Réflexions autour de la méthodologie de vérification des circuits multi-horloges : analyse qualitative et automatisation

Mejid Kebaili

Communauté
UNIVERSITÉ Grenoble Alpes

THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITE GRENOBLE ALPES

Spécialité : **Nano Electronique et Nano Technologies (NENT)**

Arrêté ministériel : 25 mai 2016

Présentée par

Mejid KEBAILI

Thèse dirigée par **Katell MORIN-ALLORY**

préparée au sein du **Laboratoire TIMA**
dans **l'École Doctorale Electronique, Electrotechnique, Automatique et Traitement du Signal**

Réflexions autour de la méthodologie de vérification des circuits multi-horloges : analyse qualitative et automatisation

Thèse soutenue publiquement le **25 octobre 2017**,
devant le jury composé de :

M. Bruno ROUZEYRE
Professeur des Universités, Université Montpellier 2, Président

Mme Emmanuelle ENCRENAZ-TYPHENE
Maître de Conférences, Université Pierre et Marie Curie, Rapporteur

M. Sébastien PILLEMENT
Professeur des Universités, Université de Nantes, Rapporteur

M. Régis LEVEUGLE
Professeur des Universités, Grenoble INP, Membre

M. Jean-Christophe BRIGNONE
Ingénieur, STMicroelectronics, Co-encadrant

Mme Katell MORIN-ALLORY
Maître de Conférences, Grenoble INP, Directrice de thèse

Remerciements

Avant toute chose, je tiens à témoigner toute ma reconnaissance à ma directrice de recherche, Madame Katell Morin-Allory pour m'avoir suivi, guidé et encouragé pendant ces trois années ainsi que pour son implication permanente durant la rédaction de ce mémoire.

Je tiens également à exprimer toute ma gratitude à Monsieur Jean-Christophe Brignone, mon encadrant de thèse en entreprise, pour m'avoir fait confiance pendant cinq ans tout en me délégant un certain nombre de responsabilités.

Je tiens, bien évidemment, à remercier Madame Emmanuelle Encrenaz Typhène et Messieurs Sébastien Pillement, Régis Leveugle et Bruno Rouzeyre pour avoir accepté d'être membres de mon jury, ainsi que pour leurs retours ayant permis d'améliorer la qualité de ce manuscrit.

Je tiens, par ailleurs, à exprimer mon estime la plus sincère aux fournisseurs d'outils qui ont accepté de mettre en place une collaboration et de me supporter durant les travaux d'évaluation. Je pense notamment à Messieurs Jean-Philippe Binois et Guillaume Plassan de Synopsys, Ayari Abdelouahab de Mentor Graphics ainsi que Raik Brinkmann, Sven Beyer et Bruno Geldreich de Onespin.

Je tiens aussi à témoigner toute mon affection à celles et ceux sans qui cette thèse ne serait pas ce qu'elle est. Je pense en particulier à Mesdames Dominique Borrione et Irène Serre, qui m'ont fait l'honneur de travailler à mes côtés, ainsi qu'aux différentes personnes ayant vérifié les CDC dans la division CPU-GPU : Eric Charlet, Lionel Picandet, Amaury Breme, Hugo Roynette et Fabio Fernandez.

Je tiens, en outre, à faire une dédicace spéciale à toutes les merveilleuses personnes cottoyées au quotidien et grace à qui l'environnement professionnel est devenu une seconde maison. Je pense en particulier à Mesdames Carmela D'Arca et Valérie Vidal-Henry ainsi qu'à Messieurs Marc Beaujoin, Laurent Boudou, Patrick Cochennec, Gabriel Duffy, Sébastien Ferroussat, Laurent Figadère, Thierry Karger, Stéphane Martin, François Oswald et Xavier Robert.

Pour finir je tiens à dédier ce manuscrit à Mesdames Colette Aurières et Iéva Dogramadjian sans qui je n'en serait pas là aujourd'hui.

Résumé

Depuis plusieurs années, le marché des circuits intégrés numériques requiert des systèmes de plus en plus complexes dans un temps toujours plus réduit. Afin de répondre à ses deux exigences, les industriels de la conception font appel à des fournisseurs externes proposant des circuits fonctionnant sur des signaux d'horloge dédiés. Lorsque ces derniers communiquent entre eux, les horloges d'emission et de réception ne sont pas les mêmes, on parle de « Clock Domain Crossing » (CDC).

Les CDC correspondent à des communications asynchrones et peuvent provoquer des dysfonctionnements critiques. Par ailleurs, ces problèmes étant intermittents et complexes à analyser, ils ne peuvent pas être exhaustivement vérifiés avec des méthodes telles que l'analyse de timing ou la simulation fonctionnelle. Avec l'augmentation du nombre de CDC dans les circuits, les industriels de la conception assistée par ordinateur (EDA) ont proposé des solutions logicielles spécialisées dans la vérification statique des CDC.

Cependant, les circuits développés étant en constante évolution, les outils ne sont pas en mesure de s'adapter. Pour pallier ces problèmes, la vérification industrielle des CDC est basée sur la spécification de contraintes et d'exclusions par l'utilisateur. Ces actions, qui se substituent aux outils, peuvent masquer des bugs. De plus, l'effort humain requis par cette approche n'est pas compatible avec le temps alloué au développement de circuits industriels.

Nous avons donc cherché à automatiser la vérification en proposant des solutions basées sur des propriétés formelles. Les travaux ont consisté à analyser les différentes techniques de conception et de vérification des CDC à travers l'évaluation des principaux outils du marché. A partir des résultats obtenus, nous avons formalisé les problèmes pratiques et proposé des modèles permettant d'obtenir des résultats exhaustifs automatiquement. Les essais ont été réalisés sur un sous-système à base de processeurs (CPUSS) développé chez STMicroelectronics.

L'adoption de nos modèles permet une vérification complète des CPUSS de manière automatique ce qui est essentiel dans un environnement industriel compétitif. En effet, le nombre d'informations devant être spécifiées par l'utilisateur a été réduit de moitié pour chacun des outils évalués. Par ailleurs, ces travaux ont montré que l'axe de développement des outils CDC avec l'ajout de fonctionnalités telles que les flots hiérarchiques ou l'injection de fautes n'améliore pas la qualité de résultats. Une collaboration ayant été mise en place avec les principaux fournisseurs outils, certaines solutions seront probablement intégrées aux outils dans les années à venir.

Abstract

For several years now, the digital IC market has been requiring both more complex systems and reduced production times. In this context, the semiconductor chip maker companies call on external IP providers offering components working on dedicated clock signals. When these IPs communicate between them, the source and destination clocks are not the same, we talk about "Clock Domain Crossing" (CDC).

CDC correspond to asynchronous communications and can cause critical failures. Furthermore, due to the complexity and the random nature of CDC issues, they can not be exhaustively checked with methods such as timing analysis or functional simulation. With the increase of CDC in the digital designs, EDA tools providers have developed software solutions dedicated to CDC static verification.

Whereas, the designs are subject to continuous change, the verification tools are not able to be up to date. To resolve these practical issues, the CDC industrial verification is based on the specification of constraints and exclusions by the user. This manual flow, which replaces the tools, can mask bugs. Moreover, the human effort required by this approach is incompatible with the time allowed to industrial designs development.

Our goal has been to automate the verification submitting solutions based on formal properties. The work consisted in the analysis of the different CDC design and verification approaches through the evaluation of main CDC checker tools. From the results obtained, we have formalized the practical problems and proposed models to obtain automatically exhaustive results. The tests have been performed on a processor-based subsystem (CPUSS) developed at STMicroelectronics.

Adopting our models enables a complete checking of CPUSS in an automatic way, which is essential within a competitive industrial environment. Actually, the amount of information to be specified by the user has been reduced by half for each one of the evaluated tools. Otherwise, this work has shown that the development axis of the CDC tools despite the addition of functionalities such as hierarchical flows or fault injection, doesn't improve the quality of results (QoR). Since a collaboration has been established with the main tool providers some solutions would probably be included into the tools over the coming years.

Table des matières

Introduction

Depuis plusieurs années, le développement des circuits intégrés numériques se concentre sur l'optimisation des performances et la réduction de la consommation électrique : les systèmes sur puce (SoC) se complexifient et intègrent de plus en plus de composants (IP). En parallèle, le temps de mise sur le marché (Time-to-Market) imposé aux concepteurs de circuits se réduit. Afin de satisfaire ces deux exigences, les industriels de la conception font appel à des fournisseurs externes [80]. Ces derniers proposent des composants fonctionnant sur des signaux d'horloge dédiés. Au niveau du système, lorsque les composants communiquent, les horloges ne sont pas les mêmes, on parle de traversée de domaines d'horloges ou de "Clock Domain Crossing" (CDC).

Les CDC correspondent à des communications asynchrones et peuvent générer des dysfonctionnements critiques [84]. Il existe différentes manières pour assurer un transfert de données correct entre des composants cadencés sur des horloges distinctes, la plus connue est l'utilisation de structures de synchronisation. Cependant, ces techniques sont complexes à mettre en place, leur comportement doit donc être validé. Avec l'augmentation du nombre de CDC dans les circuits, les industriels de la conception assistée par ordinateur (EDA) ont proposé des solutions logicielles spécialisées dans la vérification statique des CDC. La technique de vérification s'appuie sur la reconnaissance d'un modèle de synchonisation et la validation de propriétés formelles associées au modèle reconnu.

Malgré une volonté d'automatiser la vérification des CDC, les outils présentent, en pratique, des limitations majeures. Ces problèmes ne sont pas dus aux algorithmes de détection et de validation mais au désalignement entre ce qui est vérifié par les outils et ce qui est implémenté dans les circuits. Les SoC étant en constante évolution, les techniques de synchronisation le sont aussi et les outils ne sont pas en mesure de s'adapter : certaines structures ne sont pas reconnues. Pour éviter d'être dépendants des circuits à vérifier, les fournisseurs d'outils ont proposé des modèles génériques mais cette approche provoque la détection de fausses structures. Afin de faire un compromis entre des modèles trop spécifiques et des modèles trop génériques, les solutions logicielles ont intégré des jeux de paramètres permettant de configurer les modèles à détecter en fonction des circuits à vérifier.

La vérification des CDC à l'aide d'outils de vérification statique ne requiert donc pas uniquement une expertise en conception de communications asynchrones. La personne en charge de l'analyse des CDC doit avoir une connaissance approfondie

des solutions logicielles utilisées afin de pouvoir les configurer en fonction du circuit à vérifier et déterminer si les informations rapportées sont correctes ou sont dues à une limitation outil. Pour pallier ces problèmes pratiques tout en réduisant l'effort humain, la vérification industrielle des CDC est généralement basée sur la spécification des structures de synchronisation et l'exclusion des faux problèmes par l'utilisateur [26, 78]. Cependant, ces actions se substituent aux outils en termes d'analyse : les cas spécifiés corrects par l'utilisateur ne sont pas analysés par les outils. Cette approche, qui présente un risque d'erreur humaine, peut donc masquer des bugs jusqu'à leur apparition sur silicium.

Dans ce contexte industriel où la vérification des CDC consiste à analyser les circuits quasi-manuellement pour éviter les limitations outils, nous avons développé une méthodologie basée sur l'utilisation de contraintes. Cette mission, d'une durée de dix-huit mois, a été effectuée dans le cadre d'un Diplôme de Recherche et d'Innovation (DRI) [48]. Ces travaux préalables ont consisté à spécifier les principaux problèmes pratiques rencontrés avec un outil de vérification statique, à définir la procédure d'analyse devant être adoptée pour éviter de masquer des problèmes et à proposer des solutions pour obtenir des résultats conformes aux spécifications techniques [50]. Cependant, bien que cette approche permette de réduire le temps dédié à la vérification des CDC tout en assurant une couverture complète des problèmes et une qualité de résultats optimale, elle est basée sur des interventions manuelles. Par ailleurs, la compréhension de cette méthode requiert une connaissance des approches d'analyse implémentées dans les outils de vérification et, malgré la présence de plusieurs solutions sur le marché, très peu d'informations sont publiées sur le sujet.

À partir de ce constat, nous avons souhaité automatiser la méthodologie en proposant des solutions basées sur des propriétés formelles. Une collaboration a donc été mise en place, à travers un contrat de thèse CIFRE, entre STMicroelectronics et le laboratoire TIMA. Par ailleurs, les travaux étant dépendants des outils de vérification, nous avons dû proposer aux industriels de l'EDA un accord pour pouvoir évaluer leurs solutions logicielles et leur proposer des solutions à implémenter.

Ce manuscrit va donc débuter par un préambule décrivant le contexte de la thèse à savoir les différents types de communication considérés comme des CDC, le circuit de STMicroelectronics utilisé pour les travaux pratiques, et la place de la vérification des CDC dans le flot de conception.

Ensuite, afin de définir la problématique associée à la vérification statique des CDC et d'orienter nos travaux, un état de l'art sur les techniques de conception et une analyse des différentes approches de vérification des CDC seront présentés dans une première partie. Des résultats d'évaluation des outils du marché et de l'effort humain requis pour obtenir une qualité de résultats maximale seront étudiés.

Dans une seconde partie, nous proposerons des modèles permettant d'automatiser la vérification des CDC dans un contexte industriel de plus en plus compétitif. Un comparatif des résultats obtenus avec et sans les solutions proposées sera exposé.

Préambule : contexte, conventions et définitions

Le sujet de ce manuscrit est l'analyse et l'optimisation des methodes de vérification des traversées de domaines d'horloges ou "Clock Domain Crossing" (CDC). Les tests sont réalisés sur un sous-système à base de processeurs développé chez STMicroelectronics. Cette partie est dédiée à la définition des éléments du contexte :

— La place des CDC dans les communications entre horloges ;

— Les sous-systèmes à base de processeurs de STMicroelectronics ;

— La place de la vérification des CDC dans le flot de conception.

1 Les communications entre horloges

Une communication correcte entre deux composants nécessite la stabilité des données transférées lors de leur capture par la destination [27]. Généralement, l'ordre d'apparition des phases d'émission et de réception des données est réalisé à l'aide d'une référence temporelle appelée signal d'horloge. Ce signal périodique permet de cadencer le transfert des données.

1.1 Les différents types de communication

L'implémentation des communications entre des composants dépend de la relation entre les signaux horloges sur lesquelles ils sont synchronisés. Dès 1990, Messerschmitt propose une classification des communications en deux catégories principales (synchrone et quasi-synchrone) et étudie les méthodes de conception adaptées [72]. Avec le développement des circuits au début des années 2000, un troisième type a fait son apparition : les communications asynchrones [6, 27, 102].

Communication synchrone Si le signal d'horloge est le même entre la source et la destination, on parle de communication synchrone. L'horloge arrive en même temps sur les différents composants. Le temps de propagation des données n'étant pas nul, la réception ne peut pas se faire au même cycle d'horloge que l'émission. Les phases d'envoi et de capture sont donc exclusives et la stabilité des données est assurée.

Communication asynchrone Si la source et la destination sont cadencées par des signaux d'horloge différents, on parle de communication asynchrone. Les deux

références de temps n'ayant pas de relation entre elles, l'ordre d'apparition des phases d'émission et de réception est aléatoire. Dès lors, des dysfonctionnements tels que la perte de données ou la propagation de données incohérentes peuvent apparaître.

Communication quasi-synchrone Si des signaux d'horloge différents sont connectés à une source commune, on parle de communication quasi-synchrone (ou librement synchrone). La stabilité des données peut être obtenue en assurant que les temps d'arrivée des horloges, sur les composants dédiés à l'émission et à la réception, sont identiques. On parle alors d'horloges alignées en phase, c'est-à-dire que leurs fronts respectifs arrivent au même instant sur les composants source et destination.

La figure 1 présente les différents types de communication entre horloges : synchrone 1a, asynchrone 1b et quasi-synchrone 1c. Elle montre le transfert d'un signal entre deux registres dont les signaux d'horloge respectifs peuvent être identiques (Clk) ou différents (Clk_S et Clk_D). Lorsqu'ils sont différents, la figure présente le cas d'une transition montante sur la donnée transférée qui n'est pas capturée par la destination. La communication quasi-synchrone est illustrée avec deux horloges contrôlées par des signaux fonctionnels indépendants. Dans ce manuscrit, toutes les figures représentant des types de circuit sont réalisées selon les conventions définies dans le tableau 1.

