Titre: Title:	Conception et réalisation d'un module de test d'intégrité des signaux pour un circuit intégré de grande surface
Auteur: Author:	Daniel Picard
Date:	2004
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Picard, D. (2004). Conception et réalisation d'un module de test d'intégrité des signaux pour un circuit intégré de grande surface [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8928/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/8928/
Directeurs de recherche: Advisors:	Yvon Savaria, & Claude Thibeault
Programme: Program:	Unspecified

UNIVERSITÉ DE MONTRÉAL

CONCEPTION ET RÉALISATION D'UN MODULE DE TEST D'INTÉGRITÉ DES SIGNAUX POUR UN CIRCUIT INTÉGRÉ DE GRANDE SURFACE

DANIEL PICARD
DÉPARTEMENT DE GÉNIE ÉLECTRIQUE ET DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉLECTRIQUE)
Août 2004

Your file Votre référence
ISBN: 0-612-97976-8
Our file Notre référence
ISBN: 0-612-97976-8

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

CONCEPTION ET RÉALISATION D'UN MODULE DE TEST D'INTÉGRITÉ DES SIGNAUX POUR UN CIRCUIT INTÉGRÉ DE GRANDE SURFACE

présenté par : PICARD Daniel
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. AUDET Yves, Ph.D., président
M. SAVARIA Yvon, Ph.D., membre et directeur de recherche
M. THIBEAULT Claude, Ph.D., membre et codirecteur de recherche
M. KHOUAS Abdelhakim, Ph.D., membre

REMERCIEMENTS

J'aimerais adresser mes premiers remerciements à mon directeur, le professeur Yvon Savaria, pour m'avoir permis de travailler sur ce projet. Yvon a su me guider et me donner de précieux conseils tout au long de ce projet. Je voudrais également remercier la collaboration de mon co-directeur Claude Thibeault. Grâce à ses connaissances, Claude a apporté à ce projet des concepts très profitables. Je tiens également à souligner la participation essentielle des membres du jury, soit le professeur Yves Audet, et le professeur Abdelhakim Khouas.

Avoir eu la chance de réaliser ma maîtrise en entreprise fût une expérience plus que profitable. Le fait de côtoyer des gens expérimentés m'a permis d'accroître mes connaissances et ce, autant sur le plan technique qu'humain. Je tiens à exprimer ma gratitude à Hyperchip Inc. et particulièrement à David Chamberlain, Richard Norman, Ara Talasian et Karl Fecteau. J'aimerais également remercier tous ceux qui m'ont apporté leur appui : Boris, Meng, Bing et Stéphane.

Je ne pourrais oublier de remercier profondément mes parents, Réjean et Anny, qui m'ont encouragé depuis le tout début de mes études et dans les moments importants. Également, un gros merci à Annie Pelletier pour son support inestimable, le temps qu'elle a consacrée à la révision de ce mémoire et plus. Finalement, je tiens à remercier mon frère Stéphan qui fût un modèle à suivre tout au long de mes études. Il a toujours su m'appuyer, m'encourager et m'aider depuis le début. Merci.

RÉSUMÉ

Les systèmes électroniques doivent répondre à une demande croissante de bande passante et de rapidité. Par conséquent, les concepteurs ont besoin de circuits intégrés possédant une plus haute densité et fournissant un plus grand nombre de portes logiques. Jusqu'à présent, l'obtention de circuits intégrés de plus en plus denses a été possible principalement via la miniaturisation électronique. En réduisant la taille des transistors, des cellules de mémoire et des interconnexions électriques, les concepteurs peuvent obtenir une densité plus élevée.

L'adoption d'une technologie électronique miniaturisée permet d'améliorer les performances d'un système. Cependant, pour donner un nouveau souffle à la miniaturisation, le présent projet propose d'augmenter la surface utilisable du dé de silicium. Le projet LAIC (Large Area Integrated Circuit) utilise une approche technologique de jonction inter-réticulaire qui permet de répondre aux besoins croissants. Les concepteurs pourront ainsi utiliser des circuits intégrés offrant un plus grand nombre de portes logiques et ce, sans nécessairement adopter la dernière technologie offerte.

Le projet LAIC est tout d'abord une réalisation permettant l'étude de faisabilité de la technologie de jonction inter-réticulaire. Ceci permettra d'obtenir des données sur cette technologie. Ces paramètres fourniront des renseignements provenant des 3 sections qui composent le projet LAIC. D'une part, la section *jonction inter-réticulaire* (ST) permet de relier les réticules voisins les uns avec les autres. Cette partie comporte une section permettant de trouver plusieurs possibilités de défectuosités physiques et de les identifier. La seconde partie est la section *tolérante aux pannes* qui permet de configurer

des modules redondants dans le système principal lorsque les circuits primaires sont défectueux. Les remplacements peuvent être utilisés pour les modules ainsi que les lignes de branchements entre les modules. La dernière section dite *d'intégrité des signaux* permet de quantifier des délais associés aux problèmes agissant sur les connexions numériques. En plus, elle permet de réaliser une re-synchronisation des données lors de passages entre différents réticules. Cette section permet également de créer des délais artificiels afin d'ajuster la phase des signaux et de calibrer leurs temps d'arrivée.

La dernière section qui étudie l'intégrité des signaux sera l'objet de ce mémoire. L'objectif de ce projet est de concevoir une puce utilisant une technologie CMOS à 0.35 micron, afin de valider et de fournir des informations quantitatives sur l'intégrité des signaux sur de longues lignes. Toutes les étapes nécessaires à la création d'une puce ont été réalisées, en partant de l'établissement des spécifications, à la création de l'architecture, en passant par la conception des modules, le dessin des masques et la vérification par simulation, pour aller jusqu'à la fabrication du prototype.

Le circuit LAIC, fruit du projet du même nom, a donc été fabriqué et encapsulé afin de réaliser les tests de caractérisation en laboratoire. Au moment de la rédaction de ce mémoire, la phase de test en laboratoire était débutée, mais suspendue pour un certain temps. Suite aux résultats obtenus, il sera ensuite possible d'envisager une pré-commercialisation d'un nouveau projet utilisant les fondements développés lors du projet LAIC.

ABSTRACT

The increase of demand for bandwidth and operating frequency imposes increasingly severe requirements on electronic systems. This results in IC designers needing higher densities of integration and a larger number of electronic gates. So far, the response to these needs was electronic miniaturisation: by reducing the size of transistors, memory cells and electrical interconnections, designers can obtain higher integration densities.

However, technology scaling approaches its limits and it is worth exploring other approaches to increase electronic system density. This project proposes to increase the effective area of the silicon die. The LAIC (Large Area Integrated Circuit) project uses inter-reticule stitching technology in response to the growing needs for progressing toward SOC (systems on a chip). The designers will be able to use an integrated circuit with relaxed area limitations, while keeping the same density. According with the project specifications, the designer could adopt the most recent technology or some older more mature technology.

The LAIC project is a feasibility study to demonstrate inter-reticule stitching technology.. Possible operating parameters were validated experimentally through the LAIC project. The first part, the *inter-reticule stitching* (ST) module, makes possible to connect a reticule with its neighbours. This part includes a section that can detect several possible physical defects and allows identifying them. The second part is the *fault-tolerant* section, which makes it possible to configure the use of redundant modules, when primary modules are defective. Spares are available for modules as well as for connection between them. The last section, the *signal integrity* module, allows quantifying delays associated with digital connections. Moreover, it allows

resynchronizing data that passes through reticules boundaries. This section also allows adding delays to tune phase relationships between signals.

The *signal integrity* module is the main focus of this thesis. The objective of this project is to design a chip using a 0.35-micron CMOS technology to validate methods used to implement long interconnects and to provide quantitative information on the signal integrity of long lines. The steps necessary to the create a prototype chip were: establishment of the specifications, creation of the architecture, module design, layout drawing, verification and simulation.

This project led to manufacturing of a chip called the LAIC chip. At the time of completing this thesis, the test phase was partly completed. Based on the partial test results obtained, the project is considered a success and possible commercial use of LAIC technologies can be considered.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ABRÉVIATIONS ET SYMBOLES

ABBRÉVIATIONS

CMOS	Complementary Metal Oxyde Semiconductor
CSM	Control skew module
DUS	Device under simulation
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
LAIC	Large Area Integrated Circuit
LCLK	Local Clock
MOS	Metal Oxyde Semiconductor
MUX	Multiplexeur
PFD	Phase frequency detector
PLL	Phase Locked Loop
RCLK	Received clock
SoC	System-on-Chip
TFT	Temporal fine tuning
VLSI	Very Large Scale Integration
VOS	Voltage oscillator signal
WSI	Wafer Scale Integration

PRINCIPAUX SYMBOLES

A	Air des plaques parallèles
ε	Permittivité du diélectrique
$\Delta\phi$	Variation de phase
ε_O	Permittivité dans le vide = 8.8542 x 10-12 (F/m)
ε_R	Permittivité relative du matériel = 4.2 pour le SiO2
C	Condensateur
CLK	Signal d'horloge
d	Distance entre les deux plaques parallèles.
i(t)	Courant en fonction du temps
T_{CP}	Période maximale de l'horloge entre deux registres
T_{C-Q}	Temps horloge vers la sortie
T_{INT}	Temps de propagation de la ligne
T_{PD}	Temps de propagation
T_{rq}	Temps de remise à zéro vers la sotie
T_{SETUP}	Temps de stabilisation de la donnée à l'entrée du second registre
V(t)	Tension en fonction du temps
V_{IH}	Tension d'entrée haute
V_{IL}	Tension d'entrée basse
V_{IN}	Tension d'entrée
V_{MS}	Tension de métastabilité
V_{OUT}	Tension de sortie
Z	Impédance

LISTE DES ANNEXES

CHAPITRE 1

INTRODUCTION

La création de nouveaux projets peut être issue de deux sources. La première permet de répondre à un besoin spécifique. La seconde surgit d'une idée qui peut engendrer un besoin. Les télécommunications, les systèmes temps réels, les consoles de jeux, et les mémoires embarquées de grande capacité sont des exemples d'applications qui ont des besoins croissants dans le domaine des nouvelles technologies.

La chirurgie à distance est un exemple concret qui nécessite de nouvelles technologies. Le plus gros centre de téléchirurgie, le IRCAD, a vu le jour en 1994[7]. La téléchirurgie a progressée en trois phases. Premièrement par la laparoscopie[1], ensuite par la chirurgie assistée par ordinateur et finalement la chirurgie à distance. Cependant, lors de la première expérience, sur un animal, l'équipe de travail s'est aperçue de certains obstacles de la technologie. Le temps de traitement des données (visuelles, contrôles et auditives) ajouté au temps de propagation devient un enjeu important lorsque la distance entre le patient et le chirurgien s'accroît. Ainsi, le traitement en temps réel est un aspect critique dans ce domaine.

Dans un autre ordre d'idées, le domaine récréatif, tel que les jeux vidéos, connaît actuellement une poussée croissante de popularité. Les animations à trois dimensions, les graphiques de plus en plus perfectionnés et la complexité des jeux font en sorte d'augmenter sans cesse la demande en bande passante, en mémoire et ce, tout en conservant la plus petite taille possible.

[1] La laparoscopie opératoire permet de pratiquer, dans un but thérapeutique, des interventions chirurgicales mineures et majeures à l'aide d'instruments de chirurgie miniaturisés.

Également, l'entrée d'Internet dans plusieurs foyers[2] [8], l'explosion d'Internet à haute vitesse lors des dernières années et la venue de la voix sur le protocole Internet sont d'autres exemples ayant des besoins croissant des nouvelles technologies. La transmission des données, de la voix et de l'image font en sorte d'augmenter la quantité de donnés sur un réseau de communication. Les exigences concernant la compatibilité ainsi que l'ajout de nouveaux protocoles de communication de plus en plus performants nécessitent un accroissement des débits de traitement des données. Par exemple, les échanges entre les mémoires et les puces de traitements (CPUs, contrôleurs, dédiées) s'exécutent à des fréquences de plus en plus élevées et sur des largeurs de bus de plus en plus grandes. Pour satisfaire les exigences techniques de ces trois exemples, les concepteurs se tournent vers la microélectronique.

La microélectronique est un domaine qui se situe en avant plan dans tous les exemples cités auparavant. La demande du marché requiert sans cesse des composants de plus en plus rapides et petits. L'intégration des circuits est une voie de plus en plus adoptée dans les nouvelles technologies. Cette industrie progresse à une vitesse prodigieuse. En 1965, Gordon Moore, directeur de la recherche et développement chez Fairchild Semiconductor, a écrit un papier intitulé : « Cramming more components onto integrated circuits. » Dans cet article, monsieur Moore avance: « The complexity for minimum component cost has increased at a rate of roughly a factor of two per year. » Cette observation devient popularisée sous la loi de Moore. Plus tard, cette loi fut ajustée pour être : « Le nombre de composants par IC doubles à tous les 18 à 24 mois. » [2]. La figure 1.1 dénote le nombre de transistors en fonction du temps sur plusieurs composants offerts sur le marché [3].

[2] En 1998 15,6 % des ménages comptait au moins un utilisateur régulier d'Internet à la maison comparativement à 42,2% en 2002.

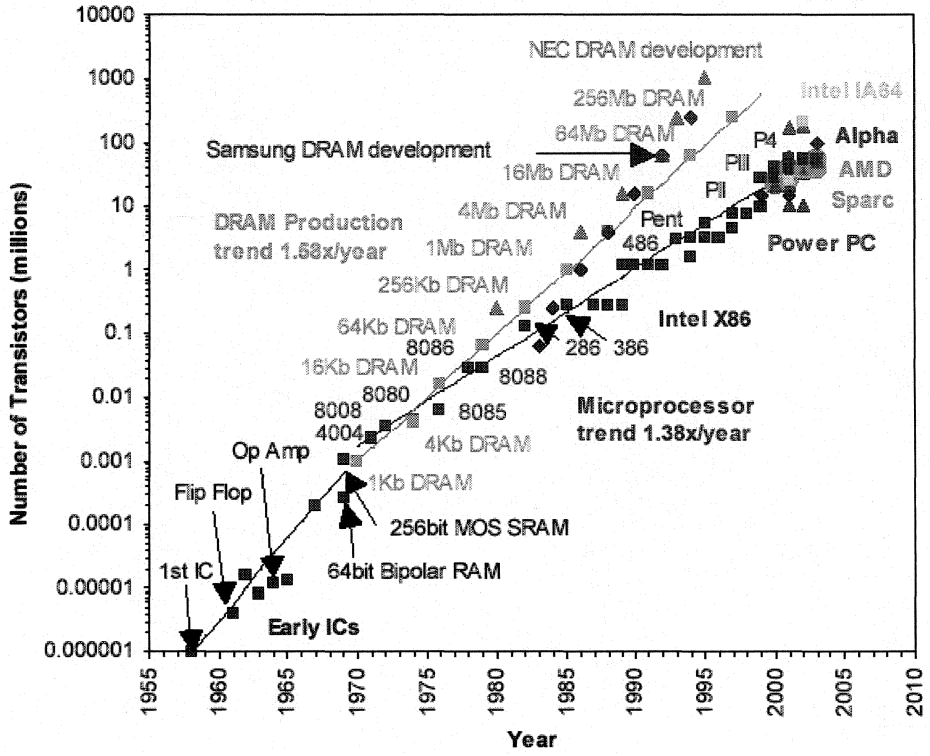

Figure 1.1 : Évolution du nombre de transistors par puce tiré de icknowledge [3]

Les possibilités les plus envisagées afin de satisfaire la soif de l'industrie sont : la réduction à l'échelle et/ou l'augmentation de l'aire des circuits. La demande de performance accrue envers les systèmes intégrés est de plus en plus difficile à satisfaire. Un passage graduel de la conception VLSI vers la conception à l'échelle de la gaufre devient ainsi une option.

1.1 Historique de l'intégration à l'échelle de la tranche

Le premier essai d'intégration à l'échelle de la tranche a été effectué au milieu des années 1960 chez Texas Instruments. Il visait le développement d'un multiplicateur

parallèle de 8 bits par 8 bits implémenté sur une tranche de 38.1 mm (1.5") [4]. La conception à l'échelle de la tranche vise généralement à exploiter une technologie afin de fabriquer des circuits intégrés dont la surface est la plus grande possible. Les avantages espérés sont : augmenter la performance, réduire la puissance dissipée, réduire la longueur des interconnexions et le nombre de connexions aux boîtiers, réduire les effets parasites et augmenter la densité d'intégration. Cependant, certains obstacles tels que le rendement, les effets associés aux longues interconnexions, le défi d'encapsuler des puces surdimensionnées et les problèmes thermiques ont constamment freiné l'émergence des systèmes intégrés à l'échelle de la tranche. La figure 1.2 illustre la tranche réalisée dans le cadre de ce projet.

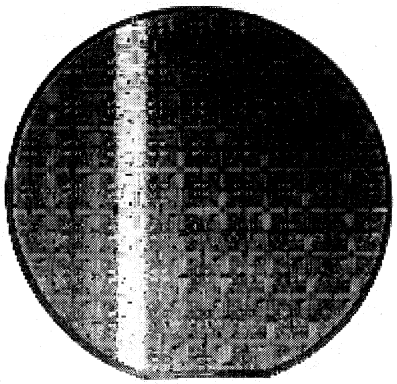

Figure 1.2 : Tranche réalisée dans le cadre de ce projet

Une nouvelle approche permettant d'atteindre un niveau d'intégration supérieur aux techniques actuelles a été expérimentée. Le groupe de recherche et développement d'Hyperchip Inc. et ses partenaires ont mis au point une technologie visant à faire évoluer les méthodes de fabrication actuellement utilisées. Les techniques expérimentées permettent de fabriquer, pour une technologie donnée, un dé dont la taille est quatre fois supérieure à la superficie maximale de ceux couramment fabriqués [5] [6]. Le dé est une partie de la tranche refermant la logique qui sera par la suite encapsulée pour constituer

le circuit intégré. La figure 1.3 et 1.4 représente respectivement le dé du LAIC et le LAIC encapsulé.

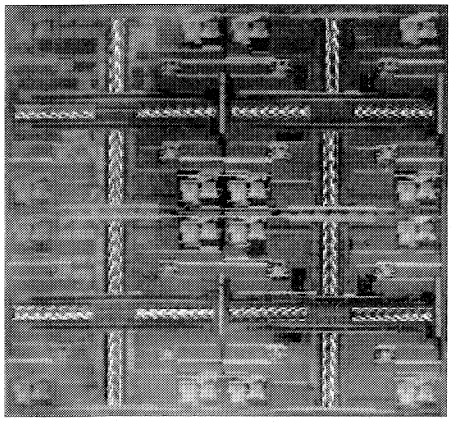

Figure 1.3 : Dé du LAIC

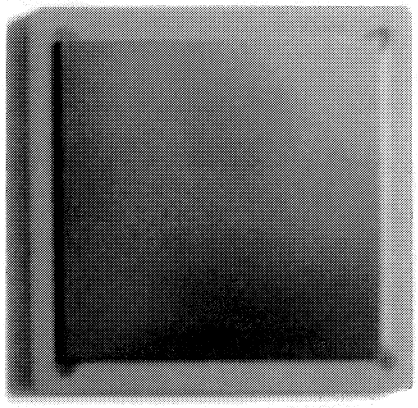

Figure 1.4 : LAIC encapsulé

Le projet LAIC (Large Area Integrated Circuit) porte sur la conception et l'implantation d'un circuit intégré à l'échelle de la tranche. Tel que spécifié auparavant, il représente une étape intermédiaire où pour une technologie donnée, la surface utilisable du dé est de 4 fois supérieure à celle des plus grands circuits normalement réalisés. Par une telle implantation, plusieurs facteurs deviennent essentiels à analyser. Ce travail porte sur l'un de ces facteurs, soit l'analyse de l'intégrité des signaux. Cette analyse devient un aspect essentiel destiné à satisfaire adéquatement les exigences technologiques actuelles. Ce mémoire présente les différents effets associés à l'intégrité des signaux. Il portera particulièrement sur l'analyse des biais de synchronisation, des moyens permettant d'évaluer quantitativement les résultats et la synchronisation.

La méthodologie soutenue dans ce projet consiste à approfondir la science de la conception d'un circuit intégré de grande surface opérant à haut régime. Un chapitre

présentera l'étude des notions directement reliées au projet. Ces notions portent plus particulièrement sur des problématiques reliées à l'intégrité des signaux tel que la diaphonie entre les longues lignes de communication, les biais de synchronisation et la synchronisation. Cette partie enveloppe également les spécifications reliées à la réalisation du projet LAIC. L'explication des spécifications portera également sur une vue d'ensemble de la structure architecturale. Une brève explication de la technologie utilisée pour augmenter la surface utilisable, la technique de jonction inter-réticulaire, sera présentée.

Par la suite, un second chapitre portera sur la réalisation des différents modules qui forment la cellule SI. Cette conception a été réalisée à partir de cellules normalisées utilisant la technologie CMOS 0.35 microns et par la fabrication de nos propres cellules. Les cellules normalisées, de type « P », ont été réalisées sur une tranche de silicium. À l'aide de ces cellules, nous avons créé les différents modules permettant d'interfacer avec la cellule SI et les fonctions utilisées afin de générer et d'effectuer la mise en forme des résultats.

Le chapitre 2 propose la description de la réalisation des simulations afin d'évaluer les corrélations avec les spécifications. Une explication de la méthodologie utilisée pour la simulation de la cellule sera fournie. Par la suite, les simulations hiérarchiques de tous les modules qui composent la cellule SI seront présentées. Finalement, la simulation de la cellule SI entière permettra de valider son bon fonctionnement.

CHAPITRE 2

ANALYSE ET ÉVALUATION DES BIAIS DE SYNCHRONYSATION À L'ÉCHELLE DE LA TRANCHE

2.1 Introduction : Problématique de l'intégrité de signaux

L'intégrité des signaux est un thème de plus en plus préoccupant lors de la conception de circuits intégrés à grande échelle. Plusieurs facteurs amènent les concepteurs à se soucier de la qualité des signaux. La progression rapide de la commutation des transistors, l'augmentation de la surface utile et la longueur des connexions ne sont que quelques exemples qui peuvent affecter la qualité des signaux. L'objectif est d'être en mesure de fournir, à un récepteur, un signal possédant une qualité suffisante afin que les informations reçues soient telles qu'elles étaient à la transmission.

2.2 Définition de la terminologie et revue théorique de l'intégrité des signaux

Pour une technologie donnée, deux moyens s'offrent aux concepteurs afin de remplir l'objectif cité à la section 2.1. Le premier moyen concerne les techniques ou règles de conception. Par exemple, il est possible d'effectuer une partie de la conception à un niveau hiérarchique plus bas, à l'aide de cellules normalisées, contrairement à une conception logicielle. On peut aussi réaliser manuellement les connexions critiques au lieu d'utiliser des logiciels automatiques. Il est important de porter une attention particulière aux signaux d'horloge, d'alimentation ainsi qu'aux signaux analogiques.

Le second moyen consiste à mettre en place des mécanismes permettant de satisfaire les exigences de l'application, tout en minimisant le déficit de performances engendré au système. Un des mécanismes simples, proposé par *Rabaey* [16], serait par exemple l'utilisation de répéteurs situés à des endroits stratégiques afin de réduire le délai associé aux lignes de transmission. Un autre pourrait être de procéder à l'ajout d'un système de re-synchronisation afin de permettre la communication entre deux domaines d'horloge sans trop de biais de synchronisation et de métastabilité.

Plusieurs facteurs peuvent nuire à la qualité d'un signal tels que les biais de synchronisation, la diaphonie (cross-talk) et les différences de phases. Ces facteurs limitent la fréquence d'horloge maximale et peuvent induire des erreurs de communication. Ainsi il ne faut pas les négliger lors de la réalisation de circuits intégrés à très grande échelle. Nous allons expliquer, brièvement, différentes sources de problèmes pouvant affecter la qualité des signaux dans les sections suivantes.

2.2.1 La diaphonie entre les lignes de communication

La diaphonie (*crosstalk*) est un problème particulièrement important sur les longues lignes de communication adjacentes. Les sources de diaphonie incluent le couplage capacitif (a) et les inductances mutuelles (b) (voir figure 2.1). *Jhang et Ha* [11] donnent la définition suivante de la diaphonie: *Lorsque deux lignes alimentées par un courant électrique sont placées à proximité l'une de l'autre, les interactions électromagnétiques entre les lignes peuvent affecter les données transmises le long de ces lignes.* Ainsi, les champs électriques ou magnétiques, selon le cas, créés par une variation de courant ou de tension sur une ligne se reflètent par couplage sur la ligne voisine. Ceci a comme conséquence de modifier le niveau de tension ou de courant de celle-ci. Dans le cas où le niveau atteint une valeur supérieure au niveau de seuil, le niveau logique sera interprété de manière erronée. Certains facteurs qui influencent la diaphonie sont le coefficient de

perméabilité du diélectrique utilisé entre les lignes, la fréquence de communication, la puissance de l'émetteur, l'épaisseur et la longueur de la ligne. Une capacité parasite ou une inductance mutuelle est formée. Ceci fournit un canal qui permet des transferts de charges.

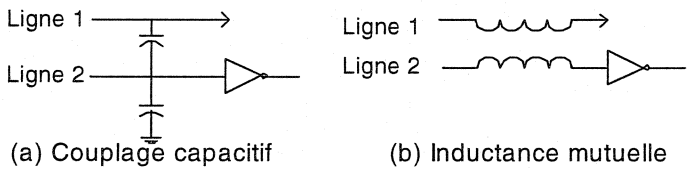

Figure 2.1 : Les sources de la diaphonie

En plus de ces interférences indésirables, un aspect qui influence la qualité des signaux est la longueur des lignes de communication. Plusieurs modèles de lignes ont été étudiés, à un niveau comportemental, afin de caractériser l'accumulation d'éléments passifs due à l'allongement de la ligne. L'objectif est de converger vers un modèle qui représente le mieux la réalité de notre connexion. Les modèles d'interconnexions intégrées (voir figure 2.2) les plus communs sont : le modèle en π, le modèle en T et celui d'une combinaison de segments RC. Ces modèles représentent une ligne comme un ensemble de composants RC qui permettent de caractériser la constante de temps du réseau.

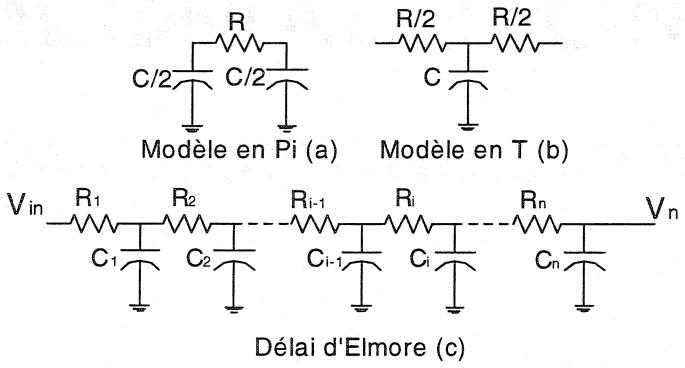

Figure 2.2 : Modèles de distribution des lignes RC

Il ne suffit pas que de calculer les délais des lignes de transmission, mais également d'être en mesure de réduire leurs effets. Selon le modèle distribué, la ligne est représentée par un ensemble de résistances et de capacités. Sur la base de ce modèle, il est possible de cibler l'endroit sur une longue ligne où une intervention est nécessaire. Par exemple, *Rabaey* démontre mathématiquement que l'augmentation de la longueur d'une ligne par un facteur deux a comme conséquence de quadrupler le délai de propagation. Ainsi, en réduisant la longueur des connexions, nous réduisons de manière significative le délai qui lui est associé. Une méthode présentée par *Rabaey* propose d'insérer des répéteurs (buffer) à des endroits stratégiques afin de réduire le délai causé par la longueur de la ligne. Cette méthode a été proposée en exemple précédemment.

2.2.2 Biais de synchronisation « skew »

La section précédente a souligné certains effets des longues lignes et des charges parasites. Ces sources de bruit sont également observables sur le signal d'horloge. La propagation de l'horloge à travers de nombreux registres et bascules a pour effet de présenter à l'émetteur une charge importante. Ainsi, en augmentant le nombre de registres, la charge vue par l'émetteur augmente. Ceci a pour conséquence d'augmenter la constante de temps et d'avoir un effet négatif sur la rapidité du système.

Tous les systèmes synchrones doivent respecter des règles de conception afin de s'assurer de leur bon fonctionnement. Une distribution en « H » de l'horloge, présenté par *Eby G. Friedman* [1], combinée avec l'adaptation des charges, peut être utilisée lorsque l'horloge d'un circuit séquentiel est connectée à un grand nombre de nœuds. La figure 2.3 représente un réseau de distribution d'horloge classique.

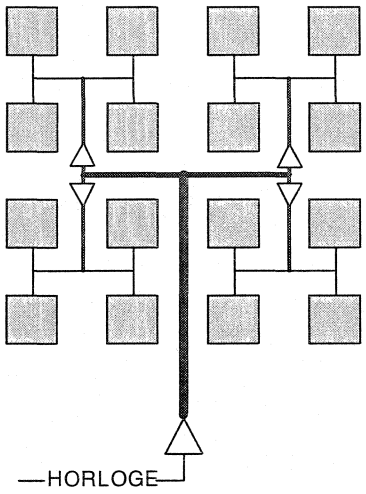

Figure 2.3 : Réseau de distribution de l'horloge en « H » avec adaptation d'impédance.

Lorsqu'il est nécessaire d'éviter les réflexions aux différents points de branchement, les lignes qui se séparent en deux doivent posséder une impédance caractéristique deux fois plus grande que l'impédance de la ligne entrante. Cette situation est illustrée par l'exemple de la figure 2.4 qui peut être modélisé par l'équation 1.

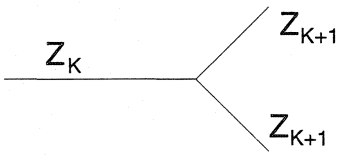

Figure 2.4 : Adaptation d'impédance

$$Z_K = Z_{K+1} \,//\, Z_{K+1} = \frac{Z_{K+1}}{2} \qquad (1)$$

Ainsi, si l'impédance de Z_K est 2 fois plus petite que Z_{K+1}, on diminue ainsi la réflexion dans les lignes causées par une mauvaise adaptation.

Le signal d'horloge franchit souvent de grandes distances, ce qui engendre l'accumulation de charges parasites. Les connexions qui distribuent ce signal peuvent être modélisées comme une ligne à délai RC que nous avons examinée dans la section 2.2.1. Les composants séquentiels possédant le même signal d'horloge ont tendance à observer la transition de l'horloge à différents moments, en raison des différentes longueurs des lignes requises pour atteindre chacun des registres. Ce phénomène est appelé biais de synchronisation de l'horloge, « *clock skew* ». De manière plus précise, le phénomène s'applique seulement lorsque deux composants séquentiels communiquent entre eux. Ce phénomène ne s'applique pas dans le cas où deux registres échangeraient des données au travers un troisième registre (voir figure 2.5), situé entre les deux étant donné que le registre situé au milieu synchronise la donnée et réduit l'effet du biais de synchronisation. *Eby G. Friedman* [1] explique ce théorème plus en détail. *Wann et Franklin* [19] présentent les causes pouvant générer un biais. Parmi les causes identifiées, il cite la différence de longueur des fils, les délais de propagation différents des amplificateurs et la différence des seuils propres aux composants. L'accumulation de tous ces facteurs peut occasionner un biais plus ou moins significatif.

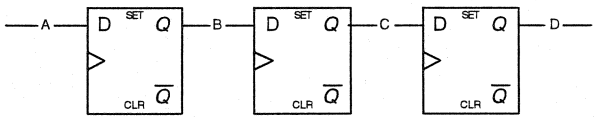

Figure 2.5 : Structure de registres où le biais peut être significatif ou non.

Dans la littérature, on identifie deux types de biais de synchronisation: les biais positifs, voir figure 2.6(a) et les biais négatifs, voir figure 2.6(b). Tout d'abord, une cause qui engendre un biais positif versus un bias négatif est la direction de propagation du signal d'horloge en regard de la propagation des données. Le cas où la direction du signal d'horloge est le même que la trajectoire des données tend à induire un biais positif. Dans le cas contraire, nous tendons à avoir un biais négatif.

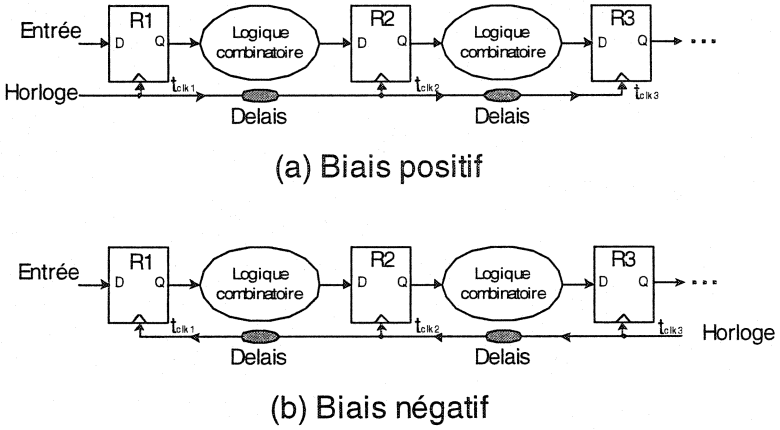

Figure 2.6 : Situations qui tendent à conduire à des biais de synchronisation négatifs et positifs.

Afin de rechercher la période minimale entre deux registres, Friedman [1] a établi l'équation (2). La période minimale de l'horloge entre deux registres, T_{CP}, doit être plus grande que la somme du temps de propagation T_{PD} total entre les deux registres, additionné au biais entre ces deux registres. Le paramètre T_{PD} peut être calculé à l'aide de l'équation (3) qui est la somme du temps de propagation du registre T_{C-Q}, du temps de propagation dans la logique entre les deux registres, du délai de propagation de la ligne et du temps de stabilisation de la donnée à l'entrée du second registre.

$$\frac{1}{f_{Clk\max}} = T_{CP}(\min) \geq T_{PD} + T_{skew} \qquad (2)$$

où :

$$T_{PD} = T_{C-Q} + T_{Logique} + T_{\text{int}} + T_{setup} \qquad (3)$$

Le biais de synchronisation influence grandement la fréquence maximale pouvant être appliquée entre ces deux registres. Par contre, à l'échelle système, il faut considérer la plus petite fréquence à laquelle on peut cadencer le circuit. Par exemple, pour un bus de données, il peut y avoir des différences de phases entre les registres utilisés sur ce bus. Les phénomènes sont de plus en plus importants lorsque nous augmentons la taille d'un circuit, ce qui produit de longues lignes de communication. L'implantation de mécanismes, afin de vérifier l'effet de ces longues lignes sur le système, est l'un des objectifs de ce mémoire.

2.2.3 Synchronisation

La synchronisation peut être définie comme l'action de cadencer les événements d'un système. Voici des exemples qui peuvent inciter les concepteurs à synchroniser les événements :

- le passage d'un système asynchrone vers un système synchrone;
- l'échange de données entre deux systèmes qui n'ont pas la même fréquence et/ou phase d'horloge;
- l'échange de données entre des systèmes très éloignés qui utilisent la même fréquence d'horloge; dans ce cas, l'utilisation d'une boucle à verrouillage de phase (Phase Locked Loop) permet de réduire le délai associé à une longue ligne.

Le dernier exemple correspond davantage à un système de grande taille comme un système intégré à l'échelle de la tranche (Wafer scale) ou au cas de signaux propagés entre différents composants d'une carte électronique.

La synchronisation doit être effectuée à un bon moment, afin de s'assurer que les données échantillonnées soient stables et valides. L'opération de synchronisation est assurée par les éléments synchrones d'un circuit, à savoir les bascules et les registres. Ces éléments sont communément modélisés comme des échantillonneurs/synchroniseurs (*sample and hold*). L'échantillonneur capture, à un instant donné, la valeur présentée à son entrée afin de la rendre disponible au synchroniseur. Le synchroniseur est un élément bi-stable, et lors de la transition d'un état stable à l'autre, il passe par un état instable. Ainsi, pour la très courte période de temps durant laquelle la transition a lieu, il est probable que la sortie soit dans une zone non définie.

Nous pouvons poser l'hypothèse qu'un synchroniseur, bi-stable, peut être modélisé par une paire d'inverseurs interconnectés, (figure 2.7). Dans cette même figure, l'échantillonneur est représenté par une porte de transmission contrôlée par le signal d'horloge et son inverse. Le comportement de ce bi-stable est caractérisé à la figure 2.8, où nous pouvons voir la tension de sortie de l'échantillonneur en fonction du temps.

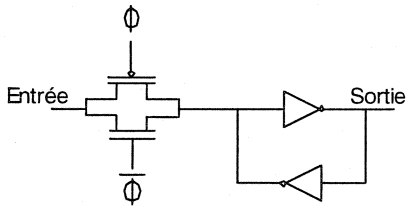

Figure 2.7 : Structure d'un échantillonneur/synchroniseur.

