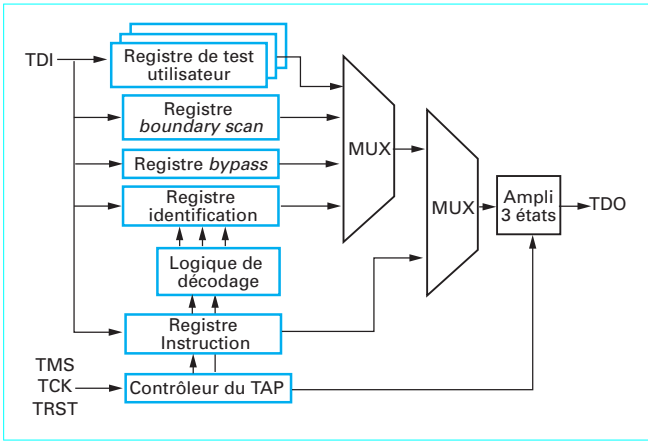

Figure 13 – Principe de l'architecture matérielle *boundary scan* au niveau circuit

Comme indiqué précédemment, le **port d'accès** aux éléments de test (TAP) comporte quatre ou cinq signaux nécessaires au test « *boundary scan* » :

— **TDI** (*Test Data Input*) et **TDO** (*Test Data Output*) : entrée et sortie de la chaîne de test. La sortie des données se fait par l'intermédiaire d'un amplificateur trois états permettant, le cas échéant, l'existence de plusieurs chaînes parallèles sur la carte ;

— **TCK** (*Test Clock*) : horloge permettant de séquencer les différentes phases du test indépendamment des horloges utilisées dans le système en mode opérationnel ;

— **TMS** (*Test Mode Select*) : entrée de commande permettant de définir le déroulement du test ;

— **TRST** (*Test Reset*) : entrée optionnelle permettant l'initialisation asynchrone de la logique de test. Cette entrée est obligatoire si le circuit n'est pas équipé d'une réinitialisation automatique à la mise sous tension (*power-up reset*).

Le **registre Instruction** permet le chargement et le stockage du code binaire de l'instruction de test dans le circuit. L'instruction mémorisée détermine le mode de test activé et permet, après décodage, l'accès au registre de données concerné par l'instruction.

Une machine à états finis initialisée par TRST, commandée par TMS et changeant d'état sur les fronts montant de TCK assure le séquencement des phases de test (contrôleur du TAP, ou « *TAP controller* »). Il s'agit d'une **machine de Moore** à 16 états, dont le graphe, complètement spécifié par la norme, ne sera pas détaillé ici. L'état courant de cette machine permet, éventuellement en fonction de l'instruction chargée dans le registre Instruction, de déterminer le registre à connecter entre TDI et TDO (registre instruction ou registre de données) et les commandes à envoyer à chaque cycle sur les différents éléments de l'architecture (décalage, chargement parallèle, activation de BIST, etc.).

Registres de données

- Les **registres de données** de test comprennent tout d'abord le **registre de périphérie** (registre *boundary scan*), qui permet d'accéder aux entrées-sorties du circuit par l'intermédiaire de la chaîne série.
- Le **registre Bypass**, de 1 bit doit obligatoirement être implanté dans un circuit *boundary scan* et est connecté entre TDI et TDO lorsque l'instruction BYPASS est exécutée (cf. § 2.2).

Un **registre d'identification** peut aussi être intégré dans le circuit. Ce registre contient un code sur 32 bits permettant d'identifier le composant, son fabricant et la version du programme dans le cas d'un composant programmable. Il est connecté entre TDI et TDO lors de l'exécution des instructions IDCODE et USERCODE.

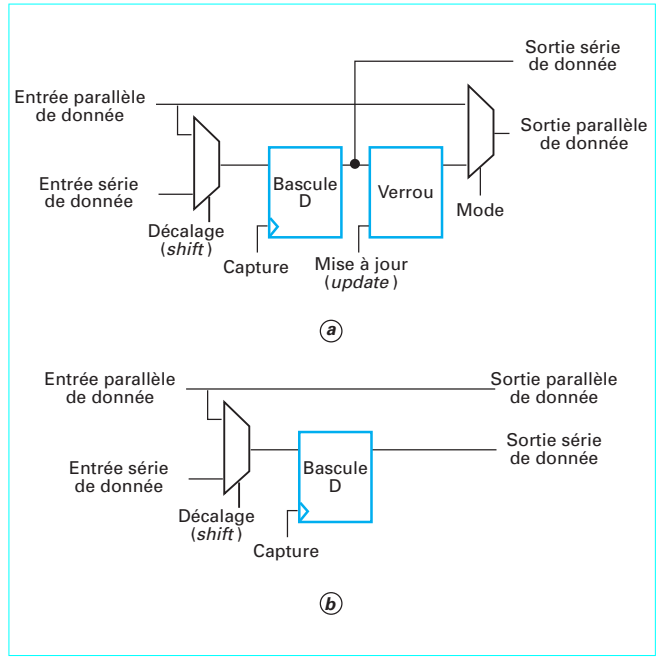

Figure 14 – Cellule du registre *boundary scan* complète (*a*) ou simple observation (*b*)

Enfin, le concepteur du circuit peut connecter des **registres de test supplémentaires** permettant de faciliter le test du circuit sur une carte. En particulier, les registres de *scanpath* implantés dans le circuit peuvent être connectés de façon à s'intégrer directement dans la chaîne série sur la carte (cf. figure **12**).

2.3.2 Le registre de périphérie

Le registre *boundary scan* est le registre de périphérie permettant d'observer et de contrôler les entrées-sorties du circuit. Il est constitué d'une cellule par interface d'entrée ou de sortie. Dans le cas des interfaces de sortie trois états et des interfaces bidirectionnelles, une cellule supplémentaire doit être ajoutée pour observer et contrôler le signal de sélection.