1.2 Les "Clock Domain Crossing"

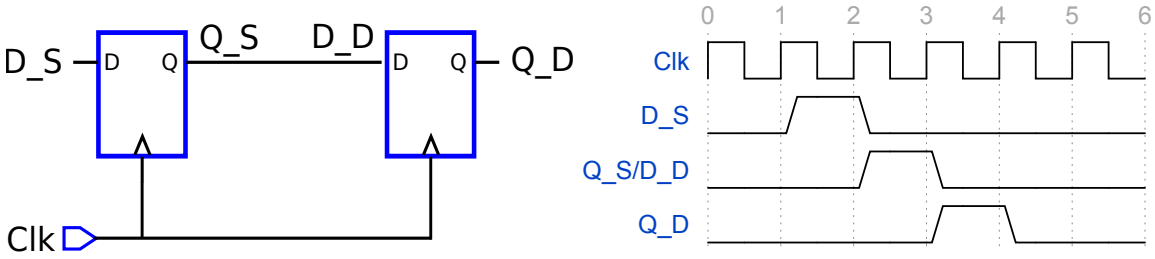
La théorie des domaines d'horloges Des horloges alignées en phase appartiennent à un même domaine (Clock Domain). Les communications entre horloges d'un même domaine sont synchrones. À l'inverse, des horloges déphasées sont réparties dans des domaines différents [27]. Quand des données traversent des domaines d'horloge, on parle alors de « Clock Domain Crossing » (CDC). Les CDC correspondent à des communications asynchrones.

Les gestion des interfaces quasi-synchrones Pour éviter de gérer tous les CDC de manière asynchrone, les horloges connectées à une source commune peuvent être alignées en phase. Ce travail d'alignement, réalisé par les ingénieurs en charge de l'implémentation physique, consiste à définir et implémenter des délais sur les arbres d'horloge. Les horloges appartiennent alors au même domaine et la communication entre les registres est synchrone. L'étape de vérification temporelle statique (STA) va ensuite vérifier la stabilité des données lors de leur capture par la destination. On peut distinguer trois types de communication entre horloges déphasées [27] pouvant être gérés de façon synchrone :

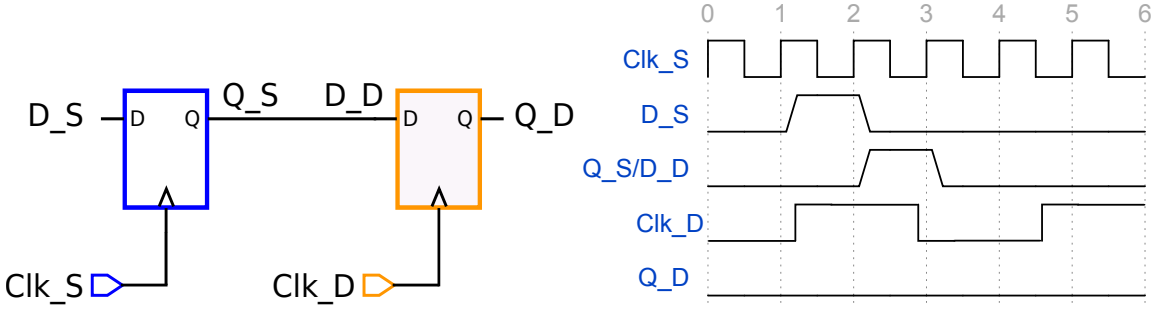
— Mésochrone : même fréquence et déphasage constant (traversée de logique combinatoire) ;

— Périodique : fréquences différentes et déphasage périodique (traversée de diviseurs d'horloge) ;

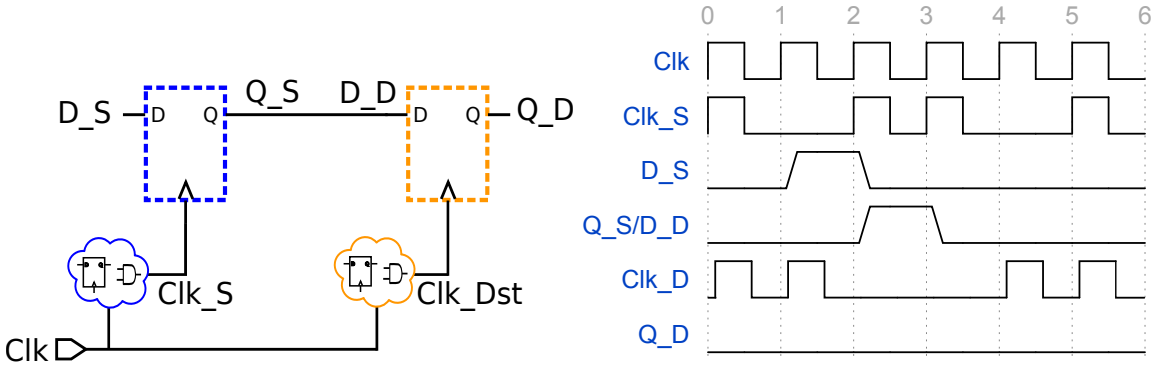
— Plésiochrone : presque la même fréquence donc déphasage variable mais faible (sorties de PLL).



(a) Communication synchrone



(b) Communication asynchrone



(c) Communication quasi-synchrone

FIGURE 1 – Les types de communication

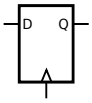
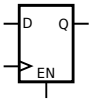
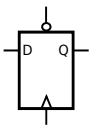
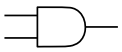
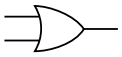
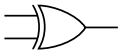
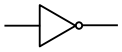
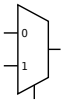
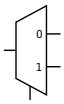
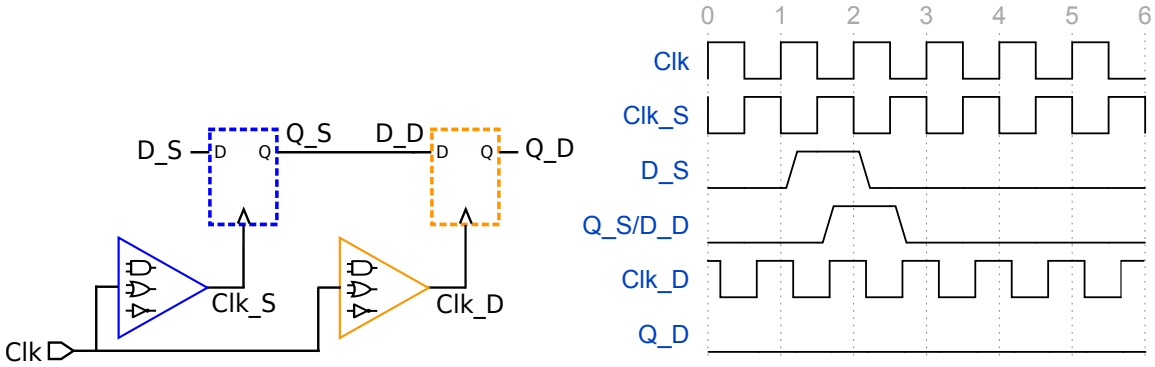
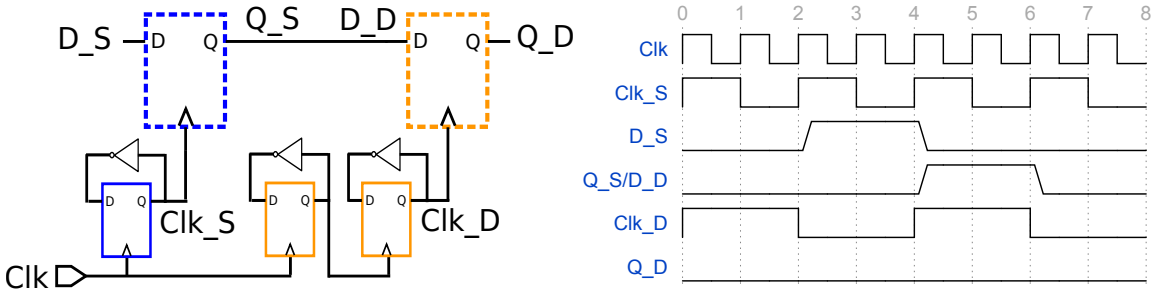
Type	Objet	Convention
Couleur	Domaine d'horloge source	Bleu
	Domaine d'horloge destination	Orange
	Problème	Rouge
	Solution	Vert
Nom	Donnée d'entrée de registre	D
	Horloge d'entrée de registre	Clk
	Reset d'entrée de registre	Rst
	Contrôle d'entrée de registre	En
	Donnée de sortie de registre	Q
	Sortie de logique combinatoire	C
	Domaine d'horloge source	Suffixe "_S"
	Domaine d'horloge destination	Suffixe "_D"
Symbole	Verrou D	
	Bascule D	
	Bascule D avec contrôle	
	Bascule D avec reset	
	Porte ET	
	Porte OU	
	Porte OU exclusif	
	Inverseur	
	Port d'entrée	
	Port de sortie	
	Multiplexeur	
	Démultiplexeur	
	Bloc combinatoire	
	Bloc combinatoire et séquentiel	

TABLE 1 – Conventions d'écriture des figures

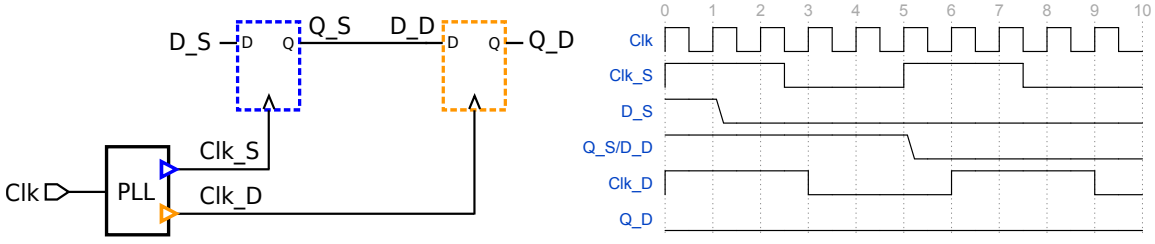
La figure 2 présente les différentes communications pouvant être gérées par un alignement des arbres d'horloge. Chaque cas est illustré par le transfert d'un signal entre deux registres dont les horloges (Clk_S et Clk_D) ont une source commune (Clk). L'interface mésochrone (2a) est représentée avec des blocs de logique combinatoire différents dans les cônes d'entrée respectifs des horloges source (Clk_S) et destination (Clk_D). La communication périodique (2b) est illustrée par le fait que Clk traverse des diviseurs d'horloge différents entre le registre d'envoi et celui de réception. Le cas plésiochrone (2c) présente deux horloges en sortie d'une même PLL.



(a) Communication mésochrone



(b) Communication périodique



(c) Communication plésiochrone

FIGURE 2 – Les communications quasi-synchrones

Les compromis pratiques Dans l'industrie, un circuit peut contenir plusieurs dizaines d'horloges différentes. La spécification des délais d'horloge doit faire l'objet d'une attention particulière dans la mesure où cette approche présente un coût de surface et de consommation électrique. À l'inverse, considérer les interfaces quasi-synchrones comme asynchrone ne présente qu'un coût lié au temps de conception de structures de synchronisation. Dès lors, il arrive que certaines de ces communications ne soient pas gérées à travers un alignement des phases des signaux d'horloges : elles

sont alors considérées comme des CDC. La vérification des communications quasi-synchrones peut être réalisée soit par la STA (synchrone) soit par la vérification CDC (asynchrones) suivant les cas. La figure 3 présente les trois types de communication entre horloges (synchrone, asynchrone et quasi-synchrone) et s'ils sont analysées par la STA ou par la vérification CDC.

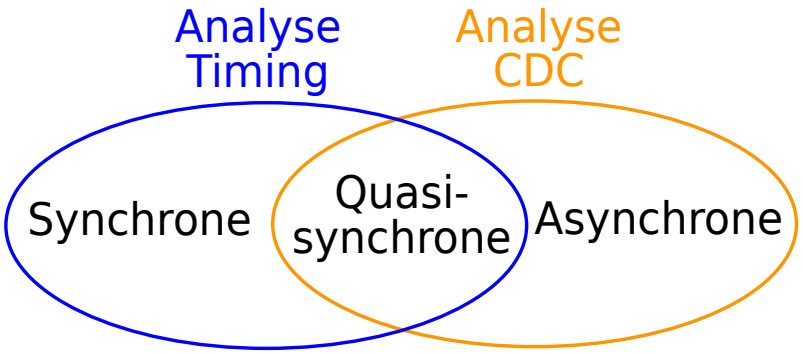


FIGURE 3 – Type de vérification en fonction de la communication

Les communications purement asynchrones Les communications sans horloge ou "purement asynchrones" ne seront pas étudiées. Ces dernières utilisent des protocoles de type « poignée de main » (handshake) pour le séquençage et des porte C (ou cellules de muller) pour la mémorisation [41] : elles ne font pas partie du spectre des CDC.

2 Description du circuit de ST : le CPUSS

Les circuits à base de processeurs sont présents dans de nombreuses applications telles que l'automobile, les objets connectés ou le spatial. Chez STMicroelectronics, les équipes sont réparties en fonction du type d'application. Afin d'optimiser le développement des circuits, une équipe est dédiée à la conception de sous-systèmes génériques à base de processeurs (CPUSS). Ces composants sont ensuite fournis à tous les groupes chargés du développement de SoC intégrant des processeurs.

2.1 L'architecture des sous-systèmes

Les composants Les CPUSS sont basés sur des processeurs haute performance provenant de fournisseurs externes. La particularité des sous-systèmes est qu'ils permettent de faire fonctionner les processeurs avec des performances maximales tout en limitant la consommation d'énergie. Ils intègrent des composants de mesure des performances, de la consommation électrique et des caractéristiques du circuit. Le travail de conception réside dans le développement de blocs de contrôle gérant les différents composants en fonction des besoins.

L'architecture Pour automatiser l'implémentation de structures dédiées à la réduction de la consommation d'énergie, les sous-systèmes sont découpés par domaines de tension. Un domaine de tension correspond à la liste des composants ayant une

alimentation commune. L'alimentation peut prendre différents niveaux, par exemple 0.8V ou 1.1V. Par ailleurs, afin de délivrer des circuits pouvant s'intégrer facilement, les ports d'entrée/sortie sont définis dans le domaine de tension du client (SoC). En considérant qu'il y a au moins un domaine d'horloge par domaine de tension, l'architecture des CPUSS est Globalement Asynchrone et Localement Synchrone (GALS) [92, 102].

Le circuit choisi Les circuits CPUSS sont donc composés de ports primaires, d'un processeur, de composants de mesure et de blocs de contrôle répartis dans des domaines de tension respectifs. Parmi les différents sous-systèmes CPUSS, le circuit utilisé pour les travaux de thèse présente les caractéristiques techniques suivantes [92] :

- Architecture GALS ;
- 4 domaines de tension ;
- 1 CPU multi-coeurs ;
- 4 blocs de contrôle ;
- 2 composants dédiés aux mesures ;
- 7 interfaces avec l'extérieur.

2.2 Les interfaces asynchrones

Les CDC Les circuits de type CPUSS intègrent plus de 100 domaines d'horloge dont une dizaine est sujette à CDC et plus de 150000 chemins asynchrones répartis en une vingtaine d'interfaces. Afin d'empêcher l'apparition d'éventuels dysfonctionnements au niveau des CDC, des structures de synchronisation sont utilisées. Par ailleurs, les sous-systèmes intègrent des particularités comme les communications asynchrones contrôlées à l'extérieur du circuit.

Les structures sur-mesure Chaque interface contrôlée à l'intérieur du circuit est composée de plusieurs techniques de synchronisation en fonction de la taille des données transférées et des performances requises. Du fait des spécificités des sous-systèmes, certaines techniques de synchronisation sont modifiées [90, 93]. Les changements dépendent des caractéristiques de l'interface asynchrone et peuvent être classés en deux catégories :

- Les interfaces entre deux domaine de tensions : les structures varient en fonction du niveau d'alimentation ;
- Les interfaces asynchrones avec le processeur : les structures sont modifiées afin d'assurer des performances maximales.

Le circuit choisi En matière de CDC, le sous-système choisi présente les spécificités suivantes :

- 12 domaines d'horloge sujets à CDC ;
- 24 interfaces asynchrones ;
- 14 interfaces synchronisées sur-mesure ;
- 5 interfaces synchronisées à l'extérieur.

La figure 4 présente un sous-système de type CPUSS à haut niveau. Les quatre domaines de tension sont délimités par des pointillés de couleur (noir, bleu, rouge et vert). À chaque bloc correspond une couleur représentant respectivement un domaine d'horloge. Les interfaces asynchrones sont symbolisées par des flèches. Les CDC synchronisés par des techniques sur-mesure sont mis en évidence par des flèches de couleur rouge. Les interfaces contrôlées à l'extérieur ne sont pas représentées.