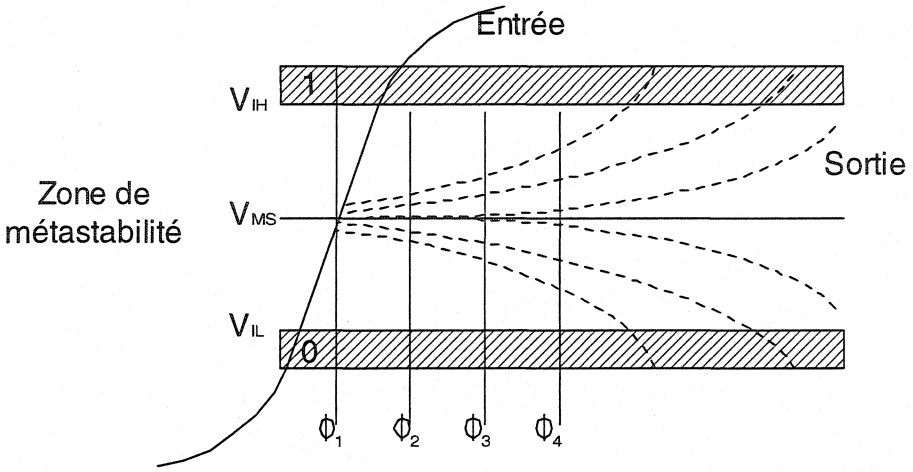

Figure 2.8 : Phénomène de métastabilité

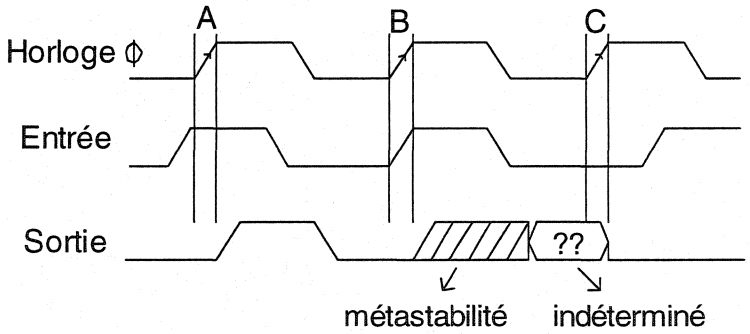

Figure 2.9 : Trois exemples de synchronisation

Nous pouvons observer, à la figure 2.8, que plus le moment d'échantillonnage Φ_i, est retardé, plus le niveau de tension de sortie (la sortie est représentée par des pointillés sur la figure 2.8) s'éloignera de la zone de métastabilité et ce, de façon exponentielle. De cette façon, nous pouvons réduire les erreurs de synchronisation due à la plage de tension indéfinie. Ceci révèle qu'il faut laisser un certain temps à l'entrée du synchroniseur avant de faire l'échantillonnage, afin que la tension d'entrée de celui-ci soit dirigée vers un niveau 1 ou 0 et ainsi réduire le temps de réponse du bi-stable. Alors que sur la figure 2.9, nous pouvons remarquer trois exemples de synchronisation. La première, A, illustre une bonne synchronisation. L'entrée est suffisamment stable avant

d'être échantillonné par la bascule. La seconde exemple, B, illustre un de phénomène de métastabilité. Ce phénomène de métastabilité peut survenir lorsqu'il y a une violation en ce qui concerne la bascule. Par exemple le temps maintient ou de positionnement qui ne sont pas respecté. Ainsi, dans ce cas, la variation du signal d'entrée se fait presque simultanément avec une transition sur le signal d'horloge. Alors, la sortie de la bascule sera momentanément dans un état non-stable (de métastabilité) et ce pour une durée indéterminée. Une donnée instable se caractérise par une valeur se situant entre V_{IH} et V_{IL}. Par la suite, le signal de sortie sera dirigé vers un niveau logique bas ou haut. Cependant, ce niveau est indéterminé. Le troisième exemple, C, illustre une bonne synchronisation. Le niveau logique respecte amplement le temps de maintient de la bascule.

La microélectronique est un domaine qui progresse très rapidement. Quelques éléments clef de la progression sont : la vitesse, la complexité, la fiabilité et le coût. Le projet propose de s'attaquer à l'un de ces éléments clefs, soit la complexité. Elle peut être reflétée sur la réduction à l'échelle ou par l'augmentation de la surface utilisable. L'élément étudié propose une nouvelle méthode de fabrication permettant d'atteindre un niveau d'intégration supérieur aux procédés actuels. Il consiste à réaliser un prototype démontrant la faisabilité d'un ensemble de concept qui permettent de concevoir un circuit intégré à l'échelle de la tranche.

2.2.4 Vue d'ensemble du prototype LAIC

Le prototype réalisé à l'échelle de la tranche est développé dans le but de valider des concepts ainsi que certains mécanismes répondant à diverses problématiques rencontrées lors de l'intégration de systèmes microélectroniques. Ces concepts et mécanismes seront présentés plus en détails dans ce mémoire. Les problématiques traitées ici ont été examinées dans la première partie de ce chapitre. Le projet de recherche réalisé chez

Hyperchip permet de valider 3 aspects essentiels du prototype LAIC. Le premier aspect vise à analyser et tester la technique de jonction inter-réticulaire. Le second aspect porte sur un mécanisme permettant d'être tolérant aux pannes à l'échelle de la tranche. Le dernier aspect concerne l'intégrité des signaux d'un circuit intégré à l'échelle de la tranche. La méthodologie permettant de valider tous ces concepts passe par la réalisation d'une puce, un circuit intégré. Ces trois parties seront implantées sur la même puce. Les 2 premiers aspects font partie de la recherche effectuée par Hyperchip et ne font pas partie de ce mémoire. Cette recherche porte donc essentiellement sur l'intégrité des signaux.

L'objectif général de la partie intégrité des signaux est de caractériser le comportement d'un circuit intégré à grande surface utilisant un procédé d'avant-garde, soit la jonction inter-réticulaire. Cet objectif général peut être fragmenté en objectifs spécifiques propres à la partie intégrité des signaux. Les objectifs spécifiques de cette partie sont les suivants:

- être en mesure de générer de courts délais sur un bus parcourant une longue distance;
- permettre la synchronisation individuelle de chacun des bits d'un bus de données sur différentes phases de l'horloge à la suite d'un long parcours;
- caractériser de manière quantitative le délai accumulé sur un bus de données pour différentes configurations de parcours.

Le segment de la puce qui vise à tester l'intégrité des signaux possède des modules réalisant des fonctions qui permettent de répondre aux objectifs de l'intégration à

l'échelle de la tranche. Une description détaillée de chacun des ces modules sera faite au chapitre suivant.

2.2.5 Technique de conception, augmentation de la surface de conception

Parmi les facteurs déterminant dans la conception d'une puce on retrouve: la miniaturisation des transistors, la fréquence d'opération ou de commutation, la largeur des bus, la puissance dissipée, la densité et la surface efficace. L'interaction de ces facteurs influe sur la qualité des signaux. Par exemple, l'augmentation de la fréquence d'opération est limitée par l'existence des biais de synchronisation.

La réduction à l'échelle permet d'augmenter le nombre de transistors par unité de surface. Toutefois, cette technique possède ses limites. Une des limites associées à la réduction de la taille des transistors provient des courants de fuite entre les drains et sources. Un second facteur qui limite la réduction des dimensions est l'augmentation correspondante de la densité de courant qui a comme conséquence d'augmenter l'électromigration [17].

La technique d'intégration utilisée par ce projet permet de bénéficier d'une superficie utile considérable. Elle permet d'atteindre une superficie disponible 4 fois supérieure aux techniques actuellement employées, pour une technologie donnée. L'augmentation de l'aire des circuits est rendue possible par la jonction inter-réticulaire. Il en sera discuté dans les deux sections suivantes.

2.2.6 Jonctions inter-réticulaires

L'objectif des jonctions inter-réticulaires est de créer des liens métalliques entre les réticules avoisinants. Le réticule normalement utilisé pour produire une image dont l'aire couvre un ou plusieurs dés est alors utilisé pour définir une fraction du dé. La zone normalement réservée aux lignes de coupure (« dicing ») qui permettent de séparer les dés pour l'encapsulation est alors utiliser pour réaliser des lignes de jonction. Ces lignes sont étirées jusqu'à la limite de la zone de coupure pour permettre un chevauchement avec d'autres lignes de jonction lors de la seconde exposition du réticule. Le chevauchement peut être effectué pour les quatre côtés du réticule et ce pour toutes les couches de métallisation de la technologie utilisée (voir figure 2.10). Le nombre d'exposition du réticule dépend du nombre de copies que nous voulons créer. Ainsi, il y aura autant d'expositions que de réticules interconnectés.

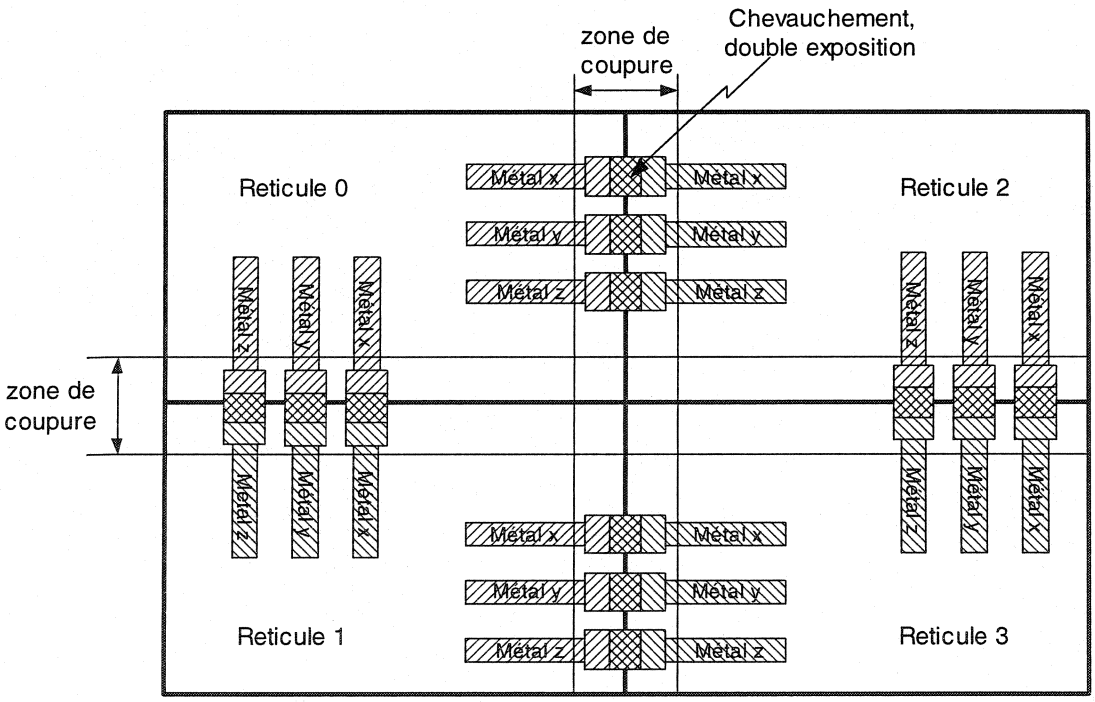

Figure 2.10 : Jonction inter-réticulaire

Dans le cadre de ce projet, quatre expositions seront nécessaires afin de relier quatre réticules d'une même puce. À l'aide de cette technique, il est donc possible d'établir des communications entre des modules de 2 ou plusieurs réticules voisins. Un élément de complexité découlant de cette nouvelle méthode de fabrication est que les réticules doivent être identiques. L'aspect d'orthogonalité des bus d'interconnexions est un enjeu important lors de la conception du réticule, il en sera discuté davantage dans la section suivante qui porte sur la spécification technique et les contraintes du projet. Il est toutefois possible de créer un dé à l'aide de différents masques, en conséquence, différents réticules. Plusieurs applications du domaine de la microélectronique pourraient jouir de cette nouvelle avenue.

Actuellement, la surface maximale d'un dé pouvant être fabriquée est de 15 x 15 mm à 17 x 17mm selon l'usine de fabrication [5]. La limitation de ces dimensions est principalement due à l'outil de fabrication des dés. L'outil de fabrication permet d'exécuter des sauts « stepping » pouvant aller jusqu'à 17mm [6]. Ainsi, la surface pouvant recevoir le circuit, dé, peut aller jusqu'à une surface de 17 x 17mm. À l'aide de la technique de jonction inter-réticulaire, nous pouvons dépasser les limites de l'outil de fabrication et créer des surfaces de conception inégalées, pour une technologie donnée.

2.2.7 Spécifications techniques et contraintes du projet

Le projet LAIC utilise une technologie 0,35 micron à trois couches de métallisations. Tel que mentionné précédemment, la validation de la technique de jonction inter-réticulaire et ses effets sur l'intégrité des signaux de même que la tolérance aux pannes sont les objectifs premier du projet de recherche réalisé chez Hyperchip Inc. Le choix du procédé technologique, CMOS 0,35 micron, découle de cet objectif premier. Par l'utilisation d'une technologie robuste et mature, nous limitons les risques associés au projet.

De manière plus précise, l'aire du dé réalisée dans le cadre du projet est de 1074.7906mm^2 soit de 32.784mm x 32.784mm, ce qui, rappelons-le, correspond à une superficie quatre fois plus grande que les limites normales des techniques de fabrication actuelle [6]. Rappelons également que le composant réalisé se divise en quatre réticules identiques. La grandeur d'un réticule est de 17.5mm x 17.5mm. Les dimensions pouvant accueillir de la circuiterie sont d'approximativement[3] 16mm x 16mm. Nous allons exclure de ce mémoire les détails relatifs à la création des jonctions inter-réticulaires, qui ont été développées par d'autres membres de l'équipe. La figure 2.11 illustre le dé réalisé pour ce projet. Lorsque nous analysons la topologie 2 x 2 réticules identiques, nous pouvons observer les contraintes de conception qui doivent être respectées pour produire les communications inter-réticulaires.

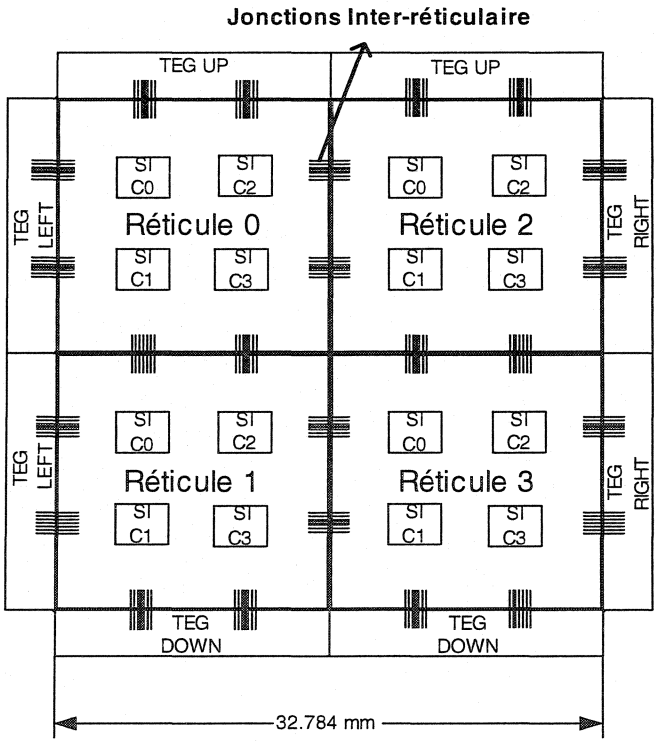

Figure 2.11 : Représentation du dé, 2 x 2.

[3] Les dimensions exactes pouvant accueillir la circuiterie sont de 15.992mm x 15.992mm

Également, la figure 2.11 illustre l'interdépendance de chacun des modules, connexion et terminaison et doivent respecter une grande compatibilité. Un des défis présenté pour la conception de ce projet est que les réticules doivent être identiques en tout point. Il sera question de cet élément dans un chapitre subséquent.

2.3 Résumé

La progression des tailles des transistors, l'augmentation de la surface et la longueur des connexions sont des facteurs qui amplifient l'importance que l'on doit accorder à la qualité des signaux. À l'aide de la technique de jonction inter-réticulaire, un grand pas a été accompli vers les niveaux d'intégration à l'échelle de la tranche (Wafer Scale Integration). Nous pouvons ainsi créer des dés d'une surface jamais réalisée auparavant pour une technologie dont la densité exige l'utilisation de réticules répétés selon un pas régulier. La section suivante présentera les différentes parties de la cellule conçue pour tester l'intégrité des signaux, la cellule SI.

CHAPITRE 3

DESCRIPTION FONCTIONNELLE DU MODULE DE TEST D'INTÉGRITÉ DES SIGNAUX

3.1 Introduction

Comme nous avons observé au chapitre précédent, l'intégrité des signaux (SI) est un élément important dans bien des systèmes microélectroniques. L'objectif premier de ce projet est d'évaluer les éléments qui influencent la qualité du signal. Pour répondre à cet objectif, nous avons proposé et conçu différents modules qui permettront de déterminer, de manière quantitative et qualitative, la dégradation des signaux et d'apporter au besoin certains correctifs.

Le présent chapitre présentera les avantages de l'architecture utilisée. Suivront ensuite les explications relatives aux différents modules qui composent le module de test de l'intégrité des signaux.

3.2 Architecture du circuit intégré

L'architecture du circuit intégré est à la fois une architecture simple et qui répond parfaitement aux objectifs du module de test de l'intégrité des signaux. Tout d'abord, le circuit intégré est formé de quatre réticules identiques regroupant chacun quatre cellules SI. Ceci permet de fabriquer un circuit ayant 16 cellules. Les cellules SI peuvent être

utilisées seules, mais pour extraire toutes les possibilités, elles doivent être utilisées en interaction avec les cellules SI voisines. Chacune des cellules est donc connectée avec ses voisines immédiates (voir figure 3.1).

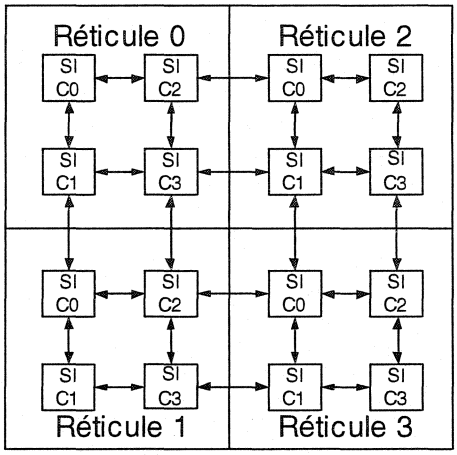

Figure 3.1 : Connexions simplifiées des cellules SI.

En se referant à la figure 3.1, prenons le cas de la cellule C0 du réticule 0. Elle peut être reliée aux cellules C2 et C1 du même réticule. Dans le cas de la cellule C3 du réticule 0, elle peut être reliée aux cellules C2 et C1 du réticule 0, C2 du réticule 1 et C1 du réticule 2. En résumé, chaque cellule peut être reliée à un maximum de 4 cellules voisines. Les cellules près d'un autre réticule peuvent communiquer avec celles d'un réticule voisin. Les branchements entre les réticules utilisent la technique de la jonction inter-réticulaire expliquée au chapitre précédent. Les branchements se font horizontalement ou verticalement, il n'y a aucune liaison entre les cellules situées en diagonal. Celles qui se retrouvent aux extrémités extérieures du circuit, ne peuvent qu'être reliées à 2 ou 3 cellules, selon le cas. Tel que souligné précédemment, une contrainte lors de la conception est que nous devons réaliser quatre réticules identiques. Cette contrainte est dictée par la technique de jonction inter-réticulaire. Ainsi, si deux réticules peuvent être reliés ensemble par des jonction inter-réticulaires, il faut, réciproquement que les entrées

du réticule « A » soient reliées aux sorties du réticule « B ». Il faut alors que le nombre de sorties du réticule « A » soit le même que le nombre d'entrées du réticule « B » et vice versa. Ceci est une conséquence de la reproduction du réticule. Cette restriction s'applique à toutes les cellules d'un réticule, que ce soit pour les signaux horizontaux et verticaux. Pour cette raison, chaque cellule possède le même nombre de bus reliant les réticules entrant que le nombre de bus sortant. Une attention particulière au « routage » a été apportée pour les bus reliant les cellules, afin de respecter la projection orthogonale de ceux-ci.

L'architecture adoptée dans le cadre de ce projet permet l'échange des données sur un large éventail de combinaisons. À l'aide de la topologie de connexion point à point, nous sommes en mesure de créer des chemins de longueurs différentes. Nous pouvons également utiliser diverses fonctions des cellules SI au moment voulu. La connexion point à point nécessite cependant une plus grande surface de « routage » qu'une autre topologie, par exemple, une topologie à bus partagés. Ce besoin d'espace n'est pas une contrainte pour ce projet. Étant donné que nous analysons un circuit à très grande surface, nous pouvons utiliser tout l'espace disponible pour supporter ce type d'architecture. Le dessin de masque de la cellule SI à été réalisé complètement manuellement à l'exception du décodeur (voir en annexe A à la figure A.1). Les sections suivantes décrivent la composition d'une cellule SI (voir le schéma bloc de la cellule SI à la figure 3.2). Le décodeur d'adresse, les interfaces d'entrées et de sorties, le module de départ, le module de délai fin, le synchroniseur et le détecteur de phases et de fréquences seront expliqués l'un à la suite de l'autre dans ce chapitre.

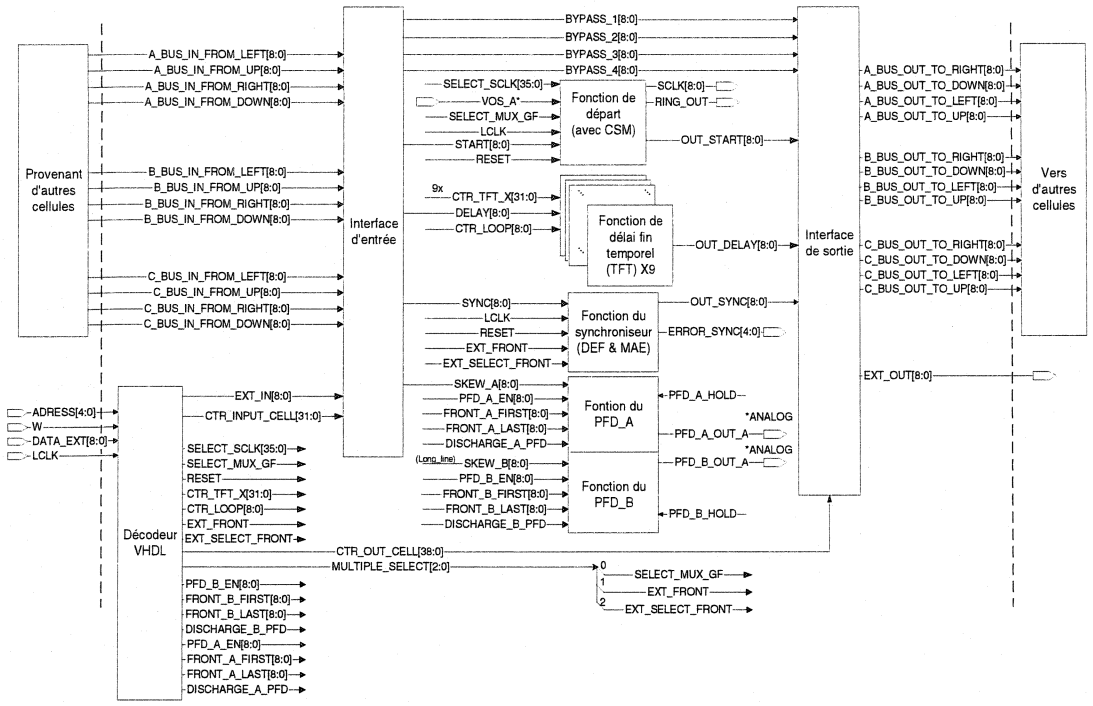

Figure 3.2 : Schéma bloc d'une cellule SI.

3.2.1 Conception du décodeur

Comme nous avons expliqué précédemment, la cellule SI possède plusieurs fonctions et offre plusieurs possibilités de branchement. Il est donc nécessaire de configurer cette dernière de manière à utiliser les fonctions et les interconnections désirées. La conception du décodeur est telle qu'il agit comme interface permettant de sélectionner et de configurer la cellule selon les besoins. Il agit comme interface d'entrée avec le monde extérieur de la cellule SI. Le décodeur, présenté à la figure 3.3, est un élément qui s'est ajouté en cours de conception de la cellule SI. Son implémentation est due, en partie, à une contrainte tardive sur le nombre limité de plots de la puce. Les signaux de sortie du décodeur sont branchés vers l'interface d'entrée de la cellule SI. Seulement les signaux de contrôle propres aux fonctions sont reliés directement aux différents modules. De cette façon, aucun intermédiaire n'est nécessaire à l'approvisionnement des contrôles. Tous les modules possèdent des contrôles provenant du décodeur. Une première ébauche

du décodeur a été effectuée au niveau porte logique. Cependant, pour des raisons de simplicité et de rapidité, l'utilisation du VHDL a été adoptée pour la version finale du décodeur. L'ajout de contrôles pour certains modules, tels que le détecteur de phases et de fréquences (PFD) et une partie du vernier temporel (TFT) ont également été conçus en VHDL. Ces deux modules font partie du décodeur, mais ils constituent deux entités distinctes.

La conception VHDL du contrôleur a été réalisée par un collègue. Le contrôleur, entité du décodeur, regroupe tous les registres. Ceux-ci permettent de contenir l'information pour configurer la cellule SI. Une description de la plage d'adresse des registres se retrouve en annexe B. Le contrôleur a le rôle de gérer la remise à zéro des registres de la cellule. Il alimente également les deux autres entités, soit le contrôleur PFD et le contrôleur TFT. Le premier, le contrôleur PFD, permet de générer des signaux de contrôle pour le module PFD et pour le convertisseur analogique à numérique. Ces signaux permettent la configuration du PFD afin d'en activer les différentes options. Le second, le contrôleur TFT, permet de générer les signaux de contrôle permettant la sélection d'un délai plus ou moins long du module TFT. Également, le module du contrôleur TFT a été réalisé par un collègue. La génération de ces signaux est représentée sur neuf bus de contrôle, soit un bus pour chacun des neuf modules TFT. Une explication propre aux fonctions sera présentée dans des sections ultérieures.

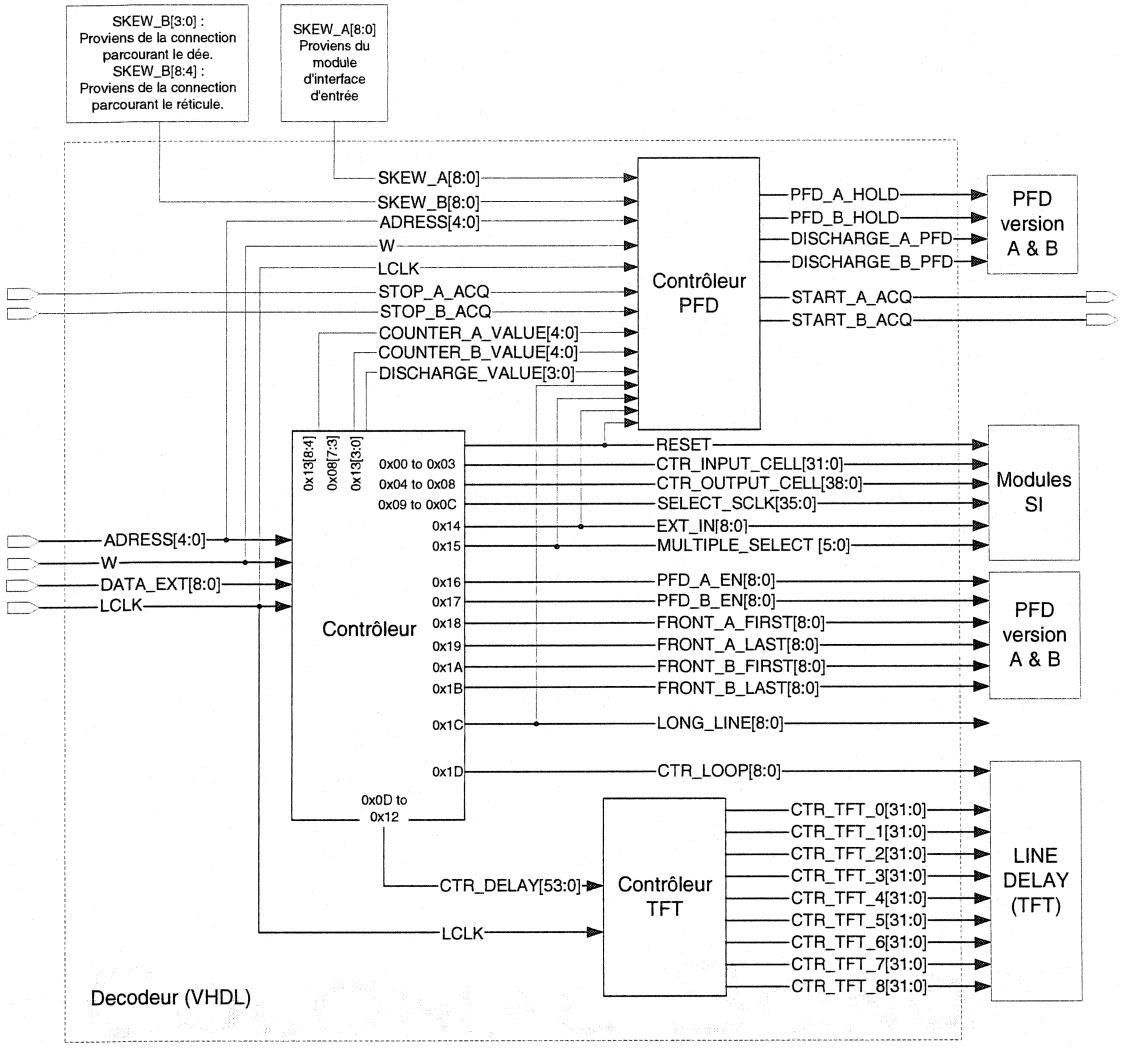

Figure 3.3 : Schéma bloc du décodeur.

Le principe de fonctionnement du décodeur est en fait une utilisation cascadée de deux étages de registres (voir figure 3.4). Ces registres sont utilisés pour configurer les modules de la cellule SI et la configuration des parcours des données à travers la cellule SI. Le premier étage de registre est utilisé pour conserver la donnée. Le second étage de registre permet l'application instantanée de tous les contrôles vers les différentes fonctions de la cellule SI. Tous les signaux de contrôle sont générés par le décodeur via

la configuration effectuée de ces registres. Le décodeur configure donc le chemin choisi, la phase du signal d'horloge du module de départ et toutes les autres configurations nécessaires dans les différents modules.

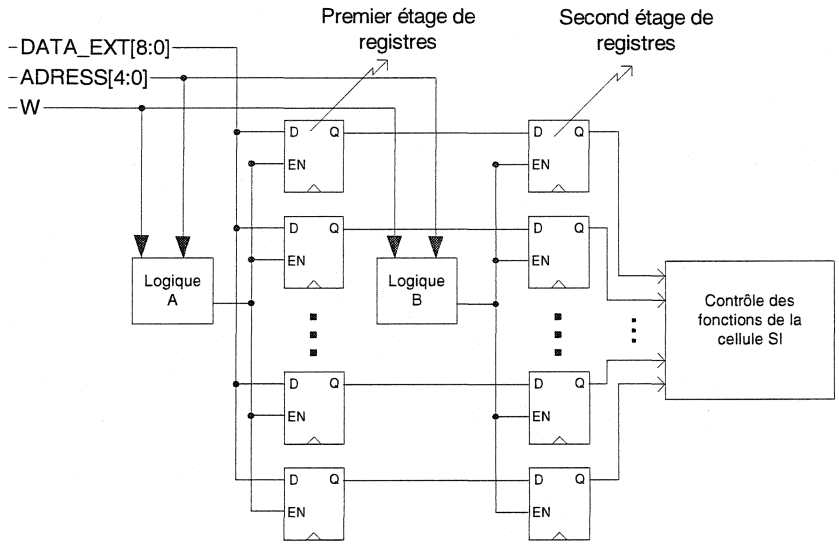

Figure 3.4 : Cascade des registres du décodeur.

3.2.2 Cycle d'écriture du décodeur

Le décodeur peut être interfacé à l'aide d'un FPGA. L'interface utilise un bus de données de 9 bits, un bus d'adresses de 5 bits, une ligne d'écriture et des signaux de contrôle spécifiques aux modules. Une particularité du décodeur est que nous pouvons écrire dans les registres, mais nous ne pouvons pas lire directement leurs contenus. Il aurait été possible de lire le contenu des registres du décodeur en ajoutant un signal de lecture et configurer le bus de données en mode «bidirectionnel ». De cette façon, un troisième état, soit haute impédance, aurait été nécessaire pour récupérer et écrire dans les registres en utilisant le même bus. Cependant, nous voulions garder l'interface de la cellule SI simple. Il est toutefois possible de lire le contenu en utilisant une

fonctionnalité de la cellule. Pour ce faire, les données doivent être dirigées vers le bus de sortie du module interface de sortie.

L'écriture des registres doit se faire en respectant une séquence propre au décodeur. La séquence est la suivante : au départ, il est essentiel d'écrire dans chacun des 32 registres. Une fois que cette étape est effectuée, il est possible de modifier le contenu d'un registre sans avoir à réécrire dans tous les registres. Un chronogramme du cycle d'écriture est présenté à la figure 3.5. Les conditions nécessaires pour effectuer une écriture dans un registre sont les suivantes :

1. Présenter une adresse et une donnée valide sur les bus respectifs.
2. Lorsque le signal d'écriture est à un niveau logique haut, la valeur présente sur le bus de données sera écrite au registre de l'adresse présente sur le bus d'adresses. Le tout sera exécuté lors du prochain front montant de l'horloge.

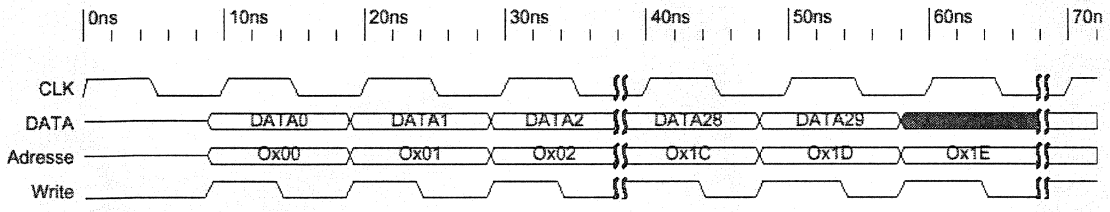

Figure 3.5 : Chronogramme d'écriture.

Une fois que tous les registres de contrôle du premier étage ont été configurés, l'écriture à l'adresse Ox1E permet de faire une écriture globale. L'écriture globale permet le passage des données du second étage des registres vers les fonctions. De cette façon, la

configuration de la cellule SI est réalisée en une seule période d'horloge. La configuration de tous les registres nécessite 32 périodes d'horloge, correspondant aux 32 registres.

Le décodeur est utilisé, entre autres, pour synchroniser les données provenant du monde extérieur vers la cellule SI. Le décodeur sert à la fois à la configuration de la cellule et à faire parvenir les données aux différentes fonctions de la cellule. Afin de satisfaire les deux conditions, un mécanisme doit distinguer s'il s'agit d'une configuration ou d'une transmission de données. Pour ce faire, nous avons utilisé le même principe de la configuration des registres. Lorsqu'il s'agit de faire parvenir des données aux fonctions, le procédé est le même que lorsque nous voulons écrire dans les registres. Une écriture dans un registre dédié, « EXT_IN », à l'adresse Ox14 permet le transfert de la donnée dans le premier étage de registres. Par la suite, une écriture globale permet le passage de la donnée vers la fonction voulue. Cette dernière commande est activée de manière logiciel en écrivant dans le registre à l'adresse Ox1E.

3.2.3 Remise à zéro de la cellule SI

La remise à zéro est essentielle pour un système de communication. Cette remise à zéro permet d'interrompre l'exécution normale des événements et de forcer le système dans un état connu. Il est possible d'interrompre les événements de plusieurs façons. Premièrement, l'application d'une remise à zéro peut se faire de manière matérielle en appliquant un changement sur une ligne dédiée à cet effet. Deuxièmement, il est possible de forcer une remise à zéro de façon logicielle. Par une condition particulière, l'interruption des événements sera activée. Troisièmement, il est possible de combiner une remise à zéro matérielle et logicielle. Dans le cas de notre circuit, nous avons opté pour une remise à zéro logicielle en raison d'une économie de plots.

Le contrôleur inséré dans le décodeur est responsable de la remise à zéro de la cellule SI. Chaque cellule SI est responsable de la remise à zéro de ses fonctions. En d'autres termes, lorsque nous appliquons une remise à zéro sur une cellule SI, les quinze autres cellules de la puce ne sont pas mises à zéro. Il faut obligatoirement exécuter la remise à zéro pour chacune d'elles pour les mettre toutes dans un état connu.

Le principe de remise à zéro se fait de la même façon qu'une écriture dans un registre. Cependant, la nuance de cette écriture est qu'il n'est pas nécessaire de faire une écriture globale pour activer la remise à zéro. L'écriture se fait à l'adresse Ox1F du décodeur. La valeur présente sur le bus de données n'est pas importante. Un signal haut de la ligne d'écriture et la valeur Ox1F sur le bus d'adresse est suffisant. Une fois que cette commande est exécutée, le signal de remise à zéro dirigé vers toutes les bascules de la cellule, est maintenu pour une durée de sept périodes d'horloge. Cette période est nécessaire pour que toute la logique combinatoire ait le temps de revenir à un état connu.

3.2.4 Conception des interfaces d'entrées et sorties

Les interfaces d'entrée et sortie de la cellule SI permettent l'aiguillage des données. L'interface d'entrée accepte les données provenant de deux sources. La première est une autre cellule SI via les multiplexeurs 4:1. La deuxième source est un bus branché au décodeur qui fournit les données provenant du monde extérieur de la puce. Une fois que la source est sélectionnée, les données sont dirigées vers une ou plusieurs fonctions de la cellule grâce aux multiplexeurs 8:1. Les données peuvent toutefois être destinées à une autre cellule SI via les 4 bus de « bypass », sans passer par l'une ou l'autre des fonctions de la cellule SI. La configuration de ces multiplexeurs est faite à l'aide des registres situés aux plages d'adresses Ox00 à Ox03 (voir l'illustration du module d'interface d'entrée à la figure 3.6). Le dessin de masque de l'interface d'entrée à également été réalisé manuellement (voir la figure A.2 de l'annexe A).

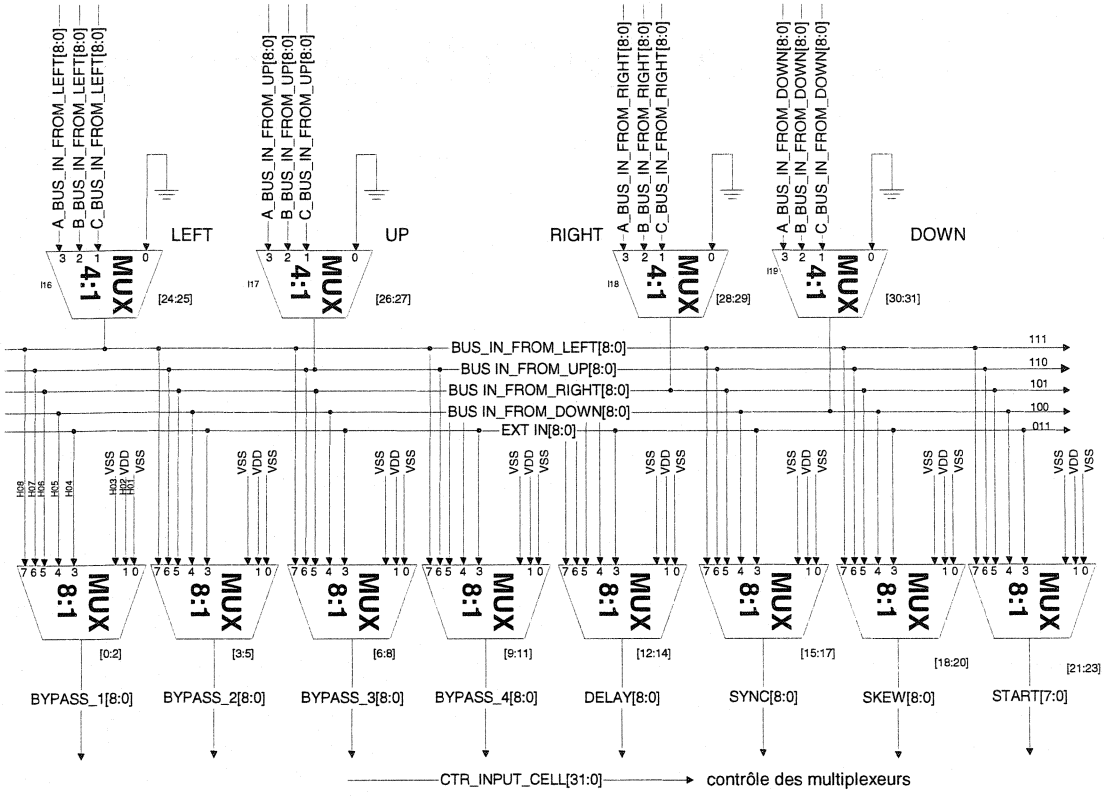

Figure 3.6 : Schéma bloc de l'interface d'entrée.