Le schéma de principe d'une cellule complète du registre *boundary scan* est donné en figure **14*a***. Cette cellule permet à la fois de contrôler et d'observer le signal de donnée. Elle est constituée d'une bascule D (registre à décalage 1 bit), d'un verrou permettant de maintenir constante la sortie parallèle de donnée pendant les opérations de décalage et de deux multiplexeurs. Les signaux de commande *Shift*, *Capture*, *Update* et *Mode* sont générés par la logique de décodage d'instruction, en fonction de l'état courant du contrôleur du TAP.

Si aucune instruction INTEST (cf. § 2.2) n'est implantée, une cellule dite « simple observation » peut être employée pour les interfaces d'entrée. Cette cellule, représentée en figure **14*b*** permet d'éviter le retard dû au multiplexeur sur le chemin direct de donnée.

La figure **15** illustre la configuration du registre de périphérie en fonction des phases de test :

— dans le cas du test interne d'un circuit (cf. **15*a***), une instruction de type INTEST (cf. § 2.2) est chargée dans le registre Instruction du circuit concerné. Les registres Instruction des autres circuits peuvent être chargés avec l'instruction BYPASS afin d'accélérer l'accès au circuit sous test. Le registre de périphérie de chacun de ces circuits est alors inactivé (positionnement du signal Mode pour con-

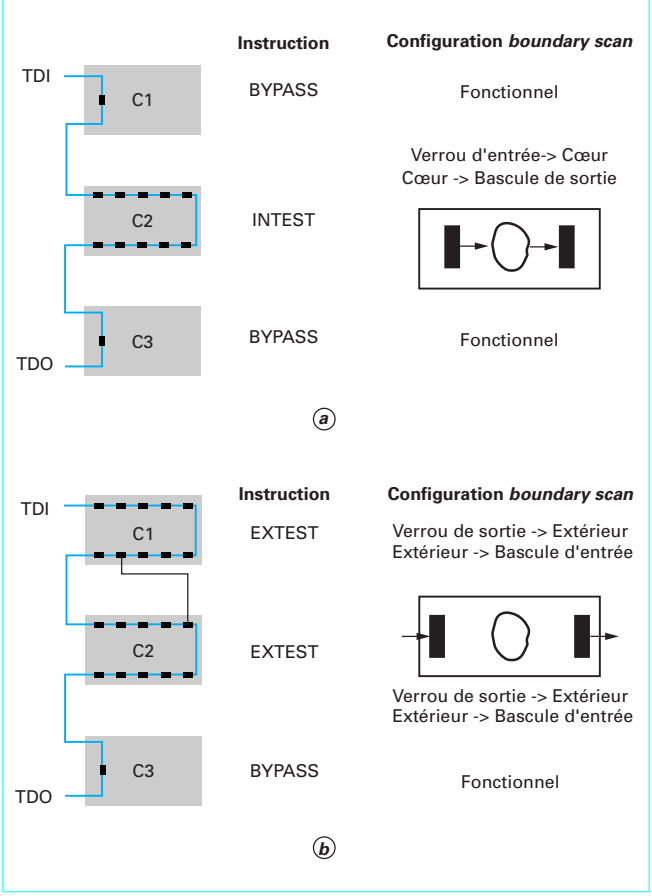

Figure 15 – Exemples de configurations sur une carte avec trois circuits C1, C2, C3 pour le test interne du circuit C2 (*a*), et le test d'une interconnexion entre une sortie de C1 et une entrée de C2 (*b*)

necter les entrées parallèles de données aux sorties parallèles de données, et positionnement au niveau inactif des signaux *Capture* et *Update*). Dans le circuit sous test, les cellules d'entrée du registre *boundary scan* sont positionnées pour appliquer le contenu du registre en entrée du circuit, et les sorties du circuit sont chargées dans les cellules correspondantes du registre *boundary scan*. Les décalages et chargements sont commandés en fonction de la succession des états atteints par le contrôleur du TAP, donc en fonction des évolutions du signal externe TMS ;

— dans le cas du test d'une interconnexion (cf. figure **15*b***), deux circuits (au moins) sont concernés et sont configurés avec l'instruction EXTEST. Pour ces circuits, les cellules de sortie du registre *boundary scan* sont positionnées pour appliquer le contenu du registre en sortie du circuit, tandis que les cellules d'entrée permettent d'échantillonner les valeurs lues sur les interconnexions.

2.3.3 Test des éléments de test

L'ensemble de la logique ajoutée pour le test *boundary scan* doit elle-même être testée en fin de fabrication. Si le test du circuit utilise l'interface *boundary scan*, une grande partie des fautes de la logique de test est implicitement détectée ; toutefois, des vecteurs supplémentaires peuvent être requis pour obtenir un taux de couverture satisfaisant. Par ailleurs, si le test du circuit n'utilise pas directement cette interface, la logique de test doit être testée au même titre que n'importe quelle autre fonctionnalité du circuit. Les capacités intrinsèques du *boundary scan* (contrôlabilité et observabilité liées aux possibilités de décalages) permettent de tester facilement un grand nombre d'éléments. Les blocs les plus critiques sont le contrôleur du TAP et la logique de décodage d'instruction, en particulier lorsqu'un grand nombre d'instructions est implanté. Les méthodes de conception pour la testabilité doivent dans certains cas être appliquées sur ces blocs.

3. Vers le test des SoC

L'évolution des modèles de fautes pour les technologies récentes a été présentée dans l'article [E 2 460], § 2.3. En parallèle de cette évolution, les technologies submicroniques profondes (DSM) induisent, pour les circuits les plus pointus (SoC ou système sur une puce), de nouveaux problèmes liés :

— à l'augmentation de la complexité des éléments intégrés ;
— à l'augmentation de la densité d'intégration ;
— à la forte croissance des fréquences d'horloges ;
— à la diminution requise des coûts.

L'**augmentation de la complexité des circuits** induit essentiellement quatre types de problèmes.