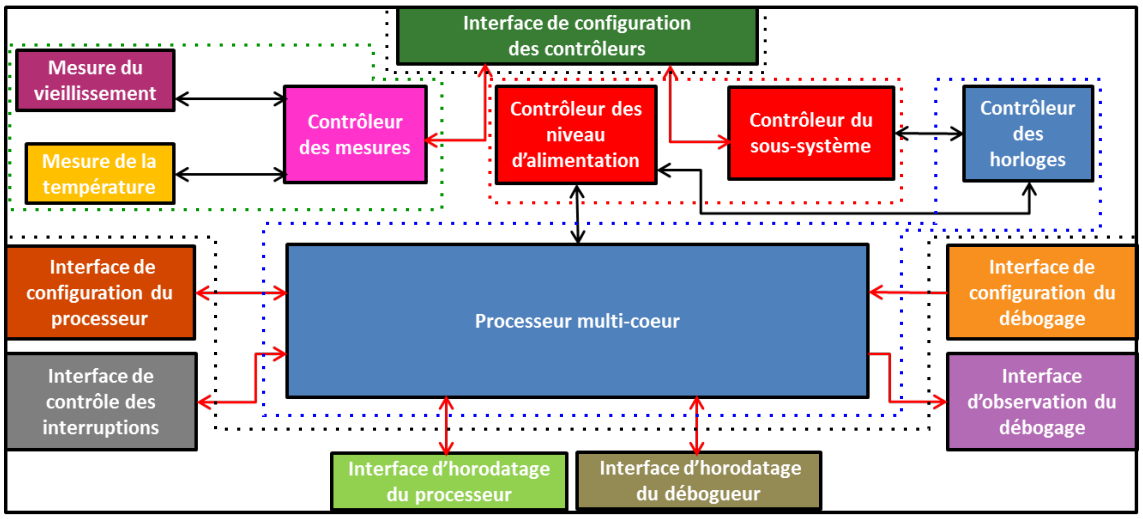


FIGURE 4 – La CDC au seins des circuits CPUSS

3 La vérification CDC dans le flot de conception

La conception d'un circuit intégré numérique démarre avec la mise en place des spécifications techniques et se termine par l'envoi des masques en usine de fabrication des plaquettes de silicium. Le développement se fait en plusieurs étapes correspondant à différents niveaux de modélisation du circuit. Chaque étape est découpée en deux parties : la transformation du modèle de circuit et, en parallèle, la validation. L'ensemble des étapes est le flot de conception des circuits intégrés.

3.1 Le flot de conception

La conception Comme illustré dans la figure 5, la conception des circuits intégrés démarre avec la modélisation et la configuration des composants au niveau transfert entre registres (RTL), à partir des spécifications techniques. Il n'y a pas de notion de délai, les seules références de temps sont les signaux d'horloge. Le circuit est ensuite modélisé au niveau porte logique (GATE), contenant les temps de traversée des différents composants. La dernière étape consiste à placer et router tous les composants, on parle de modélisation au niveau implémentation physique (IMPL). À partir de cette description, il est possible de générer les masques (LAYOUT).

La validation Les modélisations du circuit doivent être conformes aux spécifications techniques. Pour s'en assurer, et éviter toute erreur de conception, des étapes de validation sont effectuées. De même, après l'étape de synthèse d'un niveau de

modélisation à un autre, une preuve d'équivalence structurelle entre les deux est réalisée. Pour chaque niveau de modélisation, des versions du circuit sont produites tant que la validation rapporte des problèmes. À titre d'exemple, lors du développement de sous-systèmes à base de CPU, une nouvelle version du circuit est générée chaque semaine.

La figure 5 présente le flot de conception des circuits intégrés numériques avec la place de la vérification CDC à chaque étape. Les différents niveaux de modélisation sont représentés par des rectangles, et les étapes de validation par des losanges.

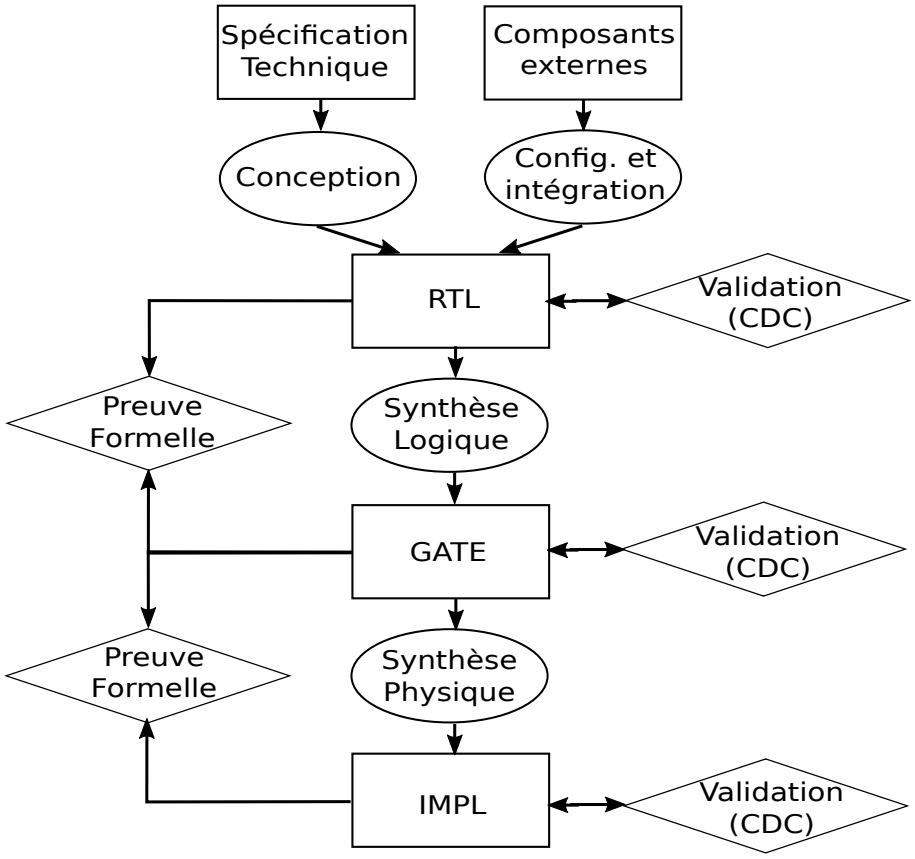


FIGURE 5 – La vérification CDC au sein du flot de conception

3.2 La vérification des CDC

Les CDC et les niveaux de modélisation La vérification des CDC fait partie des étapes de validation. Elle s'effectue sur les trois niveaux de modélisation du circuit : RTL, GATE et IMPL. Au niveau RTL, les temps de traversée des composants n'étant pas encore définis, le pire cas doit être envisagé. Les communications quasi-synchrones sont alors considérées comme asynchrones. Aux niveaux GATE et IMPL, la définition des domaines d'horloge doit être redéfinie à partir des informations provenant des personnes en charge de l'implémentation physique des arbres d'horloge.

Le flot de vérification horizontal des CDC En théorie, une vérification des CDC doit être effectuée sur chaque version du circuit quel que soit le niveau de modélisation. Pour un sous-système CPUSS, la vérification CDC prend entre six et douze semaines. Pour assurer une implémentation correcte des interfaces asynchrones, la vérification CDC doit être effectuée sur, au moins, six versions majeures du circuit (deux pour chaque niveau de modélisation). Cependant, une analyse complète au niveau IMPL ne peut être réalisée avant la mise en fabrication du circuit. La vérification complète n'est réalisée que sur le circuit RTL validé fonctionnellement. Les vérifications aux niveaux GATE et IMPL se limitent respectivement à un test de non-régression par rapport aux modélisation RTL et GATE : c'est le flot de vérification horizontal des CDC [91].

La figure 6 présente le flot de vérification horizontal des CDC appliqué chez ST-Microelectronics avec les deux types d'analyse : la vérification complète au niveau RTL et les tests de non-régression aux niveaux GATE et IMPL. Au niveau RTL, la vérification CDC doit assurer qu'aucun problème de communication ne peut survenir. Au niveau GATE et IMPL, il faut que les résultats d'analyse soient équivalents à ceux du RTL, sinon le circuit doit être corrigé.

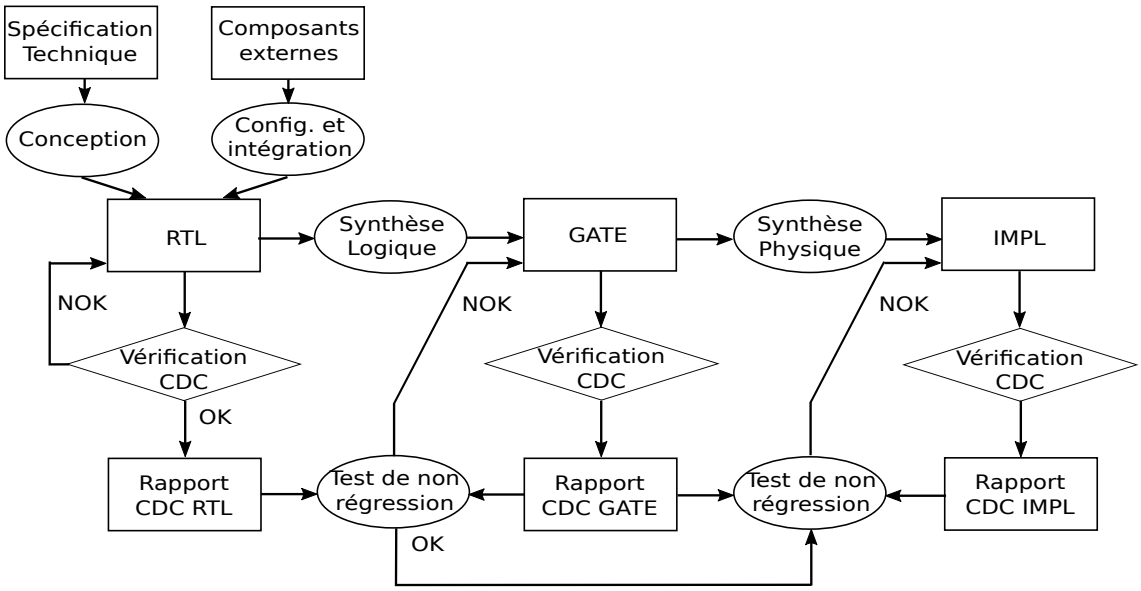


FIGURE 6 – Le flot horizontal de vérification des CDC

4 Conclusion

Les CDC correspondent à des communications asynchrones entre horloges. Contrairement aux communications synchrones, le transfert de données entre domaines d'horloge peut provoquer des dysfonctionnements. La vérification des CDC doit donc être réalisée à chaque étape du flot de conception des circuits intégrés numériques. Cependant, en pratique, la validation exhaustive des CDC n'est réalisée qu'au niveau RTL. Dans ce manuscrit, les travaux présentés seront effectués uniquement au niveau RTL.

Le circuit choisi pour illustrer les problématiques et les solutions proposées est un CPUSS développé chez STMicroelectronics. Ce sous-système est représentatif des circuits industriels car il intègre plusieurs spécificités pour ce qui est des interfaces asynchrones. Le nombre de CDC est conséquent, les structures associées sont spécifiques et, du fait de l'architecture GALS, la vérification ne peut être réalisée qu'à haut niveau, c'est-à-dire sur le circuit complet.

Première partie

Analyse qualitative des techniques de vérification CDC

Chapitre 1

Les problèmes CDC

Les dysfonctionnements relatifs aux communications asynchrones sont dus au déphasage entre l'horloge de l'émetteur et celle du récepteur. Ces problèmes peuvent être groupés en trois catégories :

— Les problèmes dus au temps de propagation des données ;

— Les problèmes dus à la différence de fréquences entre les horloges ;

— Les problèmes dus au fonctionnement des éléments séquentiels.

1 Le temps de propagation des données

Lors d'une communication synchrone, l'horloge arrive en même temps sur les composants d'émission et de réception. Le délai entre l'envoi et la capture des données est d'un cycle d'horloge. Dans le cas d'un CDC, les temps d'arrivée des horloges émettrice et réceptrice sont indépendants, on parle d'horloges déphasées. Le temps entre les phases d'envoi et de capture n'étant pas défini, les données réceptionnées peuvent être différentes de celles émises.

La capture de données incohérentes Lors du transfert de plusieurs signaux en parallèle (bus), entre deux domaines d'horloge, il n'est pas possible de prévoir l'instant d'arrivée de l'horloge de destination. De plus, le temps de traversée entre l'émetteur et le récepteur varie d'un chemin à l'autre. Cette différence de délais entre les canaux peut provoquer des décalages au moment de la capture des signaux. Bien que ce phénomène ne soit que temporaire, les données réceptionnées ne correspondent pas à celles envoyées : on parle de capture de données incohérentes.

La figure 1.1 présente ce problème avec le transfert d'un bus de deux bits (Q_S[1 :0]) dont les états changent au même cycle de l'horloge source (Clk_S) et sont capturés (D_D[1 :0]) à des cycles différents de l'horloge destination (Clk_D).

La capture de courses de signaux Les temps de propagation entre l'émetteur et le récepteur sont encore plus problématiques lorsque les données sont recombinées à travers des éléments de logique combinatoire. La traversée de portes logiques introduisant des délais supplémentaires, des courses de signaux peuvent survenir lorsque les données transférées changent d'état. Des valeurs transitoires (glitch) sont alors

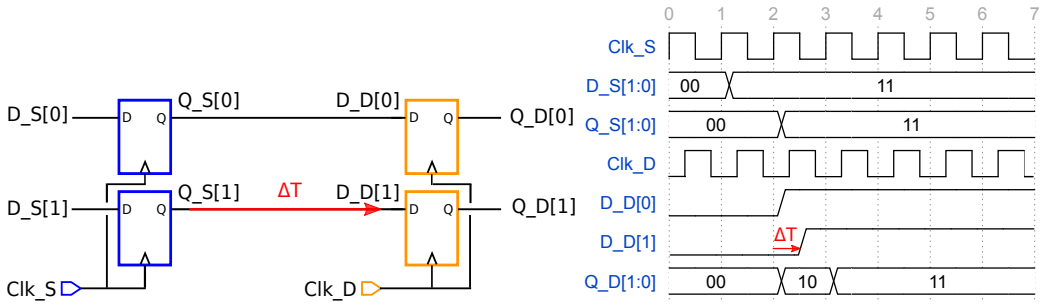


FIGURE 1.1 – La capture de données incohérentes

générées en sortie de la logique combinatoire et peuvent être capturées par la destination. Dans les systèmes synchrones, le temps de traversée des portes logiques présentes entre deux éléments séquentiels est inférieur à un cycle d'horloge (idéalement) pour éviter la capture d'états transitoires.

La figure 1.2 illustre ce phénomène à travers une course de signaux (C_S[0] et C_S[1]) au niveau d'une porte « OU Exclusif », générant un état transitoire en sortie (D_D) lors de l'arrivée du front d'horloge de destination (Clk_D). L'état transitoire est alors capturé et propagé (Q_D) dans le domaine de destination.

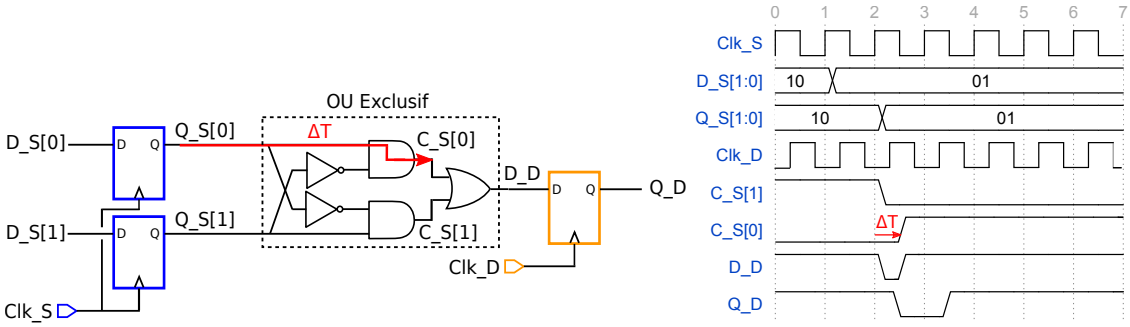


FIGURE 1.2 – La capture de courses de signaux

2 La différence de fréquence entre les horloges

Lors d'un CDC, les horloges source et destination peuvent, en plus d'être déphasées, avoir des fréquences d'arrivée différentes. Dès lors, les données transférées peuvent être corrompues. Cette problématique n'est pas spécifique aux CDC, les communications quasi-synchrones périodiques y sont aussi exposées.