L'interface de sorties capture les données provenant des différentes fonctions et les redirige vers des cellules SI voisines ou vers le monde extérieur pour fin d'analyse. La configuration de ces multiplexeurs se fait à l'aide des registres situés aux plages d'adresses Ox04 à Ox08 du décodeur. La figure 3.7 représente le schéma bloc de l'interface de sortie. Nous pouvons également observer (voir figure A.3) la réalisation du dessin de masque de l'interface de sortie en annexe A.

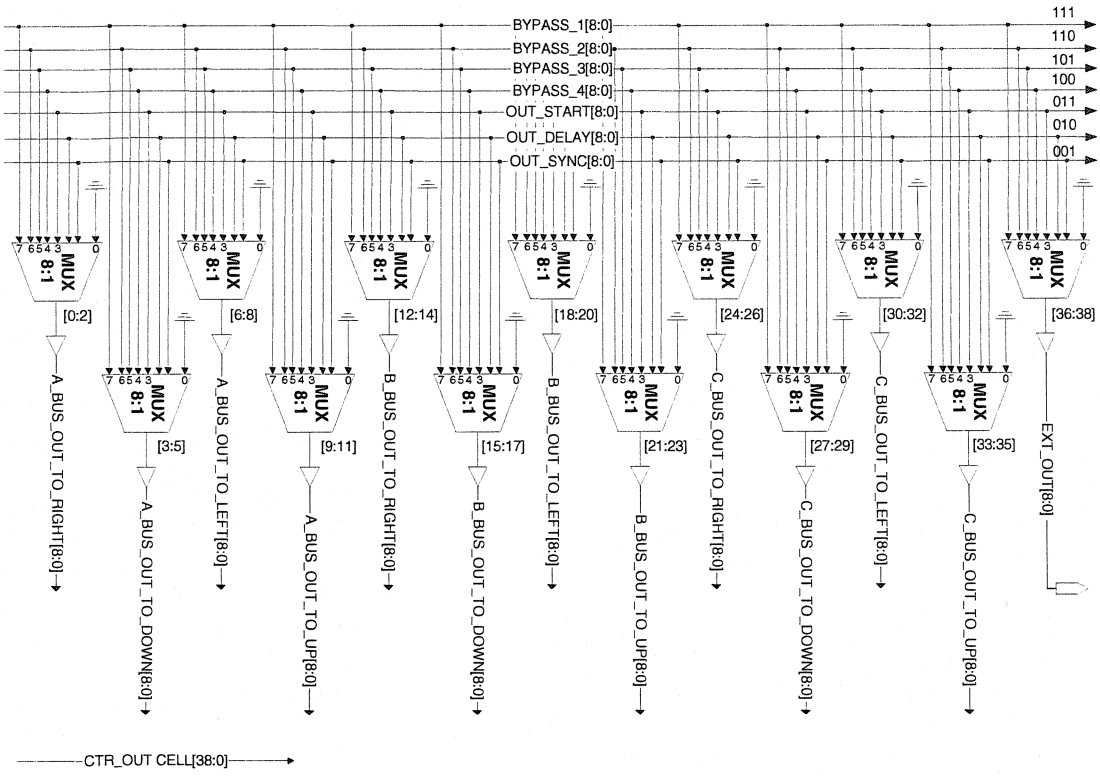

Figure 3.7 : Schéma bloc de l'interface de sortie.

3.2.5 Conception du module de départ

Le module de départ est à la base un module permettant de synchroniser les données. Il est possible, dès le départ, de diriger les données vers le module de départ ou simplement de le contourner. En cours de route, l'option de re-diriger les données provenant d'une autre cellule SI vers ce module est également possible. Nous pouvons introduire le module de départ en deux parties (voir figure 3.8). La première partie contribue à synchroniser les données et la seconde permet de générer les signaux d'horloge qui serviront à cadencer les données. La représentation du dessin de masque manuelle est représenté en annexe A (voir figure A.4).

La première partie du module de départ saisit les données provenant de l'interface d'entrées et les synchronise avec les signaux fournis par le module de contrôle des biais. Chacun des bits de données est synchronisé par une bascule D possédant ses propres lignes d'horloge. Ces lignes d'horloge sont générées par le module de contrôle des biais. Cette partie sera détaillée plus loin. La sortie du module de départ est dirigée vers l'interface de sortie. La sortie de données comporte une largeur de 8 bits qui sera combinée au signal d'horloge formant ainsi le bit le plus significatif, soit le 9^e bit.

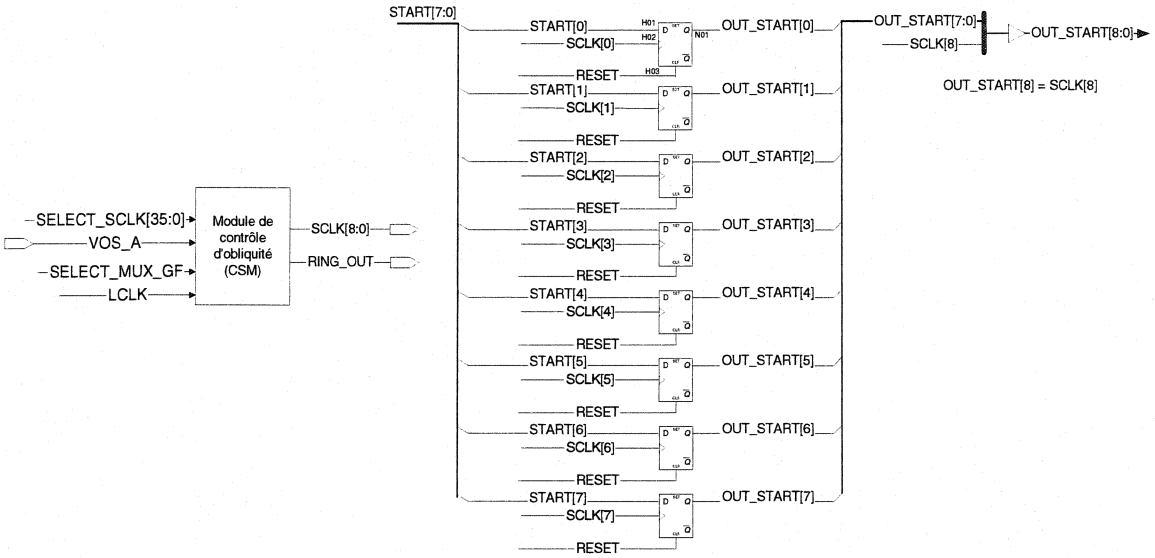

Figure 3.8 : Module de Départ

La seconde partie est le module de contrôle des biais (CSM). À l'aide de ce module, nous sommes en mesure de générer des signaux à différentes phases pour chacun des bits du bus de données. Ces signaux seront utilisés comme horloge pour cadencer les données au module de départ. Ainsi, le CSM est un générateur d'horloge à phases multiples permettant de simuler l'effet du biais de synchronisation sur les données. Le générateur d'horloge provient d'une conception du groupe de recherche chez Hyperchip. Il a été utilisé tel quel pour nos besoins. Il est en mesure de créer une différence de

phases entre deux fronts identiques suivant la somme des temps de propagation d'un inverseur et de deux portes « NON-ET » (voir la figure 3.9). Cette différence de phases peut être ajustée en fonction du délai désiré. La configuration de ces délais se fait à l'aide des registres du décodeur à la plage d'adresse suivante : Ox09 à Ox0C. L'écriture à ces registres permet de configurer 9 multiplexeurs qui choisiront, séparément, parmi 16 signaux d'horloge différents. Les 8 premières horloges générées attaqueront les bascules de synchronisation. La 9^{e} horloge combinera le bus de données situé au bit le plus significatif, tel que spécifié précédemment.

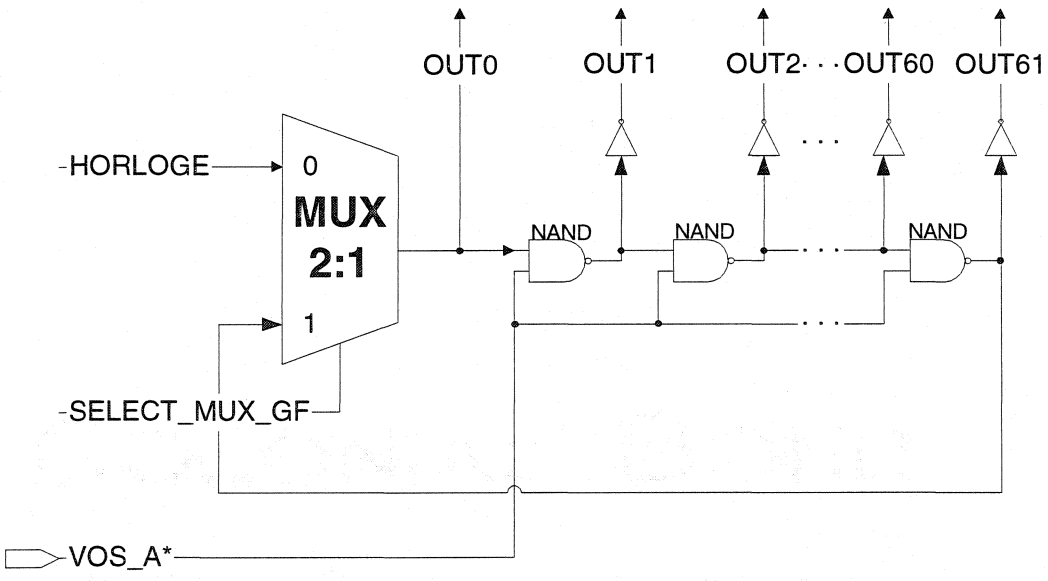

Figure 3.9 : Générateur d'horloge

À titre d'exemple, l'utilisation du signal horloge «out 2 » et «out 60 », permet de générer un délai maximum de 5.31 nanosecondes considérant le temps de propagation de la porte « NON-ET » qui est de 90 picosecondes. Sachant que l'utilisation d'une fréquence de 100MHz est envisagée pour le test de la puce, nous pouvons ainsi créer une différence de phases d'une demi-période entre chaque horloge. Pour respecter la

similarité des fronts, uniquement les sorties paires du CSM sont utilisées pour synchroniser les données. L'utilisation des sorties impaires aurait également pu satisfaire les exigences. La plage dynamique possible entre les phases d'horloge se situe entre 0 et 5.31 nanosecondes. Le pas entre deux phases consécutives est égal au délai au travers de 2 portes « NON-ET ». Puisque nous utilisons uniquement les sorties paires, le délai est théoriquement de (2 * 90=) 180 picosecondes. L'utilisation de l'horloge locale permet de cadencer le générateur d'horloge. Ce circuit peut également être configuré en boucle fermée. De cette façon, la sortie « 61 », avant l'inverseur, est dirigée vers l'entrée du multiplexeur. Suite à la configuration, nous pouvons utiliser le circuit en boucle ouverte alimentée par l'horloge locale ou en boucle fermée à l'aide de la sortie « 61 ».

En plus d'être programmable, un autre élément permet la variation de la phase des signaux d'horloge. Une entrée analogique permet de raffiner la différence de phases entre les signaux en agissant sur le temps de propagation des portes « NON-ET ». En effet, le temps de propagation de la porte « NON-ET » sera plus rapide lorsque la tension appliquée à l'entrée est de 3,3 volts comparativement à un niveau de tension plus bas, par exemple de 3,0 volts. Modifier le niveau de tension a pour effet de modifier la dynamique du système et la phase des signaux générés. La plage de variation du signal analogique « VOS » se situe entre la tension de seuil de la porte « NON-ET » et 3.3 Volts. À l'aide du module CSM, nous sommes en mesure de générer des différences de phase très petites. La précision que nous pouvons atteindre est le temps de réponse d'une porte logique selon le niveau de tension qui attaque une de ces entrées. Il est important de pouvoir influencer les données du circuit à l'aide de délai fin lorsque nous effectuons des mesures de très grande précision. Le module de départ le permet.

3.2.6 Conception du vernier temporel fin TFT (Temporal Fine Tuning)

Le vernier temporel fin, TFT, est un module de la cellule SI. Il est semblable au module de départ pour ce qui est de son utilisation. Par contre, il est composé uniquement de logique combinatoire. Aucune synchronisation n'est réalisée par ce module. Il peut être employé dès le départ ou à tout moment lors des échanges entre les cellules. Le TFT est reproduit 9 fois dans chaque cellule SI. Ainsi, chaque bit du bus de données possède son propre module TFT.

Avant la description détaillée du module TFT, une notion de propagation des portes logiques doit être clarifiée. Le délai de propagation d'une porte logique est un élément important dans ce module. Le temps de propagation d'une porte logique n'est pas identique pour toutes ses entrées. En effet, pour une porte logique à plusieurs entrées, le temps nécessaire pour provoquer un changement à la sortie diffère d'une entrée à une autre. L'écart de temps de propagation entre l'entrée la plus rapide et la plus lente est directement rattaché aux nombres d'entrées de cette porte logique.

Prenons le cas d'une porte « NON-ET » à 3 entrées utilisée dans le module TFT pour illustrer l'énoncé précédent (voir la figure 3.10). Lorsque toutes les entrées de la porte sont à un niveau logique « 1 », excepté une entrée, le temps de propagation, pour modifier l'état de sortie de « 1 » à « 0 », sera plus long si c'est l'entrée « C » qui provoque le changement que si c'est l'entrée « A ». Ceci est dû au temps de propagation du niveau logique qui doit passer à travers un plus grand nombre de transistors branchés en série pour modifier d'état à la sortie. Étant donné l'utilisation de cellules normalisées dans le cadre de ce projet, chaque entrée respective de toutes les portes logiques possède approximativement le même délai de propagation que les autres portes de même type. Les différences qui subsistent peuvent être occasionnés par la température et/ou par les

variations du procédé de fabrication. Dans le cas présent, nous allons tirer profit de cette «lacune» des portes logiques pour créer des délais très fin.

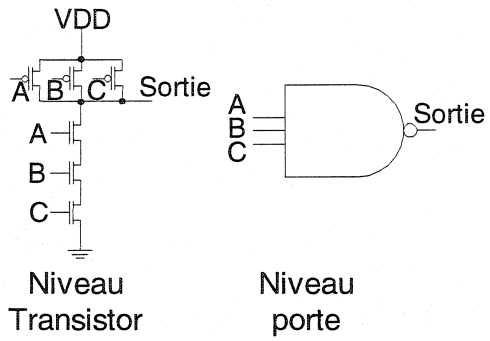

Figure 3.10 : Porte « NON-ET » au niveau transistor et au niveau porte logique

L'objectif de ce module est d'ajouter un délai propre à chaque ligne. Ceci s'accomplit en configurant les portes « NON-ET » de telle sorte que le signal de données soit en mesure de changer d'entrée aux portes suivantes. La sortie d'une porte peut être dirigée vers l'entrée « B » ou vers l'entrée « C ». De cette façon, la propagation du signal de données qui attaque la porte suivante pourrait se faire avec un délai plus grand, selon la configuration choisie. Nous nous sommes basés sur les travaux de *Olivieri et Trifiletti* [14] pour la réalisation du module TFT. Avec cette technique, nous arrivons à créer des délais de l'ordre des picosecondes. La figure 3.11 illustre le module TFT utilisé dans la cellule SI. Comme nous l'avons spécifié auparavant, ce module est reproduit 9 fois pour toutes les cellules SI. Chaque bit de données parcourt un des 9 modules TFT. Nous pouvons retrouver en annexe A le dessin de masque réalisé manuellement des 9 TFT (voir la figure A.5). Alors, tous les bits auront leur propre délai de propagation. Nous pouvons remarquer sur la figure 3.11 que des portes logiques ont été ajoutées au module TFT. Ceci est pour adapter la charge, afin que toutes les portes perçoivent la même capacité d'entrée. Une attention particulière a été apportée au dessin des masques de ce module, afin que la longueur des lignes soit sensiblement la même.

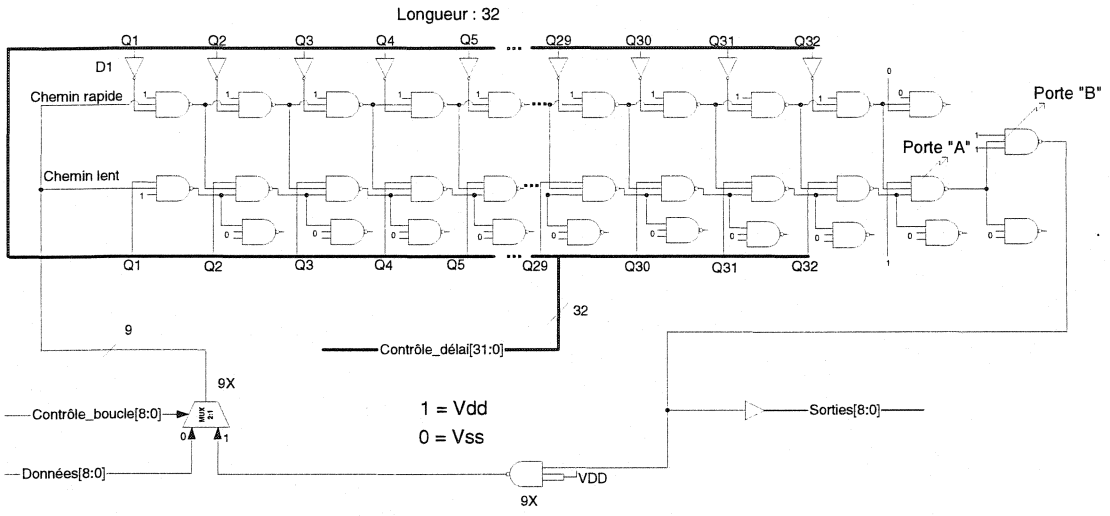

Figure 3.11 : Un module TFT

Le principe de fonctionnement du module TFT peut être interprété comme un circuit qui retarde la transmission de la donnée entre l'entrée et la sortie du module. Ainsi, la donnée est routée soit par un passage rapide ou un passage plus lent. La notion de passage rapide et lent est relative. Le temps de passage sera expliqué plus en détail dans cette section. Le choix du passage est fait selon la programmation du délai voulu. La configuration des registres du décodeur à la plage d'adresse Ox0D à Ox12 fera en sorte de sélectionner le parcours de la donnée lors du passage dans le module TFT. Il faut noter que le signal de donnée traverse 34 portes « NON-ET ». Les deux portes additionnelles localisées à la fin de la série sont placées pour des raisons bien spécifiques. Étant donné que nous choisissons entre deux chemins, l'avant-dernière porte, porte « A », a été ajoutée pour concentrer le signal vers une seule sortie. Pour ce qui est de la dernière porte, porte « B », elle est utilisée afin d'obtenir un nombre pair de portes. De cette façon, le signal a la même polarité à l'entrée qu'à la sortie.

Lorsque que le module est analysé dans son ensemble, c'est-à-dire que chacun des bits du bus de données passe à travers son module TFT, il en résulte plusieurs délais. Ces délais sont soit différents ou semblables selon la configuration de 9 TFT choisie. Ils sont reflétés sur chacun des bits du bus de données. Pour affirmer que les délais seraient semblables, nous devons éliminer de notre analyse tous les délais qui ne sont pas générés par le module TFT. C'est-à-dire qu'il faut éliminer les différences de délais associées aux longueurs de fils, à la température et à tous les autres facteurs hors de notre contrôle. Il est possible de créer 32 chemins différents par module TFT, dont les délais se situent entre celui du chemin rapide et celui du chemin lent. Le chemin parcouru par la donnée dépend des signaux de contrôle imposés. Par exemple, la donnée peut débuter son parcours à partir du chemin rapide et en cours de route traverser vers le chemin lent selon la valeur sur le bus de contrôle.

Le module TFT permet de créer plusieurs délais. Cependant, avant de faire l'analyse des délais qu'il crée, étudions le temps de propagation d'une porte logique « NON-ET ». Selon les spécifications, fournies par le fabriquant, les temps de propagation de la porte « NON-ET » utilisée sont respectivement de 80 picosecondes et 95 picosecondes pour les entrées « B » et « C ». Alors, la différence de temps de propagation entre deux entrées est d'environ 15 picosecondes. Maintenant que nous avons examiné les délais des entrées utilisées de la porte « NON-ET », voyons comment trouver le délai maximum et minimum d'un module TFT. Posons le délai nominal comme étant le délai associé au temps de propagation de la donnée à travers la série de portes « NON-ET » du module TFT. Selon le temps de propagation d'une porte NON-ET », le délai nominal minimum est de 80 picosecondes, multiplié par 34 portes ce qui donne 2.72 nanosecondes. Pour ce qui est du délai nominal maximum, nous devons tenir compte que les deux dernières portes utilisent l'entrée « B » à 80 picosecondes. Alors, ce délai est de 95 picosecondes. multiplié par 32 portes plus 80 picosecondes multiplié par 2 portes, soit 3,2 nanosecondes.

Une fois les délais minimum et maximum du module TFT caractérisés, il est possible de trouver les différences de phases entre les modules. En soustrayant le délai nominal maximum au délai nominal minimum, nous obtenons une différence de délai maximum pouvant être observé entre deux bits du bus de donnée. Cette différence de délai est de 3,2 nanosecondes moins 2.72 nanosecondes, soit 480 picosecondes. Une autre façon de trouver ce délai est de prendre la différence de temps de propagation entre les entrées d'une porte « NON-ET », soit 15 picosecondes et le multiplier avec le nombre de portes, soit 32 portes. Ce qui donne le même résultat soit 480 picosecondes. Le pas de notre système est de 15 picosecondes qui correspond à la différence des temps de propagation entre les deux entrées. De plus, nous pouvons obtenir une plage de différence de délai se situant entre 0 à 480 picosecondes. En d'autres termes, nous avons la possibilité de générer un délai entre chacun des bits du bus de données variant de 0 à 480 picosecondes. Il faut noter que ces valeurs découlent des spécifications du manufacturier.

Sachant les délais pouvant être générés par le module TFT, voyons comment les configurer. L'obtention d'une des valeurs de délais se fait en configurant le module TFT. Le bus de contrôle, CONTRÔLE_DELAI[31:0], provient de l'entité du contrôleur TFT. En fonction de la valeur des registres, associés aux adresses du contrôleur TFT, une nouvelle valeur sera générée par le contrôleur TFT et transmise aux différents modules TFT. La table de vérité (voir annexe C) illustre les valeurs générées sur le bus de contrôle en fonction de la valeur dans le registre. Par exemple, la valeur « 000000 » au registre du TFT permet de configurer le module TFT pour une propagation au travers du chemin le plus lent. Comparativement avec la valeur « 1xxxxx »[4] qui permet de configurer le module TFT pour qu'il propage le signal par le chemin le plus court. Le

[4] Le « x » représente une valeur pouvant être un « 1 » ou un « 0 » logique, sans importance.

module TFT permet la configuration en mode boucle fermée. Ceci permet de générer des valeurs alternant entre 1 et 0 sans appliquer de nouvelles données.

À l'aide de ce module, nous sommes en mesure de générer des délais de l'ordre de 15 picosecondes entre les bits du bus de données. Ceci est fait à l'aide d'un module complètement combinatoire. La génération de délais fins est un grand avantage lorsque nous voulons observer les effets causés sur de longues lignes de communication. Nous pouvons intervenir artificiellement afin d'observer le comportement de celui-ci pour différentes situations. Lorsque les données sont générées par une cellule et traversent un second réticule, il peut arriver des erreurs de synchronisation. Un module permettant de re-synchroniser les données a été introduit dans la cellule SI. Il sera expliqué en détail à la section suivante.

3.2.7 Conception du synchroniseur

Les circuits séquentiels possèdent un mécanisme permettant de cadencer les échanges de données. Ce mécanisme, qui effectue la synchronisation, fait en sorte que tous les événements de transfert se font à un instant bien précis et périodiquement. Pour une cadence donnée, lorsque la somme des délais entre deux systèmes est élevée, la synchronisation peut causer des erreurs dans les échanges de données. Le module de synchronisation utilisé dans la cellule d'intégrité des signaux permet de remédier, en partie, aux erreurs que peut provoquer la synchronisation. Ce circuit initialement proposé par le professeur Claude Thibeault comporte d'infimes modifications afin d'être adapté pour ce projet. La présentation du dessin masque du synchroniseur, réalisé manuellement, se retrouve en annexe A (voir la figure A.6).

Le synchroniseur est en fait un module qui reçoit les données, provenant d'une autre cellule, en les synchronisant avec l'horloge contenue dans les données, RCLK. Une fois

la synchronisation effectuée, une re-synchronisation avec l'horloge locale de la cellule SI est exécutée. Deux choix s'offrent à nous. Le premier est de re-synchroniser les données sur un front montant du signal d'horloge locale. Le deuxième choix est de re-synchroniser les données sur le front descendant du signal d'horloge locale. La méthode utilisée pour la sélection du choix sera expliquée plus tard.

Le principe de fonctionnement du synchroniseur est le suivant. Il utilise deux signaux d'horloge pour effectuer la prise de décision. La première horloge est l'horloge locale de la cellule. Cette horloge est propagée à toutes les bascules de la cellule SI. La seconde horloge provient du bus de données entrant dans le module de synchronisation. Comme nous avons expliqué dans la partie du module de départ, le bit le plus significatif du bus de données contient le signal d'horloge. À cet instant, cette horloge, introduite dans le bus de données, est l'horloge locale de la cellule. Une fois que le bus de données atteint le synchroniseur, dans une autre cellule, cette horloge devient l'horloge reçue, RCLK. Ainsi, ce signal d'horloge aura parcouru le même trajet que les données. Alors le bit le plus significatif, l'horloge reçue, aura accumulée les mêmes délais que les données. Les délais, engendrées par les différents modules et ceux associés au biais de synchronisation, se retrouvent ainsi sur les données et sur le signal d'horloge. Il agit comme «représentant» des signaux de données. Il faut noter que le synchroniseur ne peut être utilisé que si les données ont passé par le module de départ. Seulement le module de départ ajoute l'horloge locale au bus de données. Une analyse de proximité de l'horloge locale et de l'horloge reçue sera effectuée. Le traitement se fait sur ses deux signaux. La conséquence du verdict sera telle que le synchroniseur utilisera le front montant ou le front descendant pour re-synchroniser les données. Par défaut, le synchroniseur utilise le front montant, il faut que toutes les conditions soient satisfaites pour que le synchroniseur utilise le front descendant.

Suite à l'explication générale du synchroniseur, une division des parties de celui-ci sera faite afin de porter une attention particulière à chacune de parties de ce module. Le module de synchronisation se divise en trois parties. La première partie est un module permettant de détecter la proximité du front montant et du front descendant du signal d'horloge locale au signal d'horloge reçue. Ce module est le DEF. La deuxième partie, MAE (machine à états), est celle qui détermine si les signaux doivent être synchronisés sur le front montant ou le front descendant du signal d'horloge. La dernière partie est le synchroniseur en tant que tel qui utilise les directives provenant du MAE pour cadencer les signaux sur le front montant ou le front descendant.

3.2.8 Conception du DEF

Le module DEF est le cœur du synchroniseur. Il se divise en deux parties. La première partie sert à évaluer si le front descendant du signal d'horloge reçu se réalise juste avant la transition du front montant du signal d'horloge locale. Le principe de fonctionnement est que le signal d'horloge reçue, RCLK, agit comme donnée à notre système. Ce signal attaque deux bascules synchronisées par l'horloge locale, LCLK (voir figure 3.12). L'entrée de la première bascule est précédée d'un délai combinatoire formé de 4 amplificateurs. La comparaison effectuée par cette partie du DEF permettra d'affirmer si la transition d'un front montant de l'horloge locale a lieu durant la plage de temps entre le front descendant de l'horloge reçue RCLK et celui de RCLK retardé. La plage de temps est définie par le temps de propagation du signal d'horloge à travers les 4 amplificateurs. Ainsi, la donnée, représentée par l'horloge reçue, doit parcourir ce délai avant d'atteindre la première bascule. La seconde bascule reçoit directement la donnée, l'horloge reçue, sans délai. Les sorties de ces deux bascules sont dirigées vers une porte «ET» possédant une entrée inversée. Le signal dirigé vers l'entrée inversée provient de la bascule ne contenant pas de délai à son entrée.

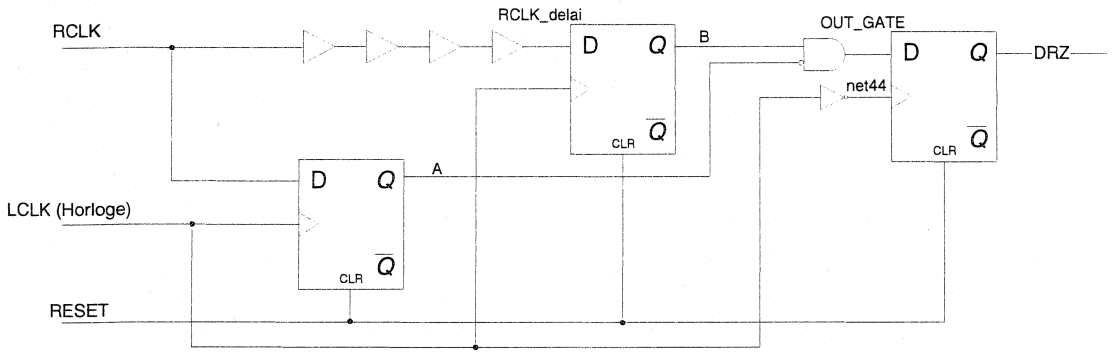

Figure 3.12 : Première partie du DEF, DRZ

L'élément important dans ce module est le délai créé par les 4 amplificateurs. Ce délai d'une durée « t_1» est en fait la plage de temps accordée pour affirmer que les signaux sont près l'un de l'autre. Prenons 5 événements susceptibles de se produire lors de l'utilisation du synchroniseur. Les 5 événements sont présentés dans un chronogramme temporel à la figure 3.13.

1. Le premier événement permet d'illustrer l'effet de la présence de deux fronts montants du signal RCLK et RCLK retardé (RCLK_delai) avant l'arrivée du front montant du signal LCLK. Ceci à comme conséquence d'avoir deux niveau haut à chacune des entrées de la porte « ET » avec l'entrée « A » inversée. Alors les conditions ne sont pas satisfaites pour avoir un niveau logique haut à la sortie de cette porte et d'avoir un niveau bas sur le signal DRZ.

2. Le deuxième événement est pour illustrer l'arrivée de deux fronts descendant du signal RCLK et RCLK retardé (RCLK_delai) avant le front montant du signal LCLK. Cet évènement à pour conséquence d'avoir deux niveau logique bas aux entrées de la porte « ET » avec l'entrée « A » inversée. Ainsi, le signal de sortie DRZ demeure à un niveau logique bas.

3. Le troisième événement illustre l'arrivée du front montant du signal RCLK avant le front montant du signal LCLK et l'arrivé du front montant RCLK retardé après

le front montant du signal LCLK. Semblable aux deux premiers évènement, celui-ci ne satisfait pas les exigences pour l'obtention d'un niveau logique haut à la sortie de la porte « ET » avec l'entrée « A » inversée. Ainsi, la sortie DRZ reste à un niveau logique bas.

4. Le quatrième événement présente l'arrivée du front descendant du signal RCLK avant le front montant de LCLK et l'arrivée du front descendant du signal RCLK retardé après le front montant du signal LCLK. Cet évènement permet d'obtenir au signal « A » un niveau logique bas et à l'entrée « B » un niveau logique haut. Les conditions font en sorte de changer la sortie de la porte « ET » pour un niveau logique haut. De cette façon, au prochain front descendant du signal d'horloge LCLK, la sortie DRZ sera à un niveau logique haut.

5. Le dernier événement illustre deux fronts montant des signaux RCLK et RCLK retardé après le front montant du signal LCLK. Ceci à pour effet d'obtenir deux niveaux logique bas aux entrées de la porte « ET » et ainsi, un niveau bas à la sortie DRZ.

Il faut noter que ce chronogramme a pour but de simplifier l'explication fonctionnelle du circuit. Il ne s'agit pas d'un résultat de simulation ou d'un résultat de mesure.

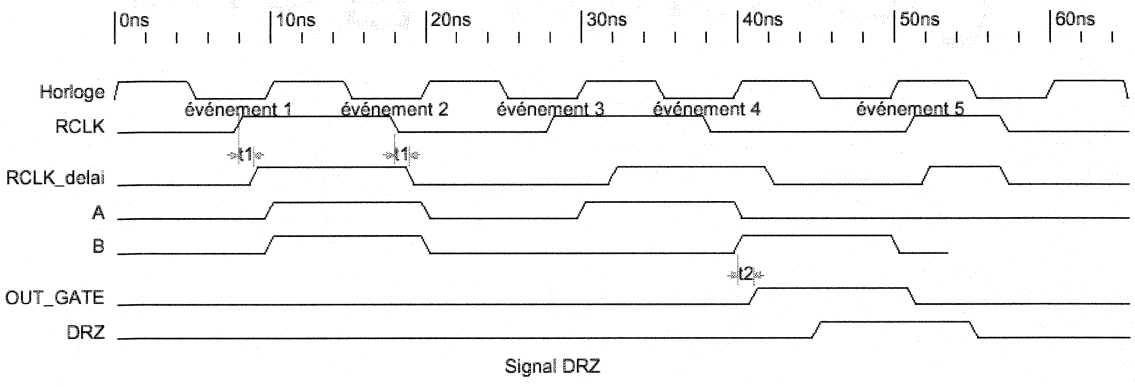

Figure 3.13 : Chronogramme temporel du DRZ

Nous pouvons observer que sur le chronogramme, le quatrième cas illustre la condition nécessaire à détecter un front montant du signal LCLK précédé d'un délai très court du front descendant du signal RCLK. Maintenant que la première partie du DEF a été expliquée, analysons la seconde partie de ce module.

La seconde partie du DEF sert à évaluer si la transition du front descendant du signal d'horloge reçue, RCLK, se réalise juste avant celle du front descendant du signal d'horloge locale, LCLK. Le principe de fonctionnement de la seconde partie servant de détection est très similaire à la première. Le circuit est identique à la première partie à l'exception que les bascules sont synchronisées sur le front descendant (voir figure 3.14).

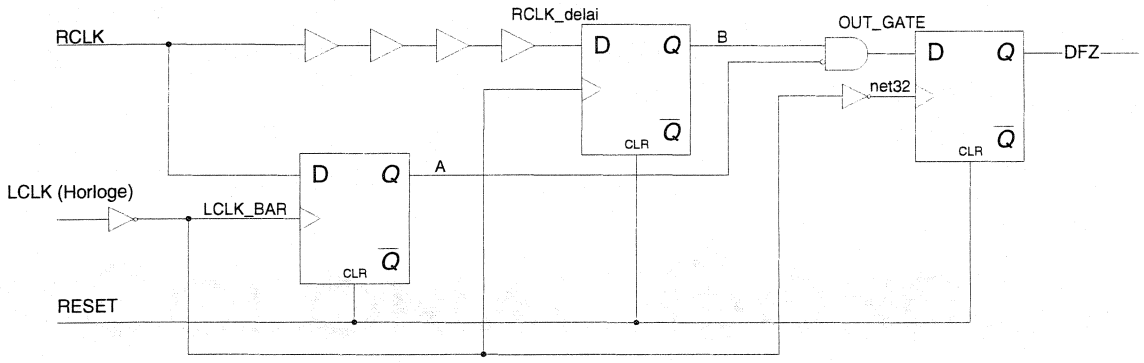

Figure 3.14 : Deuxième partie du DEF, DFZ

Nous allons étudier les 5 événements qui risquent de se retrouver lors de l'utilisation du synchroniseur. Ces 5 événements sont identiques à ceux décrits précédemment (voir chronogramme de la figure 3.15).

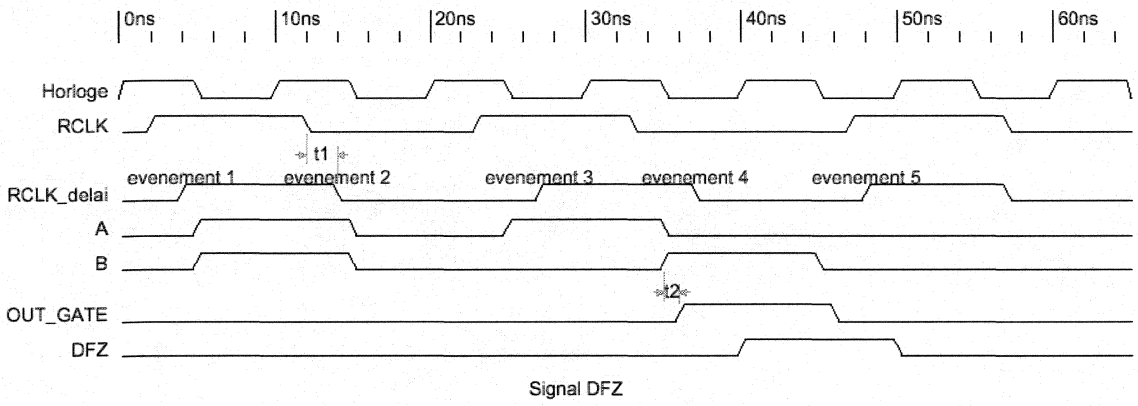

Figure 3.15 : Chronogramme temporel du DFZ

Nous pouvons remarquer que la condition nécessaire à activer la seconde partie du DEF est la même que pour la première partie. Comparativement à la première partie du DEF, cette condition est satisfaite sur un front descendant du signal d'horloge local. Cet événement est représenté au quatrième événement.

Le DEF est divisé en deux parties, tel que présenté. La première partie active le signal DRZ lorsque la transition du front descendant du signal d'horloge reçue, RCLK, survient juste avant celle du front montant du signal d'horloge locale, LCLK. La seconde partie active le signal DFZ lorsque la transition du front descendant du signal d'horloge reçue, RCLK, se manifeste juste avant celle du front descendant du signal d'horloge locale, LCLK. Les signaux DRZ et DFZ sont dirigés vers le module MAE qui agit comme jury pour déterminer lequel des fronts, montant ou descendant, sera utilisé par le synchroniseur.