● **Fossé croissant entre les bandes passantes externe et interne d'un circuit**

La **bande passante externe** est définie comme le produit entre le nombre de broches (en excluant les broches d'alimentation) et la fréquence de commutation des signaux externes.

La **bande passante interne** est définie comme le produit entre le nombre de transistors (ou de portes) dans le circuit et la fréquence de commutation des signaux internes.

Ces deux derniers paramètres augmentent beaucoup plus rapidement que les précédents. Il est ainsi prévu d'ici 2010 une croissance de trois ordres de grandeur de la bande passante interne alors que la bande passante externe n'augmenterait que dans un rapport 10. Ceci montre clairement que l'accès aux éléments internes depuis les broches du circuit sera considérablement plus difficile qu'aujourd'hui, réduisant l'efficacité attendue d'un test externe, et augmentant dans des proportions inacceptables les temps d'application des tests par des équipements externes.

Il faut ajouter à ceci qu'une part non négligeable des éléments intégrés correspondent à des blocs séquentiels (machines à états finis ou blocs équivalents). L'augmentation du nombre de transistors se traduit donc par une très forte croissance de l'espace des états devant être exploré lors du test, ce qui complique encore la tâche de génération des vecteurs et tend également à augmenter le temps de test en augmentant la longueur des séquences à appliquer.

Enfin, il faut noter que la différence croissante entre les fréquences de commutation interne et externe rend impossible l'application par un équipement externe d'un test à vitesse nominale.

En conséquence, les méthodes de test ne peuvent qu'évoluer rapidement vers une nouvelle répartition entre test interne et test externe. Engagée à la fin des années 1990 pour les circuits les plus complexes, la mutation vers une forte intégration de dispositifs de test dans les circuits ne peut que s'accélérer. Du test exclusivement externe, les habitudes industrielles ont évolué vers un test externe facilité par des éléments de DFT comme des *scanpath*. L'utilisation de blocs d'autotests est devenue classique dans le cas des mémoires embarquées. Les circuits futurs devront intégrer une part encore plus importante de logique de test, permettant de profiter de la grande bande passante interne pour réduire le temps d'application des vecteurs de test, autorisant un test à vitesse nominale et limitant les accès à un équipement externe à l'échange d'informations clés (type de test à réaliser, résultats obtenus, données de diagnostic…). Ces informations clés étant peu nombreuses, la bande passante externe ne sera alors plus une limitation.

● **Cohabitation dans un même circuit de types de blocs multiples**

Chaque type de bloc (numérique, analogique, mémoires statiques et dynamiques, mémoires flash, réseaux programmables, blocs radiofréquence (RF), microsystèmes…) nécessite des méthodes spécifiques. Par ailleurs, chaque type d'élément induit de nouveaux types de défauts et nécessite des caractéristiques différentes de l'équipement de test. L'utilisation en fin de fabrication d'un testeur différent pour chaque type de bloc induit des coûts élevés et des temps de test prohibitifs. L'utilisation d'un « testeur universel », capable de tester l'ensemble des blocs, induirait des coûts quasiment aussi importants.

Là encore, la solution la plus efficace sera probablement d'intégrer au niveau de chaque type de blocs des éléments de test spécifiques, capables de communiquer leurs résultats à un testeur externe relativement simple. Toutefois, si des méthodes de DFT efficaces existent aujourd'hui pour les parties numériques, il n'en est pas de même pour les parties analogiques et *a fortiori* pour les microsystèmes. De nombreux travaux visent à mettre au point et à standardiser de telles approches.

● **Réutilisation de blocs complexes**

L'avènement du système sur une puce a impliqué un changement de méthodologie de conception et le développement rapide de la réutilisation de blocs, seule approche permettant de concevoir des circuits aussi complexes dans un temps réduit, compatible avec les contraintes économiques. Ces blocs réutilisables (ou composants virtuels, ou « cœurs » de circuits, ou IP pour « *Intellectual Property* ») sont le plus souvent achetés chez un fournisseur, qui conserve la propriété intellectuelle du bloc… et la plupart des informations détaillées le concernant. Une fois le bloc inséré dans un circuit, il faut pourtant être capable de le tester efficacement. Ceci pose donc de nombreux problèmes, à la fois au niveau de la transmission des informations nécessaires et suffisantes pour le concepteur utilisant le bloc, et au niveau de l'application du test en fin de fabrication. Des travaux, débutés en 1995 visent à la standardisation de méthodes dans ce domaine, sous l'appellation IEEE1500. Ceci inclue un langage de description des informations de test (CTL : *Core Test Language*) et une méthode d'accès au bloc à l'intérieur d'un SoC (*Core Test Wrapper*) (cf. [Doc. E 2 462]).

● **Maintien d'un rendement acceptable pour des mémoires de grande taille**

Pour les circuits très réguliers comme les mémoires SRAM ou DRAM, des techniques de diagnostic et de réparation sont classiquement utilisées pour augmenter le rendement de fabrication et donc diminuer les coûts de production. Des éléments redondants (lignes ou colonnes de cellules mémoires) sont alors intégrés dans le circuit. Le test en fin de fabrication est suivi, si la mémoire est défectueuse, par un diagnostic (localisation des cellules défectueuses) et une réparation est effectuée lorsque les éléments redondants peuvent remplacer les éléments défectueux (reconfiguration électrique, ou plus souvent par rayons laser).

Si ces méthodes sont habituelles dans des lignes de production de boîtiers de mémoires, elles ne sont classiquement pas appliquées lorsque la mémoire est embarquée comme un bloc dans un circuit complexe. Hors, les blocs de mémoire vont représenter un pourcentage très important de la plupart des SoC, et leur rendement conditionnera fortement le rendement de production de ces systèmes.