L'augmentation du risque de capture d'états transitoires Le transfert d'un bus de données d'un domaine d'horloge lent vers un rapide (Slow-to-Fast Crossing) augmente le risque de capture de données incohérentes et de courses de signaux. Si l'horloge de destination est plus rapide que l'horloge source, les états transitoires peuvent être capturés pendant plusieurs cycles d'horloge de destination.

La figure 1.3 présente ce problème avec le transfert d'un bus de deux bits (Q_S[1 :0]) dont les états changent au même cycle de l'horloge source (Clk_S) et sont capturés (D_D[1 :0]) à des cycles différents de l'horloge destination (Clk_D).

Le registre de réception capture des données incohérentes pendant deux cycles d'horloge de destination.

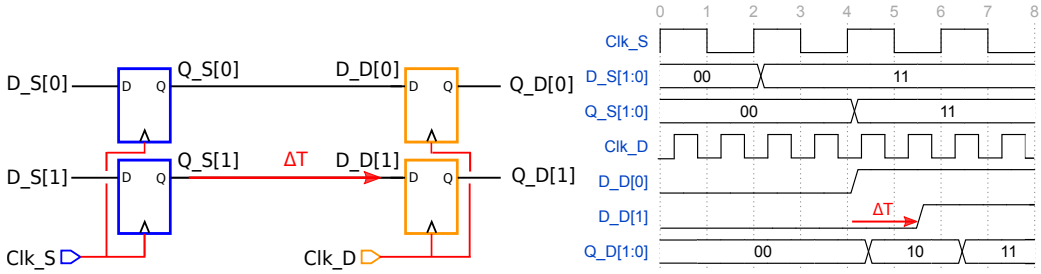


FIGURE 1.3 – Le transfert d'une horloge lente vers une horloge rapide

La perte de données Dans le cas d'un CDC allant d'un domaine rapide vers un domaine lent (Fast-to-Slow Crossing), le signal transféré peut changer d'état plusieurs fois pendant un cycle d'horloge de destination. Les données transférées étant binaires, si elles changent un nombre de fois multiple de deux avant leur capture, des informations sont perdues. Ce problème est indépendant du déphasage, si deux horloges sont alignées (communication synchrone) mais que la source est au moins deux fois plus rapide que la destination, il peut survenir.

La figure 1.4 illustre ce phénomène par un signal (Q_S/D_D) maintenu à '1' pendant un cycle de l'horloge source (Clk_S) et qui n'est pas capturé par l'horloge destination (Clk_D).

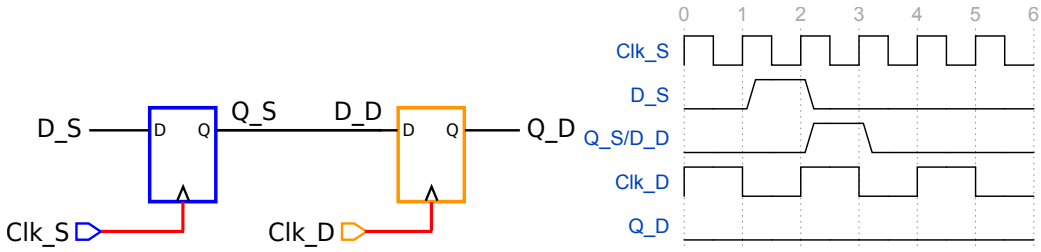


FIGURE 1.4 – La perte de données

3 Le fonctionnement des éléments séquentiels

Les éléments séquentiels sont des composants permettant la synchronisation des données d'entrée sur une horloge. Pour assurer un fonctionnement correct, il faut que le signal d'entrée reste stable pendant une certaine durée autour du front d'horloge de capture. La période en amont est le temps de stabilisation (setup), et celle en aval est le temps de maintien (hold) [4,14,25,32,34,58,71,107]. Un dysfonctionnement des éléments séquentiels peut causer un bug critique dans le cadre de la communication entre horloges.

3.1 Etude du fonctionnement des éléments séquentiels

Le fonctionnement d'une bascule D Une bascule D (flip-flop) est composée de deux verrous D (latch) afin de propager la donnée d'entrée uniquement sur un front

d'horloge. Le premier verrou propage la valeur du signal d'entrée lorsque l'horloge est inactive. Dès que l'horloge est active, il bloque et mémorise la valeur d'entrée. Le second verrou fait la même chose mais pour des états d'horloge inversés. Une donnée est propagée en sortie de la bascule quand l'horloge est active. Aucune nouvelle donnée n'est transmise tant que l'horloge n'est pas désactivée, la sortie de la bascule conserve la précédente valeur.

La figure 1.5 montre les représentations d'une bascule D à différents niveaux : transfert entre registres, verrou D, porte logique et symbolique.

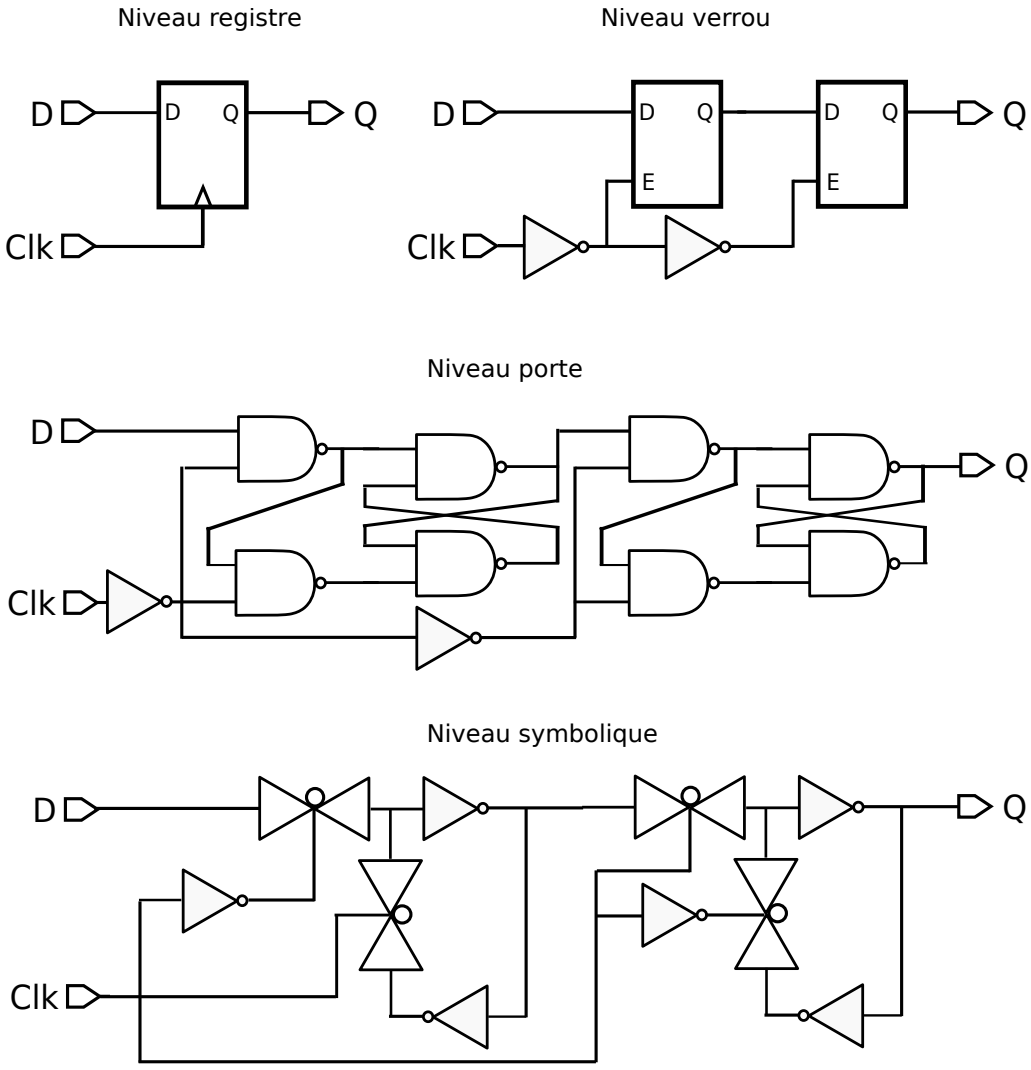


FIGURE 1.5 – La bascule D

Les éléments bistables Pour chaque verrou, la mémorisation est faite avec une boucle composée d'inverseurs couplés. Lorsque la boucle est fermée (blocage et mémorisation), le circuit se gère de manière autonome. Sa sortie ne peut pas être contrôlée et il existe deux états de stabilité possibles (0 et 1), on parle d'élément bistable. Cependant le passage d'un état à l'autre nécessite de l'énergie qui, si elle n'est pas suffisante, génère un état de sortie intermédiaire, comme décrit en 1975 par Hurtado et Elliott [40]. Ce phénomène est analogue à celui du pendule inversé de Kapitza [43, 71] qui permet de faire tenir un pendule inversé en équilibre à la

verticale à l'aide d'un support vibrant. Le support vibrant correspond à une valeur de tension d'entrée ne permettant pas aux inverseurs de commuter, elle est entre les états logiques 0 et 1. Ce niveau de tension ne correspondant à aucune valeur logique est appelé état métastable. La figure 1.6 présente la boucle de mémorisation et la superposition des fonctions de transfert des inverseurs avec les trois états possibles en sortie : 0, 1 et métastable.

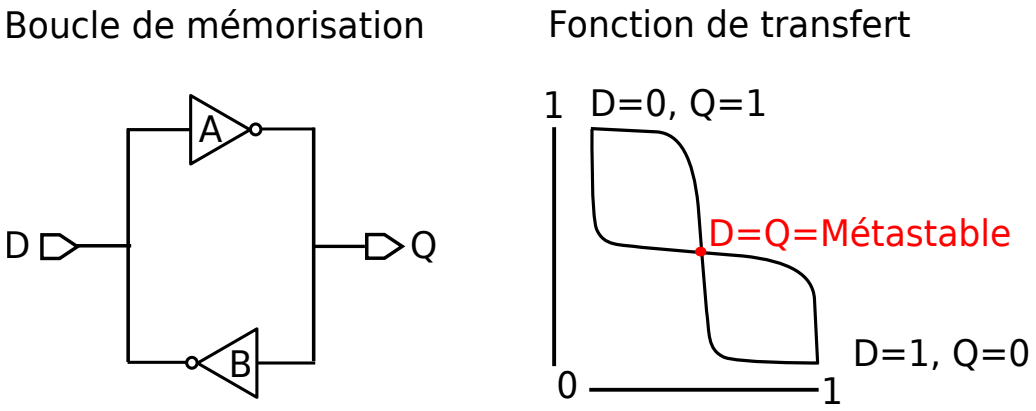


FIGURE 1.6 – Les éléments bistables

Les CDC et la métastabilité Si un signal, à l'entrée d'un élément séquentiel, change d'état proche du front d'horloge d'échantillonnage, la boucle de mémorisation peut se fermer alors que le signal est entre les états logiques 0 et 1. Dès lors les inverseurs couplés pourront rester dans cet état métastable indéfiniment. C'est la raison pour laquelle le bon fonctionnement des éléments séquentiels est conditionné par la stabilité du signal d'entrée pendant une période autour du front d'horloge. Dans le contexte des CDC, la non-relation entre les horloges peut provoquer la génération d'un état métastable en sortie pendant une durée indéterminée (potentiellement infinie) [25,32,34,107]. Ce phénomène était connu des spécialistes du domaine dès les années 50 [14] mais n'a été présenté pour la première fois qu'en 1973 par Chaney et Molnar [15].

Les trois scénarios de résolution de la métastabilité En pratique, les différentes sources de bruit du circuit permettent à la boucle de se stabiliser à une valeur logique mais ni le temps, ni l'état de résolution de la métastabilité ne sont prédictibles [40, 58, 71]. Par ailleurs, les sources de bruit peuvent tout autant faire passer la boucle dans un état métastable [107]. Trois scénarios sont envisageables en aval de l'élément séquentiel métastable :

— La « bonne » résolution de la métastabilité, c'est à dire l'augmentation du temps de traversée de la bascule ;

— La « mauvaise » résolution avant la capture, c'est à dire la perte de la donnée ;

— La capture d'un état métastable (non logique), c'est à dire la propagation du problème.

La figure 1.7 présente une donnée transférée (Q_S/D_D) qui change d'état lors de l'arrivée du front d'horloge de destination (Clk_D). La sortie entre alors dans un état métastable (Q_D) qui se résout de trois manières différentes en sortie : retard (Q_D[0]), perte de donnée (Q_D[1]) et propagation d'un état non logique (Q_D[3]).

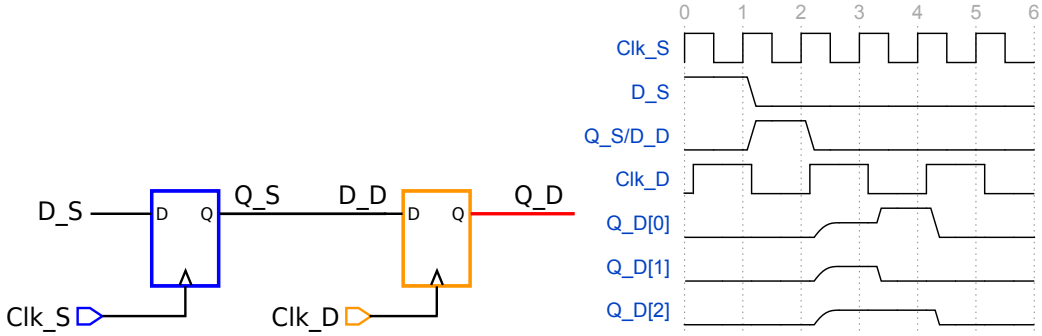


FIGURE 1.7 – Le phénomène de métastabilité

Les convergences Si le scénario de capture d'un état métastable est problématique par définition, la résolution aléatoire de la métastabilité peut aussi provoquer des dysfonctionnements. Dans le cas du transfert d'un bus de données, la génération d'états métastables sur plusieurs canaux en parallèle est dangereuse. Si les états métastables se résolvent à la bonne valeur pour certains bits, et à la mauvaise pour d'autres, des données incohérentes sont propagées. Lorsque les données sont recombinées à travers de la logique combinatoire, des états non voulus peuvent être générés dans le domaine de destination : ce problème s'appelle la convergence.

La figure 1.8 présente un problème de convergence avec le transfert d'un bus de deux bits (Q_S[1 :0]) recombinés dans le domaine de destination à travers une porte "ET". Les signaux changent d'état au moment de l'arrivée de l'horloge de destination (Clk_D), générant alors des états métastables en sortie des registre de destination (Q_D[1 :0]). Le bit 1 se résout à la bonne valeur alors que le bit 0 se résout à la mauvaise. La sortie de la convergence à travers la porte "ET" prend alors une valeur non voulue.

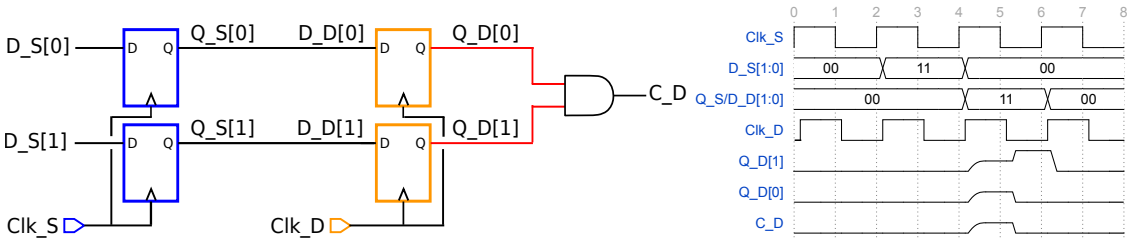


FIGURE 1.8 – La convergence de signaux après un CDC

3.2 Observation du phénomène de métastabilité

Dans l'industrie, le phénomène de métastabilité est pris en compte lors des étapes de conception des bibliothèques de composants (Library Cells). Les ingénieurs en

charge de la caractérisation des cellules extraient les temps de résolution des éléments séquentiels dans les différentes conditions de fonctionnement du circuit (processus de fabrication, tension d'alimentation, température) [94].

Le banc de test Les mesures effectuées lors de la caractérisation des éléments séquentiels mettent en évidence l'augmentation du temps de résolution de la bascule. Ce phénomène correspond uniquement au scénario de bonne résolution de la métastabilité. Il est nécessaire d'étudier le comportement d'un signal métastable pour vérifier que les deux autres scénarios (mauvaise résolution et non-résolution) sont réels. Pour pouvoir observer un état métastable à la sortie d'une bascule D, deux évènements doivent se produire : la génération du phénomène et sa propagation. Le signal d'entrée D doit donc changer d'état pendant les temps de stabilisation et de maintien autour du front d'horloge, générant un signal métastable en sortie du premier verrou. Ce signal métastable doit durer jusqu'à ce que qu'un front d'horloge opposé arrive pour fermer la boucle du second verrou et le propager en sortie, soit un demi-cycle d'horloge au minimum.