3.2.9 Conception du MAE

Le schéma du MAE est présenté à la figure 3.16. Lorsque la sortie du MAE est à un niveau « 1 » logique, le synchroniseur utilisera le front descendant de l'horloge locale

pour re-synchroniser les signaux. Dans le cas contraire, lorsque la sotie du MAE est à un niveau logique « 0 » alors le synchroniseur utilisera le front montant pour re-synchroniser les données.

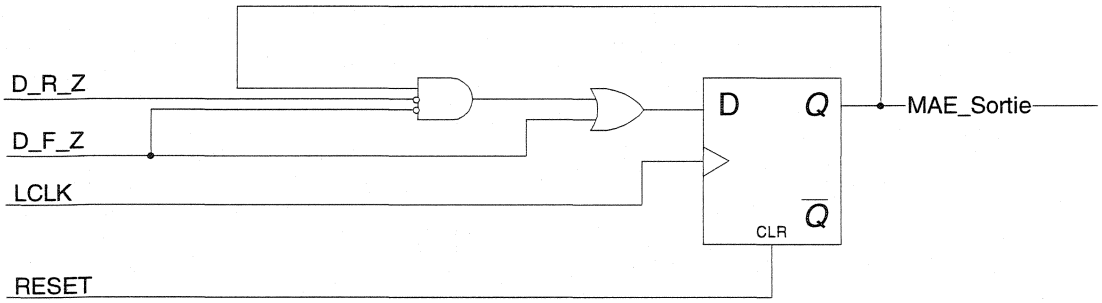

Figure 3.16 : MAE

Les deux conditions qui permettent l'utilisation du front descendant par le synchroniseur sont les suivantes :

1. Il faut que la sortie du DEF du signal DFZ soit à un niveau « 1 » logique. C'est donc dire que le front descendant de l'horloge locale arrive tout de suite après celui de l'horloge reçue.

2. Cette deuxième condition implique trois éléments. Premièrement, il faut que la sortie de DFZ soit à un niveau logique bas. Il faut également que la sortie DRZ soit à un niveau logique bas. Finalement, il faut qu'à l'horloge précédente, nous utilisions le front descendant. C'est–à-dire que la sortie du MAE doit être à un niveau logique haut.

Dans le cas où ces deux événements ne sont pas rencontrés, le MAE commandera la sélection du front montant pour la re-synchronisation des données.

3.2.10 La prise de décision par synchroniseur

Le synchroniseur, présenté à la figure 3.17, effectue la re-synchronisation avec l'horloge locale pour les deux cas possibles, soit sur le front montant ou descendant. Dans ce cas, le module est pret à toute éventualité et il peut réagir plus rapidement. Dans le cas où les résultats du DEF et du MAE concluraient que la re-synchronisation devrait se faire sur l'un ou l'autre des fronts, le synchroniseur aura déjà fait ces deux synchronisations. Toutefois, une seule des deux re-synchronisations sera utilisée à la sortie de ce module, selon la décision du MAE. La sortie du MAE est dirigée vers le contrôle de 8 multiplexeurs 2 à 1. Chacun des bits de données sera re-synchronisé à l'aide du même front d'horloge locale. Cependant, le circuit permet l'intervention manuelle pour sélectionner le front utilisé pour la re-synchronisation. Il est possible via une écriture aux bits 1 et 2 au registre Ox15 du décodeur de forcer la re-synchronisation soit sur un front ou l'autre de l'horloge locale. Cette configuration doit être effectuée avant l'utilisation du synchroniseur.

Les données doivent provenir du module de départ, tel que spécifié précédemment, pour que le bus de données transporte l'horloge RCLK. Lorsque les données entrent dans le synchroniseur, elles sont aussitôt synchronisées avec le front descendant de l'horloge reçue, RCLK. De cette manière, nous aurons la chance d'échantillonner les données au milieu de la période et de minimiser les erreurs de synchronisation dues au chemin différent parcouru par les données et l'horloge. Toutefois, l'objectif n'est pas de synchroniser la donnée avec l'horloge reçue, mais plutôt avec l'horloge locale. Alors, c'est pour cette raison que nous faisons la détection de proximité des deux signaux d'horloge. Nous utilisons l'information fournie par le MAE, à savoir si la transition du

front descendant du signal d'horloge reçu est réalisée avant celle du front montant et descendant de l'horloge locale. Une fois la sélection du front effectuée, les données sont cadencées une autre fois sur le front descendant de l'horloge locale. Ceci permet de réduire les chances d'une synchronisation erronée du module qui recevra les données du synchroniseur.

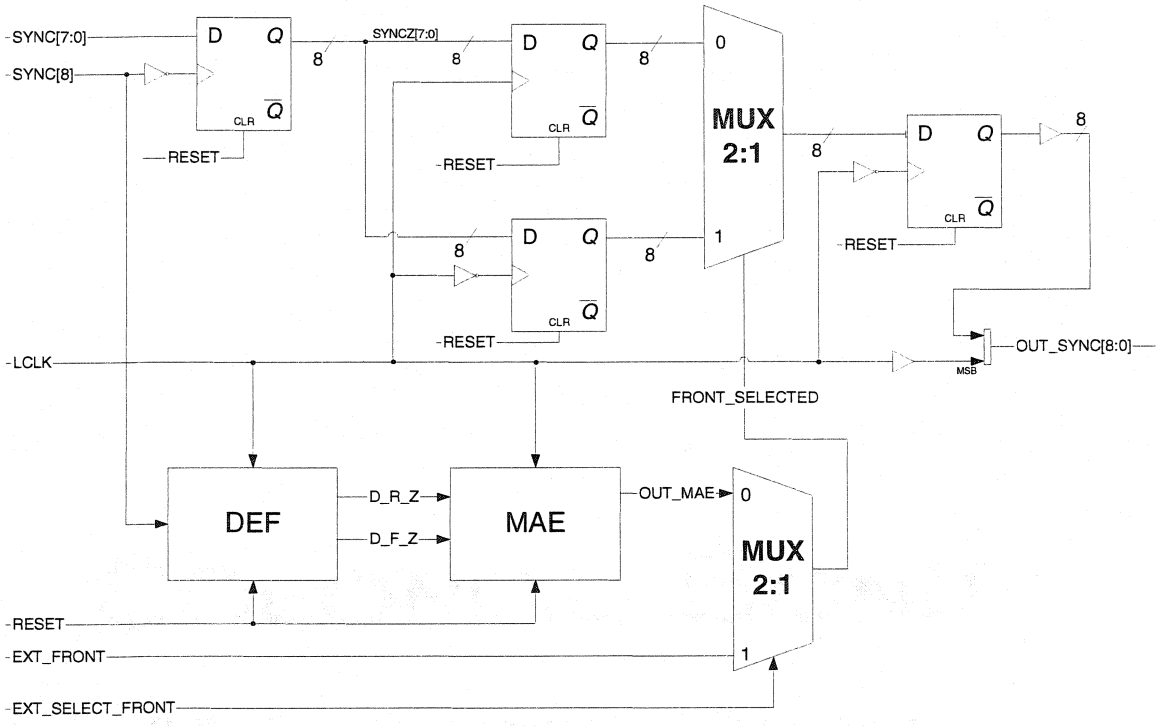

Figure 3.17 : Synchroniseur

Le synchroniseur permet de cadencer les données avec l'horloge locale de la cellule SI. Ceci est utilisé dans le cas où les données seraient propagées d'une cellule vers d'autres. Dans ce cas, la phase du signal d'horloge locale peut différer de celle des données reçues, car le signal d'horloge et les données n'utilisent pas le même parcours. La re-synchronisation est très importante dans ces cas-là. Par contre, il est important de connaître l'étalonnage que nous devons faire au circuit de re-synchronisation afin

d'optimiser l'apport de celui-ci à la cellule SI. La différence de phases et les délais pourront être détectés et quantifiés à l'aide du PFD présenté à la section suivante.

3.2.11 Conception du PFD (*Phase Frequency Detector*)

L'ajout intentionnel de délais permet de reproduire des cas probables pouvant survenir dans un système. Ceci permet de vérifier plusieurs situations pratiques et de confirmer certaines hypothèses. L'objectif du circuit de détection de phases et de fréquences, le PFD, est d'établir des liens quantitatifs au sujet de la différence de phases. Le PFD est un des modules le plus important de la cellule SI, c'est pour cette raison que nous allons porter une attention particulière à celui-ci. Nous pouvons retrouver en annexe A (voir la figure A.7) le dessin de masque du PFD. Le dessin du miroir de courant (partie 3 du PFD) a été réalisé en collaboration avec un collègue d'Hyperchip.

Dans un système intégré, un délai significatif entre deux signaux peut être la cause d'une erreur du système. Une conception à l'échelle de la tranche nécessite des connexions de très grande longueur. Ces longues lignes causeront des accumulations de délais. L'objectif du PFD est de détecter et de quantifier la différence de phases entre les signaux d'un bus de données.

Voici les trois grands points de la spécification du circuit PFD :

1. Le circuit doit être en mesure de détecter et de quantifier une différence de phases entre deux lignes d'un bus de 9 bits.
    a. La détection devra pouvoir être réalisée sur deux transitions hautes des signaux, sur une transition haute et une transition basse de l'un ou l'autre des signaux ou, sur deux transitions basses.
    b. La sélection des transitions, énoncée en « a », devra être appliquée individuellement sur chacun des bits du bus de données.
    c. La possibilité d'activer ou de désactiver individuellement les signaux qui seront comparés. Seuls les signaux sélectionnés seront utilisés lors du calcul de la variation de phases.

2. Le circuit devra être en mesure de détecter une différence de phases entre deux signaux avec une résolution de 500 picosecondes.

3. La sortie du circuit devra fournir une tension proportionnelle à la différence de phases détectée. Cette tension devra :
    a. Accumuler un nombre fini de détections.
    b. Pouvoir être réinitialisé en tout temps.

Pour résumer, les différentes caractéristiques du circuit PFD sont qu'il doit être en mesure de faire la détection sur une sélection de signaux d'un bus de données de 9 bits. Cette détection peut être faite soit : sur deux fronts montants, deux fronts descendants ou un front montant et un descendant de l'un ou l'autre des signaux. La résolution du circuit

devra être de 500 picosecondes. C'est à dire que le détecteur devra être en mesure de fournir l'information quantitative à la sortie lorsque les signaux ont une différence de phases de 500 picosecondes ou plus, voir figure 3.18. Finalement, l'information fournie sera présentée sous forme de tension proportionnelle à la différence de phase.

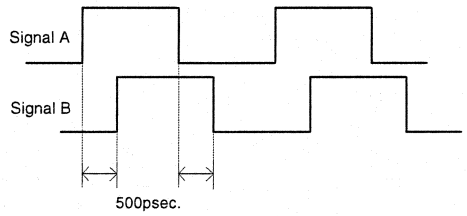

Figure 3.18 : Résolution du PFD

Pour répondre à ces exigences, l'étude de circuits de comparaison et de détection de phases a été effectuée. Plusieurs travaux ont été réalisés sur ce sujet, entre autres [12], [13], [15] et [18]. Par la suite, l'analyse de ce qui a été fait nous a permis d'opter pour un PFD en y ajoutant une partie contrôle. Ceci permet de répondre à certaines exigences précédentes.

Une explication simplifiée du fonctionnement du circuit de détection de phases et de fréquences sera faite avant de procéder aux explications des différentes parties qui composent le PFD. Nous voulons ainsi faciliter la compréhension lorsque nous décrirons chacun des modules se rattachant au détecteur de phases et de fréquences. Lorsque nous envoyons des données à travers un certain nombre de réticules, plusieurs facteurs font en sorte que les transitions des différents signaux s'éloignent les unes des autres. Une fois que les données reviennent au point de départ, le PFD effectue une mesure de la différence de phases sur les signaux du bus de données pré-sélectionnés. Étant donné que chacun des réticules n'a pas le même domaine d'horloge, il est nécessaire de former

une boucle. Ceci a pour effet d'éliminer l'impact des biais sur l'horloge d'un réticule à un autre. La détection se fait sur deux bits provenant d'un bus de 9 bits. Un circuit permet de trier ses bits de façon à faire ressortir le premier et le dernier bit qui aura une transition montante ou descendante, selon la configuration choisie. Alors une fois que le premier bit change d'état, le circuit de détection s'active et une pompe de charge chargera une capacité interne. Cette charge sera faite jusqu'au moment où le dernier bit changera d'état. Lorsque ceci survient, le détecteur de phases et de fréquences se désamorcera et provoquera une ré-initialisation du circuit de détection. La ré-initialisation arrête la pompe de charge et la capacité sera chargée à un potentiel proportionnel à la durée de l'intervalle de temps entre les transitions sur les deux signaux d'entrée. Une explication détaillée sera faite ultérieurement.

3.2.11.1 Caractéristique du contrôleur :

Suite à l'explication simplifiée du détecteur, nous pouvons discuter ses différentes parties. Le contrôle permet de satisfaire plusieurs des objectifs du circuit. Durant la phase de conception, deux prototypes du contrôleur du PFD ont vu le jour, une version en porte logique et une autre en VHDL. Pour des raisons de rapidité d'implémentation, l'option VHDL a été préférée. Cette entité fait partie d'un sous-ensemble du décodeur discuté précédemment. Les fonctions que le contrôleur PFD gèrent sont : l'accumulation de plusieurs acquisitions, la génération des signaux de contrôle associés au PFD et finalement la possibilité de modifier le temps de décharge de la capacité.

Le principe de fonctionnement d'accumulation est assez simple. Étant donné que les données, SKEW[8:0], proviennent du même module, le décodeur, il est simple de savoir si une donnée est envoyée. Pour chaque donnée envoyée, un compteur est incrémenté et lorsque le nombre de données envoyées est identique au registre préalablement

configuré, alors deux signaux s'activent. Le compteur s'incrémente si et seulement si le détecteur de phases est activé et que la donnée envoyée est identique à la donnée reçue. Une copie des données envoyées permettra de faire la comparaison. L'activation du détecteur de phases se fait en écrivant dans un registre à l'adresse Ox08 ou l'adresse Ox13 selon le PFD choisi. Une distinction des PFDs sera faite dans une section suivante de ce chapitre. Il est possible de configurer le contrôleur pour accumuler entre 1 et 32 comparaisons de phases sans décharger le condensateur. Cependant, il se peut que les différences de phases excèdent la capacité du condensateur et que celui-ci se sature. Si une telle saturation est observée, il faut alors changer le contenu du registre « PFD_COUNTER_VALUE » ce qui permet de définir le nombre de cycles d'accumulation.

Lorsque le nombre d'acquisition effectué est le même que celui configuré, deux signaux s'activent. Le premier signal permet d'informer que toutes les détections ont été réalisées avec succès et de signaler la possibilité d'effectuer le traitement. Le traitement sera fait à l'aide d'un convertisseur analogique/numérique situé à l'extérieur de la puce. Le second signal permet de maintenir le circuit de détection à l'état où il est. Ceci est par mesure de précaution, pour ne pas activer le détecteur à un moment non souhaité. Ces signaux seront actifs jusqu'à ce que le traitement soit terminé. Un signal devra être généré, afin d'aviser la fin du traitement. Ce signal sera généré par le convertisseur analogique/numérique. À ce moment, la décharge de la capacité interne pourra être effectuée. Le contrôleur permet également de régler le temps de décharge de la capacité. L'écriture des bits les moins significatifs au registre de l'adresse Ox13 permet de configurer le temps de décharge. Ce registre possède 4 bits. Il est donc possible de faire varier le temps de décharge entre 1 à 16 périodes d'horloge. Cette option a été ajoutée pour répondre au besoin éventuel de l'ajout d'une capacité additionnelle à l'extérieur de la puce.

Le contrôleur permet également de générer les vecteurs permettant de commander l'activation ou la désactivation des signaux qui seront détectés. Il se doit de faire une sélection sur tous les signaux du bus de 9 bits qui seront évalués. De cette façon, il est possible de désactiver des lignes que nous ne voulons pas comparer. Le « masque » de ces lignes est effectué seulement à la réception du PFD pour que tous les effets néfastes à la transmission des signaux puissent affecter le décalage entre les bits lors du parcours. Les registres permettant de réaliser cette commande se situent aux adresses Ox16 et Ox17 selon le PFD choisi. Nous avons également la possibilité de sélectionner individuellement le front qui activera la détection du premier signal et du dernier signal du bus de données.

Le contrôle du détecteur de phases et de fréquences étant expliqué, voyons maintenant en détail le circuit de détection. Le circuit de détection de phases et de fréquences se divise en trois parties. La première permet de faire ressortir le premier et le dernier bit qui se manifestent à l'entrée du PFD, sur le bus de données (1^{re} partie sur la figure 3.19). La seconde partie est le détecteur qui permet d'activer et de désactiver la pompe de charge (2^{e} partie sur la figure 3.19). La troisième partie est la pompe de charge composée d'un miroir de courant qui permet de charger la capacité interne (3^{e} partie sur la figure 3.19). La figure 3.19 illustre chacune des trois parties du PFD. Les références des 3 parties ont également été ajoutées sur le dessin de masque à la figure A.7 situé en annexe A.

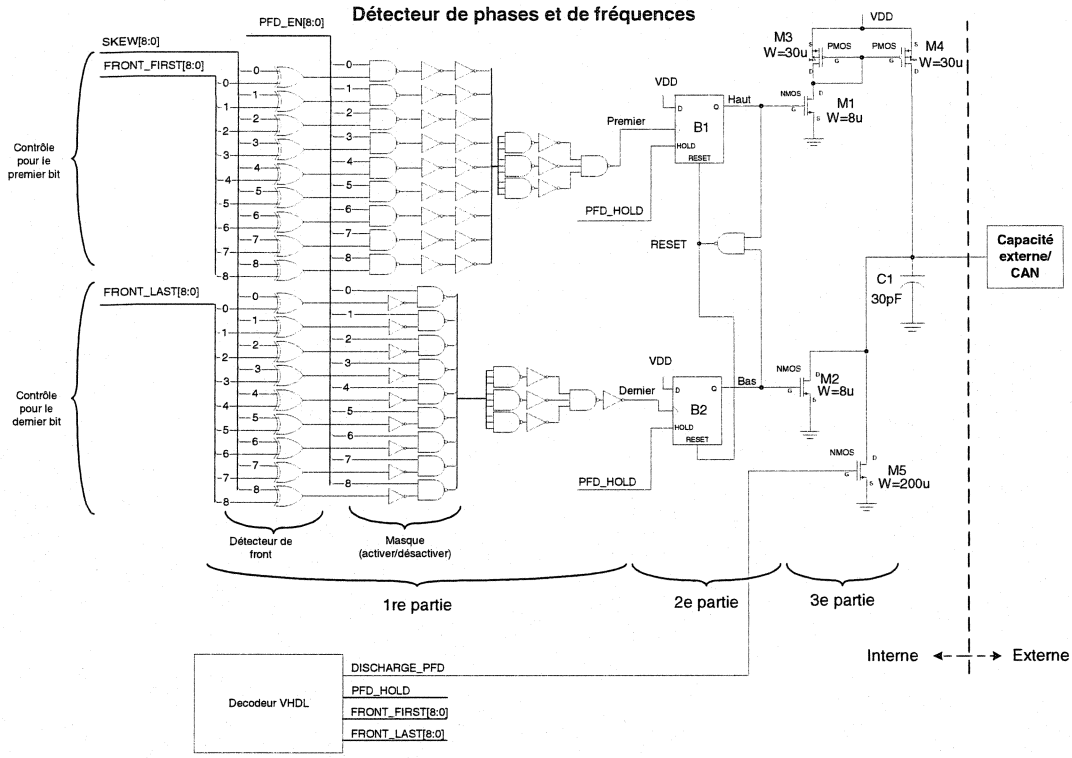

Figure 3.19 : Le détecteur de phases et de fréquences, PFD

La première partie du PFD permet de diviser le bus de données en deux chemins distincts pour le premier et le dernier bit arrivé. Chacun de ces chemins possède son propre bus de contrôle permettant de sélectionner le front montant ou descendant qui activera la détection. Le bus de données et le bus de contrôle sont partagés pour les deux chemins. Le premier chemin est pour sélectionner le bit du bus de données qui change d'état le premier. Le second chemin est pour sélectionner le dernier bit du bus de données qui change d'état.

Au départ, nous avons conçu le circuit à l'aide de portes logiques pour répondre aux exigences, sans porter d'attention particulière aux délais de propagation. Par la suite, nous avons gardé l'idée des portes logiques, mais une fois que le circuit fut fonctionnel,

nous avons fait des manipulations de portes. Nous avons utilisé une combinaison de portes « NON-ET » et d'inverseurs, afin d'obtenir un délai de propagation similaire pour les deux chemins. Nous remarquons que le nombre d'étages d'inverseurs et le nombre de portes « NON-ET » sont identiques pour les deux chemins sur la figure 3.19. Ceci permet d'obtenir un temps de propagation le plus proche possible pour les deux chemins. C'est très important que les deux chemins aient le même temps de propagation pour ne pas fausser les résultats dus à la méthode de détection.

Une fois que les données ont été divisées en deux, la seconde partie, la détection, sert à détecter une variation sur les deux signaux. Ceci permettra de fournir une mesure sur la différence entre la première et la dernière variation. La détection de phases et de fréquences consiste à utiliser deux bascules et une porte «NON-ET ». Les entrées des deux bascules sont reliées au niveau « Vdd ». Les deux signaux provenant des sélections du premier et du dernier bit sont respectivement branchés aux entrées d'horloge des deux bascules. La porte « NON-ET » sert pour la remise à zéro de la fonction de détection. En raison de la logique de contrôle qui dissocie le premier et le dernier bit arrivé, ce sera toujours la bascule « B1 » qui sera activée en premier. Aussitôt que l'entrée « Premier » devient active, par une transition haute, le détecteur envoie un signal d'un niveau logique 1 à la sortie de la bascule « B1 ». Ce signal est relié à l'entrée d'horloge de la bascule. La durée du signal sur la ligne « Haut » varie en fonction du temps d'arrivée du second signal, « Bas ». La seconde bascule, « B2 », sert à détecter que le dernier bit de comparaison est arrivé. Une fois que le deuxième signal survient, la porte « NON-ET » effectue une remise à zéro et remet les deux signaux « Haut » et « Bas » à un niveau logique bas via l'entrée « reset » des bascules. Ce circuit permet donc d'activer et de désactiver un miroir de courant à l'arrivée de ses signaux. Le chronogramme est présenté à la figure 3.20.

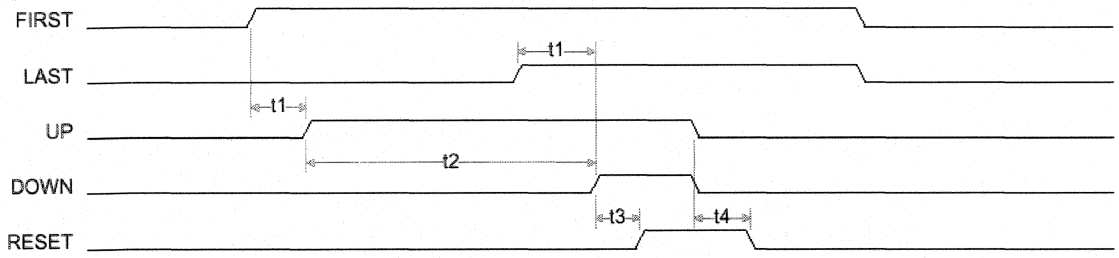

Figure 3.20 : Chronogramme du détecteur de phases

La dernière partie est la pompe de charge. Elle est composée d'un miroir de courant et de deux transistors NMOS. Ces transistors permettent d'activer et de désactiver le miroir de courant. La pompe de charge comprend également une capacité interne de 30 pF et d'un transistor NMOS pour décharger cette capacité. Cette partie analogique a été réalisée manuellement. Elle n'utilise aucune cellule normalisée de la bibliothèque. Lorsque le signal « haut » devient actif, il permet la commutation du transistor M1 et par le fait même d'activer le miroir de courant. Le miroir de courant est formé par les transistors M3 et M4 sur la figure 3.19. Une fois que le miroir de courant est activé, celui-ci produit un courant qui permet de charger le condensateur. Le condensateur se charge jusqu'au moment où le signal « Bas » devient actif. À ce moment, la porte « NON-ET » s'active et ré-initialise les deux bascules à zéro. À ce même instant, le transistor M2 est activé et agit comme un court-circuit, pendant un bref instant, vu par le miroir de courant.

Cette particularité survient lorsque les deux transistors conduisent en même temps. Le temps entre le moment où les deux bascules sont à un niveau logique haut et celui ou elles sont ré-initialisées introduit un temps mort, « dead-zone ». Le temps mort provient de la durée de propagation à travers la porte « NON-ET » additionné au temps de propagation du signal reset de la bascule pour obtenir une sortie à un niveau zéro, Trq. Ce temps, non désiré, introduit une erreur de mesure, car le transistor M1 conduit en

même temps que le transistor M2, ainsi la capacité continue à être chargée. Pour être en mesure de connaître les spécifications reliées au temps de charge, la section suivante explique les détails permettant d'obtenir la valeur de la capacité.

3.2.11.2 Calcul du condensateur

Une des spécifications de la cellule SI est de permettre d'obtenir des informations quantitatives sur la différence de phases entre deux signaux allant de 500 psec à 25 nsec. Ceci se reflète directement sur le temps de charge de la capacité. La tension maximale aux bornes de celle-ci ne doit pas dépasser 3.0 Volts, pour un courant de 3 mA fourni par le miroir de courant. Lors de nos calculs, une marge de sécurité sera insérée pour ne pas saturer le condensateur. À l'aide des équations (4) et (5), nous pouvons caractériser la valeur de la capacité obtenue.

L'équation aux bornes d'un condensateur est donnée par :

$$v(t)=\frac{1}{C}*\int_0^T id\ \tau+v(0) \qquad (4)$$

À l'aide de l'équation (4), nous pouvons trouver la valeur de la capacité en fonction des autres paramètres. L'équation (5) présente le détail du calcul.

$$C=\frac{i*(t-0)+v(0)}{v}=\frac{3\,mA*(25\,n\sec .-0)+0}{3.0\,V}=25\text{pF} \qquad (5)$$

Le condensateur sera réalisé à l'aide des capacités parasites entre les couches de métallisation. Nous allons créer une capacité à plaques parallèles utilisant les trois couches de métallisation. La disposition des plaques parallèles est illustrée à la figure 3.21. Le calcul détaillé de l'aire de chacune des plaques est présenté aux équations (6) à (9). À l'aide de [10] nous avons déterminé les caractéristiques du condensateur à plaque parallèle.

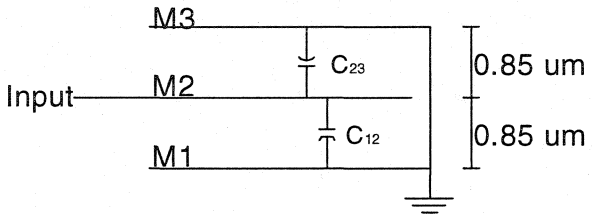

Figure 3.21 : Condensateur à plaques parallèles

$$C = \frac{\varepsilon}{d} * A = \frac{\varepsilon_R \varepsilon_O}{d} * A \qquad (6)$$

où:

C : La capacité entre les deux plaques parallèles;

ε : La permittivité du diélectrique;

ε_O : La permittivité dans le vide = 8.8542 x 10-12 (F/m);

ε_R : La permittivité relative du matériel = 4.2 pour le SiO_2;

A : L'aire des plaques parallèles;

d : La distance entre les deux plaques parallèles.

La capacité par unité de surface obtenue entre le métal 1 et le métal 2 (M1-M2) et entre le métal 2 et le métal 3 (M2-M3) est présentée à l'équation (7):

$$C = \frac{4.2 * 8.8542 \times 10^{-12} (F/m)}{0.85 \times 10^{-6} m} = 43.75 \; aF/\mu m^2 \qquad (7)$$

Pour obtenir une capacité de 25 pF, nous devons fractionner la capacité en deux parties, soit celle entre le métal 1 et le métal 2 (le condensateur M1-M2) et celle entre le métal 2 et le métal 3 (le condensateur M2-M3). Il nous faut une plaque parallèle ayant une surface de

$$Aire\ plaque = \frac{12.5pF}{43.75a\,F/\mu m^2} = 274285.71\mu m^2 \qquad (8)$$

La surface calculée en (8) est la surface totale d'une plaque parallèle. Pour avoir les dimensions de la largeur et de la longueur de cette plaque, il suffit de prendre la racine carrée :

$$\text{Dimension (largeur x longueur)} = \sqrt{274285.71\mu m^2} = 523.72\,\mu\text{m} \times 523.72\,\mu\text{m} \qquad (9)$$

Pour des raisons pratiques, nous avons limité les dimensions du condensateur à 500µm x 500µm. Ceci nous donne une capacité totale de 21.875pF. D'autres facteurs aideront à augmenter la valeur du condensateur. Premièrement, la capacité du plot Cp est branchée en parallèle et elle contribue environ 2.5pF, selon les simulations fournies par le fabriquant des boîtiers. Deuxièmement, la capacité de la trace C_t de la carte de circuit imprimée peut atteindre 1 pF/cm. Selon la carte de test, nous sommes en mesure d'atteindre environ 10 pF. Finalement, l'ajout de la capacité d'entrée C_{in} d'une autre

puce influencera la capacité totale. Selon les spécifications de cette dernière, nous pouvons ajouter 5pF à la capacité totale. La figure 3.22 illustre les différentes capacités vues par le miroir de courant du PFD. Il faut noter que nous n'avons pas tenu compte de la faible contribution des effets de bords.

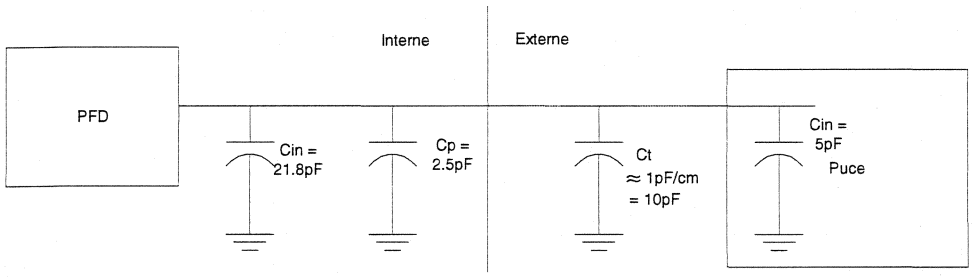

Figure 3.22 : Capacités vues par le miroir de courant du PDF

3.2.11.3 Étapes de ré-initialisation et de mise en fonction

Le détecteur de phases et de fréquences offre plusieurs possibilités. Pour cette raison, il est important de définir les étapes qui permettent de le configurer et de gérer l'échange des données une fois le PFD activé. L'étape de mise en fonction et de ré-initialisation est très importante. Lors de l'application des vecteurs de test, certaines étapes doivent être respectées, afin d'avoir les mesures désirées. Les étapes décrites au diagramme de la figure 3.23 permettent l'envoi des données au module PFD. Au départ, il est absolument nécessaire d'initialiser les bus de configuration aux options voulues avant d'envoyer les données. De ce fait, la désactivation de tous les bits de comparaison doit être effectuée en premier lieu. Par la suite, l'envoi des signaux de commande permettant de sélectionner le front agissant sur la détection du premier et du dernier bit doit être exécuté. Ensuite, nous devons configurer le nombre de répétitions d'acquisition que nous voulons réaliser sans décharger la capacité. Par la suite, c'est le moment de configurer le temps de décharge de la capacité. Ces deux configurations se font à la

même adresse du décodeur, soit l'adresse Ox13. L'étape suivante est la configuration des lignes qui activeront le détecteur. Cette étape se fait individuellement sur chacun des bits du bus de contrôle.

Une fois que ces 4 étapes de configuration sont effectuées avec succès, il est possible d'envoyer les vecteurs de données qui seront analysés. Nous devons faire parvenir le vecteur 0 à la suite du vecteur de données. Cela est dû à la logique combinatoire. Une fois ce vecteur envoyé, une comparaison entre la donnée envoyée avec la donnée reçue doit se faire avec succès. La comparaison se fait avec le vecteur dont le décodeur possède une copie que reçoit le PFD. L'étape suivante est de vérifier le nombre de données reçues avec le nombre de données configurées par le registre Ox13. Dans le cas où le nombre de données reçues par le décodeur ne serait pas le même que le nombre configuré au registre Ox13, le décodeur attendra d'autres données afin d'incrémenter son compteur interne. Avant d'envoyer d'autres données, nous avons l'option de modifier les fronts qui activeront la détection. Dans le cas où nous ne voudrions pas modifier les fronts de détection, nous pouvons procéder directement à l'envoi d'autres vecteurs de données. Cependant, s'il est nécessaire de modifier les fronts de détection, les 4 premières étapes de contrôle décrites précédemment devront être exécutées de nouveau.

Dans le cas contraire, lorsque le nombre de données envoyées correspond à la valeur du registre, le décodeur active le signal avisant le convertisseur analogique/numérique du début d'acquisition. Une fois la conversion terminée, le convertisseur analogique/numérique génère un signal de fin de conversion. Lorsque la cellule SI reçoit ce signal, ceci l'informe que la tension aux bornes de la capacité n'est plus requise et active sa décharge.

À ce stade, nous avons effectué une ou plusieurs accumulations de comparaison de phases. Il s'agit maintenant de savoir si d'autres vecteurs seront envoyés. Une réponse négative mène à la fin de l'utilisation du PFD. Une réponse positive requière un questionnement à savoir si les fronts de détection restent les mêmes. L'utilisation des mêmes fronts de détection aura comme avantage de passer directement à l'envoi de nouveaux vecteurs de données. La modification des fronts de détection nécessitera une reconfiguration des registres utilisés à cette fin. Il résulte de retourner au point de départ.

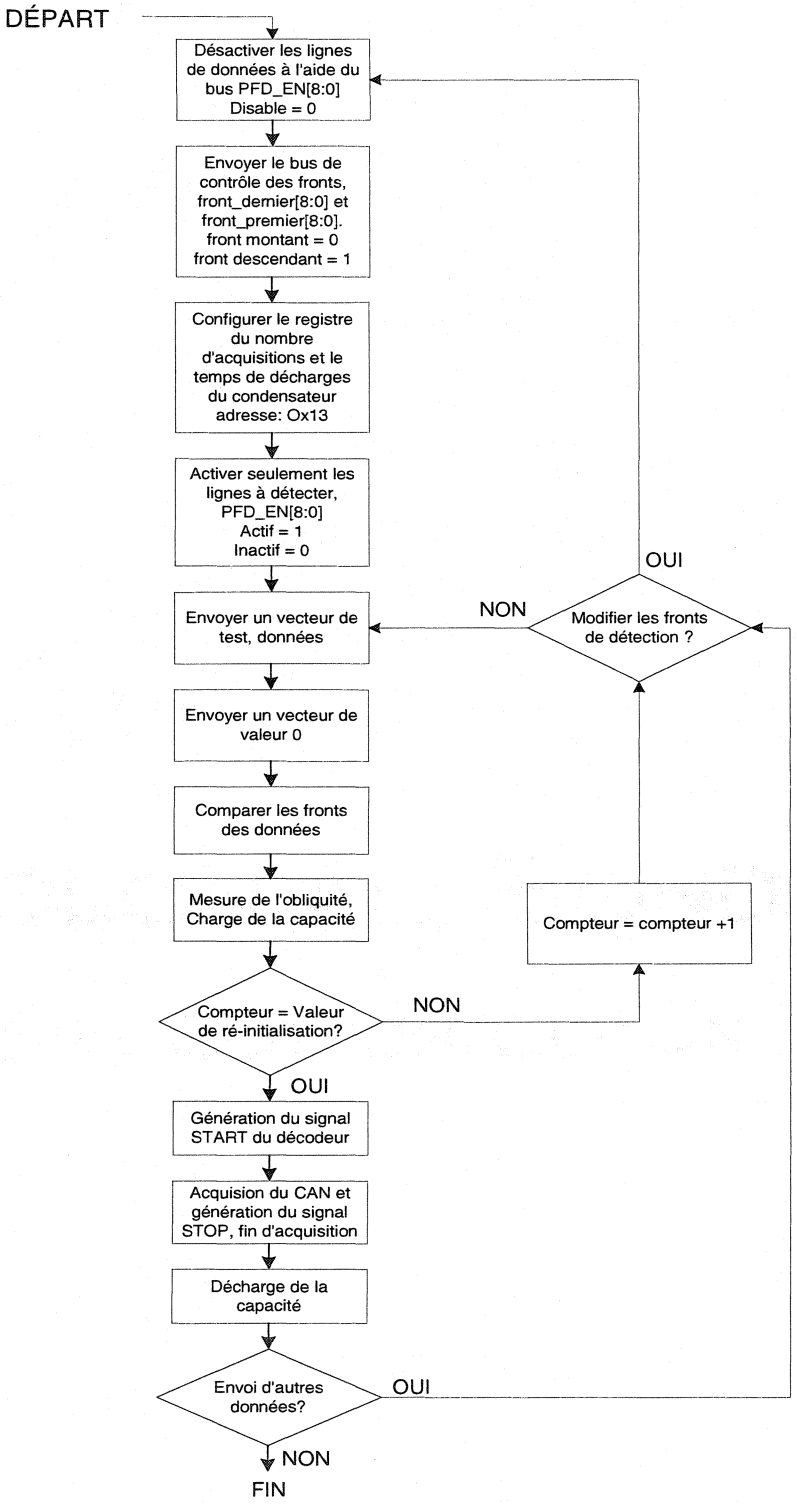

Figure 3.23 : Diagramme d'états du PFD

3.2.12 Particularité de la cellule SI

La cellule d'intégrité des signaux possède plusieurs fonctionnalités. Il a été possible, lors de la phase de conception, d'ajouter des fonctionnalités grâce à son architecture extensible. Nous avons notamment élaboré un branchement entre deux cellules SI dans le boîtier de la puce. L'objectif de ce branchement est d'examiner des solutions pouvant être adoptées dans le cadre de projets futurs.

En utilisant les cellules à l'extrémité du dé, soit les cellules 1 et 3 du réticule 3, cela permet d'optimiser l'utilisation des ressources de la puce. Nous pouvons apercevoir, à la figure 3.24, que tous les branchements de la cellule 1 sont identiques selon une perspective extérieure. La seule différence est que le bus de données de la sortie est partagé entre le monde extérieur et l'entrée de la cellule voisine, la cellule 3. De cette façon, il n'y a pas de bus de données entrant pour la cellule 3 du réticule 3.