Il faudra donc être capable, pour de tels circuits, d'appliquer sur les blocs de mémoire enterrés les mêmes techniques que celles qui sont appliquées pour les mémoires individuelles. Ceci impliquera des modifications des lignes de production (nouvelles étapes et nouveaux équipements utilisés en fin de fabrication), mais également des modifications des méthodes, des bibliothèques et des outils de conception utilisés pour l'implantation de ces blocs. Au-delà du test, il s'agira de diagnostic avec réparation, voire « d'autoréparation » (BISR : *Built-In Self-Repair*) du circuit.

■ **L'augmentation de la densité d'intégration** pose également différents problèmes. La plupart sont liés à l'évolution des modèles de faute à prendre en compte et ont déjà été mentionnés en paragraphe 2.3 de l'article [E 2 460].

La difficulté de localisation d'une faute croît environ d'un ordre de grandeur tous les six ans. Or, cette localisation est nécessaire, pour un certain nombre de circuits, afin de pouvoir analyser les causes des défauts et améliorer la fabrication, voire la conception physique (dessin des masques) du circuit. Bien que cette localisation sorte du cadre strict du test, elle peut être facilitée par les dispositifs de test intégrés (BIST ou *scanpath*, par exemple).

■ L'**augmentation des performances**, et plus particulièrement des **fréquences de fonctionnement**, a aussi un impact notable sur le test. L'importance croissante des fautes de retard a déjà été mentionnée (cf. [E 2 460] § 2.2.6 et § 2.3.1) ; en corollaire, le test à vitesse nominale voit son importance grandir. Or, un testeur externe (ATE) est classiquement construit avec des circuits intégrés ayant une génération de retard par rapport aux circuits qu'ils testent… Pour les circuits fonctionnant à la limite des possibilités de la technologie, cela signifie que les ATE sont généralement trop lents pour permettre un test à vitesse nominale, sans l'aide de dispositifs d'autotest intégrés.

■ Les **outils de conception** employés pour satisfaire les contraintes drastiques sur le temps de développement (***Time to Market***) ont aussi des conséquences sur la longueur du test. La synthèse logique (voire architecturale) est aujourd'hui incontournable dans un flot de conception de circuit numérique complexe. Toutefois, l'utilisation des outils de synthèse a pour conséquence le rétrécissement de la distribution des temps de propagation sur les chemins combinatoires entre éléments de mémorisation. En effet, les outils cherchent à réduire le chemin critique, puis à allonger les autres chemins pour optimiser la surface du circuit. La distribution des temps de propagation tend donc vers une distribution en pic de Dirac… Dans un circuit synchrone, toute faute de retard tend alors à rendre le circuit incorrect vis-à-vis de ses spécifications temporelles. Tous les chemins devenant critiques, il est de moins en moins possible de limiter le test des fautes de retard à un petit nombre de cas. Ce problème peut être contourné en effectuant tout le test à vitesse nominale.

■ Il faut noter enfin que les **contraintes économiques** vont également dans le sens d'un test intégré.

Le coût par broche des équipements de test externe est longtemps resté stable, aux alentours de 10.000 € par broche. Les contraintes extrêmes imposées par les technologies les plus récentes, en termes de vitesse (ou fréquence), de précision des mesures ou encore de nombre de générateurs de phases, font augmenter rapidement ce coût.

Ceci, ajouté à la plus grande diversité des blocs discutée ci-dessus, rend l'intégration de dispositifs de test plus économique que l'utilisation d'un testeur externe, malgré la surface occupée dans le circuit et donc l'augmentation du prix de fabrication de chaque circuit. Des études ont montré que le test intégré et le test externe ont un coût sensiblement identique pour des circuits comportant 400 000 à 500 000 portes. Pour les circuits nettement plus complexes prévus dans les prochaines années, le test intégré sera plus rentable économiquement.

De plus, ces dispositifs intégrés permettent en général un meilleur taux de couverture pour différents modèles de fautes. Ce gain en qualité, bien que difficilement quantifiable, est aussi très important économiquement, puisqu'il influe sur la satisfaction du client et donc sur les commandes futures.

Enfin, ces dispositifs intégrés peuvent être réutilisés de façon hiérarchique pour les tests de production des cartes et des équipements et même pour les autotests périodiques de l'application, augmentant ainsi la qualité des tests sur toute la durée de vie des équipements tout en réduisant leurs coûts.

Tout ceci contribue à une mutation accélérée des éléments de test de l'extérieur vers l'intérieur des circuits.

4. Techniques de conception pour augmenter la testabilité en ligne

Avec l'augmentation de la probabilité des fautes transitoires (cf. [E 2 460], § 2.3.4), les techniques de conception permettant d'implanter un test en ligne voient leurs domaines d'application se multiplier. De nombreuses approches sont proposées pour détecter différents types de fautes, avec des compromis variés entre taux de couverture, augmentation de surface et pénalités temporelles. Cette diversité est illustrée par exemple dans [4]. L'objectif de ce paragraphe n'est donc pas de détailler toutes ces approches, mais d'en présenter les principes de base.

Par ailleurs, nous ne parlerons pas des approches utilisées pour la détection de fautes catastrophiques (chiens de garde ou *watchdog timer*) ni des approches permettant de tolérer la présence de fautes.

4.1 Circuits autocontrôlables

Une approche ayant été beaucoup étudiée consiste à utiliser des codes détecteurs d'erreur pour protéger toutes les informations transitant dans le circuit ou le bloc à tester. Le principe de base est illustré par la figure **16**. Le bloc reçoit des entrées codées et génère des sorties codées (pas nécessairement dans le même code). Ces sorties codées sont transmises aux blocs suivants, éventuellement par l'intermédiaire d'un bloc de transcodage. Les sorties codées sont de plus vérifiées par un bloc supplémentaire, s'assurant que la réponse du bloc sous test est bien un mot du code de sortie. Si ce n'est pas le cas, un signal d'erreur est émis.

Le bloc de vérification (ou *checker*) doit lui-même avoir certaines propriétés, permettant d'assurer que le signal d'erreur n'est pas faussé ; ce signal d'erreur est donc en général lui aussi codé.