La simulation de la métastabilité Pour assurer l'observation d'un comportement le plus proche possible de la réalité, la cellule simulée correspond à un élément séquentiel implémenté dans des circuits en cours de développement. Le modèle du circuit est de type « dessin de masque » afin qu'un maximum d'éléments parasites soit pris en compte. La fréquence de l'horloge a été définie à partir des performances atteignables par les circuits intégrant ces cellules : 1.25 GHz. Concernant la simulation, le modèle mathématique des transistors est propriétaire à l'entreprise. Notons que les formules mathématiques ne modélisant pas le comportement réel, certains phénomènes n'apparaissent pas. Le plus connu est la métastabilité profonde qui fut présenté par Kinniment en 2006 [42, 53, 55, 109]. Comme son nom l'indique, il correspond à une augmentation exponentielle du temps de résolution lorsque la donnée est très proche du front d'horloge. La figure 1.9 présente les résultats de simulation de la métastabilité à la sortie d'une bascule D en technologie 28nm FDSOI.

Analyse des résultats Les résultats de simulation ont confirmé la théorie relative au comportement d'un signal métastable. En effet, les trois scénarios possibles après la génération d'un état métastable en sortie d'une bascule D ont été observés :

- La courbe verte présente une bonne résolution de la métastabilité : la propagation prend un huitième de cycle d'horloge ;
- La courbe jaune présente une mauvaise résolution de la métastabilité : la donnée n'est pas propagée en sortie ;
- La courbe rose présente une non-résolution de la métastabilité : l'état non logique est maintenu plus d'un demi-cycle d'horloge (0,4 ns).

4 Conclusion

Les problèmes potentiels dus aux transferts de données entre domaines d'horloges sont multiples. La capture d'états transitoires et la perte de données correspondent à un problème de synchronisation entre les horloges d'émission et de réception.

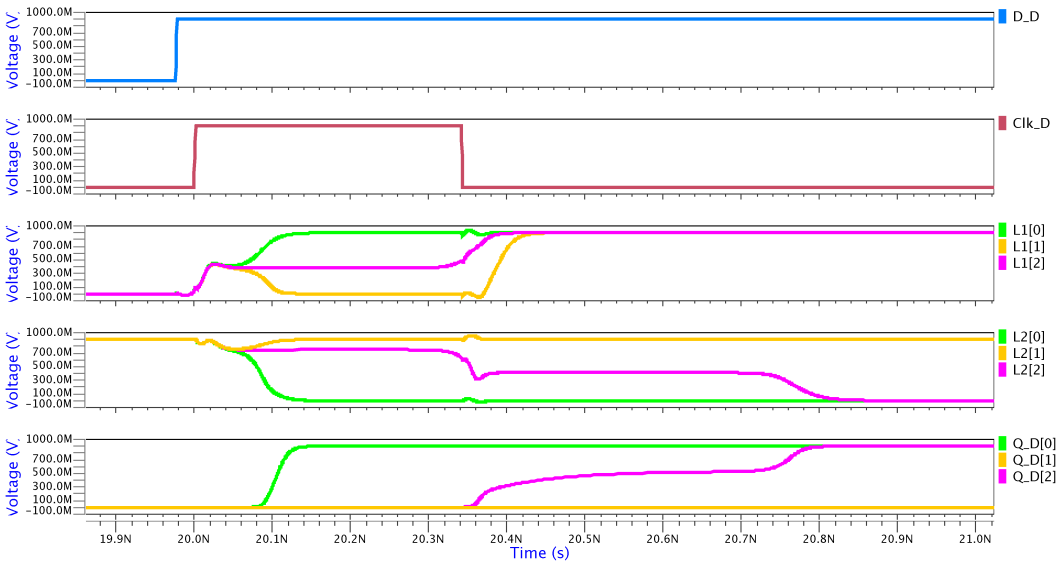


FIGURE 1.9 – Simulation de la métastabilité

Cependant la métastabilité est un phénomène physique inhérent à l'architecture des éléments séquentiels. Ce phénomène au comportement aléatoire représente le danger principal des CDC. Par définition un état métastable ne correspond pas à un état logique, sa propagation peut donc faire rentrer un système logique dans un état inconnu et potentiellement erroné.

Chapitre 2

La conception des CDC

Les ingénieurs en charge de la conception des CDC mettent en place des techniques assurant un transfert de données correct. Ces procédés sont appelés structures de synchronisation et peuvent être répartis en trois catégories :

— La gestion de la métastabilité ;

— Les protocoles de temporisation ;

— Les protocoles de séquençage.

1 La gestion de la métastabilité

Le phénomène de métastabilité est réel et son comportement aléatoire peut générer des problèmes critiques. Son apparition est inhérente au fonctionnement des éléments séquentiels. Dans le contexte des CDC, la donnée transférée étant échantillonnée par une horloge source, l'apparition du phénomène est due à la proximité entre le front d'horloge source et celui de destination. Il existe deux types de communications entre horloges pouvant être considérés comme des CDC : quasi-synchrones et asynchrones.

1.1 Les communications quasi-synchrones

La particularité des communications quasi-synchrones est que l'horloge d'émission et celle de réception ont, dans leur cône d'entrée, une source commune (sous-section 1.1, page 3). Dès lors, même si les deux horloges ne sont pas alignées en phase, leur relation est fixe : il est possible d'empêcher la génération d'états métastables.

Les détecteurs de déphasage La technique employée consiste à anticiper les moments où le déphasage entre les horloges est dangereux et à ajouter un délai sur l'horloge de réception. La détection du déphasage se fait à travers l'échantillonnage de l'horloge source par deux registres séquencés sur l'horloge de destination. Le premier registre capture l'horloge source avant le front de destination (il est retardé d'une durée correspondant au temps de stabilisation). Le second registre capture l'horloge source après le front de destination (l'horloge source est retardée d'une durée correspondant au temps de maintien). Si les sorties des deux registres sont identiques alors on peut assurer que le déphasage ne pose pas de problème. À l'inverse, le front d'horloge source arrive dans la fenêtre de métastabilité de l'horloge

de destination : il faut décaler l'arrivée de son front [27, 29, 32, 33, 72, 87]. La figure 2.1 présente à haut niveau l'approche de conception des interfaces quasi-synchrones avec le détecteur de déphasage. Les composants notés *ts* et *tm* sont des lignes de délai dont les valeurs correspondent respectivement aux temps de stabilisation et de maintien de l'horloge de destination.

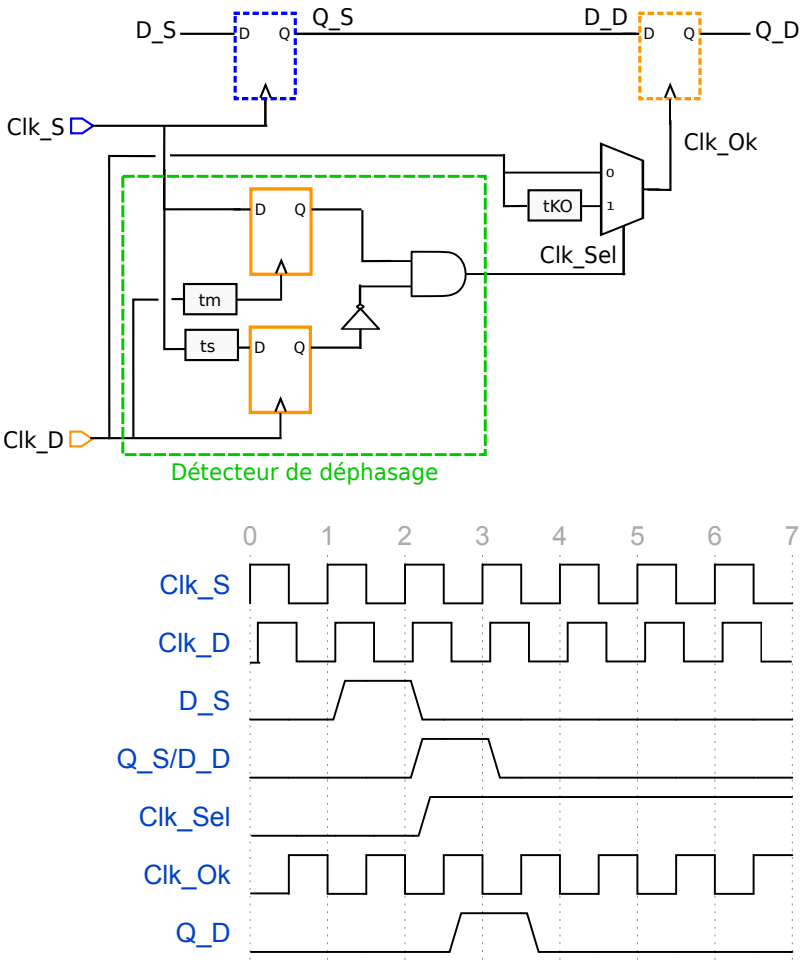


FIGURE 2.1 – Le détecteur de déphasage

Avantages et inconvénients Cette approche ramène le problème de métastabilité sur l'arbre d'horloge de destination ce qui est dangereux. Cependant, lors d'une communication quasi-synchrone, les horloges d'émission et de réception ont une relation fixe que ce soit en termes de phase (mésochrone) ou de rapport de fréquences (périodique et plésiochrone). La détection du déphasage peut donc être effectuée à l'initialisation du circuit. Pour les interfaces où le délai d'arrivée entre les fronts est périodique ou faiblement variable, un compteur peut être implémenté à partir du rapport de fréquences. Dès lors, les communications quasi-synchrones ne sont jamais sujettes à la métastabilité. Ces techniques, bien que performantes, sont sur-mesure et plus difficiles à concevoir qu'un alignement des arbres d'horloge (implémentation synchrone).

1.2 Les communications asynchrones

La particularité des communications asynchrones est que l'horloge d'émission et celle de réception ont des phases et des fréquences pouvant varier indépendamment. Les horloges n'ont donc aucune relation entre elles et il est impossible d'empêcher la génération d'un état métastable (sous-section 3.1, page 19).

1.2.1 Le "Mean Time Between Failure" (MTBF)

Sachant que la résolution de la métastabilité n'est pas prédictible, et que sa génération est inévitable, c'est un problème critique. Dès le début des années 80, la question a été étudiée et une formule mathématique permettant d'estimer le taux de défaillance des inverseurs couplés (Failure Rate) a été définie [104]. Une défaillance correspond à la génération et la propagation d'un état métastable. L'inverse du taux de défaillance est la durée moyenne entre deux défaillances ou MTBF (Mean Time Between Failure). Cette formule a ensuite été développée, à la fin des années 80, par Kleeman et Cantoni à travers un tutoriel sur le phénomène de métastabilité, le calcul du MTBF et les techniques de conception [58]. Afin d'assurer une corrélation entre les résultats théoriques et les mesures de la métastabilité, cette formule évolue en fonction du développement des circuits intégrés numériques [3,21,34,69]. Cependant l'information donnée n'est que probabiliste : un CDC présentant un MTBF important peut générer et propager un état métastable. L'équation 2.1 donne la formule la plus communément utilisée du MTBF et qui se base sur cinq paramètres :

- tr : le temps alloué à la résolution de la métastabilité. Sa valeur correspond au temps moyen que met une bascule D pour propager, lors d'un front d'horloge, une donnée en sortie.
- Tw : la fenêtre de métastabilité. C'est la période de temps maximale (autour du front d'horloge) pendant laquelle un changement d'état en entrée augmente le temps de résolution de la bascule (tr).
- Fd : la fréquence de la donnée d'entrée.
- Fc : la fréquence de l'horloge d'entrée.
- τ : la constante de résolution de la métastabilité. Sa valeur correspond au coefficient directeur de la droite représentant l'évolution de tr en fonction de l'arrivée de la donnée dans la fenêtre de métastabilité (Tw).

$$MTBF = \frac{\exp^{\frac{tr}{\tau}}}{Fc \times Fd \times Tw} \tag{2.1}$$

1.2.2 Les structures existantes

La multi-flop La plupart des recherches a consisté à réduire la probabilité d'erreurs dues à la métastabilité en augmentant la valeur du MTBF. Ces travaux ont été initiés par Kinniment à la fin des années 70 [56]. La technique la plus courante, présentée pour la première fois en 1987 par Kleeman et Cantoni [58] est la structure de type multi-flop. L'idée est de cascader des éléments séquentiels dans le domaine de destination afin d'augmenter le temps alloué à la résolution de la métastabilité

(tr). Cette approche, bien que simple à mettre en place, n'empêche pas la génération de la métastabilité. Dès lors, les problèmes liés à la mauvaise résolution de la métastabilité (perte de données et convergences) peuvent survenir. Concernant les performances de cette structure, l'ajout d'éléments séquentiels augmente le délai. La figure 2.2 présente la structure de synchronisation par multi-flop avec des chronogrammes illustrant les deux scénarios possibles de résolution de la métastabilité :bon état pour $Q\_MF[0]$ et mauvais pour $Q\_MF[1]$.

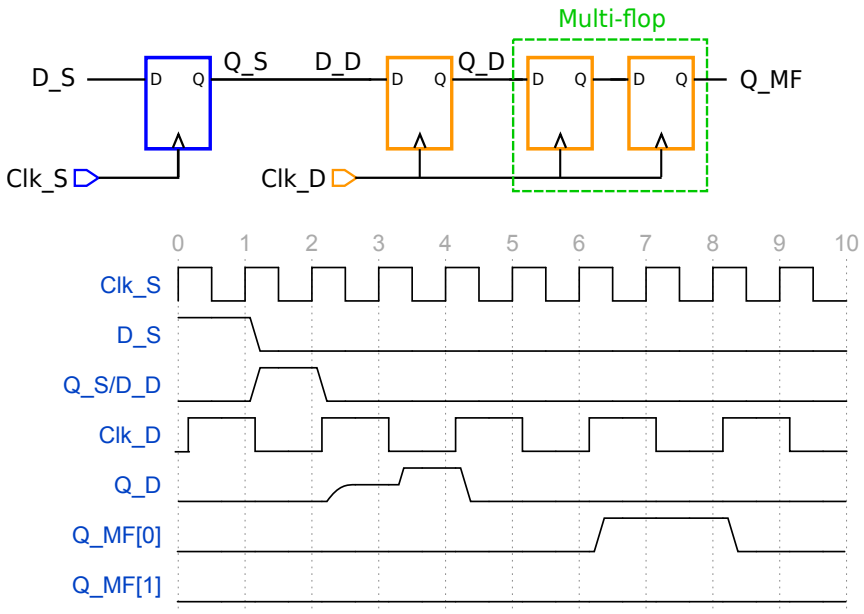


FIGURE 2.2 – La structure multi-flop

Les verrous de synchronisation Pour éviter l'impact en performances, les concepteurs ont cherché à minimiser la constante de résolution de la métastabilité (τ). Le circuit le plus couramment utilisé, appelé verrou montant (Jamb Latch) a été décrit par Kim, Cline et Dutton en 1990 [52]. Sa particularité est que la boucle de mémorisation est composée de transistors NMOS et PMOS de même taille (Wn = Wp). Notons qu'un verrou classique est composé de transistors PMOS deux fois plus grands que les NMOS (Wp = 2Wn) afin d'assurer des temps de transition montants et descendants équivalents. Ce verrou de synchronisation a été réutilisé par la plupart des spécialistes du domaine qui se sont efforcés de l'améliorer [28, 54]. Cependant, le paramètre τ est proportionnel à la différence entre les tensions de seuil des transistors et l'alimentation. Cette technique n'est donc pas suffisante lorsque les circuits sont alimentés à faible tension. Dès lors, différents circuits ont été développés, notamment le synchroniseur robuste (Robust Synchronizer) [107, 111] et le synchroniseur symétrique renforcé (Symetric Boost Synchronizer) [47]. Les deux techniques sont basées sur l'utilisation de rehausseurs de tension, comme décrit par Cline en 1998 [20]. L'idée est que lorsqu'un état métastable est détecté, un jeu de résistance vient réhausser la tension de la boucle de mémorisation pour rendre le signal stable rapidement.

Ces dix dernières années, avec l'évolution des circuits selon la loi de Moore, des paramètres comme les nœuds technologiques ou la température sont devenus des

facteurs de dégradation du MTBF [5, 7]. Des structures ont donc été proposées en fonction du contexte de fonctionnement des circuits [1, 13, 110].