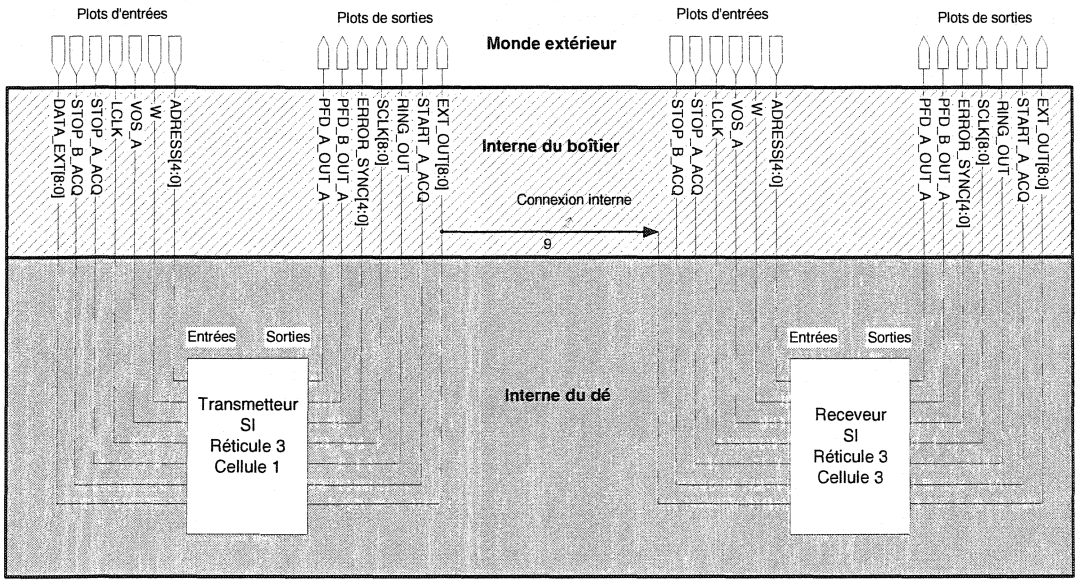

Figure 3.24 : Connexion par le boîtier

Il est simple d'ajouter cette fonctionnalité. Cependant, le branchement par le boîtier engendre une grande complexité d'utilisation de la cellule 3. Comme toutes les cellules SI, cette cellule devra être configurée, afin d'être en mesure d'observer les résultats de ce branchement. La complexité se situe au niveau du passage des données vers la cellule 3. Nous devons nous servir de la sortie de la cellule 1, préalablement configurée comme étant le bus de données d'entrée de la cellule 3. Voici les étapes simplifiées de l'échange des données.

Premièrement, il faut aiguiller les multiplexeurs de sorte que les données soient dirigées directement de l'entrée vers le bus de sortie de la cellule 1. Deuxièmement, il faut écrire à la cellule 1 la donnée que nous souhaitons transmettre à la cellule 3. La cellule 3 obtient alors une donnée valide sur son bus d'entrée. Il faut ensuite écrire à la cellule 3 l'adresse du registre que nous voulons configurer. Pour une autre écriture, il faut utiliser de nouveau la cellule 1 comme intermédiaire. La synchronisation des données et des adresses devra être tenue pour compte en raison du délai engendré par la connexion dans le boîtier. Ce branchement est une particularité de la puce. D'autres particularités sont présentes dans le module d'intégrité des signaux. Par exemple, nous aborderons à la section suivante la distinction entre les deux PFDs.

3.2.12.1 Distinction entre les PFDs

Le circuit réalisé totalise 16 cellules SI, tel qu'expliqué précédemment. Dans ses 16 cellules, un regroupement de 4 cellules sera utilisé pour être répliqué 4 fois. Parmi chaque regroupement de 4 cellules, une des cellules est différente des trois autres. La cellule nommée cellule 0 possède deux versions identiques du PFD, mais pour des utilisations diverses. La première utilisation du PFD est la même que pour les trois autres cellules, les cellules 1, 2 et 3. Il s'agit d'un détecteur de phases et de fréquences

où les données proviennent d'un bus de sortie de l'interface d'entrée de la cellule SI. Les données circulent à travers différentes cellules, fonctions, d'un ou plusieurs réticules et reviennent à la cellule de départ pour être quantifiées par le PFD.

Pour ce qui est du second PFD, un registre dédié à l'adresse Ox1C permet d'envoyer les données sur un bus de 9 bits. Ce bus de données est divisé en 2 parties. La première partie pour les 5 bits les plus significatifs et la dernière partie pour les 4 bits les moins significatifs. L'utilité de cette caractéristique sera discutée plus loin. Ce bus est branché directement à l'entrée du second détecteur de phases et de fréquences. La particularité des 5 bits les plus significatifs est que les signaux ne passent pas à travers la série de multiplexeurs comme c'est le cas pour la première version du PFD. Les données de ce bus entreprennent le parcours à partir de la cellule 0 et contournent, de l'extérieur, les cellules 1,2 et 3 et reviennent à son point de départ. Elles font donc le tour des 4 cellules sans pénétrer à l'intérieur de modules logiques qui retardent les données. Les seuls éléments de délai pour ces signaux sont les tampons. Cette configuration se retrouve dans chacun des réticules.

Une seconde particularité du deuxième PFD est que le bus de 9 bits est divisé en deux, tel qu'énoncé précédemment. L'objectif de cette division est d'avoir la faculté de créer un chemin parcourant la périphérie du dé. En fait, les quatre bits les moins significatifs du bus sont utilisés pour parcourir l'extrémité du dé (voir figure 3.25). Cette particularité se retrouve pour les 4 réticules. Cependant, uniquement la cellule 0 du réticule 0 peut utiliser cette configuration. Pour ce qui est des autres réticules, soit que les données sont terminées ou tout simplement laissées flottantes. Les données quittent la cellule 0 du réticule 0 et parcourent une topologie permettant de faire le tour de la puce et de revenir à la cellule de départ, soit la cellule identifier SI C0 du réticule 0. Afin de réaliser cette architecture, l'utilisation des zones de terminaison a été exploitée dans le but de créer des boucles de retour pour les données. À l'aide de cette configuration, nous sommes en

mesure de concevoir un chemin d'une longueur d'environ 130 mm. Comme nous avons expliqué à la section 2.2.7 la largeur du dé est d'un peu plus de 32.7 mm. Ainsi, lorsque les données débutent à la cellule SI C0 du réticule 0, elles utilisent le premier TEG[5] de gauche pour se diriger vers le TEG de haut. Une fois sorties du TEG de haut, les données entrent dans le second réticule (réticule 2). Il faut noter que les données ne font qu'emprunter un chemin dédié et n'entrent dans aucun autre module des réticules 2, 3 et 4. Ensuite les données passent par le TEG de droite pour ensuite se diriger vers le troisième réticule (réticule 3). En suivant le parcourt, les données passent par le TEG du bas pour se diriger par la suite vers le quatrième réticule (réticule 1). Ensuite, les données empruntent le TEG de gauche pour se diriger vers le réticule de départ (réticule 0). Finalement, les données arrivent à la cellule 0 du réticule 0 pour être dirigées vers le module PFD. Grâce à ce trajet, nous pouvons obtenir un chemin d'une longueur de 4 x 32.7 mm, soit un peu plus de 130 mm. Un nombre pair d'inverseurs intervient dans le parcours des données, agissant comme tampon. Le défi de cette configuration est que les réticules 0, 1, 2 et 3 soient complètement identiques. Alors, il faut être en mesure de créer le chemin permettant de parcourir le tour de la puce. Les TEGs pouvant être différents, nous avons pu diriger les signaux aux bons endroits tout autour de la puce.

[5] Les TEGs sont des zones non actives sur le pourtour du réticule utilisés pour réaliser les terminaisons des entrées.

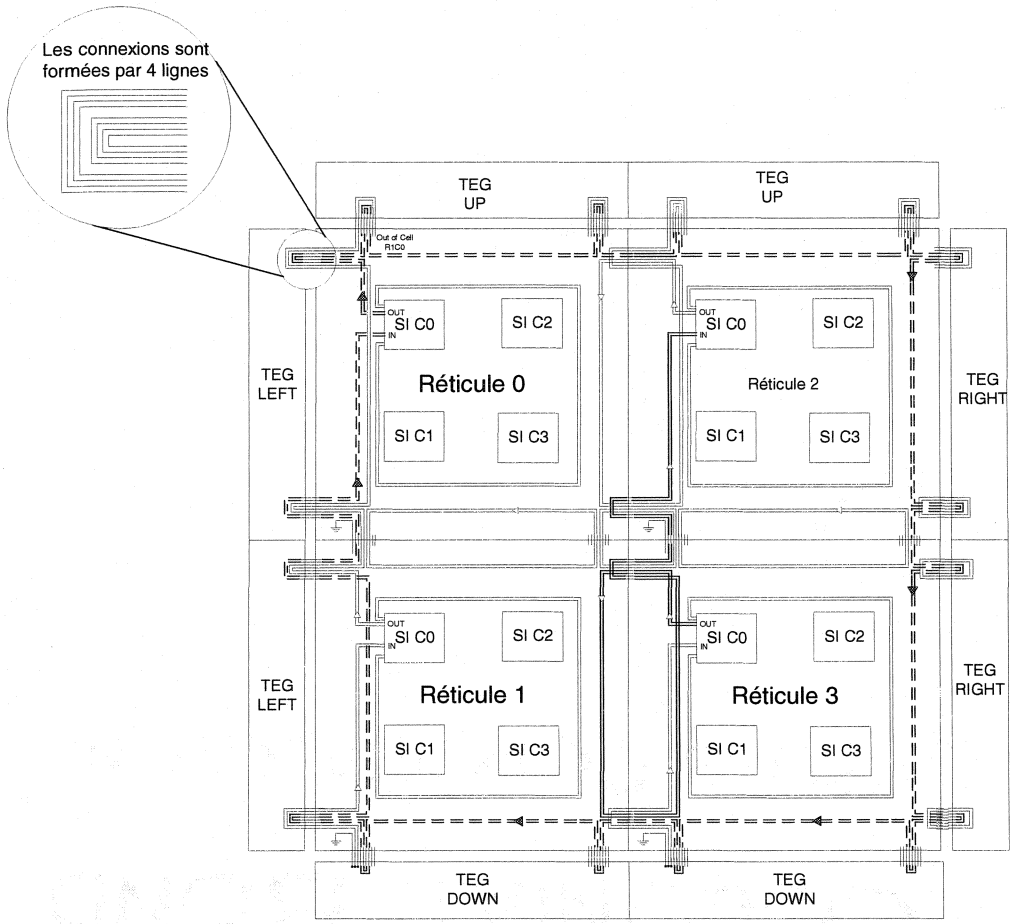

Figure 3.25 : Structure du long chemin pour le second PFD

3.2.13 Conception des topologies réalisées et envisagées.

La cellule de caractérisation de l'intégrité des signaux possède de nombreux branchements de longueurs différentes. Les longueurs de ces branchements varient de 9 mm à 130 mm de long (voir pointillé sur la figure 3.25). Il est nécessaire de varier les méthodes de branchement afin d'observer des influences sur la qualité des signaux telles qu'exposées dans le chapitre 2.

La technique de conception utilisée pour les branchements est l'utilisation d'espacement minimum entre chacune des longues lignes. Les branchements utilisés pour la conception de la cellule SI sont de type topologie b, à tampon rapproché, illustrés à la figure 3.26. Cette topologie contribue fortement à la génération de diaphonie entre les signaux.

Nous avons envisagé l'implantation de la topologie (c) à tampon inversé. Toutefois, la complexité et la probabilité d'erreur très élevée de ces branchements ont fait qu'il n'a pas été réalisé. En effet, la régularité orthogonale des cellules, des branchements, et des connexions inter-réticulaires, demande une attention particulière pour toute déviation de cette règle.

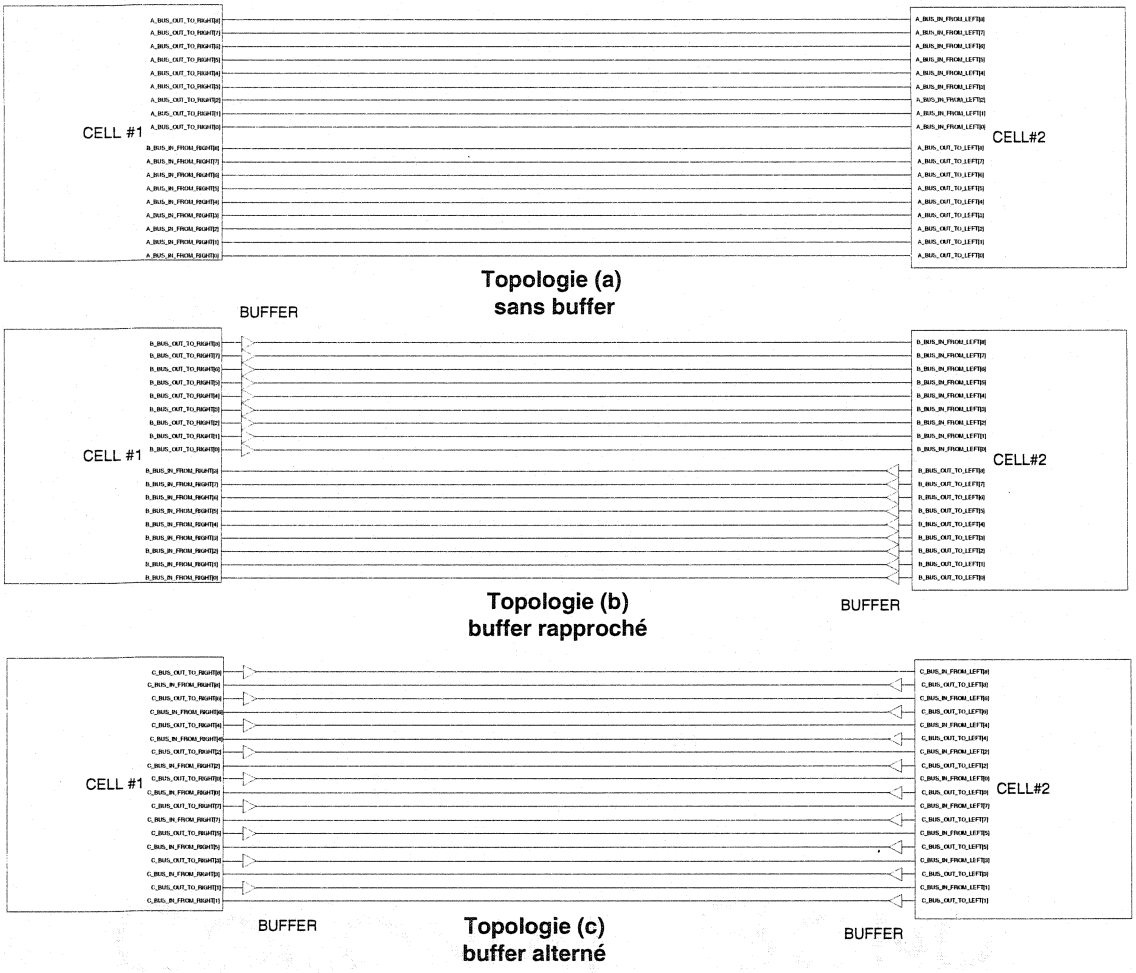

Figure 3.26 : Topologies de branchements

3.3 Résumé

La puce réalisée pour ce projet possède un module pour le test de l'intégrité des signaux. Cette partie se retrouve située stratégiquement sur le dé. Les extrémités ont été exploitées afin de créer de longues lignes. La puce est disposée de façon à recevoir 4 réticules. Chaque réticule comporte 4 cellules SI. Chacune des cellules est connectée de façon à pouvoir échanger des données avec ses voisines. Elles possèdent des interfaces d'entrées et de sorties permettant d'aiguiller les données soit vers l'extérieur ou vers différentes fonctions. Nous retrouvons des fonctions permettant de synchroniser les données, de créer des délais pour introduire des phases contrôlées et un module permettant de détecter et de quantifier les délais d'un bus de données. Ces fonctions devront être simulées et testées afin de valider leur fonctionnalité. Ceci sera l'objet du chapitre suivant.

CHAPITRE 4

VÉRIFICATION DU CIRCUIT PAR DES SIMULATIONS

4.1 Introduction

La vérification d'un circuit électronique se fait généralement par des analyses de la fonctionnalité et surtout par des simulations. Cette partie est essentielle afin de vérifier, à plusieurs niveaux, si le circuit est correct.

Tout d'abord, il faut distinguer les simulations fonctionnelles et les simulations de caractérisation. Chacune de ces simulations possède des objectifs bien spécifiques. Le but de la simulation fonctionnelle ou comportementale est de valider si le circuit répond à toutes les exigences qui ont été établies avant la phase de conception. Ces exigences correspondent aux spécifications du circuit. Cette simulation est réalisée à l'intérieur d'une période de temps assez courte, comparativement à la simulation de caractérisation. Le but des simulations de caractérisation est de pousser le circuit au maximum de ses capacités, afin d'en faire ressortir les limites. Ces simulations demandent plus de temps comparativement aux simulations fonctionnelles.

Le circuit sous simulation (DUS) devra recevoir différents stimuli. Ils peuvent prendre plusieurs formes selon le type de module à tester. Pour ce qui est des circuits numériques, on parle de vecteurs de tests. Ce terme est utilisé afin d'illustrer les différentes lignes qui attaquent parallèlement le circuit sous test. Une séquence propre au

circuit à tester doit être appliquée afin de le vérifier. Cette séquence devra être générée. Ainsi, nous parlerons de génération de vecteurs de vérification.

4.2 Génération de vecteurs de vérification

La génération des vecteurs de vérification peut se faire à l'aide d'un banc d'essai réalisé à l'aide de langages conçus à cet effet. Le langage e [9] est conçu spécifiquement pour la génération de vecteurs de vérification. Il permet de créer des bancs d'essais à un haut niveau d'abstraction. Il est conçu pour travailler avec des variables de types fréquemment utilisés en VHDL tels que les bits, les octets et les vecteurs. L'outil de simulation du langage e permet l'utilisation de générateurs pseudo-aléatoires dont les valeurs générées peuvent tenir compte de contraintes. Il possède également des outils permettant d'obtenir le taux de couverture des tests effectués du circuit sous simulation.

Un second langage, le VHDL, est celui adopté dans le cadre de ce projet. Utilisé en partie lors de la conception du circuit, ce langage a comme avantage d'être simple à utiliser. Cependant, il n'a pas été conçu spécifiquement pour la génération étendue de vecteurs de vérification. Ceci a pour conséquence d'alourdir le fichier qui génère ces vecteurs.

4.3 Environnement de simulation

Le logiciel utilisé pour la simulation de la cellule SI est ModelSim, sous l'environnement Unix. Ce logiciel a été conçu spécifiquement pour la simulation de circuits réalisés en VHDL ou Verilog. Il faut se rappeler que les modules d'intégrité des

signaux ont été réalisés en partie en logiciel (Langage VHDL) et en partie au niveau porte logique (Cadence). Nous étions limités en ce qui concerne les outils de simulation pour la vérification de la partie porte logique. Cela nous a causé plusieurs problèmes lors de la phase de vérification. Nous n'étions pas en mesure d'effectuer les simulations à l'aide de l'outil utilisé lors de la conception au niveau porte logique. La cause de ce problème est que nous ne possédions pas les fichiers de technologie indiquant les paramètres des cellules primaires utilisées. Pour remédier à ce problème, nous avons effectué la conversion de la cellule SI en fichiers Verilog. Toutefois, la conversion a du être effectuée en respectant les paramètres définis par le logiciel de simulation. Plusieurs essais ont été nécessaires afin d'obtenir les résultats souhaités tout en correspondant à la cellule SI. Par conséquent, nous étions en mesure de vérifier la cellule à l'aide de simulations des fichiers Verilog en utilisant le logiciel ModelSim.

La conversion des fichiers en format Verilog a apporté une simplicité en ce qui a trait à la vérification. Il est plus simple d'utiliser le logiciel ModelSim pour effectuer des simulations lorsque le circuit comporte plusieurs entrées/sorties. Ce logiciel permet le regroupement des signaux ayant une analogie entre eux, par exemple un bus de données. Toutefois, nous n'avons pas pu exploiter le logiciel à sa pleine capacité. Ceci est expliqué par le fait que nous n'avions pas le fichier de technologie des cellules. Il faut noter également que nous n'avions pas l'outil permettant de générer les capacités parasites du dessin de masque. Ainsi, les délais de propagation, le temps de positionnement (set-up time) et de temps de maintien (hold-time) de toutes les cellules normalisées ont été fixés à une valeur identique, soit de 1 nanoseconde. Sachant très bien que cela n'est pas le cas, le délai caractérisé par ModelSim donne des résultats qui ne doivent pas être interprétés comme nécessairement mauvais. Une explication plus détaillée sera effectuée pour chacune des parties lorsque le cas se présentera. Les simulations donnent une idée globale de ce qui peut se passer dans le circuit sans apporter de détails précis relatifs aux délais. Nous avons donc conçu notre banc d'essais de sorte qu'il puisse être générique, permettant d'apporter des modifications simples en

vue de la réutilisation des mêmes vecteurs lors du test en laboratoire. Plusieurs paramètres ont été ajoutés à nos fichiers de simulation afin d'être en mesure de modifier les délais de nos vecteurs de vérification dans le but de satisfaire les exigences et de répondre rapidement aux imprévus, toujours dans l'optique de leur réutilisation comme vecteurs de test au laboratoire.

La création des fichiers Verilog a fait en sorte d'inverser les vecteurs de format « Big Endian » à « Little Endian ». Nous ne nous sommes rendus compte de cette représentation qu'après avoir effectué la première simulation. Ceci est expliqué par le fait que l'outil de conversion de Cadence, « Schematic Composer Verilog Interface » utilise cette représentation. Nous avons essayé de formater les résultats en « Big Endian », du plus significatif au moins significatif, mais nos essais n'ont pas été fructueux. Tous nos essais causaient l'inversion de certains bits et résultaient en de mauvaises connexions. Nous nous en sommes aperçus lorsque nous voulions adresser les différents registres. Nous avons donc intentionnellement explosé, bit par bit, les vecteurs importants de chaque simulation, afin d'aider l'interprétation des résultats.

4.4 Simulations individuelles des différents modules de la cellule SI

Le module d'intégrité de signaux, SI, est divisé en 8 parties discutées au chapitre précédent. Chacune de ces parties a été simulée individuellement. Toutes les simulations individuelles sont réalisées en isolant le module à simuler du reste du système. Par la suite, nous avons simulé chaque niveau hiérarchique jusqu'à la simulation du module SI en son entier. Nous avons cru que le fait de diviser les simulations en petits groupes donnerait aux lecteurs une meilleure profondeur de chacun et ainsi favoriserait les analyses lors d'un niveau d'abstraction plus élevé. Il faut noter que les résultats de

simulations présentés dans ce projet ne sont que ceux qui ont d'écoulé d'une série de simulations et d'essais.

4.4.1 Simulations individuelles du module d'entrées et du module de sorties

Les modules d'entrées et de sorties sont des parties formées par des agencements de multiplexeurs. Ces modules permettent l'aiguillage des données en provenance ou en direction de différentes fonctions, selon le cas. En isolant ces deux modules du reste du système, leurs validations se font très aisément. Nous avons appliqué des vecteurs de données à l'entrée de ceux-ci en modifiant le contrôle pour observer leurs comportements. Ainsi, cette simulation a comme objectif de vérifier différentes valeurs du signal de contrôle, CTR_INPUT_CELL, qui affecte l'aiguillage des données aux sorties.

Selon la figure 4.1, les signaux de données d'entrées sont représentés par les bus A_BUS_IN_FROM_LEFT à EXT_IN inclusivement. Le signal de contrôle des multiplexeurs est le bus CTR_INPUT_CELL. Les signaux de sortie des différents multiplexeurs sont les bus BYPASS1 à START inclusivement. Pour la simulation du module d'entrée, nous avons appliqué 12 différentes valeurs au bus de contrôle, CTR_INPUT_CELL. Les différents contrôles permettent l'aiguillage des entrées vers les sorties des multiplexeurs. Le premier contrôle, valeur Ox00000000, sert à appliquer la valeur zéro à toutes les entrées des multiplexeurs via un branchement au Vss de l'entrée 0 des multiplexeurs. Nous pouvons remarquer qu'au départ, toutes les sorties sont à un niveau logique bas. Le deuxième contrôle, valeur OxE00000C0, permet de faire passer les données provenant du bus A_BUS_IN_FROM_LEFT vers la sortie BYPASS1. Le troisième contrôle, valeur OxE00038C0, permet d'aiguiller le bus de données A_BUS_IN_FROM_LEFT vers la sortie BYPASS1 et « SKEW » à la fois. Le quatrième contrôle, valeur OxE0003880, permet de diriger les données provenant du bus

C_BUS_IN_FROM_LEFT vers BYPASS1 et « SKEW » à la fois. Le cinquième contrôle, valeur OxE0003860, permet l'envoi du signal Vss vers BYPASS1, C_BUS_IN_FROM_UP dirigé vers BYPASS2 et B_BUS_IN_FROM_LEFT dirigé vers le bus « SKEW ». C'est de cette façon que nous avons réalisé les tests du module d'entrée. Étant donné que ce module n'est pas très complexe, nous ne nous attarderons pas plus longtemps sur les résultats qui sont présentés à la figure 4.1.

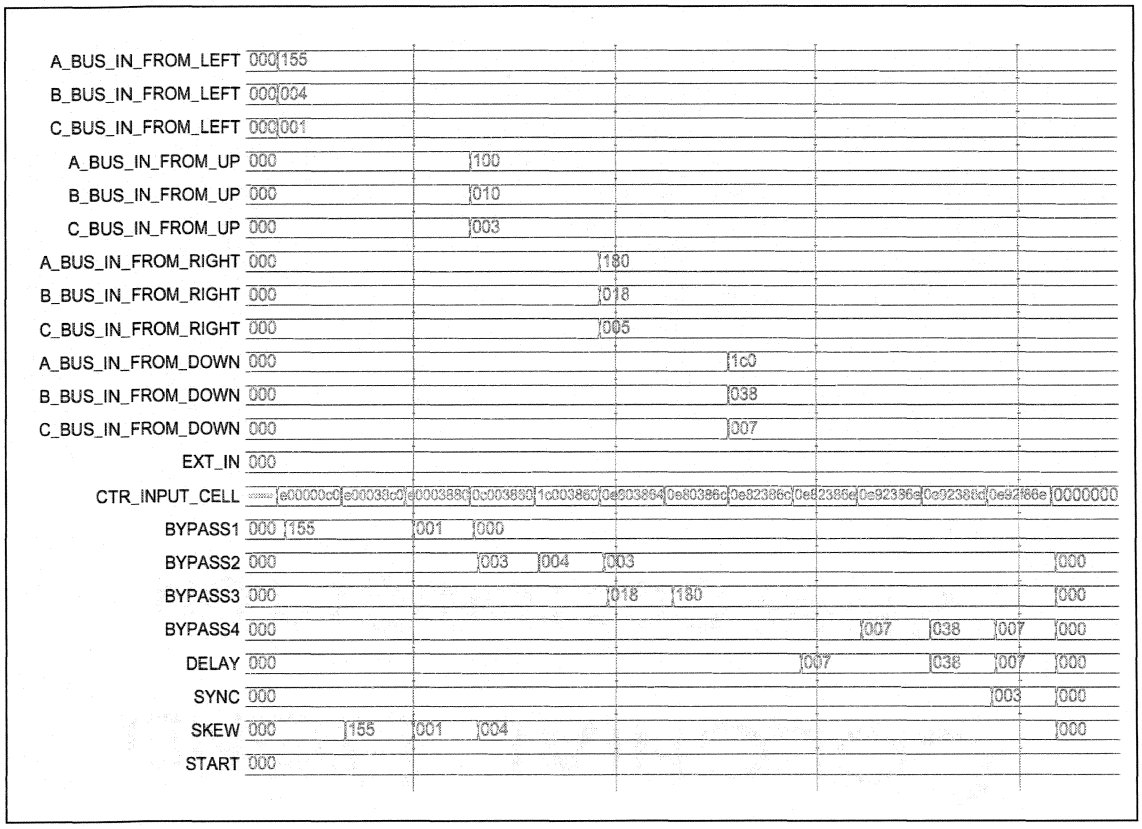

Figure 4.1 : Simulation individuelle du module d'entrée

La simulation individuelle du module de sortie permet de vérifier différents cas qui seront rencontrés lors de son utilisation. L'objectif de cette simulation est la même que pour le module précédent, soit de vérifier que l'aiguillage de données se fait correctement lors de différentes valeurs appliquées sur le bus de contrôle, CTR_OUT_CELL. Nous présentons 4 cas de figure. Selon la figure 4.2, les signaux

d'entrées du module de sortie sont de BYPASS1 à OUT_SYNC inclusivement. Le signal de contrôle des différents multiplexeurs est le bus CTR_OUT_CELL. Les signaux de sorties du module de sorties sont les bus de A_BUS_OUT_TO_RIGHT à EXT_OUT inclusivement. Le premier signal de contrôle permet le branchement des multiplexeurs à l'entrée Vss. De cette façon, toutes les sorties présentent une valeur de 0. Le deuxième contrôle, valeur Ox7FFB6DDB69, permet le branchement des signaux de sorties A_BUS_OUT_TO_RIGHT, A_BUS_OUT_TO_DOWN, A_BUS_OUT_TO_LEFT, A_BUS_OUT_TO_UP à l'entrée BYPASS1. Ce contrôle permet également le branchement des signaux de sorties B_BUS_OUT_TO_RIGHT, B_BUS_OUT_TO_DOWN, B_BUS_OUT_TO_LEFT, B_BUS_OUT_TO_UP à l'entrée BYPASS2. Également, ce contrôle permet le branchement des signaux de sorties C_BUS_OUT_TO_RIGHT, C_BUS_OUT_TO_DOWN, C_BUS_OUT_TO_LEFT, C_BUS_OUT_TO_UP au signal OUT_SYNC. Finalement, ce contrôle permet le branchement du signal EXT_OUT à l'entrée BYPASS4. Cette simulation nous a permis de vérifier le bon fonctionnement du module de sortie et de répondre aux objectifs. Les résultats sont présentés à la figure 4.2.

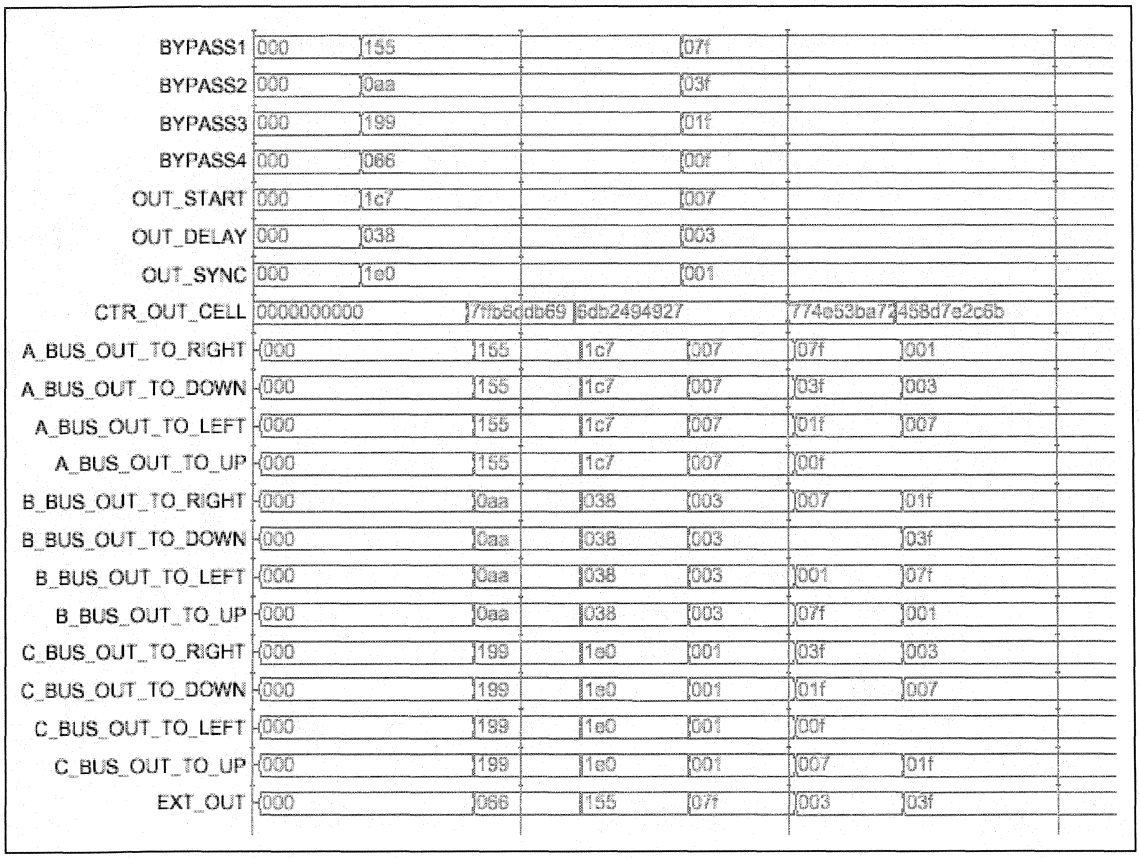

Figure 4.2 : Simulation individuelle du module de sortie

4.4.2 Simulation individuelle du module de départ

Le module de départ est utilisé dans le but de synchroniser les données à l'aide d'un générateur d'horloges. Ce générateur permet de fournir huit horloges de phases distinctes. Chacune de ses lignes d'horloges cadence respectivement les bascules de synchronisation des lignes de données. La simulation de ce module permet de vérifier le fonctionnement de la synchronisation et du générateur d'horloges.

Cependant, nous ne pouvons atteindre la précision que nous prévoyons avoir en laboratoire lors de nos simulations, étant donné que les délais fins générés par le générateur d'horloges ne sont pas représentés de façon réelle par l'outil de simulation. Ceci s'explique par l'absence du fichier de technologie. Ceci présente deux conséquences que nous observons. La première est que nous ne pouvons pas caractériser le circuit par les simulations. La deuxième est que les délais étant les mêmes, ceci engendre des résultats altérés. Par exemple, lorsque nous configurons le générateur d'horloges pour qu'il puisse générer des phases progressives entre chacune des 8 lignes d'horloges, nous rencontrons des événements de métastabilité. L'horloge locale, utilisée pour générer les 8 horloges, traverse des portes logiques dont tous les délais de propagation associés aux portes se voient automatiquement attribués une valeur de 1 nanoseconde. Dans le cas où un bit du bus de données serait appelé à changer d'état en même temps qu'une transition montante de son signal d'horloge, le simulateur forcerait la sortie de la bascule dans un état indéterminé durant toute la période d'horloge. Le signal de sortie de la bascule retrouvera un état connu au prochain front montant de son horloge si et seulement si, la valeur de l'entrée n'est pas modifiée encore une fois au même moment. Cette situation se retrouve fréquemment lorsque la génération des signaux est d'une plus grande précision que les outils de simulation que nous possédons. Il est possible que cet événement se retrouve lors des tests du circuit en laboratoire,

cependant la probabilité que cela survienne est plus mince. Ceci s'explique par le fait que le temps de maintien et de positionnement est plus petit en réalité comparativement à celui imposé par le simulateur. Alors la plage de temps nécessaire pour avoir une métastabilité est donc plus petite en pratique. Les valeurs de temps de maintien et de positionnement pour une bascule D sont respectivement de 0.3 nanoseconde et de 0.2 nanoseconde selon les spécifications du fabriquant comparativement à 1 nanoseconde pour celles associées au fichier technologique. L'erreur est plus prononcée lorsque nous analysons le temps de propagation d'une simple porte logique. En raison des contraintes, il est compréhensible d'obtenir de tels résultats en simulation. Suite à l'explication détaillée des limites de l'outil de simulation associé au fichier de technologie, nous sommes en mesure de vérifier le fonctionnement du module de départ.

L'objectif de cette simulation est de vérifier le bon fonctionnement du générateur d'horloges lorsque des données sont synchronisées avec celui-ci. Nous devons observer que les données de sortie peuvent varier en phase selon la configuration assignée au générateur d'horloges. La méthode de vérification du module de départ est la suivante. Nous avons appliqué une remise à zéro et programmé un signal d'horloge d'une fréquence de 100 MHz dès le départ de notre simulation. Par la suite, nous avons appliqué à l'entrée des bascules, bus START, des signaux de données cadencés en phases par le générateur d'horloges, bus SCLK, pour toutes les horloges. Nous avons alterné chacun des bits du bus de données pour observer la relation de phase entre ceux-ci en fonction du délai entre chacun des fronts montants des différents signaux d'horloges. Une fois la vérification du bon fonctionnement de cette partie effectuée, nous avons modifié, de façon progressive, la phase des signaux d'horloges. Lors de la modification de la phase, nous avons gardé la même séquence de données à l'entrée des bascules, bus START afin de comparer la corrélation entre les différents tests. La séquence de données, du bus start, est répétée 5 fois lors de la simulation du module départ : OxAA, Ox55, Ox00, OxFF et Ox00. Pour chacune des séquences, nous avons appliqué différentes valeurs de contrôle du bus SELECT_SCLK qui sélectionne

individuellement l'horloge des 8 bascules de données. La première sélection du bus de contrôle SELECT_SCLK choisit la même horloge pour chacune des 8 bascules. De cette manière, toutes les données seront cadencées en même temps. C'est ce que nous pouvons observer, à la figure 4.3, lorsque le signal de contrôle SELECT_SCLK est à Ox000000000. La seconde commande permet de valider la synchronisation des données à l'aide de différentes phases d'horloge. La sélection sera choisie de sorte que la première horloge, utilisée pour cadencer le bit de données 0, possède un délai minimum. Pour chacune des autres horloges, nous avons incrémenté le délai d'un pas entre chacun des bits. Ainsi, l'horloge associée au bit 1 possède un délai minimum+1 comparé au délai associé au bit 0. L'horloge associée au bit 2 possède un délai minimum+1 comparé au délai associé au bit 1 et ainsi de suite jusqu'au bit 7. Cette partie peut être observée à la figure 4.3, sur le bus d'horloge, SCLK, un escalier sur les différentes lignes, lorsque le signal de contrôle SELECT_SCLK est forcé à Ox84C2A6E1. La troisième commande permet de configurer toutes les horloges associées aux bits pairs (2,4,6 etc..) d'un délai minimum et toutes les horloges associées aux bits impairs d'une différence de délai de 2 avec les bits pairs. Les résultats de cette partie peuvent être observés lorsque le signal de contrôle SELECT_SCLK possède la valeur Ox040404040. À cette valeur de contrôle, nous pouvons observer sur le bus SCLK que tous les bits d'horloge pairs sont en phases et les bits impairs également. La quatrième commande permet d'appliquer une différence de délai maximum entre le bit 0 et le bit 1 et de réduire ce délai entre chacune des paires de bits suivants. Par exemple, la combinaison des bits 2 et 3 possède une différence de délai maximum–1, la combinaison des bits 4 et 5 possède une différence de délai maximum–2. Les résultats de cette partie peuvent être observés lorsque le signal de contrôle SELECT_SCLK est à Ox0F8F4FCF2. La dernière commande reste la même que la précédente cependant, nous avons configuré le générateur d'horloges en boucle fermée, SELECT_MUX_GF qui est forcé à 1. Les résultats de cette partie peuvent être observés lorsque le signal de contrôle SLECT_MUX_GF est à un niveau logique 1. La phase des données de sorties doivent être les mêmes que lors de la séquence précédentes. C'est ce que nous pouvons remarquer sur le bus OUT_START en observant les deux

séquences lorsque le bus SELECT_SCLK est à Ox0F8F4FCF2. Nous pouvons conclure pour cette partie que la simulation individuelle du module de départ est fonctionnelle et que le générateurs d'horloges fonctionne correctement.