Le bloc sous test et le bloc de vérification sont tous deux conçus pour assurer qu'un certain ensemble de fautes est détecté. La structure des deux blocs (et donc aussi les pénalités en surface et en vitesse) dépend donc du modèle de fautes choisi et du niveau de sûreté requis par l'application.

Les principales propriétés qui peuvent être obtenues sont :

— **bloc « sûr en présence de fautes »** : il est alors garanti qu'une faute (appartenant à l'ensemble de fautes visé), survenant dans le bloc, n'occasionne pas d'erreur sur la sortie ou bien transforme le vecteur de sortie en un mot invalide du code, quel que soit le vecteur d'entrée envoyé sur le bloc ;

— **bloc « autotestable »** : au sens du test en ligne, cette propriété correspond à l'existence, pour toute faute dans l'ensemble de fautes visé, d'au moins un vecteur d'entrée pour lequel la sortie correspond à un mot invalide du code ;

— **bloc « totalement autocontrôlable »** : le bloc est alors à la fois sûr en présence de fautes et autotestable, pour un même ensemble de fautes ;

— **bloc « fortement sûr en présence de fautes »** : pour toute faute dans l'ensemble de fautes visé, le bloc est alors soit totalement autocontrôlable, soit sûr en présence de fautes. Par ailleurs, dans ce deuxième cas, le bloc est toujours totalement autocontrôlable ou sûr en présence de fautes si une seconde faute du même ensemble survient.

Le bloc de vérification est généralement conçu pour être totalement autocontrôlable et « à code disjoint », c'est-à-dire que tout vecteur du code d'entrée est transformé en un vecteur du code de sortie et tout vecteur d'entrée en dehors du code est transformé en un vecteur de sortie invalide.

Les **circuits implantés** selon ce principe sont appelés « **autocontrôlables** » (*self-checking*) pour les différencier des circuits autotestables (*self-testing*), c'est-à-dire testés par des éléments de BIST ou ayant la propriété mentionnée ci-dessus.

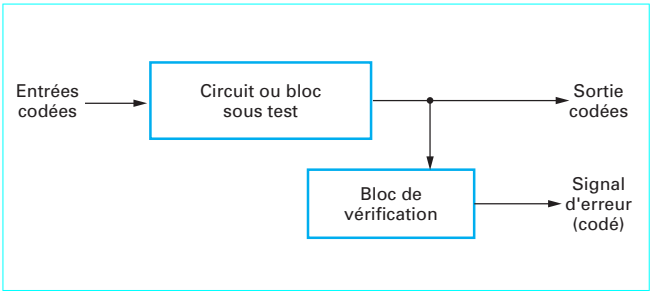

Figure 16 – Principe d'implantation d'un circuit autocontrôlable (*self-cheking*)

Différents types de codes peuvent être utilisés, en fonction des hypothèses de fautes, du type de bloc (logique aléatoire, élément de calcul arithmétique, etc.) et du compromis souhaité entre l'efficacité du test et les pénalités en surface ou en vitesse. Les codes détecteurs les plus utilisés sont la parité simple ou double, le code double rail, les codes arithmétiques séparables comme le code « résidu modulo 3 » et les codes non ordonnés comme les codes « *m* parmi *n* » et le code de Berger [5].

4.2 Test en ligne et synthèse

Une approche pour générer des circuits autocontrôlables consiste à modifier la structure électrique des cellules de base. Une autre approche consiste à utiliser des cellules classiques et à modifier le processus de conception au niveau du bloc. Des outils de synthèse spécifiques ont été proposés pour automatiser et optimiser cette deuxième approche.

La principale limitation des circuits autocontrôlables est la dépendance vis-à-vis du modèle de fautes choisi. Il a été indiqué que les modèles de fautes ne représentent qu'imparfaitement les conséquences des défauts physiques pouvant exister dans un circuit. Le comportement d'un circuit autocontrôlable est imprévisible lorsqu'un défaut non modélisé survient. Une autre approche consiste donc à fonder les modifications de structure du circuit sur des modèles d'erreurs, qui sont relatifs aux comportements aberrants du circuit, au lieu de modéliser les causes physiques de ces comportements erronés. Ainsi, des outils de synthèse ont par exemple été proposés pour détecter les erreurs de séquencement dans des blocs spécifiés sous la forme de machines à états finis. Cette approche plus fonctionnelle permet de mieux corréler les erreurs non détectées et les dysfonctionnements de l'application, donc de lier plus facilement le taux de couverture du test en ligne avec la probabilité d'évènements inacceptables lors du fonctionnement du circuit.

Des outils de synthèse ont également été proposés pour permettre non seulement l'intégration de blocs testés en ligne, mais aussi la génération optimisée de blocs tolérant certains types de fautes ou d'erreurs.

4.3 Association des dispositifs de test en ligne et hors ligne

Un test en ligne peut être continu ou périodique. Dans ce dernier cas, les dispositifs implantés pour le test hors ligne (*scanpath*, BIST…) peuvent souvent être réutilisés partiellement pour le test en ligne, ce qui permet de réduire les pénalités en surface et en vitesse occasionnées globalement par les dispositifs de test intégrés. Dans certains cas, il peut cependant être nécessaire d'adapter les dispositifs implantés pour qu'ils puissent être utilisés dans les deux contextes.

Exemple : considérons le test d'une mémoire RAM par un bloc de BIST.

Dans le cas du test hors ligne, il est possible d'écrire n'importe quelles valeurs dans la mémoire ; la contrainte majeure est le taux de couverture devant être obtenu pour un modèle de fautes donné.

Dans le cas d'un test en ligne se déroulant pendant l'exécution d'une application, le test ne peut généralement pas modifier les données de l'application qui ont été mémorisées. Un test périodique de la mémoire implique alors l'utilisation d'algorithmes devant atteindre le taux de couverture visé tout en satisfaisant cette contrainte supplémentaire ; une telle méthode de test « transparent » a été proposée par M. Nicolaidis [6].