Les bloqueurs de métastabilité Certains concepteurs se sont employés à assurer que la métastabilité ne se propage pas en sortie d'un registre : plusieurs structures telles les bloqueurs et les mélangeurs de métastabilité [31, 107] ont été développées. L'approche associée aux bloqueurs est de mémoriser la donnée d'entrée dans un verrou lorsque l'état de l'horloge est à la valeur précédant le front d'échantillonnage et de conserver la donnée mémorisée lorsque le front d'horloge survient. Dès lors la donnée échantillonnée est stable lors de l'arrivée du front d'horloge : il n'y a pas de métastabilité. Cependant, le verrou de mémorisation peut lui aussi être sujet à la métastabilité, le problème est donc simplement décalé. Le principe des mélangeurs consiste à envoyer une impulsion électrique lorsque de la métastabilité est détectée pour accélérer sa résolution. Cependant, si la métastabilité est en train de se résoudre, cette approche peut faire revenir le signal dans un état métastable. Ces techniques n'empêchent donc pas la génération de la métastabilité et augmentent même le risque de propagation. De même, l'utilisation d'inverseurs asymétriques [4] ne fait que décaler les niveaux de tension métastables, générant des oscillations.

1.2.3 Les structures utilisées en pratique

Bien que performants, les verrous de synchronisation ne fonctionnent qu'avec un dimensionnement précis de la boucle de mémorisation et des règles de conception spécifiques [5]. Dans l'industrie, le temps consacré à la conception et la caractérisation des cellules n'est pas suffisant pour pouvoir développer des solutions similaires : la majorité des cellules de synchronisation sont des multi-flops composées de verrous montants permettant une résolution plus rapide de la métastabilité [94]. La profondeur des multi-flops est généralement fixée à deux ou trois étages afin de faire un compromis entre performances, temps de conception et robustesse du circuit à la métastabilité. Concernant les faibles tensions d'alimentation, les transistors intègrent des techniques de "body biasing", permettant d'ajuster les tensions de seuil en fonction de la tension d'alimentation, et conserver la performance des cellules [108].

2 Les protocoles de temporisation

En dehors de la métastabilité, les CDC sont exposés aux problèmes de perte de données, de capture d'états transitoires et de propagation de données incohérentes (sous-section 1, page 17). Ils doivent donc être gérés à l'aide de protocoles de transfert dédiés [23, 60]. L'approche consiste à mettre en place, dans le domaine d'horloge source, des structures permettant d'empêcher l'apparition d'éventuels dysfonctionnement, on parle alors de protocoles de temporisation.

2.1 Éviter la propagation de données incohérentes

La propagation de données incohérentes survient lorsque plusieurs informations sont transférées en parallèle entre deux domaines d'horloge (bus de données). Les

incohérences peuvent être dues à deux raisons : le temps de propagation entre les éléments séquentiels ou la génération d'états métastables sur plusieurs canaux. Quelle que soit la source du problème, ce dernier n'apparaît que si plusieurs signaux (au moins deux) changent d'état au même cycle d'horloge d'émission. Pour empêcher son apparition, il faut assurer l'exclusivité des signaux (encodage Gray), c'est-à-dire qu'entre deux transferts de données successifs, un seul bit peut changer d'état.

La figure 2.3 présente un compteur Gray sur deux bits composé d'un compteur Binaire et d'un convertisseur Binaire vers Gray. Cette structure placée en entrée du registre source permet d'éviter le transfert de données incohérentes dû aux CDC.

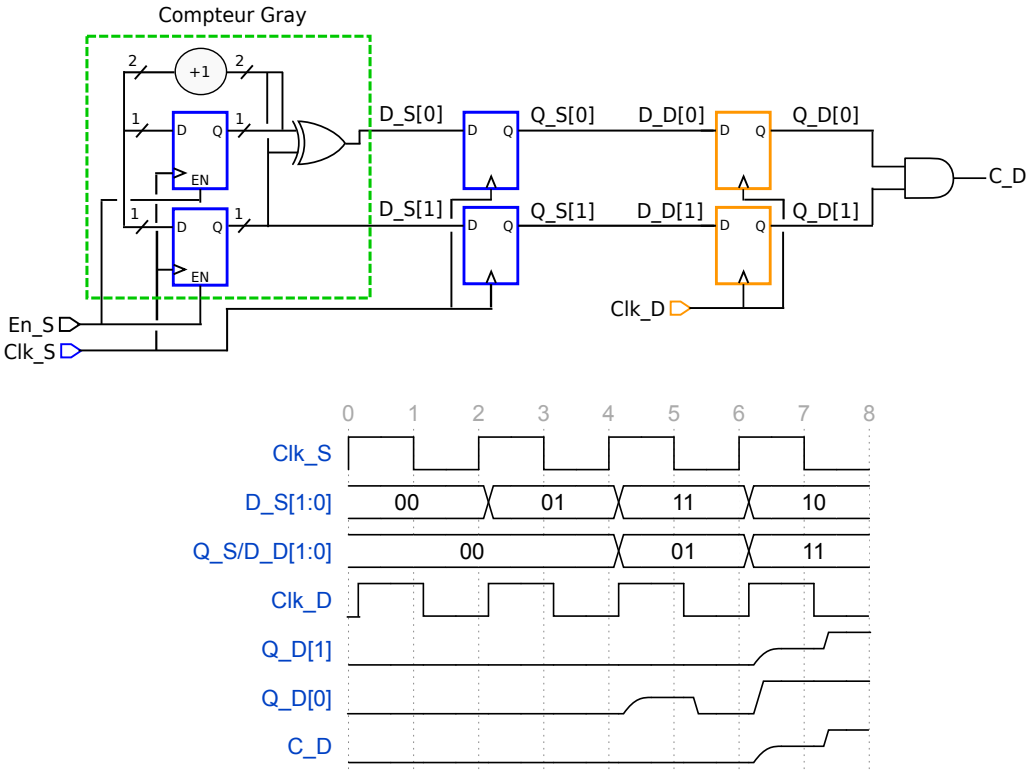


FIGURE 2.3 – L'encodage Gray

2.2 Éviter la capture d'états transitoires

La capture d'états transitoires survient lorsque plusieurs données, transférées en parallèle, convergent sur des éléments de logique combinatoire entre deux domaines d'horloge. Le problème étant dû à la dynamique des signaux transférés, il est possible d'empêcher son apparition en assurant qu'un seul signal a une influence sur la sortie de la logique combinatoire. De même, dans le cas de signaux exclusifs, étant donné qu'un seul signal change par cycle d'horloge d'émission, aucun état transitoire ne peut être généré.

La figure 2.4 présente une structure empêchant la génération d'états transitoires en sortie d'un bloc de logique combinatoire à deux entrées du type "OU Exclusif". L'idée est d'implémenter une logique assurant que les deux signaux transférés ne sont jamais sujets à des changements d'états au même cycle d'horloge source ainsi, la sortie du bloc combinatoire ne peut pas générer d'états transitoires dus aux courses.

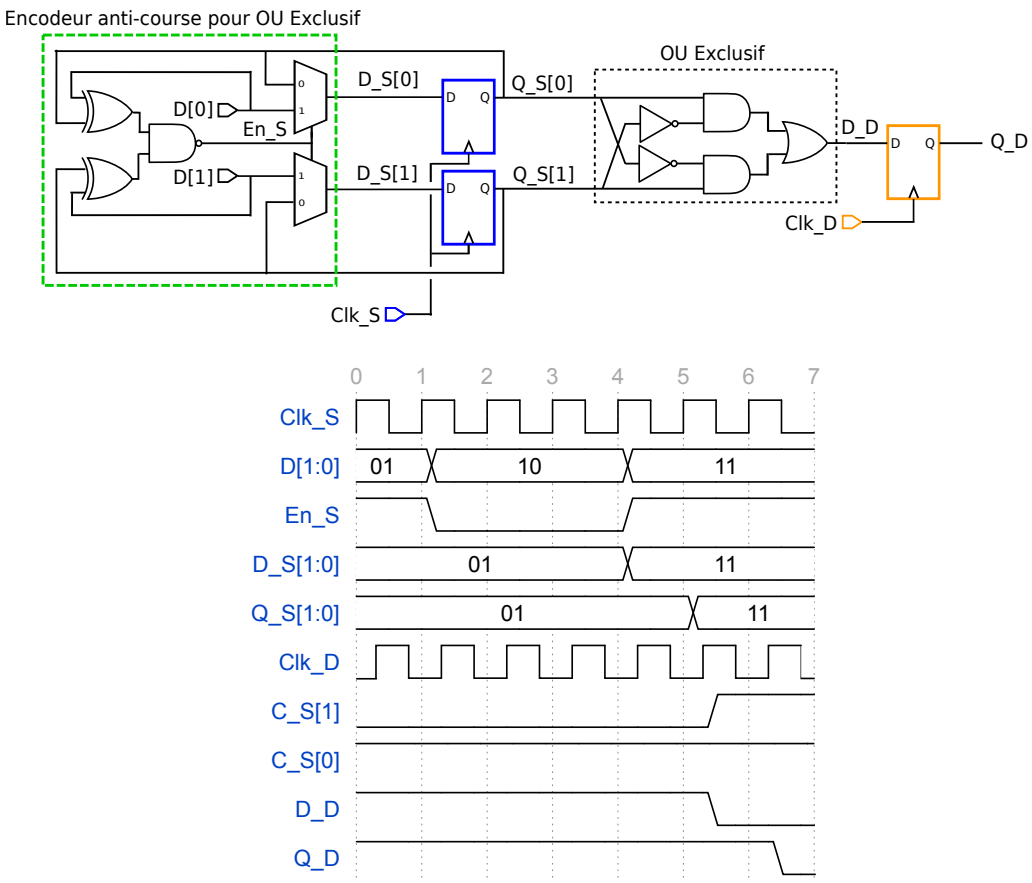


FIGURE 2.4 – La structure anti-courses

2.3 Éviter la perte de données

La perte de données est due à un des deux facteurs suivants : le transfert d'un domaine d'horloge rapide vers un lent ou la mauvaise résolution de la métastabilité. Ce problème peut être évité à travers un maintien de la donnée transférée jusqu'à sa capture par le registre de destination. Pour un CDC dont la fréquence de la source est supérieure à celle de la destination, le signal émis doit avoir une fréquence de transition inférieure ou égale à celle de l'horloge de destination. Dans le cas d'une mauvaise résolution de la métastabilité, il est possible de ramener le problème à un retard dans la propagation de la donnée : l'information envoyée doit avoir une fréquence de transition inférieure ou égale à deux tiers de celle de l'horloge de destination. Cette condition étant plus forte, c'est généralement la solution retenue lorsque la fréquence de l'horloge émettrice est supérieure ou égale à celle de l'horloge réceptrice.

La figure 2.5 présente une structure de maintien des données transférées. L'idée est d'éviter des pertes de données dues à des transitions successives en convertissant la présence de transitions en état : le signal est à l'état "1" lors d'une transition sinon il est à "0". Une fois le signal réceptionné par la destination, les états sont convertis en transitions afin de récupérer l'information émise par la source.

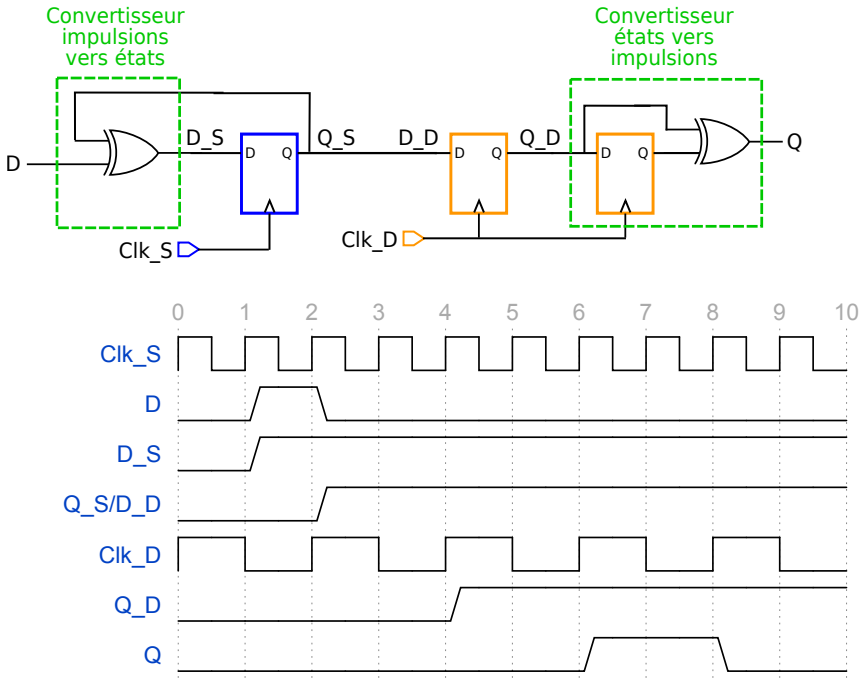


FIGURE 2.5 – Le maintien des données

2.4 Limitations pratiques

Les protocoles de temporisation sont des techniques performantes car elles sont basées sur une gestion sur-mesure des signaux dans le domaine d'horloge d'émission. Cependant, la complexité de mise en place de ces techniques augmente en fonction du nombre de données transférées en parallèle : il y a un risque d'erreur humaine. De plus, ces approches n'empêchent pas la génération et la potentielle propagation

d'états métastables : des multi-flops doivent être insérées après chaque registre de destination. Les deux inconvénients majeurs sont donc la complexité en termes de conception et le coût de surface dû au nombre d'éléments logiques à ajouter pour couvrir les différentes problématiques CDC.

3 Les protocoles de séquençage (force brute)

Le meilleur moyen d'éviter les problèmes CDC, métastabilité inclue, est d'utiliser des protocoles de séquençage. L'idée est d'assurer que les signaux transférés sont stables lors de leur capture et qu'ils ne changent d'état qu'après avoir été réceptionnés. L'approche la plus simple consiste à utiliser des signaux de contrôle entre les domaines d'horloge pour séquencer les étapes d'émission et de réception des données. Les deux principaux protocoles sont la "Poignée de main" (Handshake) et le « Premier arrivé premier sorti » (First In First Out) [27,32,85].

3.1 Synchronisation type "Poignée de main" (Handshake)

Le principe La "Poignée de main" consiste à assurer la stabilité des données à travers un protocole en quatre phases implémenté avec deux signaux de contrôle : la requête (request) et l'accusé de réception (acknowledge) [31]. Lorsqu'une donnée est émise par le domaine source, une requête est envoyée dans le domaine destination à travers une multi-flop (phase 1). Ce signal est le signal de contrôle du domaine destination puisqu'il gère la capture des données. Lorsque la requête est réceptionnée par la destination, cette dernière renvoie un accusé de réception (phase 2). L'accusé est synchronisé avec une multi-flop dans le domaine source et désactive la requête (phase 3). Une fois la requête désactivée, l'accusé l'est à son tour jusqu'à l'envoi d'une nouvelle requête (phase 4). Afin d'assurer que les données sont établies et stables tant que la requête et l'accusé ne sont pas désactivés, aucune nouvelle donnée n'est envoyé par le domaine source. La figure 2.6 présente la structure de synchronisation et le protocole fonctionnel de type « Poignée de main ». Le signal de requête est noté Req_CDC dans le domaine source et Req_D après la traversée de la multi-flop dans le domaine destination. Respectivement, l'accusé de réception est noté Ack_CDC côté destination et Ack_S en sortie de la multi-flop dans le domaine source.