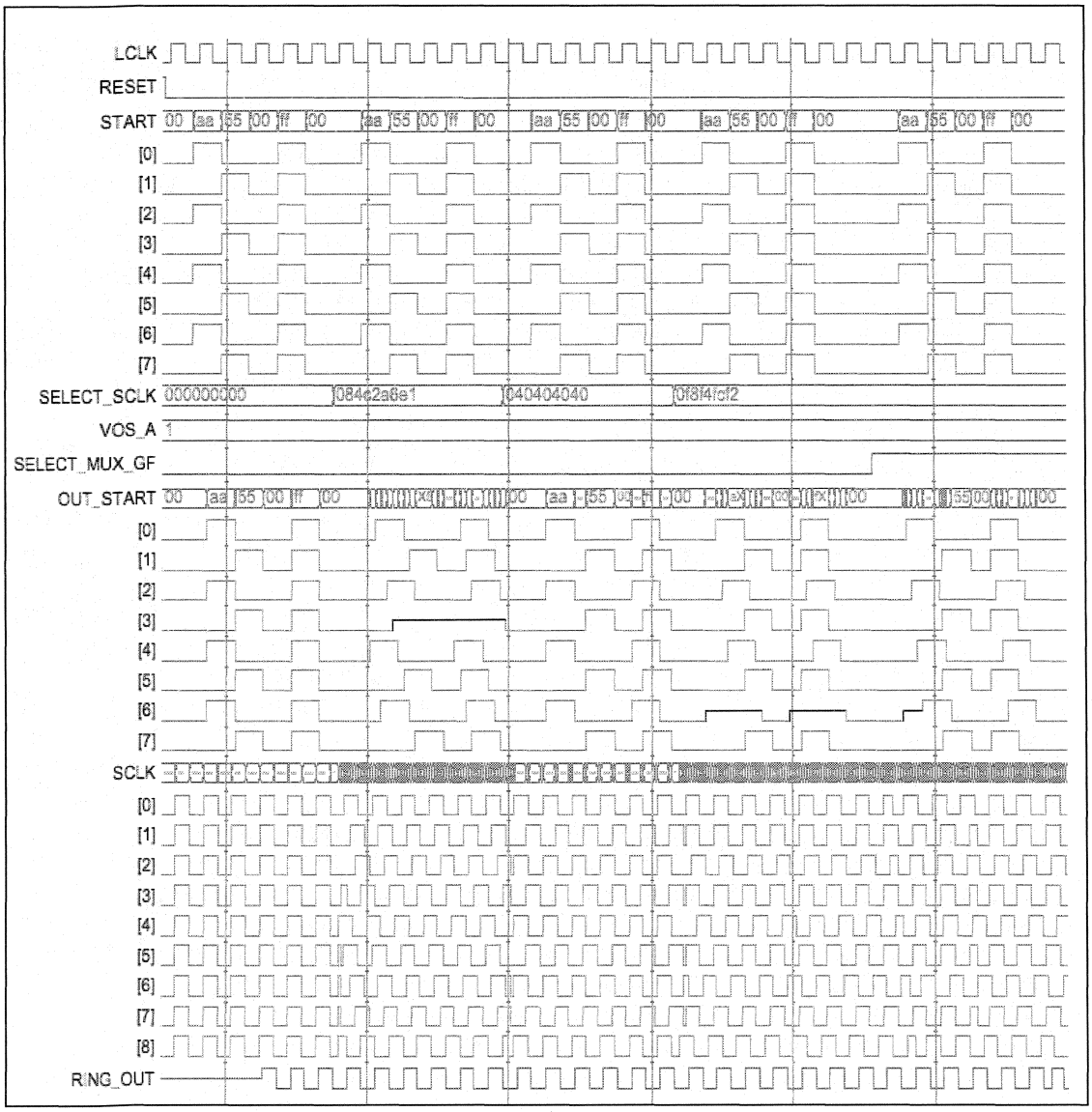

Figure 4.3 : Simulation individuelle du module de départ

4.4.3 Simulation individuelle du module de lignes à délais fins

Le module de lignes à délais fins est utilisé afin de créer une restriction lors du parcours des données à travers celui-ci. Les délais occasionnés par ce module se reflètent par une différence de phase plus ou moins grande sur chacun des bits du bus de données. La différence de phase entre les signaux de sorties du module est provoquée par la valeur associée au signal de contrôle.

Nous ne présenterons pas de résultats de simulations fonctionnels pour la partie de lignes de délais fins dans cette section. Les résultats seront présentés lors de la simulation de la cellule SI. De cette façon, nous serons en mesure d'analyser la fonctionnalité entière de ce module qui comporte, en fait, 9 copies de ce module, une pour chacun des 9 bits de données. L'analyse des différences de délais sera alors effectuée.

4.4.4 Simulation individuelle du module synchroniseur

Le synchroniseur permet la re-synchronisation des données à l'aide de l'horloge locale, suite à une prise de décision effectuée par ce module. Le résultat de cette décision fait en sorte que les données sont re-synchronisées à l'aide du front montant ou du front descendant de l'horloge locale du module. Une fois que les données ont été re-synchronisées, elles sont dirigées vers le module de sortie de la cellule. Elles peuvent être ensuite menées vers la sortie de la puce et/ou toutes autres cellules qui lui sont reliées physiquement. Ce module est décrit à la figure 3.1, intitulée *connexions simplifiées des cellules SI.*

Le synchroniseur requière deux horloges pour effectuer une re-synchronisation. La première est l'horloge locale, LCLK et la deuxième est l'horloge reçue, RCLK. Nous

avons configuré l'horloge locale de sorte qu'elle soit périodique et qu'elle ne varie ni en phase ni en fréquence. L'horloge reçue, RCLK, a dû être configurée manuellement pour simuler l'effet du biais et des délais d'une utilisation réelle de la cellule SI. Nous avons forcé le signal RCLK pour un grand nombre de conditions possibles. Nous avons appliqué cette horloge pour qu'elle soit en phase avec l'horloge locale, déphasée progressivement d'un retard de moins d'une demi-période et d'un retard de plus d'une demi-période. Ceci permet de valider le comportement des modules de décision du synchroniseur. Ainsi, l'objectif de cette simulation est de vérifier que si la transition du front descendant du signal d'horloge reçu est réalisée avant celle du front montant ou descendant de l'horloge locale, le synchroniseur prendra la bonne décision. Nous forcerons les transitions afin d'observer le plus grand nombre de cas possibles.

Les résultats des simulations sont présentés en trois parties. La première partie est une vue d'ensemble de la simulation, présentée à la figure 4.4. La deuxième partie est focalisée sur l'événement du signal DRZ à un niveau logique haut. La troisième partie est concentrée sur l'événement d'un signal DFZ à un niveau logique haut. Les deux dernières parties se retrouvent en annexe D (voir figure D.1 et D.2).

Les signaux importants de cette simulation sont le signal LCLK, l'horloge locale, et RCLK, l'horloge reçue. Le signal interne DRZ indique si le front descendant du signal d'horloge reçue survient avant la transition du front montant du signal d'horloge locale. Un second signal interne, DFZ, indique si la transition du front descendant du signal d'horloge reçue, RCLK, survient avant celle du front descendant du signal d'horloge locale, LCLK. Le signal interne OUT_MAE sélectionne le front montant ou le front descendant pour effectuer la re-synchronisation des données. Les signaux SYNC et OUT_SYNC sont respectivement le bus de données d'entrée et le bus données de sortie du module synchroniseur. La simulation débute par une remise à zéro du circuit, qui peut être remarqué sur le signal RESET. Par la suite, nous avons généré le signal

d'horloge reçue, RCLK, de façon à pouvoir rencontrer tous les cas importants. Nous pouvons observer sur le signal D_R_Z qu'il devient haut deux fois. Ceci s'explique par le fait que le front montant de l'horloge locale surviennent entre le front descendant du signal d'horloge reçue et le front descendant du signal d'horloge reçue retardée.

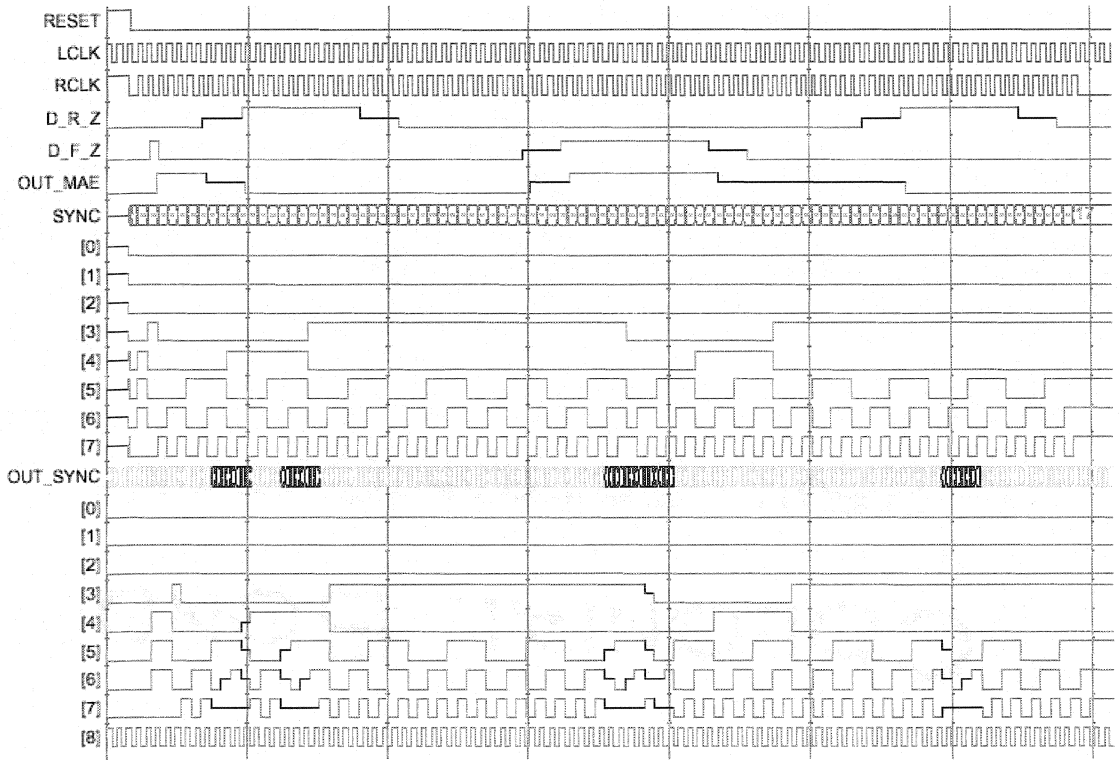

Figure 4.4 : Simulation individuelle du synchroniseur, vue d'ensemble

Dans le cas contraire, nous avons simulé le front descendant du signal d'horloge reçue, RCLK, très près du front descendant du signal d'horloge locale. De ce fait, la sortie DFZ devient active pour le signaler au module MAE. Dans cette condition, le module MAE optera pour une sélection de la re-synchronisation sur le front descendant de l'horloge locale. Nous pouvons observer ce cas lorsque le signal OUT_MAE et D_F_Z sont à un niveau haut et que le signal D_R_Z est à un niveau bas.

4.4.5 Simulation individuelle du détecteur de phases et de fréquences (PFD)

Le détecteur de phases et de fréquences permet de fournir des renseignements quantitatifs en ce qui a trait à la différence de phases entre deux signaux d'un bus de données. Il en résulte une tension plus ou moins importante en fonction du délai entre les deux signaux désignés par programmation. La simulation du module PFD nécessite une vérification en deux phases. La première phase permet la vérification de la logique combinatoire. La seconde phase vérifie la partie analogique du PFD. Chacune de ces vérifications sera faite individuellement, étant donné une restriction en ce qui concerne l'outil de simulation qui ne peut combiner la logique numérique et analogique.

4.4.5.1 Simulation de la partie contrôle, logique combinatoire

La vérification de la logique de contrôle du PFD est réalisée à l'aide de l'outil de simulation ModelSim. Une séquence bien précise doit être respectée afin d'obtenir les résultats attendus. Les détails de cette séquence ont été présentés à la section 3.2.11.3 qui décrit les *étapes de ré-initialisation et de mise en fonction* au chapitre 3 de cet ouvrage. L'objectif de cette simulation permet de vérifier le comportement de la partie contrôle du PFD. Nous enverrons des signaux de contrôle pour effectuer la sélection de certain des bits de données. Seul les bits sélectionnés devront amorcer les signaux de sorties. Le signal SKEW est le bus de données du détecteur de phases. Les bus d'entrées FRONT_FIRST et FRONT_LAST permettent la détection sur un front montant ou descendant. Le bus d'entrée PFD_EN permet d'activer ou de désactiver les bits de comparaison. Les signaux internes FIRST et LAST sont respectivement le signal qui détecte le premier et le dernier bit à changer d'état. Nous présenterons 3 détections. La première comparaison active tous les bits de données. Nous avons fait parvenir seulement le bit zéro du bus de données, ce qui active le signal FIRST et par la suite nous avons envoyé tous les autres signaux qui engendrent l'activation du signal LAST.

Nous pouvons observer les résultats lorsque le bus PFD_EN est forcé à Ox1FF. La seconde détection active seulement le bit le plus significatif et le moins significatif. Nous pouvons observer les résultats lorsque le bus PFD_EN est forcé à Ox101. La dernière détection active seulement les bits 0, 2, 4, 6 et 8 du bus SKEW. Lors de la dernière détection, nous avons intentionnellement fait parvenir 3 valeurs aux bus de données, SKEW. La première valeur (Ox100)fait en sorte d'activer le signal FIRST. La deuxième valeur, (Ox1A7) permet d'activer les bits 1, 3, 6, 7 et 8. Cependant, il manque la transition haute du bit 2 pour terminer la séquence du bus PFD_EN qui permettra l'activation du signal LAST. La troisième valeur (Ox1F7) permet de satisfaire la condition pour activer le signal LAST. Les résultats de simulation se retrouvent à la figure 4.5.

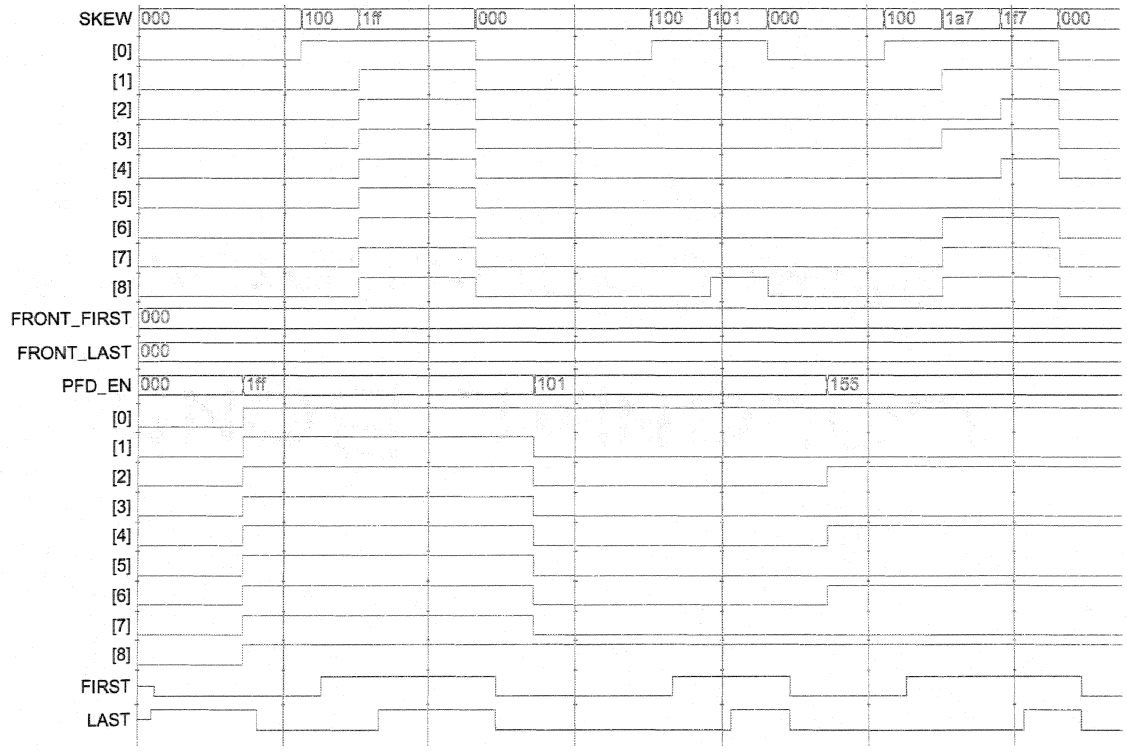

Figure 4.5 : Simulation individuelle de la logique combinatoire du circuit PFD

4.4.5.2 Simulation de la partie analogique

Les simulations ont été accomplies à l'aide du logiciel Spectre, afin de vérifier le bon fonctionnement du circuit. Les objectifs des simulations sont de vérifier : le courant de sortie du miroir de courant, la plage dynamique, le temps de décharge de la capacité et le temps mort du circuit. Le courant de sortie du miroir atteint 3 mA. Ce courant permet de créer une plage dynamique simulée de 0.103 mV/psec, comparativement à une exigence de 0.100 mV/psec. Le temps nécessaire à la décharge de la capacité est de 4 nsec. Dans le cas où le circuit serait cadencé à 125 MHz, tel qu'il le devrait, le temps nécessaire pour décharger le condensateur sera de ½ période.

Les prises de mesures effectuées sur le PFD permettront de caractériser celui-ci. Elles seront utilisées, particulièrement, pour établir la relation de phases du circuit entre la différence de temps des signaux et le niveau de la tension aux bornes de la capacité. Les mesures seront générées de sorte qu'elles permettront d'évaluer le temps mort de celui-ci. Une des spécifications du circuit de détection de phases et de fréquences est que celui-ci doit être en mesure de fournir des informations quantitatives pour des différences de phases plus grandes que 500 psec. Pour ce faire, nous avons concentré nos mesures sur une plage variant entre 0 et 2nsec. De cette façon, nous serons en mesure de trouver l'équation caractérisant notre circuit et ainsi de vérifier s'il répond aux spécifications. Les résultats de simulation sont présentés au tableau 4.1.

DIFFÉRENCE DE PHASE	TENSION DE SORTIE
0 sec.	95.5 mV
50 psec.	98.7 mV
100 psec.	103 mV
500 psec.	143.1 mV
1000 psec.	194.7 mV
2000 psec.	298.1 mV

Tableau 4.1 : Tensions de sortie du détecteur pour divers déphasages.

L'analyse des résultats de simulation nous amène à dire que la sortie ne réagit pas linéairement à l'entrée pour des courtes différences de phases, tel que cela devrait être idéalement. Lorsque nous faisons parvenir les deux signaux en même temps, aux entrées « A » et « B » du détecteur de phase, la sortie croît jusqu'à une tension de 95.5 mVolts contrairement à 0 Volt tel que souhaité théoriquement. Ceci est expliqué par le temps mort associé au circuit PFD. Une relation temporelle peut être établie entre la tension aux bornes de la capacité et la différence de phases entre deux signaux.

DIFFÉRENCE DE PHASE (T)	TENSION DE SORTIE (V)	ÉQUATION DES DROITES	
0 sec.	95.5 mV	V(t) = 0.064*t + 95.5mV	
50 psec.	98.7 mV		V(t) = 0.086*t + 94.4mV
100 psec.	103 mV	V(t) = 0.10025*t + 92.97mV	
500 psec.	143.1 mV		V(t) = 0.1032*t + 91.5mV
1000 psec.	194.7 mV	V(t) = 0.1034*t + 91.3mV	
2000 psec.	298.1 mV		

Tableau 4.2 : Modèle linéaire par morceau dérivé des simulations.

D'après les résultats de simulation du tableau 4.1, nous pouvons remarquer que lorsque la différence de phases converge vers zéro, la sortie du détecteur de phases ne réagit pas avec la même dynamique. Nous pouvons tirer de ces résultats le temps mort associé à ce circuit. Une interpolation linéaire des résultats, présentée à la figure 4.6, permet de caractériser le temps mort du détecteur de phases. La génération des droites a été faite à partir de tous les points du tableau 4.2, juxtaposés vers l'ordonnée à l'origine. Une projection de ces droites a permis de vérifier l'ordonnée de chacune d'elles. L'écart le plus important entre les ordonnées sera utilisé pour trouver le temps mort associé au circuit.

Pour trouver l'ordonnée à l'origine, nous avons sélectionné les 5 points ayant les plus petites différences de phases concentrées vers l'ordonnée. À l'aide de ces 5 points, nous avons trouvé les équations pour chacune des combinaisons de points consécutifs. La droite 1 est déterminée selon les points de 0 sec et de 50 picosecondes. La droite 2 est dérivée des points 50 et 100 picosecondes. La droite 3 est dérivée des points 100 et 500

picosecondes. La droite 4 est dérivée des points 500 et 1000 picosecondes. La droite 5 est calculée selon des points 1000 et 2000 picosecondes. Les résultats d'extrapolation sont présentés à la figure 4.6.

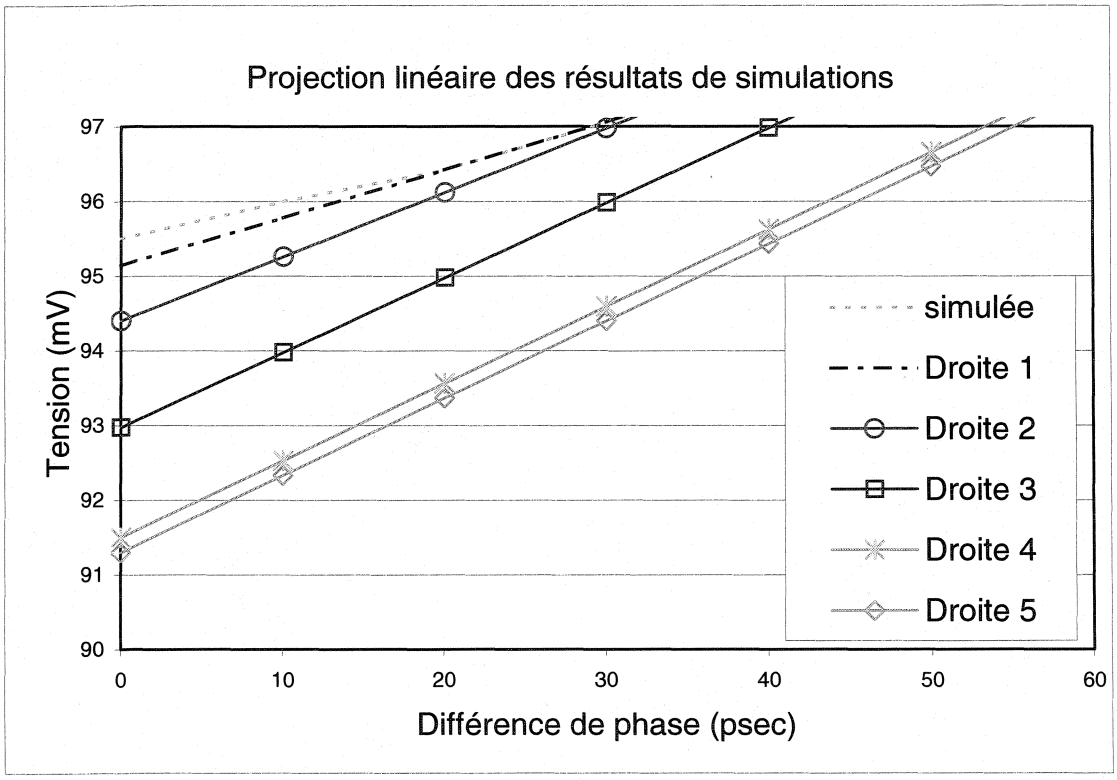

Figure 4.6 : Projection linéaire des mesures simulées

Selon les résultats présentés au tableau 4.2 et à la figure 4.6, nous pouvons déterminer le temps mort du circuit de détection de phases et de fréquences. Nous avons utilisé les deux extremums des droites à l'origine soit 91.3 et 95.5 pour les transposer sur le temps mort. La pente de la droite utilisée est celle qui décrit le circuit en régime permanent et sa valeur est de 0.103 mV/psec. Ceci donne l'équation 10 générée par itération :

$$Vout = 0.103\,\text{mV/psec.*}\,t + 95.5\text{mV} \qquad (10)$$

Le temps mort est le principal effet qui réduit la précision du circuit PFD. Cette erreur est due au temps de propagation de la porte « NON-ET » ajouté au temps de propagation de la remise à zéro de la bascule. Ce temps supplémentaire a pour effet de paralyser le miroir de courant. Durant cette période, les transistors M1 et M2 sont activés en même temps, ce qui cause un court-circuit au miroir de courant. Pour plus de détail sur le circuit, le lecteur est référé au chapitre précédent.

Les ordonnées les plus distantes affichent les résultats qui suivent : 95.5 mV et 91.3 mV. À l'aide de ces deux valeurs, nous déterminons le temps mort qui est représenté à l'équation 11:

$$\frac{(95.5-91.3)mV}{\left(0.103^{mV}/_{p\,\mathrm{sec.}}\right)} = 40.77\,p\,\mathrm{sec.} \qquad (11)$$

En raison de cette valeur, l'équation 10 n'est valide que pour des différences de phases excédant 41 psec. En dessous de cette valeur, le détecteur de phase ne réagit pas correctement et les données sont altérées. Le temps mort est semblable au régime transitoire d'un circuit. Il doit passer par cette période, indésirable dans notre cas, avant de fonctionner dans sa région linéaire. Une fois que l'accumulation des différences de phases aura atteint la limite du condensateur, pour une différence de 25 nsec, le circuit se trouvera dans la région de saturation. Le circuit ne devrait jamais se trouver dans cette région. La figure 4.7 distingue les différentes régions d'opération du PFD, soit le temps mort, la région linéaire et la région de saturation. Nous avons prévu une capacité externe lors de nos tests en laboratoire. Cette capacité additionnelle permettra d'obtenir une plage dynamique plus grande que celle rapportée ici. Ceci permettra de comparer des différences de phases supérieures à 25 nanosecondes.

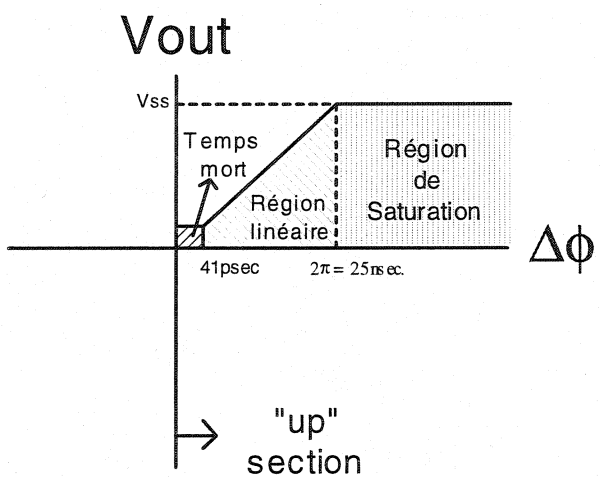

Figure 4.7 : Régimes du circuit PFD

Maintenant que la caractérisation du temps mort a été effectuée, nous allons caractériser, par simulation, le temps de décharge du condensateur. Nous avons chargé à pleine capacité le condensateur de 30 pF et forcé une décharge via un transistor d'une largeur de 200 micromètres (longueur minimale de 0,35 micron). Le temps de décharge du condensateur est de (224 – 220 = 4) nanosecondes. L'objectif de cette simulation est de vérifier le temps de décharge du condensateur s'il excède une période d'horloge, soit 8 nanosecondes. Les résultats sont présentés à la figure 4.8. Nous avons la possibilité de varier le signal de décharge pour qu'il soit actif pour un maximum de 16 périodes d'horloge.

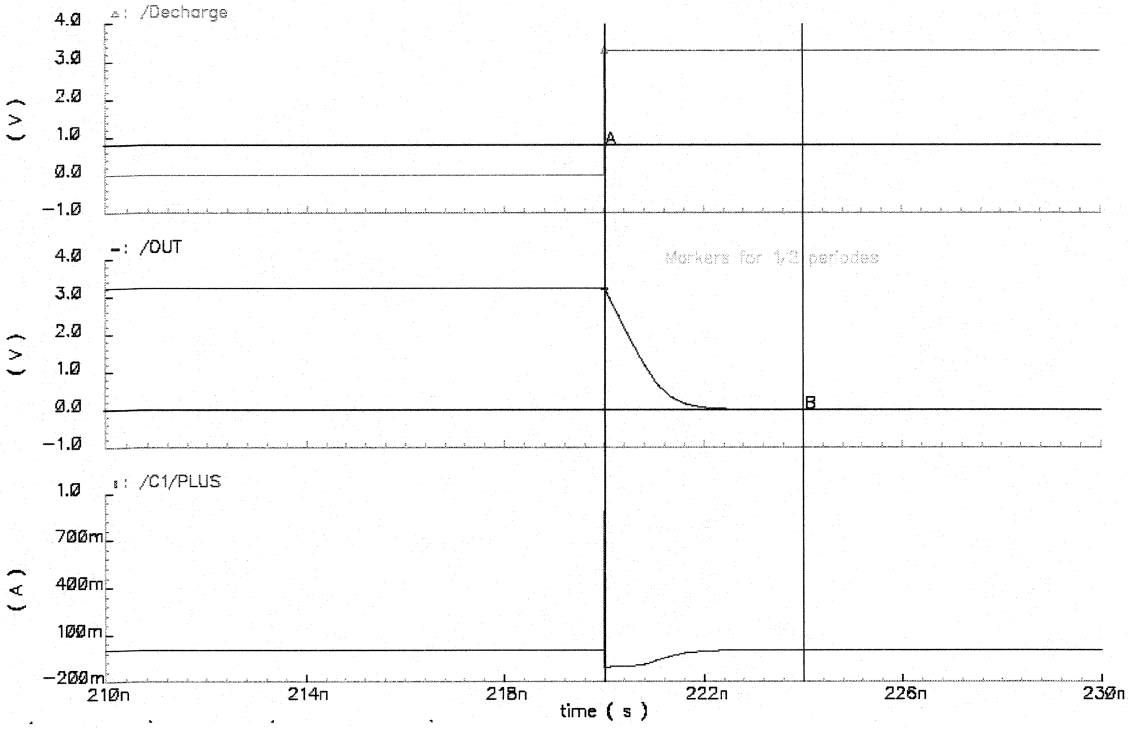

Figure 4.8 : Simulation du temps de décharge du condensateur

Une seconde simulation permet de vérifier la valeurs du courant de sortie du miroir de courant lors d'une différence de phase 1 nanoseconde entre les signaux « ina » et « inb ». Cette simulation est présentée à la figure 4.9.

Une troisième simulation permet d'observer le comportement du circuit de détection de phases et de fréquences lorsque la différence de phases est très élevée. L'objectif de cette simulation est de vérifier la tension de sortie au condensateur. Nous pouvons remarquer que les signaux « inA » et « inB » sont déphasés de 45 nanosecondes et qu'à 25 nanosecondes de différence, le condensateur est saturé à 3.3 volts tel que les spécifications l'exigent. Cette simulation est présentée à la figure 4.10.

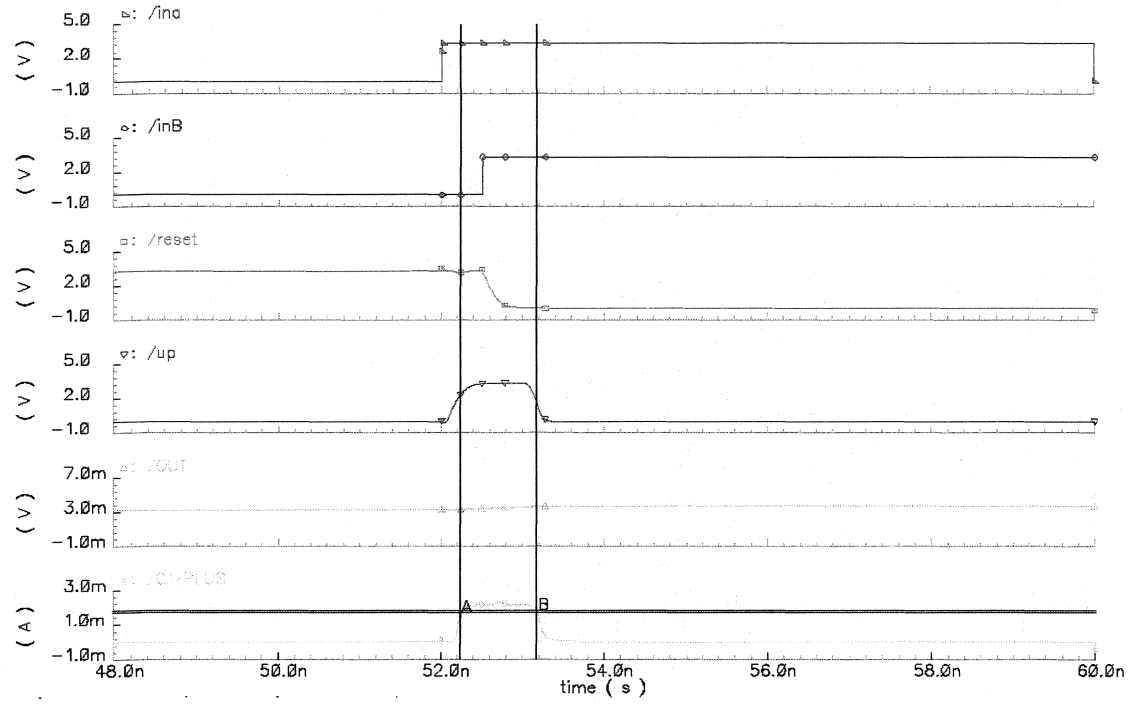

Figure 4.9 : Simulation du courant de sorite du PFD

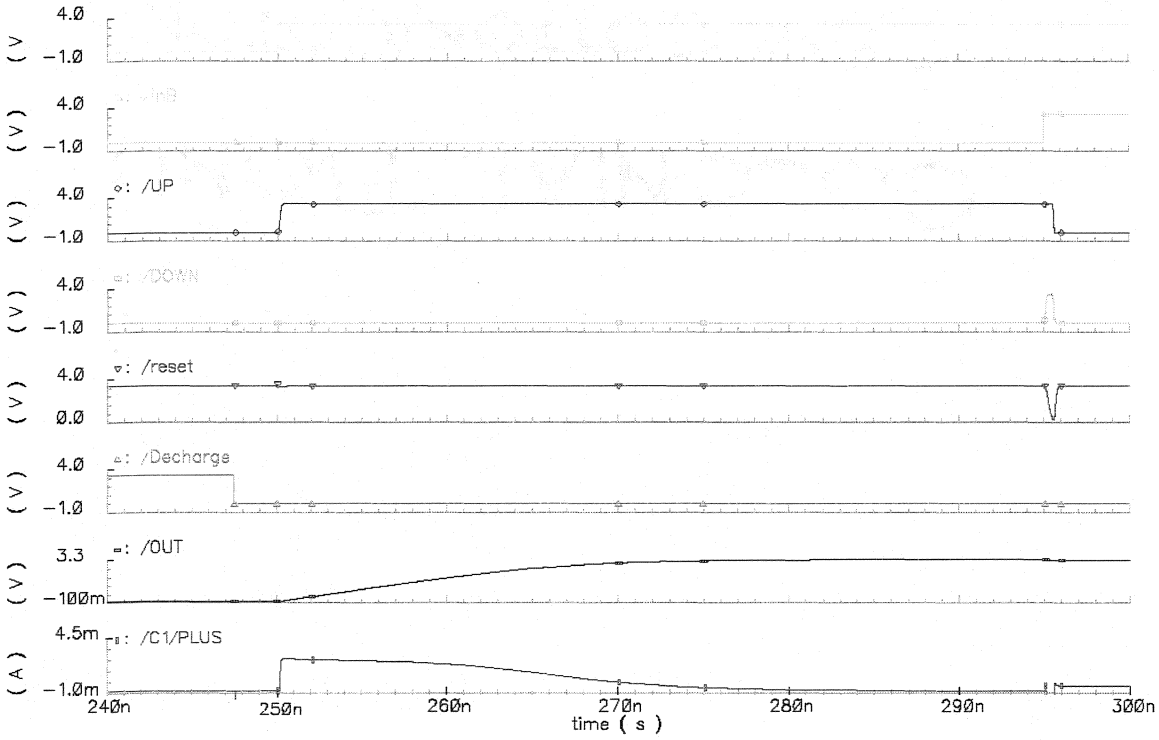

Figure 4.10 : Simulation du comportement en présence d'une grande différence de phases.

Les résultats nous mènent à dire que le circuit PFD répond aux exigences des spécifications en termes de fonctionnalité. La simulation de la cellule SI montrera entre autres la simulation du PFD parmi les autres modules.

4.5 Simulation de la cellule SI

Dans la section précédente, nous avons expliqué les simulations de tous les modules de la cellule SI. Ces simulations ont été réalisées en isolant chacun des modules. Cette section permettra d'expliquer les simulations de chacun des modules, mais cette fois-ci, étant inter-reliés. L'ajout du module décodeur et du module de ligne à délais fins complétera la validation de la cellule SI. La simulation de la cellule SI a été effectué à l'aide d'un banc d'essai programmé en VHDL. Le début des simulations sera fait par une initialisation de la cellule. Une seconde simulation validera le passage des données provenant du monde extérieur par le module de décodeur, le module d'entrées, les bus de BYPASS, le module de sorties et finalement vers le bus de sorties du monde extérieur. Une troisième simulation permettra de valider le module de départ stimulé par le décodeur et dirigé vers le module de sorties. Une quatrième simulation confirmera le fonctionnement du module de départ. Suivra ensuite la cinquième simulation qui permet de vérifier la fonctionnalité du module de délais fins. Il faut noter qu'avant toute simulation, il est essentiel de faire une remise à zéro de la cellule afin qu'elle soit au départ dans un état connu.