Plus généralement, certains dispositifs intégrés (LFSR, MISR, BIST, *boundary scan*,...) peuvent avoir des rôles différents en phase de test hors ligne et lors du test en ligne.

Exemple : le *boundary scan* et les chaînes *scanpath* peuvent dans certains cas être employés pour espionner des états internes pendant le déroulement d'une application.

4.4 Test de courant en ligne

Les différentes variantes du test de courant peuvent être employées pour du test hors ligne, avec une mesure de courant réalisée par des dispositifs externes. Cette mesure peut aussi être réalisée par des dispositifs intégrés (BICS, ou *Built-In Current Sensor*), de tels dispositifs peuvent facilement être mis à profit également dans le cadre d'un test en ligne.

Exemple : Dans le domaine spatial, ce type de dispositif peut permettre la détection d'une faute de type SEU, qui induit un pic de courant anormal. Dans d'autres domaines, les mêmes dispositifs peuvent être employés pour détecter par exemple des fautes de retard dues à divers parasites.

4.5 Surveillance d'indicateurs de fiabilité

En complément des méthodes permettant de détecter certains types prédéfinis de fautes ou d'erreurs, le test de paramètres variés peut être envisagé pour s'assurer du bon fonctionnement d'une application malgré les variations d'environnement. Le test de courant par des dispositifs intégrés est un premier pas dans ce sens. D'autres types de capteurs sont développés pour surveiller d'autres caractéristiques donnant des indications sur la fiabilité du circuit ou sur la qualité et l'intégrité des signaux électriques. Les principaux éléments existant peuvent surveiller la température interne, certains délais de propagation, la raideur des fronts d'horloge, les zones de tension intermédiaires, l'activité de certains signaux, ou encore la dose radiative cumulée.

5. Outils CAO

Pour être envisageable, l'intégration de dispositifs de test se doit de ne pas pénaliser notablement le délai de mise sur le marché. La disponibilité d'outils de conception assistée par ordinateur (CAO) bien adaptés est donc indispensable pour réduire le temps de développement.

Il n'est pas possible de dresser ici un panorama complet de l'offre CAO. En effet, un panorama deviendrait vite obsolète compte tenu des évolutions très rapides aussi bien des techniques que des sociétés commercialisant les outils. Pour un tel panorama, le lecteur intéressé pourra se référer aux adresses indiquées à la rubrique « Pour en savoir plus ». L'objectif de cette section est plutôt d'identifier les principaux types d'outils commercialisés en 2002.

Il faut noter tout d'abord que la quasi-totalité des outils commerciaux se limitent au test hors ligne. Les tests couvrent classiquement les fautes de collage simple au niveau porte (cf. [E 2 460], § 2.2.1), et selon les cas peuvent prendre en compte les fautes de retard (cf. [E 2 460], § 2.2.6) ou les contraintes pour le test de courant ; rares sont les outils considérant d'autres modèles, sauf dans le cas de macrocellules (prise en compte par exemple des fautes de couplage dans les mémoires). Dans ce cadre limité, l'offre se renforce considérablement chaque année.

■ Les outils d'évaluation de testabilité, d'ATPG et d'évaluation du taux de couverture (simulation de fautes) ont été les premiers disponibles et sont aujourd'hui très répandus. Tous les principaux fournisseurs de CAO proposent un, voire plusieurs outils dans ces catégories. De nombreux fournisseurs spécialisés dans la CAO pour le test existent également. Les différences entre les outils résident dans les modèles de fautes visés et dans le type d'algorithme employé. Par exemple, l'évaluation du taux de couverture peut être fondée sur une « vraie » simulation de fautes (simulation logique ou logico-temporelle du circuit en présence des différentes fautes du modèle) ou sur un calcul de probabilités. Des outils différents peuvent permettre des compromis entre la précision de l'évaluation et le temps d'exécution nécessaire.

■ La deuxième catégorie d'outils apparue sur le marché concerne l'implantation de chaînes série. L'utilisation de verrous selon une approche similaire au LSSD est très peu représentée. En revanche, de nombreux outils permettent de créer des chaînes de *scanpath*.

Ces outils peuvent être destinés à créer une chaîne une fois la description du circuit connue au niveau portes.

La création de la chaîne est également possible avec certains outils pendant la phase de synthèse. L'avantage de cette dernière approche est de permettre une meilleure convergence du processus d'optimisation, la synthèse pouvant immédiatement tenir compte des délais induits par les multiplexages dus à la sérialisation.

L'insertion de *scanpath* peut être complète ou partielle, selon les outils et les contraintes de l'utilisateur.

■ Bien qu'ayant existé, l'automatisation d'un partitionnement à base de multiplexeurs pour le test parallèle est très peu disponible dans les outils actuels. En revanche, l'implantation de blocs d'autotest est automatisée par différents outils, en particulier pour les blocs de mémoire (ROM, RAM statique et plus récemment RAM dynamique). Différentes variantes existent, pour différents algorithmes de génération de vecteurs, avec ou sans connexion à un *scanpath*, avec ou sans compaction des résultats du test, etc.

L'implantation d'éléments d'autotest pseudo-aléatoire pour des blocs de logique a été considérablement développée récemment. Une offre variée existe maintenant, que ce soit pour du test parallèle (connexion du générateur de vecteurs sur les différentes entrées du bloc à tester) ou du test série (connexion du générateur de vecteurs en entrée d'une chaîne de *scanpath*).

Ces fonctions d'autotest n'incluent généralement pas aujourd'hui de capacités de diagnostic.

■ Une dernière catégorie d'outils disponibles vise l'automatisation de l'implantation du *boundary scan* selon les règles de la norme IEEE 1149.1. L'automatisation peut concerner l'implantation des dispositifs de base (registre de périphérie, registre *bypass*, machine à états finis, etc.). Elle peut aussi permettre la définition d'instructions utilisateur et la connexion aux chaînes de *scanpath* ou aux autotests internes, à partir d'une description en langage BSDL.