Avantages et inconvénients Ce protocole est robuste puisqu'il assure un séquençage des étapes afin qu'aucun problème, autre que la métastabilité sur les signaux de contrôle, ne peut survenir. Cependant, cette technique présente un fort délai, lié à l'attente de signaux de contrôle traversant des multi-flops, entre deux transferts de données. Son utilisation n'est donc pas adaptée aux communications rapides. Plusieurs structures optimisées ont été développées mais le délai de synchronisation par multi-flop ne peut être définitivement supprimé. La technique la plus connue, appelée "Demi-poignée de main" (2-phase Handshake), consiste à transférer les données quand la requête et l'accusé ont la même valeur, et de les capturer lorsque leurs états respectifs sont différents. Dès lors, le protocole quatre phases n'en a plus que deux. La figure 2.7 présente les protocoles associés aux structures "Poignée de main" et "Demi-poignée de main". Les signaux de contrôle changent selon quatre

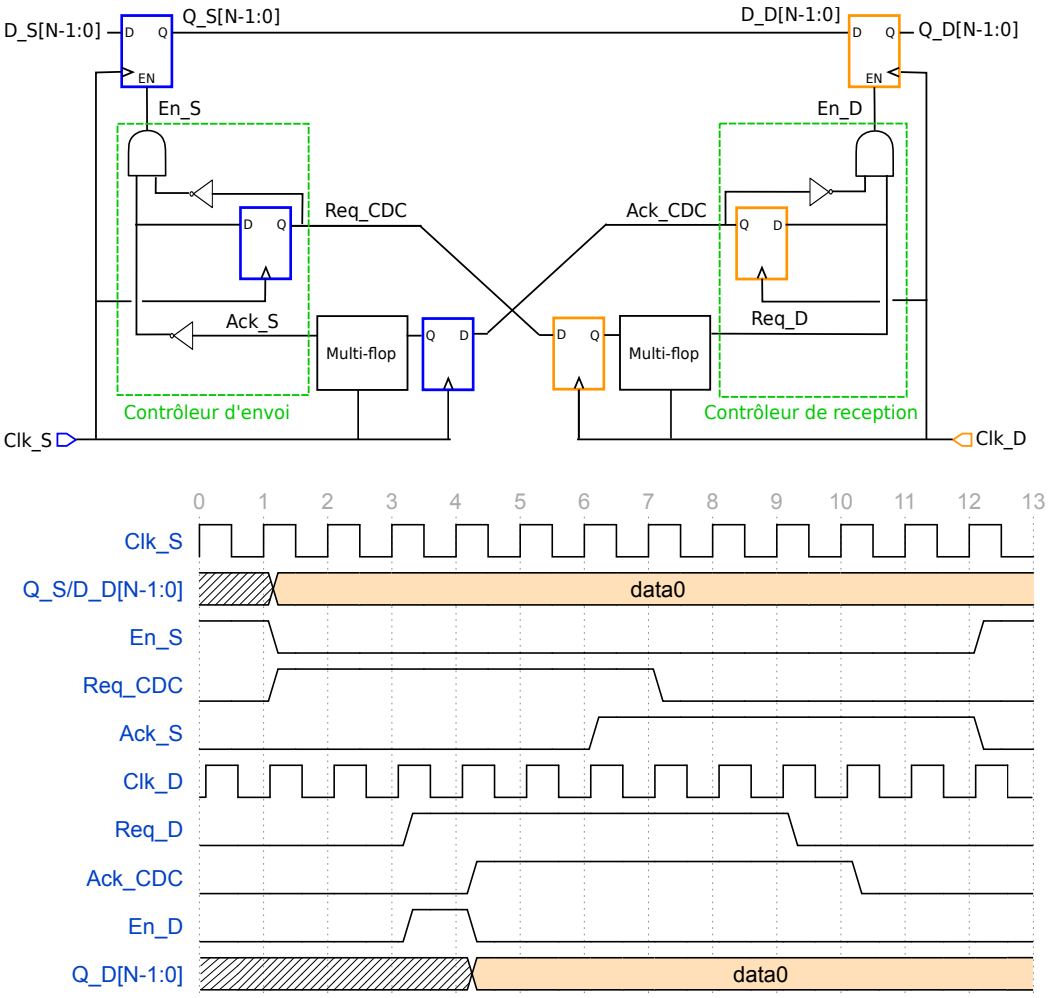
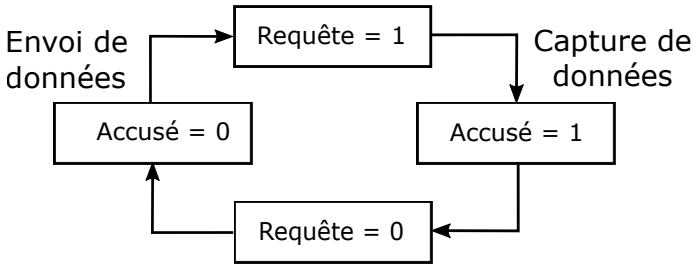
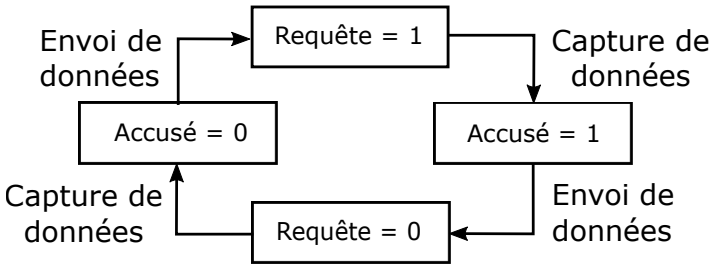


FIGURE 2.6 – La “Poignée de Main”

phases identiques pour les deux cas. Seules les conditions d'envoi et de réception des données sont différentes.



(a) Protocole "Poignée de main"



(b) Protocole "Demi-poignée de main"

FIGURE 2.7 – Les protocoles associés à la structure "Poignée de main"

3.2 Synchronisation type "Premier arrivé premier sorti" (FIFO)

Le principe La synchronisation de type « Premier arrivé premier sorti » utilise une mémoire et deux signaux de statut : mémoire vide (empty) et mémoire pleine (full). Les données émises par la source sont contenues dans une mémoire et répertoriées par une adresse d'écriture. Symétriquement, les données capturées par la destination sont lues dans la mémoire en fonction de l'adresse de lecture. Afin d'éviter les incohérences, les adresses d'écriture et de lecture sont comparées dans les deux domaines d'horloge. Lorsque l'adresse de lecture égale l'adresse d'écriture dans le domaine de destination, un signal indique que la mémoire est vide et bloque la lecture des données. Réciproquement, lorsque l'adresse d'écriture égale l'adresse de lecture précédente côté source, un signal indique que la mémoire est pleine et bloque l'écriture des données. Afin de réduire le risque de métastabilité, les adresses d'écriture et de lecture sont synchronisées (par multi-flops) respectivement dans les domaines destination et source. Les adresses étant généralement des signaux de contrôle multi-bits, elles sont encodées en Gray (protocole de temporisation) pour éviter les incohérences. La figure 2.8 présente la structure de synchronisation de type « Premier arrivé premier sorti ». L'adresse d'écriture est notée $WrAdd\_CDC$ dans le domaine source et $WrAdd\_D$ après traversée des multi-flops côté destination. Réciproquement, l'adresse de lecture est notée $RdAdd\_CDC$ côté destination et $RdAdd\_S$ après traversée des multi-flops dans le domaine source.

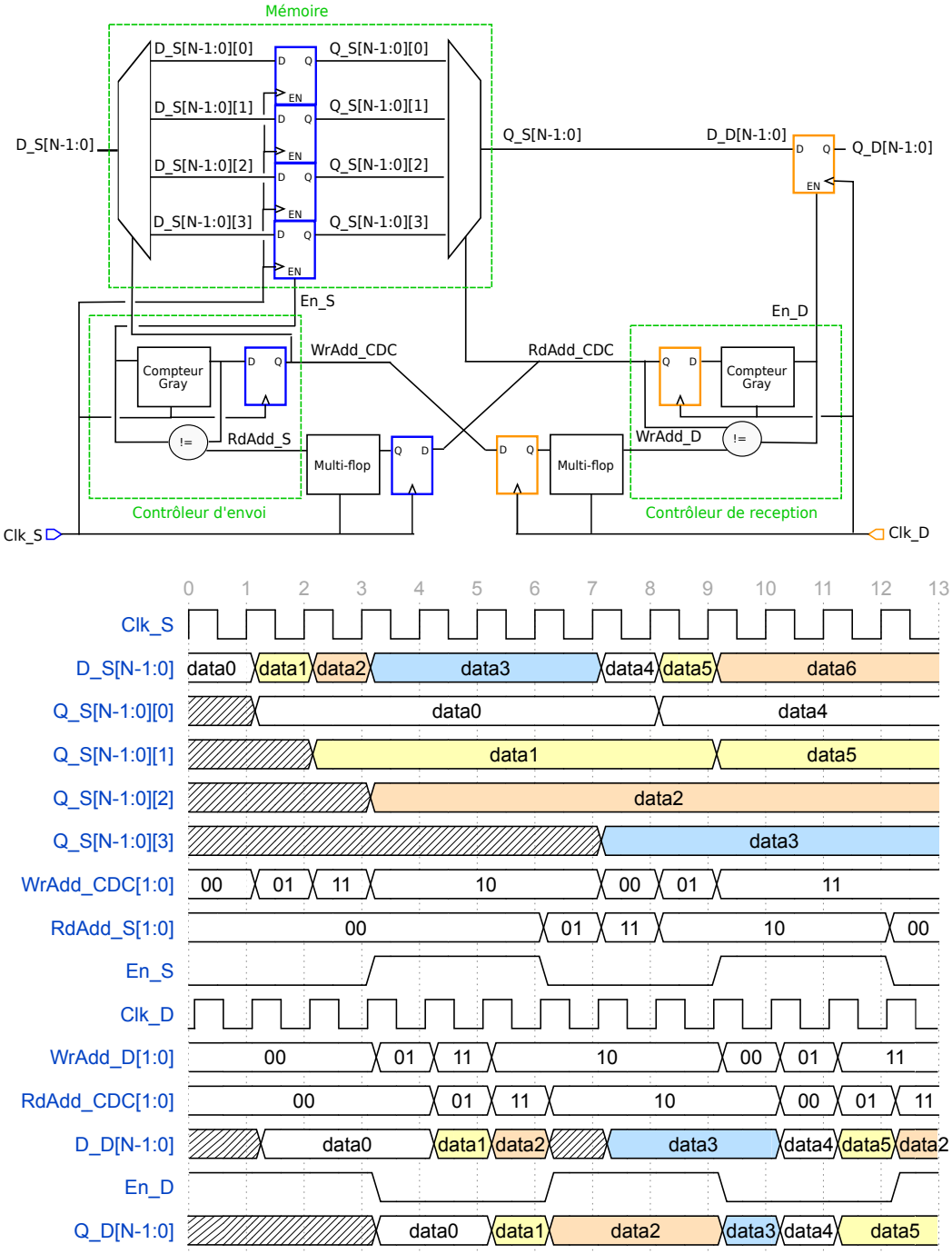


FIGURE 2.8 – Le “Premier arrivé premier sorti”

Avantages et inconvénients Cette structure est plus performante que la "Poignée de main" puisqu'il est possible d'écrire et de lire en même temps dans la mémoire excepté lorsque cette dernière est vide. Cependant, son implémentation (mémoire, logique de génération des signaux de statut et encodage gray) présente les mêmes limitations que les protocoles de temporisation.

La figure 2.9 présente le principe de fonctionnement associé à la structure "Premier arrivé premier sorti". Les données transférées sont stockées dans une mémoire. Lorsque la mémoire n'est pas vide, les phases d'écriture et de lecture sont indépendantes. Pour une adresse donnée, le contrôle de la lecture et de l'écriture correspond à un protocole de type "Poignée de main".

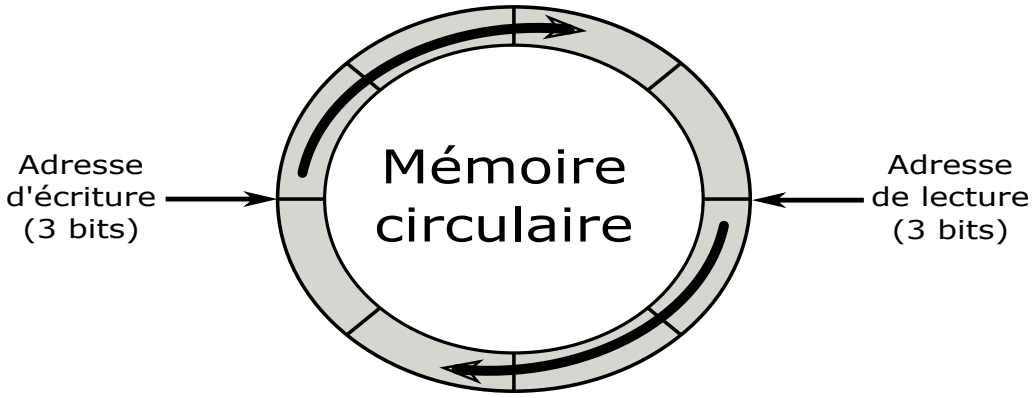


FIGURE 2.9 – Principe de fonctionnement du "Premier arrivé premier sorti"

3.3 Synchronisation de type "Contrôle Unique"

Plusieurs concepteurs ont cherché à développer des structures assurant la même facilité d'implémentation que la "Poignée de main" et les mêmes performances que la structure "Premier arrivé premier sorti". La structure la plus connue est le "Contrôle Unique" [27, 32]. L'idée est de limiter le délai entre deux transferts de données en utilisant un seul signal de contrôle pour la capture de données stables. Pour éviter la métastabilité, le signal de contrôle est synchronisé par multi-flop. Lors du transfert d'un domaine d'horloge rapide vers un domaine lent, une structure de maintien est implémentée afin d'empêcher un perte de donnée sur le contrôle. Cependant, aucun signal de retour n'indique à la source que la donnée a été capturée. Pour éviter que plusieurs données soient envoyées entre deux captures, des compteurs basés sur le temps de propagation du contrôle doivent être implémentés côté source. Pour que cela fonctionne, les dynamiques respectives des horloges source et destination doivent être fixes. Dès lors, la communication s'apparente au cas des interfaces quasi-synchrones. Bien que les performances soient satisfaisantes, l'utilisation de détecteurs de déphasage est plus efficace car il n'y a pas de délai dû aux multi-flops et la métastabilité ne peut survenir. La figure 2.10 présente une structure de type "Contrôle unique" dans le cas d'une interface de type horloge rapide vers horloge lente. Le signal de contrôle est noté En_CDC dans le domaine source et En_D à l'entrée du registre de données de destination. Afin d'éviter la perte de données, due au rapport de fréquences, une logique de maintien (protocole de temporisation) est utilisée au niveau du signal de contrôle.

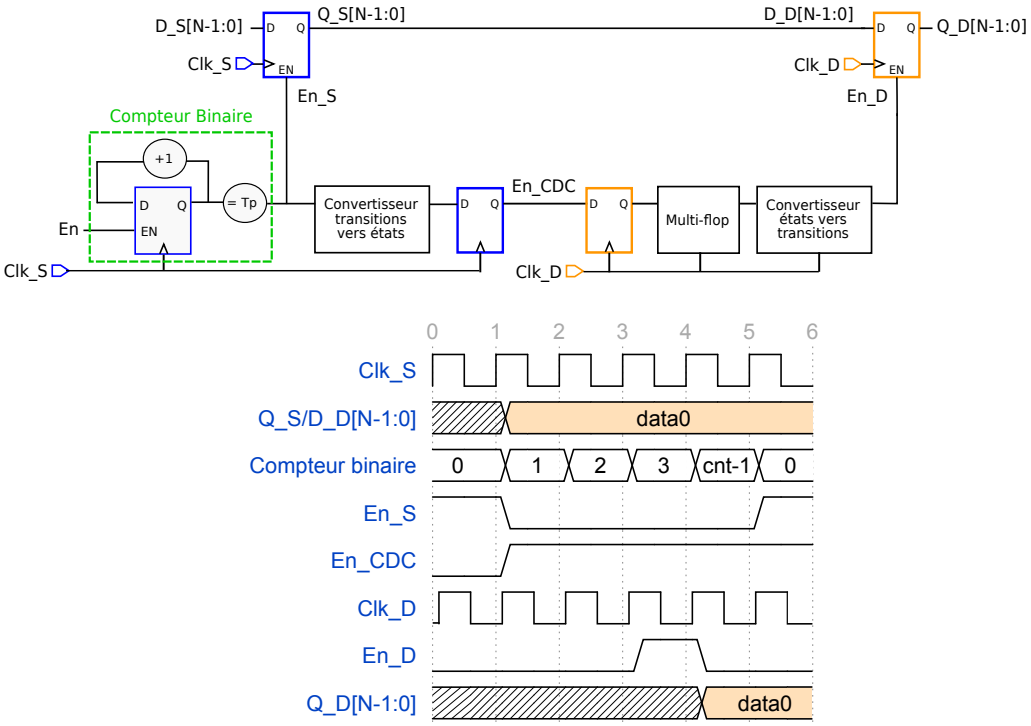


FIGURE 2.10 – Le “Contrôle unique”

La figure 2.11 présente le protocole associé à la structure “Contrôle unique”. Lorsque la donnée est émise, la phase d'envoi est arrêtée, et la phase de réception n'intervient qu'à l'arrivée du signal de contrôle synchronisé par multi-flop dans le domaine de destination. Un nouvel envoi de donnée ne sera autorisé qu'après un délai (Tp) assurant la fin de la phase de réception.