Il faut noter que lorsque les données proviennent du décodeur à travers le bus EXT_IN, le bit le plus significatif n'est pas utilisé. Ceci est du au fait que nous voulions inclure le signal d'horloge locale sur le bus de données. Alors c'est dans le module du décodeur que nous avons décidé de faire cette manœuvre. Cependant, le bit le plus significatif du

bus EXT_IN est utilisé lors de la configuration des différents registres. Le tableau 4.3 dresse une liste des signaux d'entrées et de sortie de la cellule d'intégrité des signaux.

NOMS	NOMBRE	DIRECTION	BRIEVE DESCRIPTION	TYPE
LCLK	1	Entrée	Horloge locale de la cellule	Digital
DATA_EXT	9	Entrées	Bus de données de la cellule (utilisé pour configurer les registres)	Digital
ADDRESS	5	Entrées	Bus d'adresses	Digital
W	1	Entrée	Ligne d'écriture	Digital
STOP_B_ACQ	1	Entrée	Signal indiquant la fin de l'acquisition du convertisseur Analogique à numérique (pour le second PFD)	Digital
STOP_A_ACQ	1	Entrée	Signal indiquant la fin de l'acquisition du convertisseur Analogique à numérique (pour le premier PFD)	Digital
VOS_A	1	Entrée	Signal analogique contrôlant la fréquence de l'oscillateur (0 de 3.3 Volts)	Analogue
EXT_OUT	9	Sorties	Bus de données de sorties	Digital
PFD_OUT_A	1	Sortie	Sortie du module de détection de phases et de fréquences (pour le premier PFD)	Analogue
PFD_OUT_B	1	Sortie	Sortie du module de détection de phases et de fréquences (pour le second PFD)	Analogue
START_A_ACQ	1	Sortie	Signal indiquant au convertisseur analogique à numérique de débuter la conversion (pour le premier PFD)	Digital
START_B_ACQ	1	Sortie	Signal indiquant au convertisseur analogique à numérique de débuter la conversion (pour le second PFD)	Digital
ERROR_SYNC	5	Sorties	État du synchroniseur (signaux interne)	Digital
SCLK	9	Sorties	Horloges déphasée «Skewed »	Digital
RING_OUT	1	Sortie	Horloge générée possédant le délai maximum.	Digital
VDD	Partagé	IN/OUT	Tension de la logique numérique	Puissance
GND	Partagé	IN/OUT	Masse de la logique numérique	Puissance
A_VDD	2	IN/OUT	Tension 3.3 volts analogue	Puissance
A_GND	2	IN/OUT	Masse analogue	Puissance

Tableau 4.3 : Liste des signaux de la cellule SI

4.5.1 Initialisation de la cellule SI

La méthode utilisée pour forcer la cellule dans un mode d'initialisation est d'écrire à tous les registres du décodeur. Le décodeur, module qui génère le signal d'initialisation, a été conçu de sorte qu'il doit procéder à l'écriture de tous ses registres avant d'effectuer une initialisation. Ceci permet d'adresser le dernier registre et de générer un signal de remise à zéro. L'objectif de cette simulation est de vérifier si tous les registres ont été écris avant que le signal RESET ne devienne actif. Les signaux d'entrée utiles pour l'initialisation de la cellule sont : le bus tb_adress, tb_data_ext et tb_w. Les signaux de sortie utiles sont : tb_ext_out, tb_error_sync, tb_sclk, tb_ring_out, tb_start_a_acq et tb_start_b_acq. Le signal interne généré par le décodeur indiquant une remise à zéro est le signal RESET. Nous pouvons remarquer que cette simulation répond aux exigences. La simulation de cette partie se retrouve à la figure 4.11.

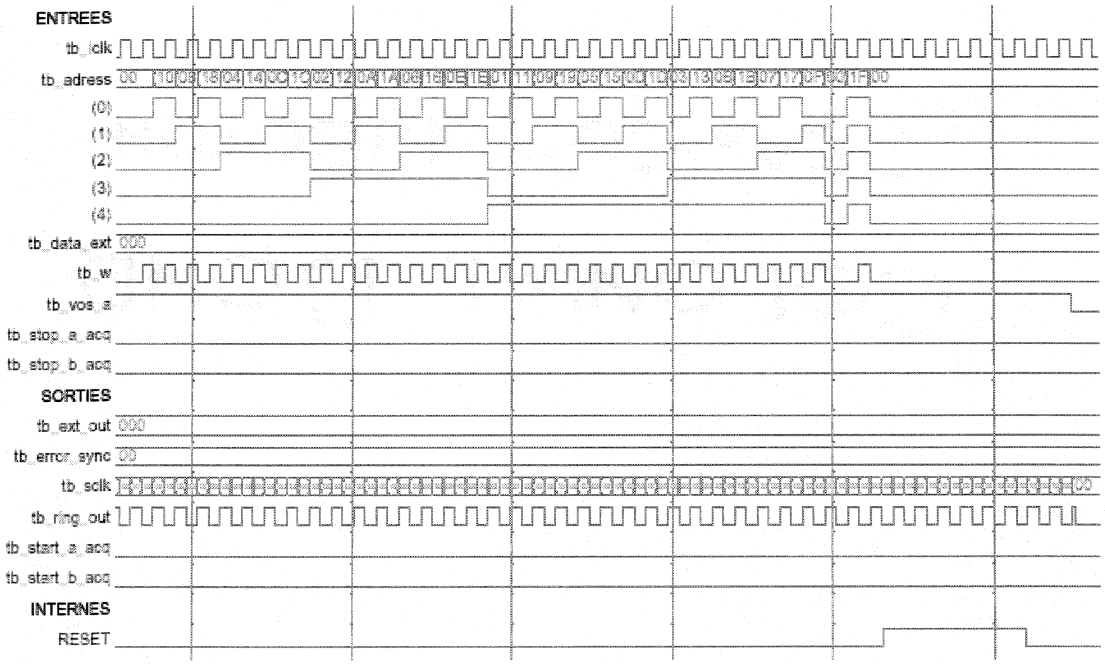

Figure 4.11 : Simulation de la remise à zéro de la cellule SI

4.5.2 Interaction des modules d'entrée et de sortie

Nous avons fait une simulation afin de valider l'échange des données provenant du module d'entrée vers le module de sortie via les bus BYPASS 1 à 4. Pour ce faire, nous avons envoyé des donnés via le bus tb_data_ext vers les différentes BYPASS. Les signaux d'entrée et de sortie de la cellule SI sont identiques à la simulation précédente. Les signaux internes utiles à la simulation des modules d'entrée et de sortie sont le bus CTRL_INPUT_CELL qui permet la configuration des multiplexeurs du module d'entrée. Le second bus de contrôle est le CTR_OUT_CELL qui permet la configuration des multiplexeurs du module de sortie. Les quatre autres bus permettent une connexion directe entre le module d'entrées et le module de sortie, BYPASS1, BYPASS2, BYPASS3 et BYPASS4. Nous avons fait parvenir une séquence de 5 valeurs provenant du décodeur et dirigées vers la sortie externe de la puce. La séquence a été répétée 4 fois pour utiliser les 4 BYPASS. Les résultats de cette simulation complète se trouvent en annexe D à la figure D.3.

Une simulation concentrée sur le bus BYPASS1 est présentée à la figure 4.12. Nous pouvons voir sur cette figure les configurations nécessaires pour aiguiller les données du décodeur vers la sortie tb_ext_out en passant par le bus BYPASS1. La première donnée envoyée vers le bus BYPASS1 est Ox1FF suivie de Ox155, Ox000 et la dernière valeur est Ox0AA. Il faut se rappeler que les valeurs du signal MSB ne sont pas envoyées par le décodeur. Ainsi, les valeurs des données de sorties sont les mêmes à l'exception du bit le plus significatif.

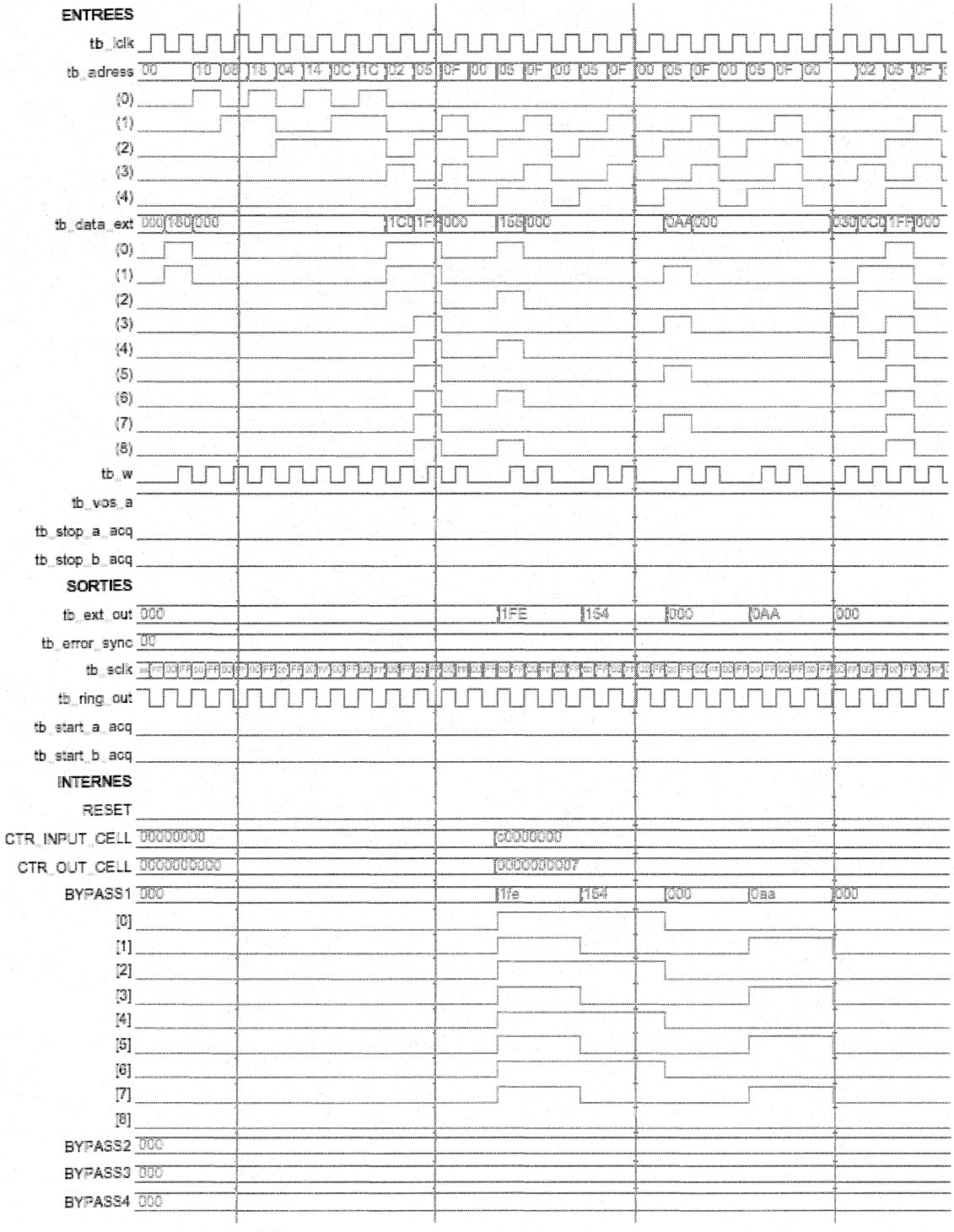

Figure 4.12 : Simulation des modules d'entrée/sortie concentrée sur le fonctionnement du bus BYPASS1

4.5.3 Simulation du module de délais fins

Les données appliquées pour valider les différents délais sont sélectionnées de façon à pouvoir observer leurs restrictions lors du passage dans le module de délai. Ceci est accompli en fonction du signal de contrôle appliqué. Les signaux de contrôle ont été choisis de façon à observer différents délais entre les lignes de données. Nous avons envoyé deux signaux de contrôle permettant d'observer plusieurs délais pouvant être générés par le module de délais fins. L'objectif de cette simulation est de vérifier la restriction du passage des données pour différents contrôles. Nous devrons être en mesure d'observer les signaux de donnée déphasée avec ceux de l'entrée et entre eux selon la valeur du contrôle choisi.

Les signaux d'entrées de la cellule sont les mêmes que dans les simulations précédentes. Les signaux de données interne du module TFT sont le bus d'entrées DELAY et le bus de sorties OUT_DELAY. Les bus de contrôle sont CTR_TFT_0 à CTR_TFT_8 qui configure, individuellement, le délai programmé pour chacun des bits de données. Le bus CTR_LOOP permet la configuration des bits de données en boucle fermée.

Nous avons procédé à deux changements de configuration associés aux délais des lignes. Chacun des bits de données possède un contrôle différent qui permet d'introduire un délai différent. La première configuration permet de valider un passage rapide et de petites variations de délais. Ceci peut être observé lorsque le bus de contrôle CTR_TFT_1 est forcé à Ox00000003.

La seconde configuration est divisée en deux parties. La première moitié du vecteur de donnée, la partie basse, permet de vérifier un passage rapide dans le module TFT et des

variations de délais petites entre eux. La deuxième moitié du vecteur de donnée, la partie haute, permet de vérifier un passage lent dans le module TFT et des petites variations de délai. L'analyse complète du vecteur de données, partie basse et partie haute, valide le comportement du module en présence de grandes différences de délais entre la première moitié et la seconde moitié du vecteur de données.

Ce module fait partie de la stratégie permettant d'ajuster les délais de ligne lorsque les délais inter-réticulaires sont supérieurs à une période d'horloge. La transmission des données dans ce module permet l'ajustement les délais en augmentant ceux-ci de façon à contourner les effets de la métastabilité. Ces délais de ligne sont ajoutés intentionnellement et sont inévitables pour un bon fonctionnement du module. Ceci s'explique par le fait que les réticules ne possèdent pas les mêmes domaines d'horloges. Ainsi, les phases des signaux d'horloge ne sont pas identiques. Alors nous ajoutons des délais, via le module TFT, afin d'ajuster la phase des données avec la phase de l'horloge pour remédier aux effets de la métastabilité. Les résultats des simulations sont présentés en annexe D (voir figure D.4).

4.5.4 Simulation du module PFD

La simulation du PFD est une étape qui a requis beaucoup d'efforts. Les possibilités de configuration de ce module sont très nombreuses. Nous avons fait varier le temps de décharge de la capacité, le nombre d'acquisition et testé différentes configurations d'activation des données. Nous avons établi une séquence du nombre d'acquisitions par le détecteur de phases et de fréquences avant de procéder aux décharges de la capacité. La séquence du nombre d'acquisition est la suivante : Ox01, Ox05, Ox09, Ox0D, Ox11, Ox15, Ox19 et Ox1D. Ceci correspond à effectuer de 1 à 8 acquisitions par valeur, selon le cas. Pour chaque valeur d'acquisition, nous avons fait parvenir la quantité de données nécessaires pour satisfaire le nombre d'acquisition tel que configuré. Par la suite, nous

avons simulé une séquence pour valider le temps de décharge de la capacité. Pour chaque valeur de temps de décharge, nous avons fait passer la séquence du nombre d'acquisition. Ainsi, pour une valeur précise du temps de décharge, nous avons réalisé 8 configurations différentes du nombre d'acquisitions.

Le signal de données d'entrées interne du module PFD est le vecteur SKEW. Le bus PFD_EN permet d'activer ou de désactiver des bits du bus de données qui ne seront pas impliqués lors de la comparaison. Les signaux FIRST et LAST indiquent respectivement que le premier et le dernier bit activé ayant été détecté. Le signal PFD_HOLD permet de suspendre le circuit analogique pendant que le convertisseur analogique à numérique effectue l'acquisition de la tension aux bornes de la capacité. Le bus Disch_value permet de configurer le temps de la décharge de la capacité en nombre de périodes d'horloge. Le bus Counter_value permet de configurer le nombre d'acquisitions qui seront effectuées avant de procéder à la décharge de la capacité. Lorsque le signal DISCHARGE_A_PFD est activé, actif haut, la capacité se décharge. Le signal START_A_ACQ indique au convertisseur analogique à numérique que le nombre d'acquisitions correspond à la configuration et que les valeurs comparées correspondent aux valeurs envoyées. Le dernier signal, Stop_a_acq, provient du convertisseur et indique que la conversion a été effectuée avec succès et que la décharge peut être amorcée. Les résultats complets de la simulation du détecteur de phases et de fréquences se retrouvent en annexe D (voir figure D.5).

Nous avons concentré une partie de la simulation complète du PFD lorsque la configuration de celui-ci présente 1 et 2 acquisitions (voir figure D.6). Lors du cas d'une acquisition, les signaux FIRST et LAST deviennent actifs une seule fois avant la transition vers le haut du signal START_A_ACQ. Pendant ce même moment, le signal PFD_HOLD force le module PFD en attente de traitement. Peu importe si des données entrent dans le module, le circuit reste en attente jusqu'à ce que le signal PFD_HOLD soit désactivé. Par la suite, nous avons simulé la réponse du convertisseur en activant le

signal stop_a_acq. Ceci indique que le convertisseur a terminé le traitement et qu'il est prêt à réaliser une autre conversion. Une fois que le module du décodeur aperçoit la réponse de fin d'acquisition provenant du convertisseur, il active le signal de décharge qui permet l'initialisation de la capacité et il remet en fonction le module PFD. Nous avons présenté en annexe D (voir figure D.7 et D.8), une configuration du module PFD pour 6 acquisitions et 10 acquisitions. Le principe de fonctionnement est identique à une acquisition, excepté le fait que nous devons faire parvenir 6 ou 10 bonnes acquisitions, selon le cas, au module PFD avant de procéder à l'échange avec le convertisseur analogique à numérique.

4.6 Sommaire

Toute conception nécessite une phase de vérification. Il existe différents degrés de vérification lors de la conception d'un circuit intégré soit la vérification fonctionnelle, la vérification spécifique et la vérification exhaustive. Dans le cadre de ce projet, nous avons effectué des vérifications fonctionnelles et spécifiques. Nous avons envisagé la vérification exhaustive, mais en raison de la complexité de la puce et du temps disponible, nous avons écarté ce degré de vérification.

Ce chapitre a décrit les étapes de vérification qui ont précédé la fabrication du circuit. Nous avons présenté les simulations effectuées. La vérification de la représentation schématique a exigé plusieurs manipulations pas toujours évidentes. Une conversion utilisant un logiciel nous a permis de créer des fichiers Verilog. À l'aide de ces fichiers, nous avons pu procéder à la vérification.

Cependant, le fait d'utiliser les fichiers Verilog à provoqué une limitation de la résolution temporelle lors des simulations logiques. En fait, l'outil de simulation

ModelSim combiné au fichier technologique fourni par le fondeur force une valeur commune pour chaque délai des cellules normalisées. Ceci fait en sorte que les résultats obtenus ne correspondent pas à ceux que nous anticipons lors du test en laboratoire. Nous avons conçu le banc d'essai de sorte qu'il puisse être générique et facilement ajustable en fonction des résultats initiaux que nous allons obtenir en laboratoire.

Les résultats de simulation illustrent une bonne fonctionnalité de tous les modules. Les exigences des spécifications de chacun d'eux ont été respectées. Les simulations ont été accomplies sur les modules d'entrées/sorties, le module de départ, le module d'insertion de délais fins, le synchroniseur, le module de détection de phases et de fréquences et finalement sur la cellule SI en entier.

CHAPITRE 5
CONCLUSION ET TRAVAIL FUTUR

5.1 Conclusion

La conception à l'échelle de la tranche est une méthode de plus en plus utilisée afin d'augmenter le niveau d'intégration. Plusieurs aspects tels que: la puissance dissipée, la tolérance aux pannes et aux défectuosités et le rendement suscitent l'intérêt des concepteurs à l'échelle de la tranche. Ce mémoire a présenté l'étude de l'intégrité des signaux à travers la conception d'un prototype de circuit de grande superficie. En raison de la taille continuellement croissante des circuits intégrés, l'analyse de l'intégrité des signaux devient un aspect essentiel pour satisfaire les exigences du marché actuel.

Nous avons, en premier lieu, étudié et analysé quelques problématiques fondamentales de l'intégrité des signaux. Les problématiques étudiées se présentent comme étant la diaphonie, la présence de longues lignes de communication, les biais de synchronisation, la synchronisation et quelques règles de conception. Cette étude permet d'avoir une idée générale des problèmes reliés à la communication numériques.

Par la suite, ce mémoire a présenté une vue d'ensemble du projet. Celui-ci présente une brève description de la technique de jonction inter-réticulaires. Cette technique permet de fabriquer un dé d'une grandeur allant jusqu'à 4 fois celle des circuits fabriqués de nos jours. Les spécifications ainsi que l'architecture concernant le projet ont également été présentées. L'architecture se devait d'être répétitive en raison d'une restriction qui découle de la technique de jonction inter-réticulaire. Le principe de base est qu'une fois le réticule conçu, 4 copies de celui-ci sont reproduites pour former le circuit intégré.

Le nombre de plots d'entrée et de sortie était un élément critique dans la réalisation de ce projet. Afin de garder toutes les fonctionnalités du circuit, nous avons réalisé une interface d'entrée. Cette interface fonctionne par le principe d'adressage direct. Chaque fonction possède ses adresses spécifiques permettant la configuration ou l'envoi de données. Le circuit réalisé possède des modules permettant de générer, individuellement, des délais fins sur chacune des lignes du bus de données. Il utilise la différence de délais associés aux entrées d'une série de portes « NON-ET ». Le délai entre les lignes est plus ou moins grand selon le chemin sélectionné. Il est également possible de synchroniser les données d'un bus de 9 bits séparément grâce à un générateur d'horloges. Le générateur d'horloges fournit au module de synchronisation 16 horloges séparées les unes des autres de 180 picosecondes. Un autre module permet la re-synchronisation des données en fonctions de l'horloge locale et de l'horloge reçue. Il faut noter que l'horloge reçue fait partie intégrante du bus de données. Également, le circuit conçu possède un détecteur de phases et de fréquences. Ce module permet de quantifier les délais générés artificiellement ou « naturellement » par le passage des données dans différentes cellules. Il est possible d'obtenir une tension proportionnelle aux délais de chacun des bits sélectionnés.

Les simulations ont été réalisées de la même méthode que la conception. Chacun des modules a été simulé afin de valider sa fonctionnalité ou de le caractériser. Les outils de simulation utilisés ont démontré très rapidement leurs limites. Les délais que nous voulions observer se situent autour de 180 picosecondes pour le générateur d'horloges et sont de moins de 500 picosecondes pour le module de détection de phases et de fréquences. Cependant, les outils utilisés lors de nos simulations, fixent des délais identiques pour tous les types de délais, tel que le temps de propagation, le temps de maintien, le positionnement à une valeur de 1 nanoseconde. Ainsi, les simulations effectuées ne peuvent démontrer hors de tout doute le bon fonctionnement relié à des

délais très fins. L'architecture utilisée pour générer les tests en laboratoire, permet de modifier facilement les paramètres et d'ajuster les délais dans le cas où cela serait nécessaire.

Le projet a été réalisé en collaboration avec Hyperchip Inc. Plusieurs événements ont fait en sorte de repousser les échéances du test, par conséquent, lors de l'écriture du présent mémoire, nous ne disposions pas de résultats de test et il n'était pas possible de poursuivre ces tests.

5.2 Travail futur

Tel que mentionné, le circuit a été encapsulé. Une carte de test a été conçue explicitement pour les tests du circuit LAIC. Le circuit LAIC peut être stimulé par un FPGA de grande taille situé sur la carte de test. Les vecteurs de tests préliminaires ont été générés à l'aide des simulations réalisées. Le code VHDL permet de faire des modifications rapides, par exemple : d'ajouter des délais ou de modifier la configuration du chemin choisi.

Une adaptation des vecteurs, réalisée par logiciel, a été produit par un autre membre de l'équipe. Ce logiciel permet d'ajuster les vecteurs en un format qui sera reconnue par la carte de test. Lors de la rédaction de ce mémoire, des essais préliminaire ont été réalisés sur la carte de tests. Les premiers essais permettaient de valider la communication entre le FPGA (situé sur la carte de tests) et le LAIC. Ces premiers tests ont été concluant. Nous sommes en mesure d'échanger des données entre le FPGA et le LAIC. La suite de ces tests fait partie du travail à réaliser.

Le circuit LAIC a démontré la faisabilité des techniques de jonction inter-réticulaire. La suite naturelle du projet serait de réaliser un circuit plus élaboré. Le futur projet ferait levier sur les connaissances acquises pour chacune des phases suivantes: les études de faisabilité, les spécifications, la conception, la réalisation, la fabrication, les simulations et les tests. En plus d'utiliser la technique de jonction inter-réticulaire, on envisage la possibilité d'interconnecter plusieurs dés standardisés disponibles en industrie. Les dés pourraient être des mémoires, des FPGA, des IP ou toutes autre forme de logique. Ces dés pourraient être assemblés sur un substrat actif de grande superficie en exploitant les connaissances acquises dans ce projet.

BIBLIOGRAPHIE

[1] FRIEDMAN, E.-G. May 2001. *«Clock Distribution Networks in Synchronous Digital Integrated Circuits. »* Proceedings of the IEEE, vol. 89, no. 5.

[2] http://www.cse.buffalo.edu/~shambhu/, University at Buffalo, historique du wafer scale integration, 11 mars 2004.

[3] http://www.icknowledge.com/, 12 mars 2004.

[4] http://www.ivorcatt.com/3ewk.htm, « *The Kernel Logic Machine. »* 11 mars 2004.

[5] http://www.knowledge-on.com/eng/default.asp, 4 février 2004.

[6] http://public.itrs.net/, « *International Technology Roadmap for Semiconductors 2002 Update.* » 2002, 194p.

[7] http://www.radiocanada.ca/actualite/decouverte/, Radio Canada, *« La téléchirurgie »*, 8 mars 2004.

[8] http://www.statcan.ca/francais/Pgdb/arts50b_f.htm, Statistique Canada, « *Ménages comptant au moins un utilisateur régulier d'Internet, selon le lieu d'accès, province. »,* 22 mars 2004.

[9] http://www.verisity.com/, « langage e », 31 janvier 2004.

[10] http://vlsi.wpi.edu/webcourse/ch04/ch04.html, *« Parasitic extraction and performance estimation from physical structure.»* Décembre 2003.

[11] JHANG, K.-S., HA, S., JHON, C.-S. 1996. *« A Crosstalk OPtimizer for Gridded Channel Routing. »* IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. pp. 424-429.

[12] LEE, H.-I., AHN, T.-W., JUNG, D.-Y., PARK, B.-H. April 2002. *« Scheme for No Dead Zone, Fast PFD Design.»* Journal of the Korean Physical Society, Vol. 40, No. 4, pp. 543-545.

[13] MANSURI, M., LIU, D., KEN YANG, C.-K. Oct. 2002. *« Fast Frequency Acquisition Phase-Frequency Detectors for GSa/s Phase-Locked Loops. »* IEEE Journal of Solid-State Circuits, Volume: 37 Issue: 10, Page(s): 1331 – 1334.

[14] OLIVIERI, M., TRIFILETTI, A. May 2001. *«An all-digital clock generator firm-core based on differential fine-tuned delay for reusable microprocessor cores.»* IEEE Iscas.

[15] PATTAVINA, J. S. August 06, 2001. *« Charge-Pump PLL Analysis. ».*

[16] RABAEY, J. M. 1996. *« Digital Integrated Circuits: A Design Perspective. »* Prentice Hall. 702p.

[17] SAVARIA, Y. 1988. *«Conception et vérification des circuits VLSI. »* Éditions de l'École Polytechnique de Montréal, 398p.

[18] TANG, Y. March 9, 2000. « *Phase-Locked Loop Based Frequency Synthesizer for Wireless Communication.* » Information Electronics Group.

[19] Wann, D., Franklin, M. Mars 1983. «*Asynchronous and clocked control structures for VLSI based interconnection networks.* » IEEE Trans. Comput., vol. C-32, pp. 284–293.