6. Exemples d'utilisation des techniques de DFT

Certaines techniques de conception pour la testabilité (DFT) sont aujourd'hui utilisées dans tous les circuits complexes, même dans des circuits produits en très gros volume. Si les techniques les plus avancées restent encore peu répandues, les autotests de mémoires et l'implantation de chemins série (*scanpath* ou LSSD) se retrouvent dans de très nombreux produits. La compatibilité, au moins minimale, avec la norme IEEE 1149.1 se généralise également. Le test des microprocesseurs d'usage général est une bonne illustration de cette évolution.

Exemple : le processeur superscalaire AMD-K7 comporte environ 100 000 bascules D, dont 90 % sont placées dans des chaînes de *scanpath* pour permettre au maximum l'utilisation des ATPG structurels combinatoires. La plupart des bascules qui ne sont pas mises dans les chemins série correspondent à des éléments de mémorisation pouvant être facilement testés en utilisant la fonctionnalité du circuit, ou inclus dans des macrocellules optimisées ayant été dessinées au micron (*full-custom*) et dans lesquelles l'insertion d'un *scanpath* est difficile. A ceci s'ajoutent les éléments de mémorisation implantés pour le *boundary scan* (IEEE 1149.1). Les bascules internes reliées en chemin série sont réparties en 48 chaînes (valeur maximum autorisée par l'équipement de test externe employé lors de la fabrication). Cette répartition permet de limiter les temps de chargement/déchargement et de conserver un temps total d'application du test en dessous de 1s. Ces chaînes peuvent toutefois être connectées bout à bout dans un mode particulier permettant de faciliter le déverminage au niveau du système. La chaîne série ainsi constituée est alors placée entre TDI et TDO et peut être utilisée par l'intermédiaire de l'interface *boundary scan*. Des dispositifs de test permettent aussi de contourner le PLL implanté pour la génération d'horloge interne, de façon à contrôler directement l'horloge depuis les broches extérieures pendant l'application par les chaînes de *scanpath* des vecteurs définis par l'ATPG. L'augmentation totale de surface pour l'implantation des *scanpath* est de l'ordre de 3,25 %.

On peut prendre comme autre exemple le premier processeur PowerPC de quatrième génération. Le MPC7400 inclut des chaînes LSSD permettant d'appliquer dans les blocs de portes logiques des vecteurs de test générés pour le test des collages simples. Les macrocellules mémoires sont testées par des éléments d'autotest. Les plus petits réseaux réguliers et la logique d'horloge de bus utilisent des vecteurs de test fonctionnels. Des vecteurs fonctionnels sont également employés pour le test des fautes de délai et pour le tri en fréquence (*speed grading*). Le *boundary scan*, implanté pour le test des interconnexions au niveau carte, permet aussi l'accès aux dispositifs de test intégré par des instructions propriétaires.

7. Conclusion

Le test (et la conception pour la testabilité) n'est pas une fin en soi, mais une tâche destinée à aider et/ou améliorer la production. Le test ne facilite pas la conception du circuit, mais au contraire la conception doit simplifier le test en fin de fabrication, en maintenance et pendant les phases opérationnelles. L'augmentation éventuelle du coût de conception du circuit est alors compensée par une baisse des coûts du test lors de la production de ce même circuit, une baisse des coûts de production des équipements qui l'utilisent, et des gains sur les ventes des circuits futurs grâce à une meilleure qualité induisant une meilleure image de marque et une plus grande satisfaction du client.

L'augmentation du ratio entre la bande passante interne et la bande passante externe rend de plus en plus difficile (et inefficace) le test d'un circuit par application de vecteurs avec un équipement externe. La tendance est donc une évolution de la répartition entre test externe et test intégré, au profit du test intégré. Les chaînes série (*scanpath* ou LSSD) ont été un premier pas dans ce sens, suivies par les autotests, en particulier pour les mémoires embarquées. Les autotests pour d'autres types de blocs commencent à se répandre, poussés en particulier par la réutilisation de blocs complexes (IP). Des blocs intégrés permettant de contrôler le test et de limiter les échanges d'information avec les équipements externes commencent à voir le jour. L'extension vers le diagnostic et la réparation pour les éléments réguliers, voire l'autoréparation (BISR) de ces blocs, est l'étape suivante. À ceci s'ajoute l'intérêt croissant pour les techniques permettant d'assurer un test continu en opération (test en ligne), de façon à pouvoir faire face aux fautes transitoires.

Au-delà du test, il faut noter également l'évolution des méthodes de conception pour prendre en compte d'autres contraintes. Des modifications de la structure et/ou de l'implantation du circuit peuvent en effet permettre d'améliorer le rendement de fabrication (DFM : *Design for Manufacturing*) ou la fiabilité du circuit (DFR : *Design for Reliability*, qui inclut mais ne se limite pas au test en ligne).

Test des circuits intégrés numériques

par **Régis LEVEUGLE**

Ingénieur de l'École nationale supérieure d'électronique et de radioélectricité de Grenoble (ENSERG)

Professeur à l'Institut national polytechnique de Grenoble (INPG)

Laboratoire des techniques de l'informatique et de la microélectronique pour l'architecture d'ordinateurs (TIMA)

Bibliographie

Références

[1] HORTENSIUS (P.D.), McLEOD (R.D.) et CARD (H.C). – *Parlalel random number generation for VLSI systems using cellular automata.* IEEE transactions on Computers, vol. 38, n° 10, p. 1466-1473 (octobre 1989).

[2] HORTENSIUS (P.D.), McLEOD (R.D.) et CARD (H.C). – *Cellular automata-based signature analysis for buit-in self-test.* IEEE transactions on Computers, vol. 39, n° 10, p. 1273-1283 (octobre 1990).