ANNEXE A

DESSIN DE MASQUE

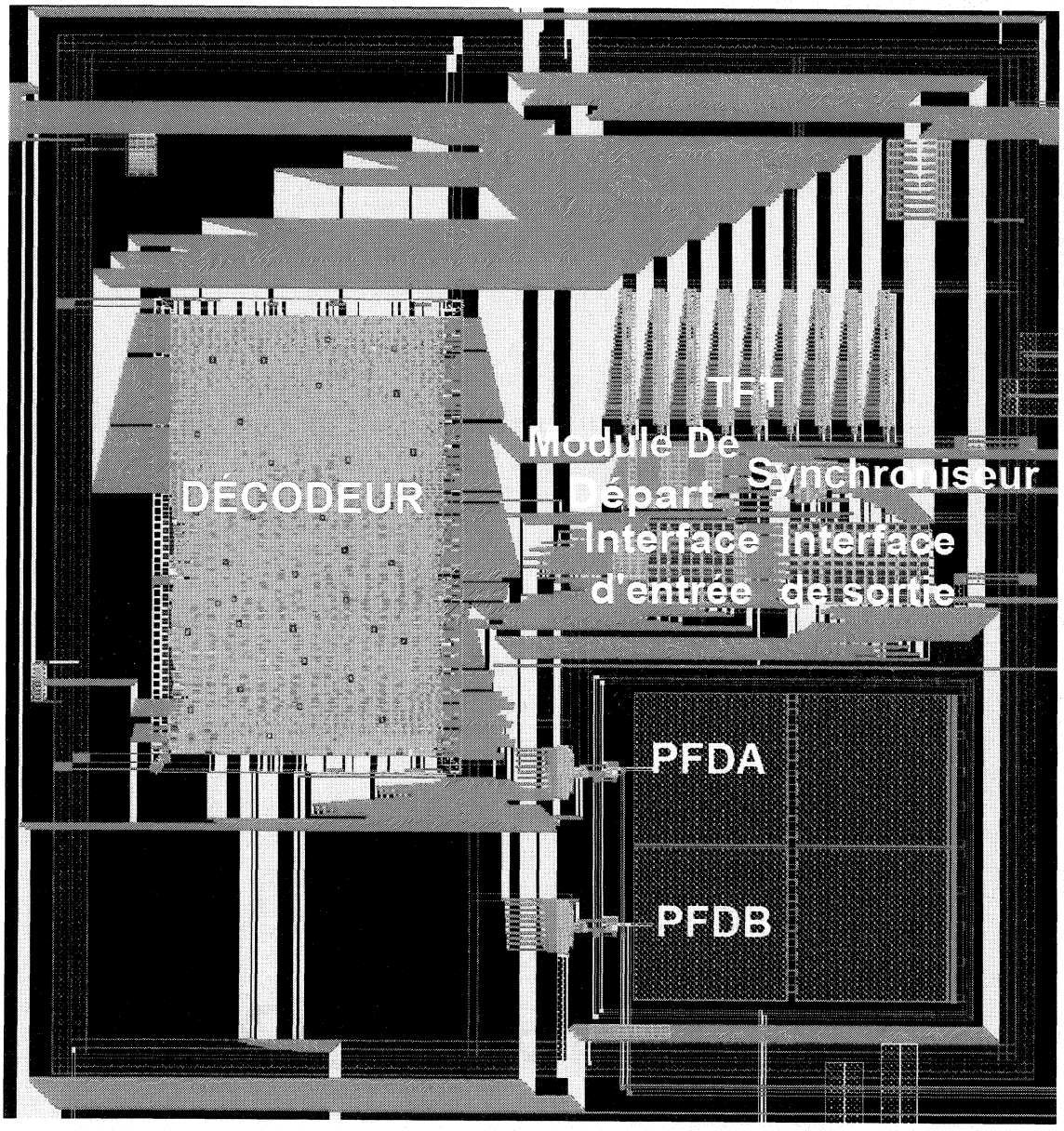

Figure A.1 : Dessin de masque de la cellule SI

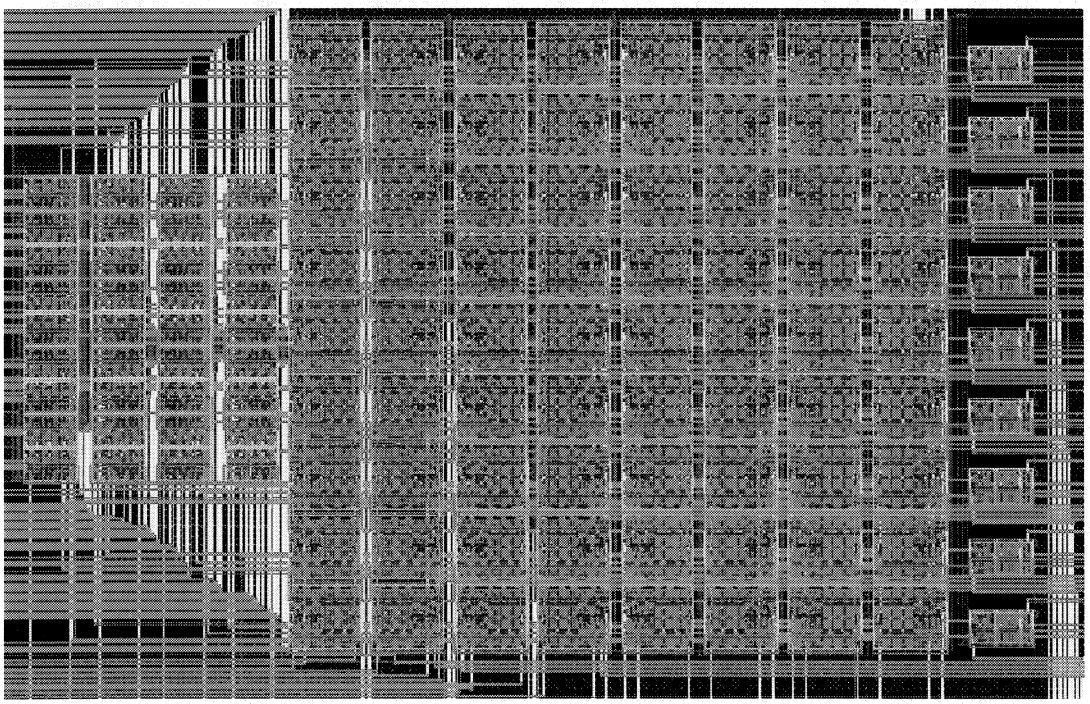

Figure A.2 : Dessin de masque de l'interface d'entrée

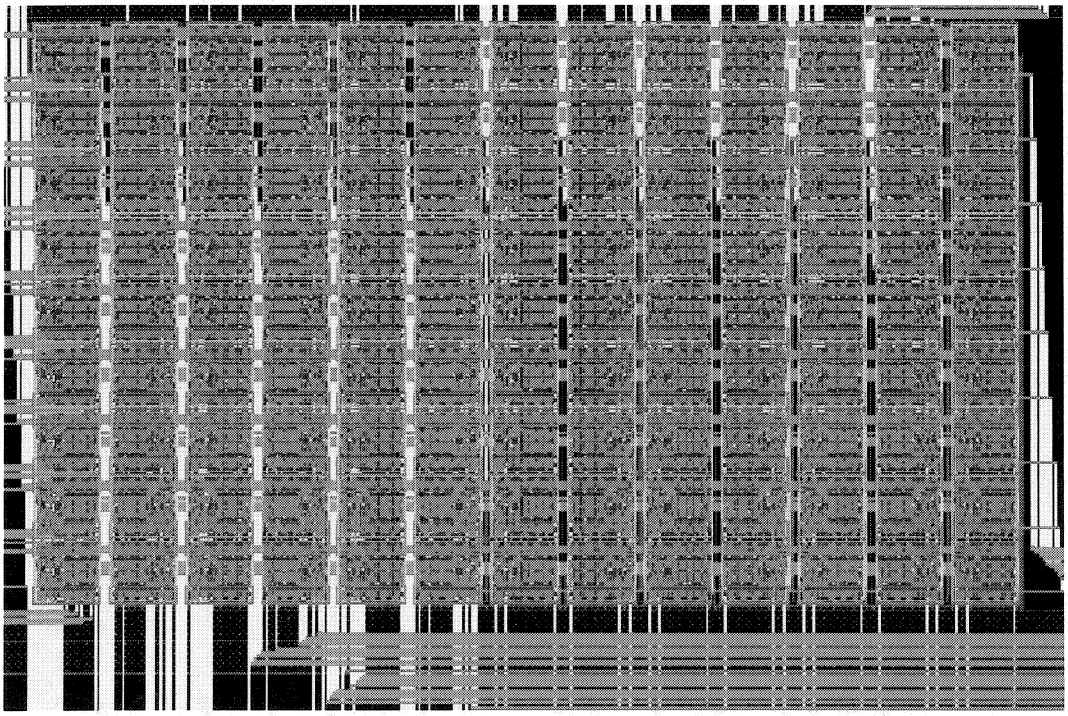

Figure A.3 : Dessin de masque de l'interface de sortie

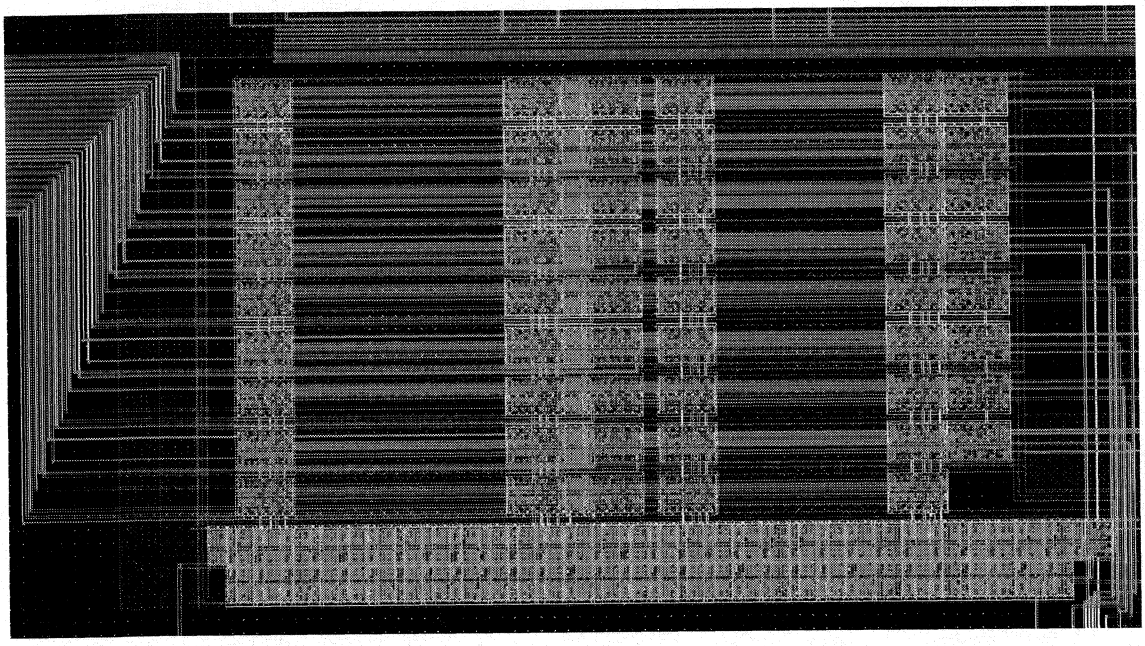

Figure A.4 : Dessin de masque du module de départ

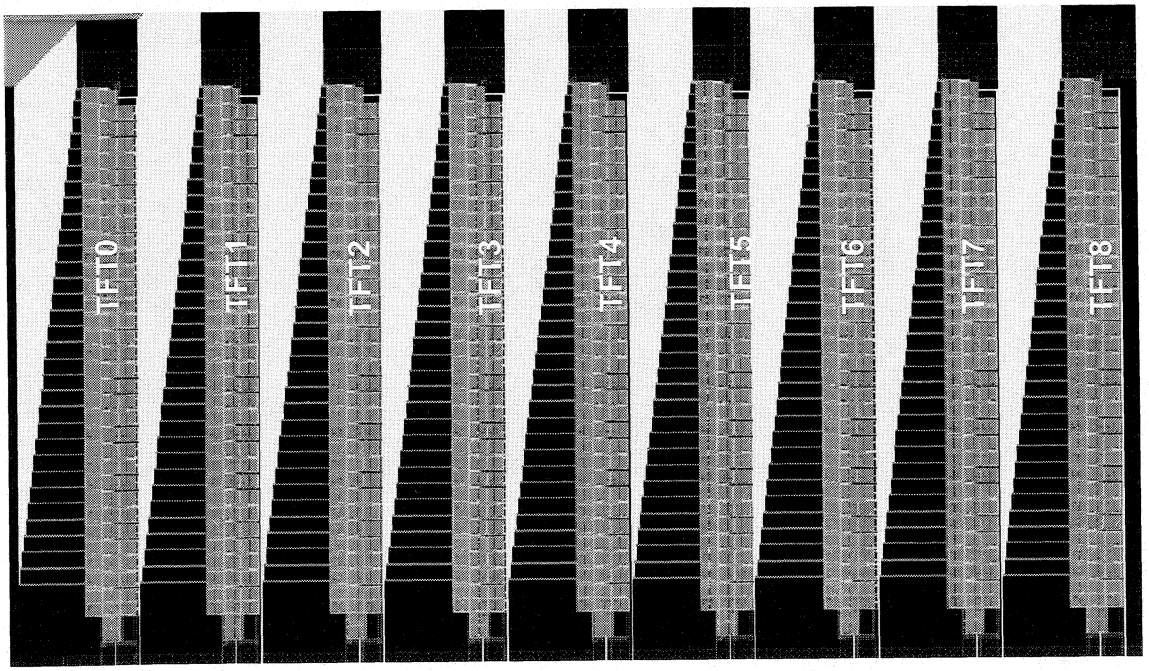

Figure A.5 : Dessin de masque des 9 modules TFT

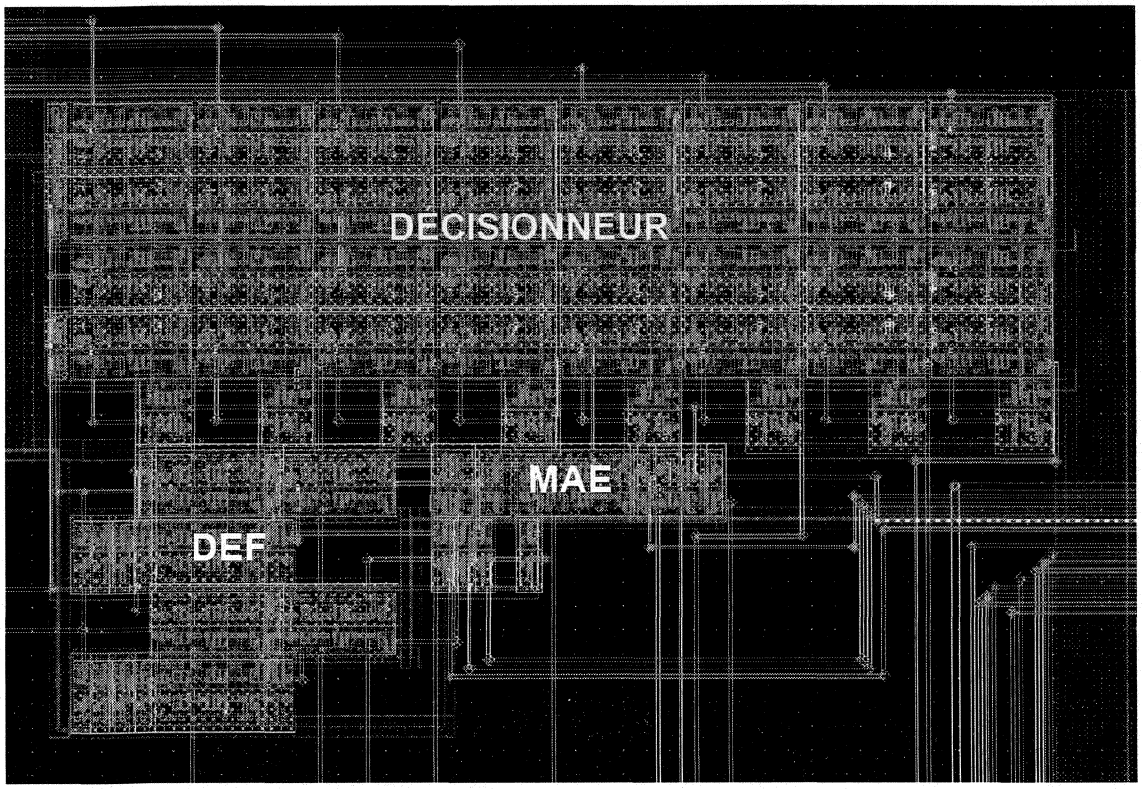

Figure A.6 : Dessin de masque du synchroniseur

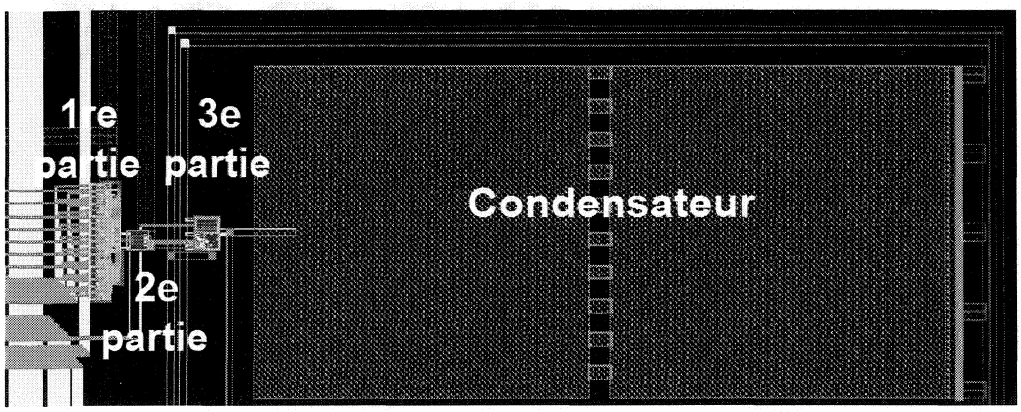

Figure A.7 : Dessin de masque du PFD

ANNEXE B

PLAGE D'ADRESSE DES REGISTRES DE LA CELLULE SI

CRT_INPUT_CELL_INTERFACE

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x00	Selection de BYPASS_3			Selection of BYPASS_2			Selection of BYPASS_1		
0x01	Selection of SYNC			Selection of DELAY			Selection of BYPASS_4		
0x02	Selection of MUX from UP (LSB)	Selection of MUX from LEFT		Selection of START			Selection of SKEW		
0x03	0*	0*	0*	0*	Selection of MUX from DOWN		Selection of MUX from RIGHT		Selection of MUX from UP (MSB)

CTR_OUTPUT_CELL_INTERFACE

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x04	Selection of A_BUS_OUT_TO_LEFT			Selection of A_BUS_OUT_TO_DOWN			Selection of A_BUS_OUT_TO_RIGHT		
0x05	Selection of B_BUS_OUT_TO_DOWN			Selection of B_BUS_OUT_TO_RIGHT			Selection of A_BUS_OUT_TO_UP		
0x06	Selection of C_BUS_OUT_TO_RIGHT			Selection of B_BUS_OUT_TO_UP			Selection of B_BUS_OUT_TO_LEFT		
0x07	Selection of C_BUS_OUT_TO_UP			Selection of C_BUS_OUT_TO_LEFT			Selection of C_BUS_OUT_TO_DOWN		
0x08	0*	PFD B counter value					Selection of EXT_OUT		

SELECT_SCLK

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x09	Selection of MUX_3 (LSB)	Selection of MUX_2				Selection of MUX_1			
0x0A	Selection of MUX_5 (LSB)		Selection of MUX_4				Selection of MUX_3		
0x0B	Selection of MUX_7			Selection of MUX_6				Selection of MUX_5 (MSB)	
0x0C	Selection of MUX_9				Selection of MUX_8				Selection of MUX_7 (MSB)

CTR_DELAYS

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x0D	Selection of CTR_TFT #2 (LSB)			Selection of CTR_TFT #1					
0x0E	Selection of CTR_TFT #3						Selection of CTR_TFT #2 (MSB)		
0x0F	Selection of CTR_TFT #5 (LSB)			Selection of CTR_TFT #4					
0x10	Selection of CTR_TFT #6						Selection of CTR_TFT #5 (MSB)		
0x11	Selection of CTR_TFT #8 (LSB)			Selection of CTR_TFT #7					
0x12	Selection of CTR_TFT #9						Selection of CTR_TFT #8 (MSB)		

DISCHARGE_VALUE

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x13	PFD A counter value					DISCHARGE VALUE COUNTER[3]			

EXT_IN

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x14	DATA FROM DATA_EXT (CAN BE ANY KIND OF DATA)								

MULTIPLE_SELECT

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x15	PFD_A COUNTER ENABLE	PFD_A COUNTER ENABLE	0*	0*	0*	0*	EXT SELECT FRONT	EXT FRONT	SELECT MUX GF

PFD_A_EN

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x16	VECTOR TO ENABLE OR DISABLE THE DATA TO BE MEASURE BY THE SKEW MODULE A								

PFD_B_EN

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x17	VECTOR TO ENABLE OR DISABLE THE DATA TO BE MEASURE BY THE SKEW MODULE B								

FRONT_A_FIRST

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x18									

FRONT_A_LAST

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x19									

FRONT_B_FIRST

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x18									

FRONT_B_LAST

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x19									

LONG_LINE

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x1C	DATA FOR THE LONG LINE								

CTR_LOOP

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x1D	MUX8	MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0

GLOBAL WRITE (Special Feature)

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x1E	0^*	0^*	0^*	0^*	0^*	0^*	0^*	0^*	0^*

GLOBAL RESET (Special Feature)

	MSB	VALEURS							LSB
Adresse	8	7	6	5	4	3	2	1	0
0x1F	0^*	0^*	0^*	0^*	0^*	0^*	0^*	0^*	0^*

NOTE:
0^* = Ce bit n'est jamais utilisé, toujours à un niveau logique bas.

ANNEXE C

TABLE DE VÉRITÉ DU CONTRÔLEUR TFT

Le code de contrôle (CTR_TFT) dont la valeur est 11111, associe le chemin le plus court. Le code de contrôle (CTR_TFT) dont la valeur est 00000, associe le chemin le plus lent.

CTR_TFT #1 to 8 du décodeur	Valeurs du bus de contrôle ctr_tft_x[31:0]																															
I6 I5 I4 I3 I2 I1	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26	Q27	Q28	Q29	Q30	Q31	Q32
0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0 0 0 0 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0 0 0 0 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0 0 0 0 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0 0 0 1 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
0 0 0 1 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
0 0 0 1 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
0 0 0 1 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1
0 0 1 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 0 1 0 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
0 0 1 0 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
0 0 1 0 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
0 0 1 1 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
0 0 1 1 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0 0 1 1 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 0 1 1 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 0 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 0 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 0 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 1 0 0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 1 0 1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 1 1 0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 0 1 1 1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 0 0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 0 0 1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 0 1 0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 0 1 1	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 1 0 0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 1 0 1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 1 1 0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0 1 1 1 1 1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1 X X X X X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Tableau C.1 : Table de vérité du contrôleur TFT

ANNEXE D

RÉSULTATS DE SIMULATIONS

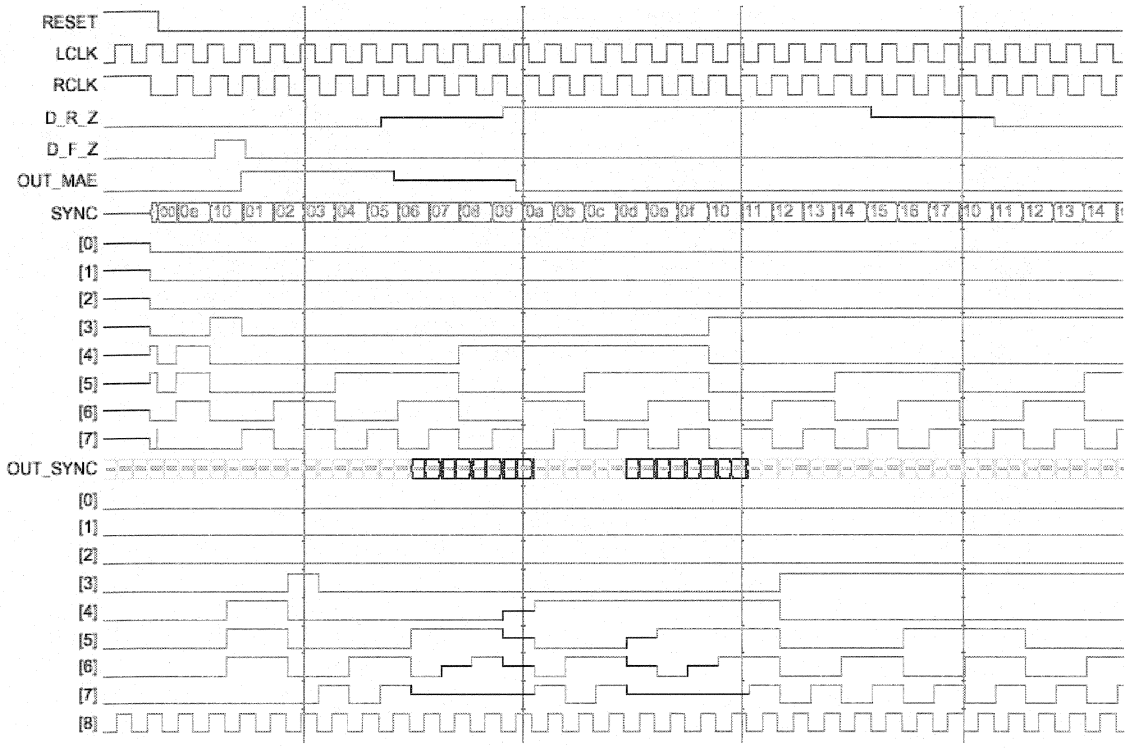

Figure D.1 : Simulation individuelle du synchroniseur, concentrée sur signal DRZ

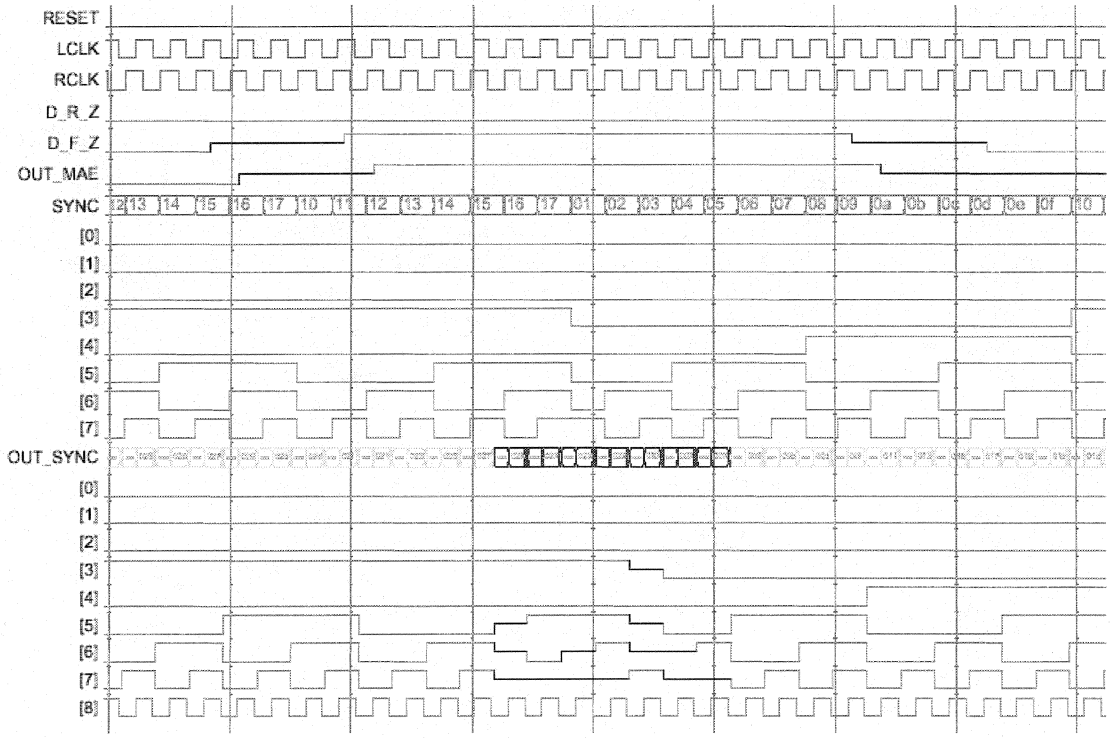

Figure D.2 : Simulation individuelle du synchroniseur, concentrée sur le signal DFZ

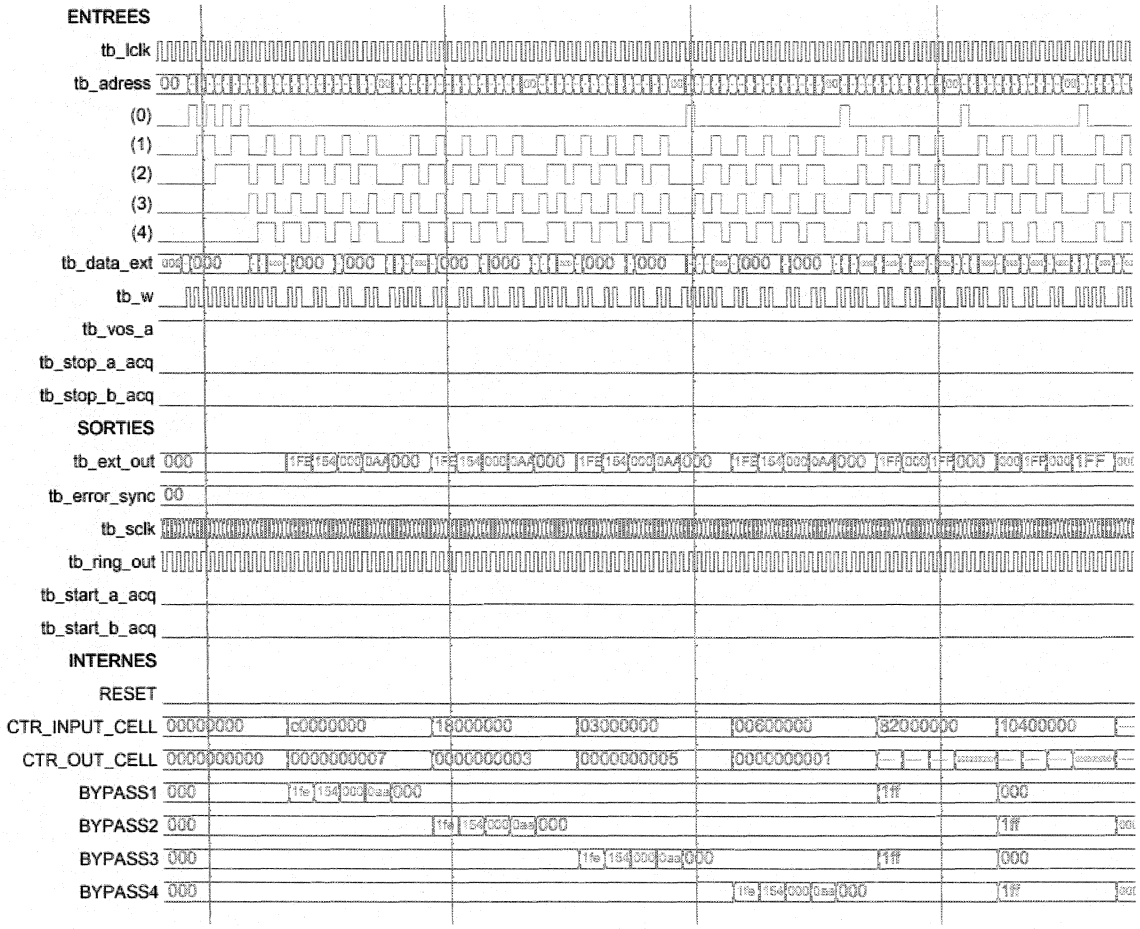

Figure D.3 : Simulation des interactions entre les interfaces d'entrée/sortie via BYPASS

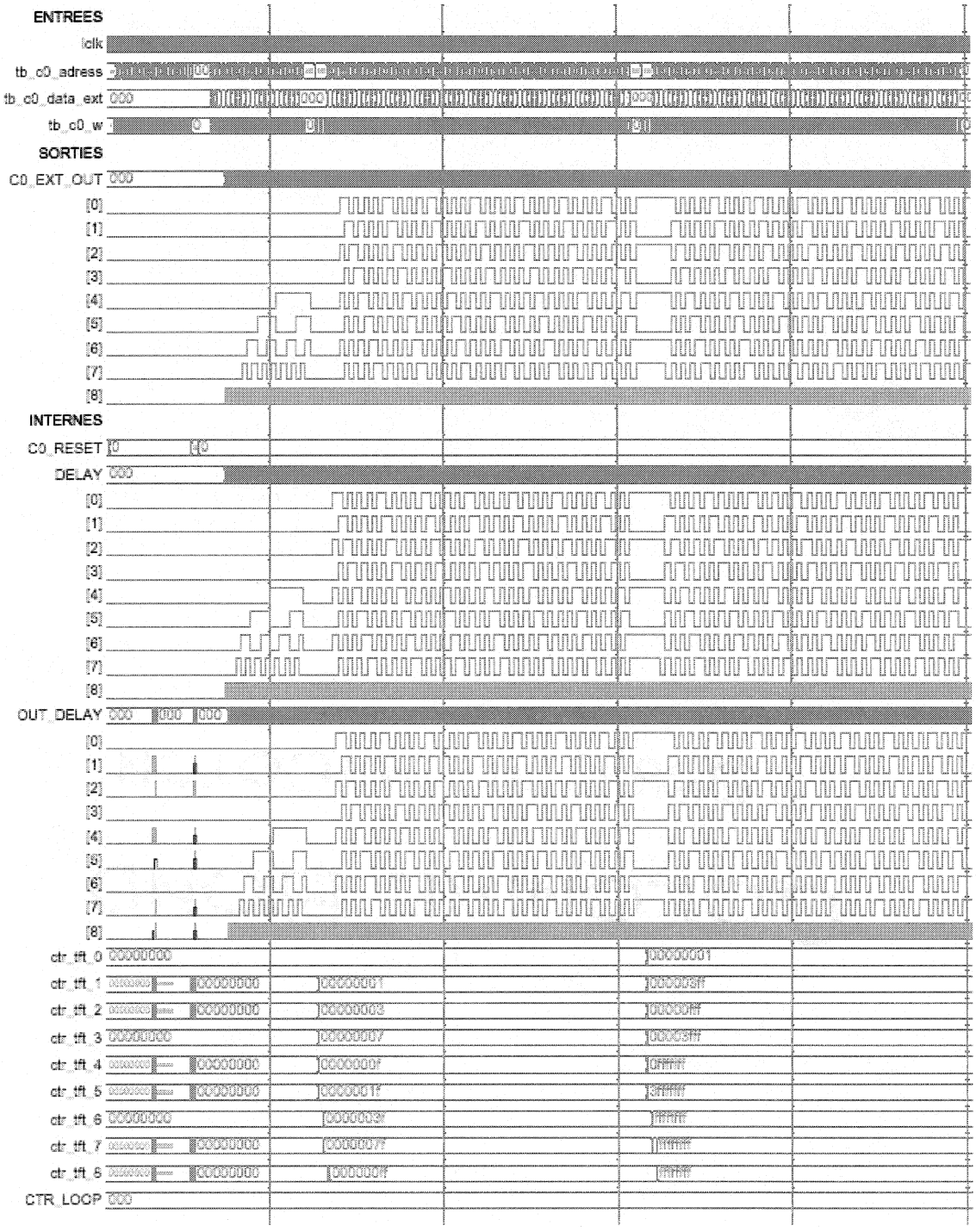

Figure D.4 : Simulation du module de délais fins

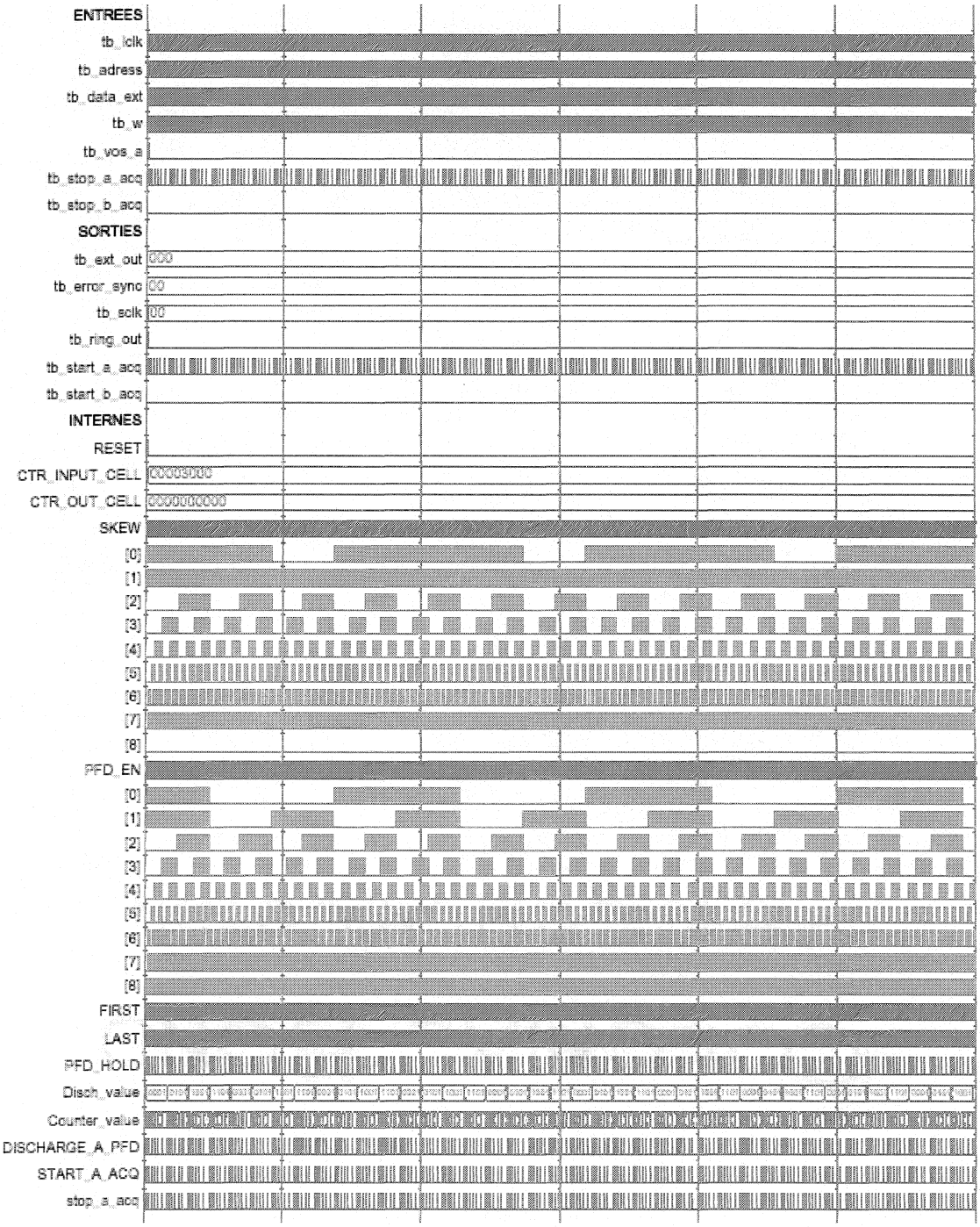

Figure D.5 : Simulation complète du module de détection de phases et de fréquences

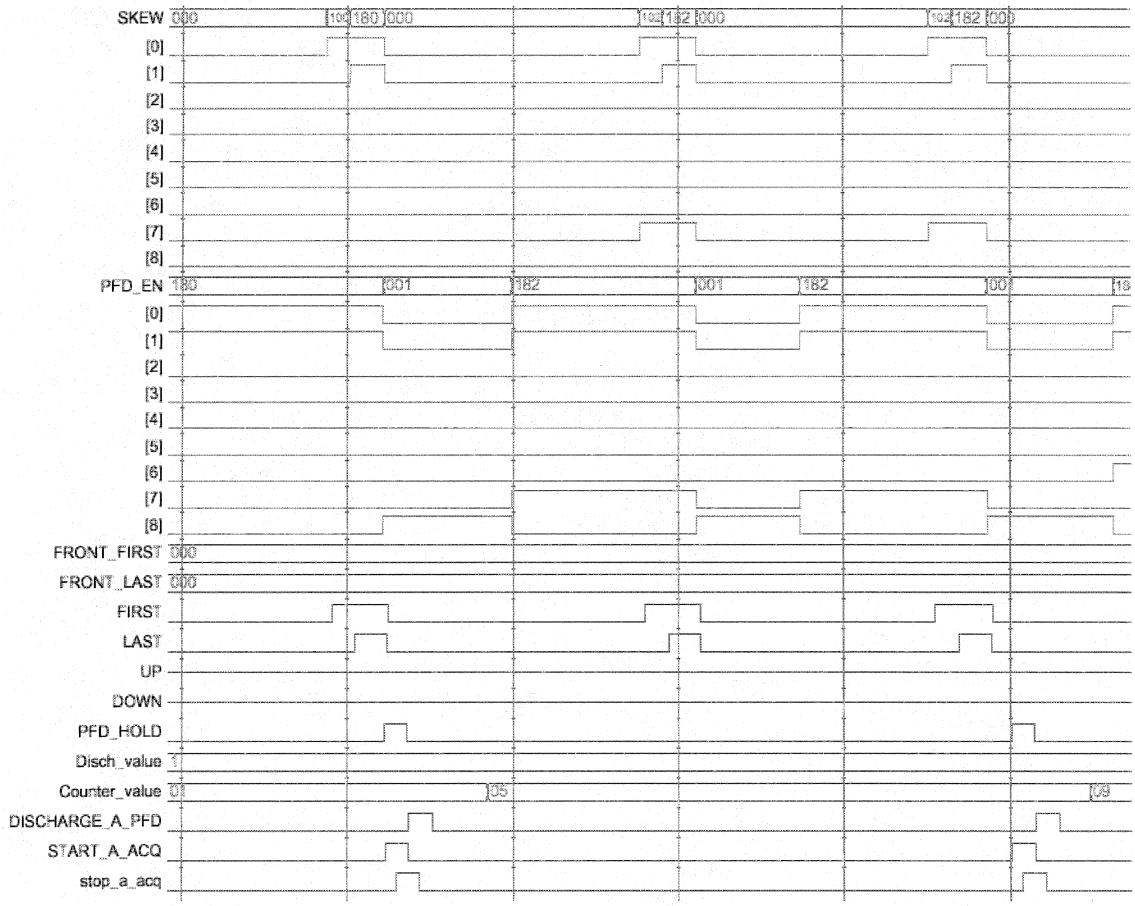

Figure D.6 : Simulation du module PFD pour 1 et 2 acquisitions.

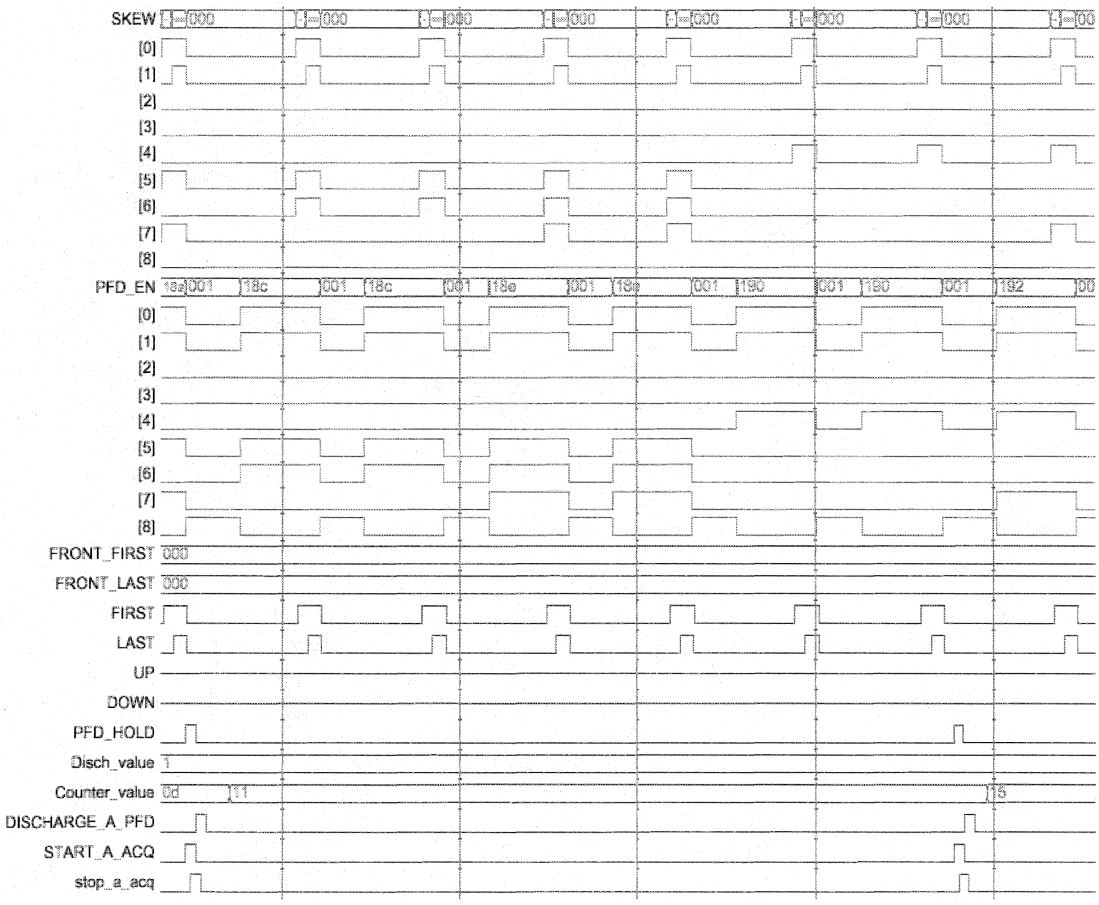

Figure D.7 : Simulation du module PFD pour 6 acquisitions

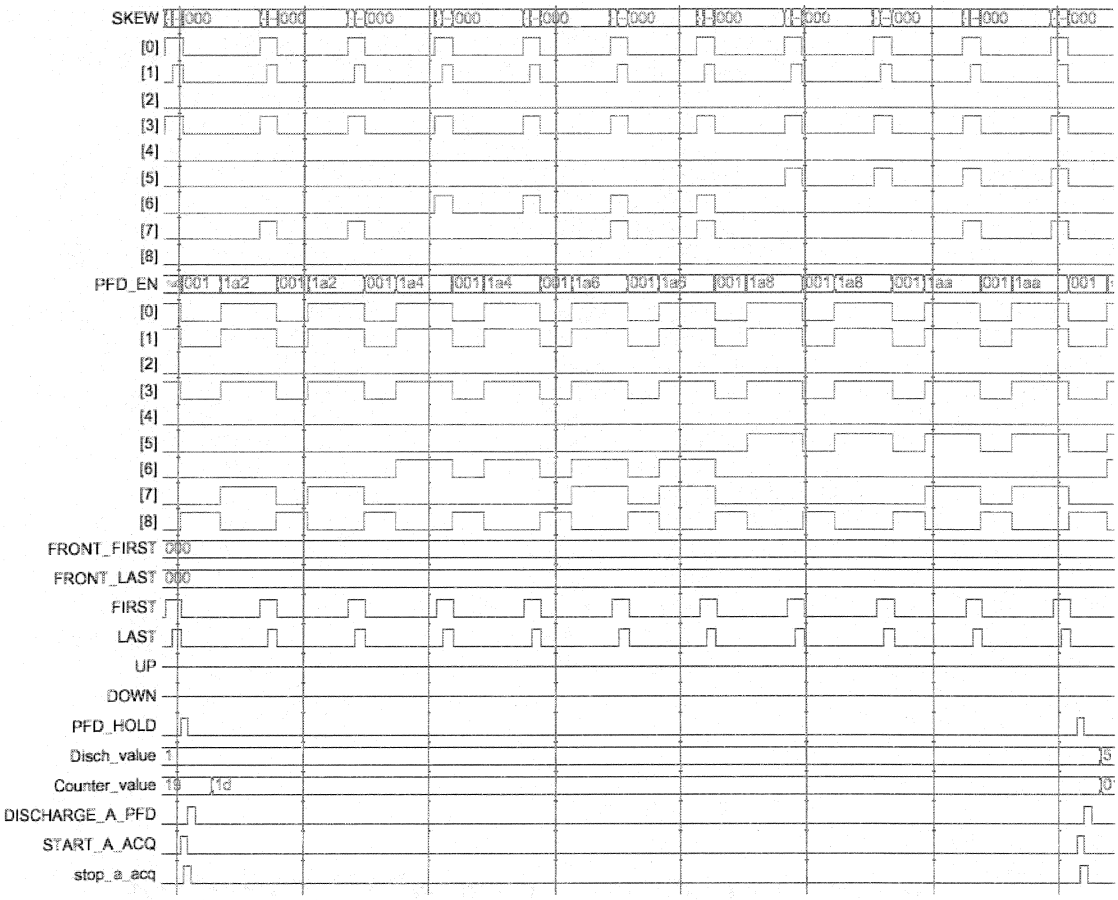

Figure D.8 : Simulation du module PFD pour 10 acquisitions