[3] LESTRAT (P.) et LEVEUGLE (R.). – *A flexible approach to built-in test and self-test implementation in VLSI circuits.* International Workshop on Defect and Fault Tolerance in VLSI Systems, Grenoble, France, p. 150-161 (novembre 1990)

[4] NICOLAIDIS (M.) et ZORIAN (Y.). – *On-line testing for VLSI – A compendium of approaches.* Journal of Electronic Testing : Theory and Applications (JETTA), vol. 12, n° 1/2, p. 7-20 (février-avril 1998).

[5] WAKERLY (J). – *Error detecting codes, self-checking circuits and applications.* Elsevier (1978).

[6] NICOLAIDIS (M). – *Theory of transparent BIST for RAMs.* IEEE Trans on computers, vol. 45, n° 10, p. 1141-1156 (octobre 1996).

Revues

IEEE design & test of computers

IEEE transactions on computers

IEEE transactions on computer-aided design

IEEE transactions on very large scale integration (VLSI) systems

Journal of electronic testing : theory and applications (JETTA).

Conférences

De nombreux colloques et conférences dans le domaine de la conception de circuits ont des sessions consacrées au test et peuvent être intéressants pour compléter l'information donnée dans cet article. On peut citer les conférences annuelles suivantes dont les actes sont édités par IEEE Computer society press :

International test conference (ITC) (États-Unis)

VLSI test symposium (VTS) (États-Unis)

Design automation and test in Europe (DATE) (Europe)

Sites internet

Informations générales

http://www.tmworld.com

(informations régulièrement mises à jour, magazine en ligne et guide d'achat avec la liste des principaux fournisseurs de matériel et d'outil CAO. Ce site ne se limite pas aux circuits intégrés numériques ; des informations peuvent aussi être obtenues pour le test analogique et le test des cartes.)

ITRS

http://public.itrs.net

(document édité sous l'égide de l'Association des industriels du semi-conducteur SIA (Semiconductor industry association) dans lequel sont détaillées les principales évolutions prévues pour la prochaine décennie).

Tests d'IP

http://grouper.ieee.org/groups/1500/

(détails sur la standardisation dans le domaine du test de blocs réutilisables (IP)).

Institute of electrical and electronics engineers Inc. (IEEE)

http://www.ieee.org

IEEE test and technology technical council (TTTC)

http://computer.org/tttc

Normalisation

Institute of electrical and electronics engineers Inc. (IEEE)

IEEE 1449.1 2001 Test access port and boundary scan architecture

TECHNIQUES DE L'INGÉNIEUR

GAGNEZ DU TEMPS ET SÉCURISEZ VOS PROJETS EN UTILISANT UNE SOURCE ACTUALISÉE ET FIABLE

Techniques de l'Ingénieur propose la plus importante collection documentaire technique et scientifique en français !

Grâce à vos droits d'accès, retrouvez l'ensemble des **articles et fiches pratiques de votre offre, leurs compléments et mises à jour,** et bénéficiez des **services inclus.**

RÉDIGÉE ET VALIDÉE PAR DES EXPERTS	MISE À JOUR PERMANENTE	100 % COMPATIBLE SUR TOUS SUPPORTS NUMÉRIQUES	SERVICES INCLUS DANS CHAQUE OFFRE

- **+ de 350 000 utilisateurs**
- **+ de 10 000 articles de référence**
- **+ de 80 offres**
- **15 domaines d'expertise**

- **Automatique - Robotique**
- **Biomédical - Pharma**
- **Construction et travaux publics**
- **Électronique - Photonique**
- **Énergies**
- **Environnement - Sécurité**
- **Génie industriel**
- **Ingénierie des transports**
- **Innovation**
- **Matériaux**
- **Mécanique**
- **Mesures - Analyses**
- **Procédés chimie - Bio - Agro**
- **Sciences fondamentales**
- **Technologies de l'information**

Pour des offres toujours plus adaptées à votre métier, découvrez les offres dédiées à votre secteur d'activité

Depuis plus de 70 ans, Techniques de l'Ingénieur est la source d'informations de référence des bureaux d'études, de la R&D et de l'innovation.

www.techniques-ingenieur.fr

CONTACT : Tél. : + 33 (0)1 53 35 20 20 - Fax : +33 (0)1 53 26 79 18 - E-mail : infos.clients@teching.com

TECHNIQUES DE L'INGÉNIEUR

LES AVANTAGES ET SERVICES compris dans les offres Techniques de l'Ingénieur

ACCÈS

Accès illimité aux articles en HTML
Enrichis et mis à jour pendant toute la durée de la souscription

Téléchargement des articles au format PDF
Pour un usage en toute liberté

Consultation sur tous les supports numériques
Des contenus optimisés pour ordinateurs, tablettes et mobiles

SERVICES ET OUTILS PRATIQUES

Questions aux experts*
Les meilleurs experts techniques et scientifiques vous répondent

Articles Découverte
La possibilité de consulter des articles en dehors de votre offre

Dictionnaire technique multilingue
45 000 termes en français, anglais, espagnol et allemand

Archives
Technologies anciennes et versions antérieures des articles

Impression à la demande
Commandez les éditions papier de vos ressources documentaires

Alertes actualisations
Recevez par email toutes les nouveautés de vos ressources documentaires

*Questions aux experts est un service réservé aux entreprises, non proposé dans les offres écoles, universités ou pour tout autre organisme de formation.

ILS NOUS FONT CONFIANCE

AgroParisTech · AIRBUS · cea · CESI Enseignement Supérieur et Formation Professionnelle · cnes Centre National d'Études Spatiales · CSTB le futur en construction · ENGIE · GE · L'ORÉAL · Leroy-Somer · École Polytechnique Université Paris-Saclay · SAFRAN Aerospace Defence Security · Sagemcom · SAINT-GOBAIN · Schneider Electric · THALES · TOTAL · Valeo · VEOLIA · Wallonie · SPW Service public de Wallonie

www.techniques-ingenieur.fr

CONTACT : Tél. : + 33 (0)1 53 35 20 20 - Fax : +33 (0)1 53 26 79 18 - E-mail : infos.clients@teching.com