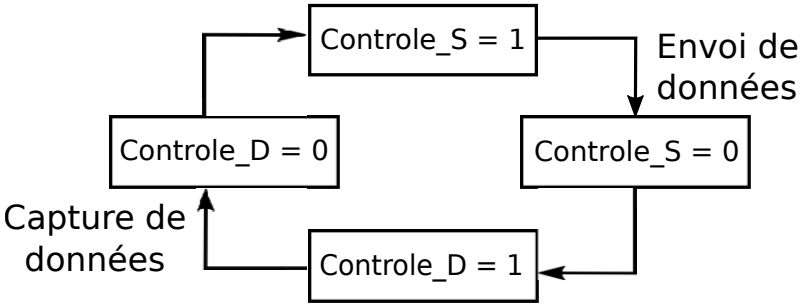


FIGURE 2.11 – Le protocole associé au “Contrôle unique”

4 Conclusion

Les CDC peuvent être conçus de différentes manières pour réduire la problématique de non alignement des horloges source et destination :

— La métastabilité est gérée par l'utilisation de verrous de synchronisation et de multi-flops ;

— Les autres problèmes CDC peuvent être gérés par des protocoles de temporisation ou de séquençage ;

— Pour les interfaces quasi-synchrones, des détecteurs de déphasage et des structures “Contrôle Unique” peuvent être implémentés ;

En pratique, il est recommandé d'utiliser des protocoles de séquençage pour la synchronisation des données et de limiter la mise en place de multi-flops et de protocoles de temporisation aux signaux de contrôle. Néanmoins, ce choix de techniques possibles ainsi que leur complexité peuvent engendrer des erreurs lors de la conception des structures. Il est donc nécessaire de vérifier les CDC afin d'assurer que les communications asynchrones sont correctement implémentées.

Par ailleurs, la propagation d'états métastables ne peut être empêchée, quelle que soit l'approche de synchronisation choisie [22, 34]. Dès lors, la vérification des autres problèmes CDC doit être optimale : tous les cas doivent être couverts et l'analyse doit être conforme avec les spécifications techniques.

Chapitre 3

La vérification des CDC

La vérification des CDC a pour but d'assurer que les communications asynchrones fonctionnent correctement. Comme expliqué dans le chapitre 2 (sections 2 et 3, pages 29 et 33), bien que la génération d'états métastables ne puisse pas être empêchée, il existe différents protocoles permettant d'éviter les autres problèmes CDC. L'objectif se limite donc à valider la présence de ces techniques de synchronisation le plus tôt possible dans le flot de conception. Au niveau RTL, les temps de traversée des portes logiques n'étant pas définis, la simulation (vérification dynamique) n'est pas applicable : une vérification statique dédiée aux CDC est donc réalisée. Cependant, comme décrit dans l'Introduction de ce manuscrit, cette approche présente, en pratique, des limitations majeures et il n'existe aucune publication la décrivant avec précision. Pour comprendre les raisons des problèmes rencontrés dans l'industrie, il faut analyser la manière dont la vérification des CDC est effectuée et ce, selon trois points de vue :

— Académique ;

— Fournisseurs d'outils ;

— Concepteurs de circuits.

1 La vérification du point de vue académique

Du point de vue académique, la vérification CDC consiste à valider, de manière automatique, les structures implémentées. L'approche dépend de l'environnement de travail : si les structures sont définies dans les spécifications, leur validation peut se faire directement sinon, elles doivent être détectées. Les techniques dédiées à la conception des CDC étant peu nombreuses, la reconnaissance par motifs (pattern matching) [103] et la vérification de modèles (model checking) [70] sont applicables. La vérification CDC se fait donc généralement en deux temps : la détection des structures et la validation des protocoles.

1.1 Le flot de vérification automatique

Détection des structures CDC La *détection des structures CDC* est basée sur la reconnaissance par motifs. Elle consiste à identifier, à l'intérieur d'un circuit, des structures spécifiques aux CDC [17, 76, 96, 98, 105]. Les circuits sont modélisés

par leur description matérielle (au niveau RTL, GATE ou IMPL). Les structures sont décrites par des composants internes aux outils, appelés modèles CDC. La détection des structures consiste à rechercher, dans la description matérielle, les motifs correspondant aux modèles à travers un algorithme appelé engin de filtrage. Lorsqu'une reconnaissance est faite, la structure est rapportée. Pour chaque cas identifié, des informations relatives aux CDC comme les domaines d'horloge ou les registres d'envoi et de capture, sont définies. On dit que la détection génère un modèle instrumenté de la structure. À l'inverse, si aucune structure n'est trouvée, le CDC est rapporté comme problématique : le circuit doit être corrigé. Par exemple, pour une structure de type multi-flop, l'outil va détecter qu'il y a un CDC et que plusieurs éléments séquentiels sont cascadés dans le domaine de destination. Les registres cascadés seront rapportés comme les éléments d'une multi-flop.

Validation des protocoles CDC La *validation des protocoles CDC* est basée sur la vérification de modèles. Elle consiste à prouver des propriétés formelles (décrivant un protocole) sur un modèle de circuit à l'aide d'un engin de preuve [44, 45, 59, 63, 65, 96]. Les modèles sont identifiés lors de la détection des structures CDC. Les propriétés sont décrites dans les bibliothèques de composants internes aux outils. Lorsque les propriétés sont prouvées, le CDC peut être considéré comme correct. Si ce n'est pas le cas, un contre-exemple est renvoyé à travers des chronogrammes : le circuit doit être corrigé. Les propriétés dépendent du type de structure de synchronisation détecté. Par exemple sur une structure de type « Poignée de main », le protocole de séquençage en quatre phases est vérifié.

La figure 3.1 présente le flot de vérification CDC académique. Lors de l'étape de détection des structures, l'engin de filtrage prend en entrée le circuit à vérifier et des bibliothèques de composants à rechercher pour générer des modèles instrumentés correspondant aux structures CDC. S'il détecte un CDC mais pas de structure associée, il rapporte un problème. Lors de l'étape de validation des protocoles, l'engin de preuve prend en entrée les modèles instrumentés et les propriétés formelles à vérifier pour générer un statut de preuve. S'il n'est pas en mesure de prouver les propriétés, il rapporte un contre-exemple.

1.2 Les limitations pratiques

La spécificité des modèles Lors de l'étape de détection, si une structure, correctement implémentée, ne correspond à aucun modèle interne, elle ne peut pas être identifiée. Le CDC est alors rapporté comme problématique et la validation du protocole associé n'est pas effectuée. Par ailleurs, les composants recherchés par l'engin de filtrage ne peuvent pas être modifiés. Si une structure spécifique est implémentée, l'utilisateur doit la définir [8, 73]. Le CDC est ainsi considéré comme correct mais aucune analyse n'est effectuée par l'outil. Cette approche n'est automatique que dans le cas de structures « classiques ».

Les explosions d'états Concernant la validation des protocoles, les modèles identifiés sont locaux mais les signaux d'horloge ne sont pas pris en compte. Or, si la

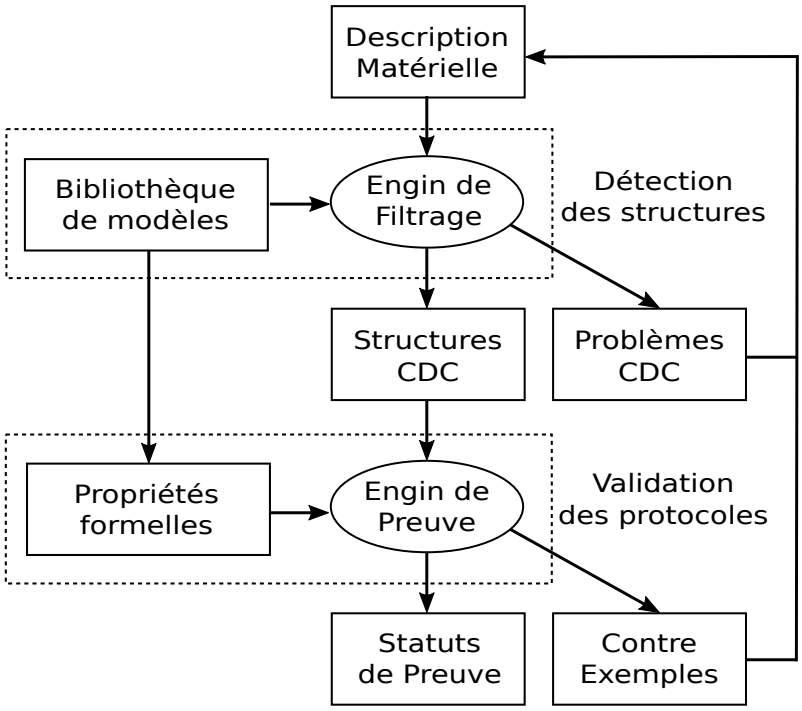


FIGURE 3.1 – Le flot de vérification académique

vérification est faite à haut niveau (top level verification) et non localement (hierarchical level verification), les arbres d'horloge entrent en compte. La complexité du modèle à valider est donc augmentée et la vérification des structures peut être sujette à des explosions d'états [2, 19, 70]. Si la structure n'a pas été vérifiée complètement et qu'aucun contre-exemple n'a été trouvé, le statut de preuve est non-conclusif. Lorsque la validation n'est pas conclusive, aucun résultat n'est renvoyé : la personne en charge de la vérification CDC doit conclure manuellement.

La figure 3.2 présente le flot de vérification académique avec les problèmes associés en rouge.

1.3 Les techniques marginales

Afin de vérifier les CDC le plus rapidement possible sans réduire la couverture des problèmes, d'autres approches de vérification ont été proposées. Ces techniques, bien qu'intéressantes en termes de qualité de résultats, ont un coût de mise en place important et nécessitent la connaissance de toutes les structures CDC d'un circuit pour être utilisées :

— L'utilisation de modèles au niveau système [10, 86]. L'idée est de modéliser les multi-flops par deux registres cascadés et de vérifier les propriétés associées aux protocoles avant le niveau RTL. Cette approche nécessite une connaissance des tous les CDC d'un circuit.

— La définition de propriétés embarquées dans le code RTL [8]. L'objectif est de vérifier les protocoles sans détection des structures. Cette technique de vérification requiert l'identification de tous les CDC d'un circuit, et la spécification

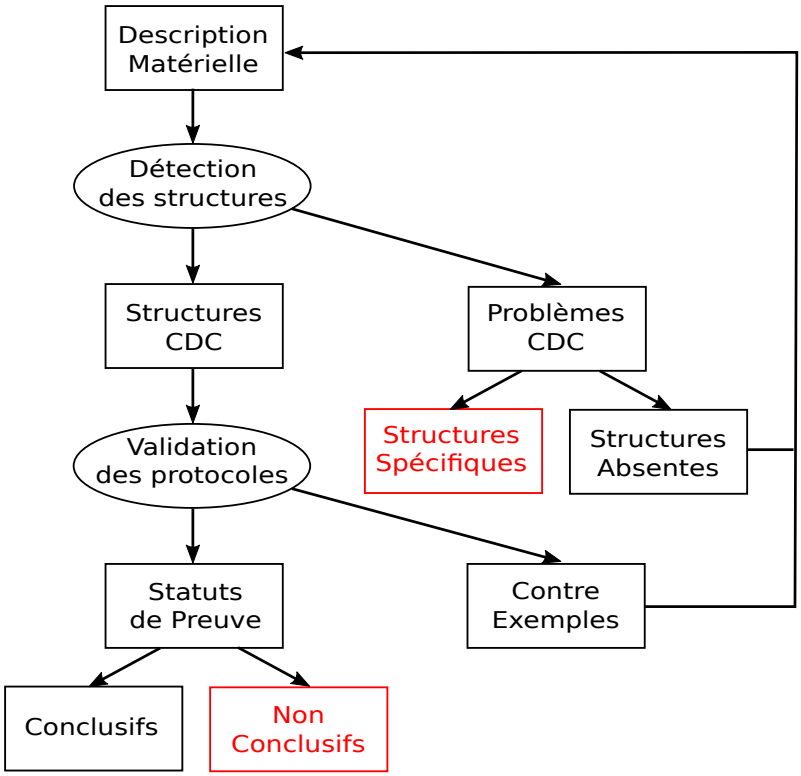


FIGURE 3.2 – Problèmes relatifs au flot de vérification académique

de toutes les propriétés permettant de valider le bon fonctionnement des interfaces asynchrones.

— L'utilisation des outils de STA [16,23]. Cette méthode permet d'identifier tous les domaines d'horloge mais demande le développement manuel de scripts permettant d'identifier les CDC.

— Le test des CDC après fabrication [46,62]. L'idée est d'implémenter de la logique d'auto-diagnostique (Built-in Self Test), dédiée aux CDC, afin de pouvoir détecter des problèmes dus à l'étape de fabrication des circuits. Cette solution présente un coût important en termes de temps de conception et de surface.

2 La vérification pour les fournisseurs d'outils

Les fournisseurs d'outils dédiés à la vérification des CDC ont généralisé la méthode académique. Pour s'affranchir des spécificités liées à la conception des circuits [73,82], ils ont développé une approche de vérification selon trois axes :

— La configuration du moteur de détection [18, 60, 73, 82] ;

— La définition de modèles génériques [50] ;

— Le développement de fonctionnalités de guidage.

2.1 Etude théorique

2.1.1 Le flot de vérification configurable

La configuration des outils Les outils industriels permettent de reconnaître les modèles CDC selon plusieurs niveaux de configuration. Cela assure la détection d'un large spectre de structures possibles. Par exemple, les conditions nécessaires à la détection des multi-flops sont un CDC et des registres en cascade au niveau de la destination. La configuration permet ensuite de définir si la communication entre les registres cascadés doit être synchrone ou si elle peut être quasi-synchrone. Cette approche permet d'adapter les outils aux circuits à vérifier, c'est l'inverse de la méthode de vérification académique.

Les modèles génériques Afin de vérifier le fonctionnement des techniques de synchronisation sans connaître les détails de l'implémentation, les outils ont développé des modèles génériques. L'idée est d'identifier l'architecture minimale requise pour effectuer la validation d'une structure CDC. Les motifs identifiés sont alors rapportés à travers des modèles partiels qui, bien qu'incomplets structurellement, sont vérifiables. Un jeu de propriétés formelles est associé afin de permettre la validation des protocoles. À titre d'exemple, il est possible d'identifier une structure de type "Poignée de main", à travers un modèle de type "Contrôle unique". La vérification du protocole quatre phases sur les signaux de contrôle est alors remplacée par l'analyse du comportement de la donnée transférée (protocole de temporisation). Cette approche permet de reconnaître et valider les structures aussi bien classiques que spécifiques de manière automatique.

Les fonctionnalités de guidage Sur les circuits industriels, la qualité des résultats rapportés dépend de l'environnement de vérification. Une détection exhaustive des structures nécessite la présence de tous les domaines d'horloge et pour un CDC donné, l'identification d'un modèle unique. Si plusieurs modèles sont identifiés, l'outil rapportera le premier rencontré. Par ailleurs, pour assurer un statut de preuve conclusif et correct, il faut que l'aire de la vérification soit réduite aux frontières de l'interface asynchrone. Or, plus la complexité des circuits augmente, plus il est difficile de mettre en place un environnement de vérification adapté. Les outils axent donc leurs développements sur l'ajout de fonctionnalités de guidage. L'idée est uniquement d'assister l'utilisateur dans les étapes de vérification, cela n'augmente pas la qualité des résultats obtenus. Les solutions les plus connues sont :

— Les analyseurs d'arbres d'horloge [74] pour assurer une couverture CDC maximale en identifiant les différents signaux d'horloge et la logique permettant de les contrôler ;

— Les flots hiérarchiques [57, 60, 74] afin de découper l'analyse au niveau bloc et de générer des modèles CDC instrumentés pour réduire la complexité du circuit à haut niveau ;

— Les flots de validation hybride [83] permettant de générer un environnement de simulation avec les propriétés lorsque le statut de validation n'est pas conclusif ;

— Les générateurs de contre-exemples locaux [77] qui comme leur nom l'indique, génèrent une trace d'erreur lorsque la validation est non-conclusive afin d'en

déterminer la raison.

La figure 3.3 présente le flot de vérification du point de vue fournisseurs d'outils. Les étapes de détection des structures et de validation des protocoles peuvent être adaptées aux circuits à vérifier à travers des paramètres d'entrée. Les modèles génériques sont illustrés par les structures partielles en sortie de l'étape de détection. Les fonctionnalités de guidage sont présentées à travers la génération de contre-exemples locaux en sortie de la validation des protocoles.

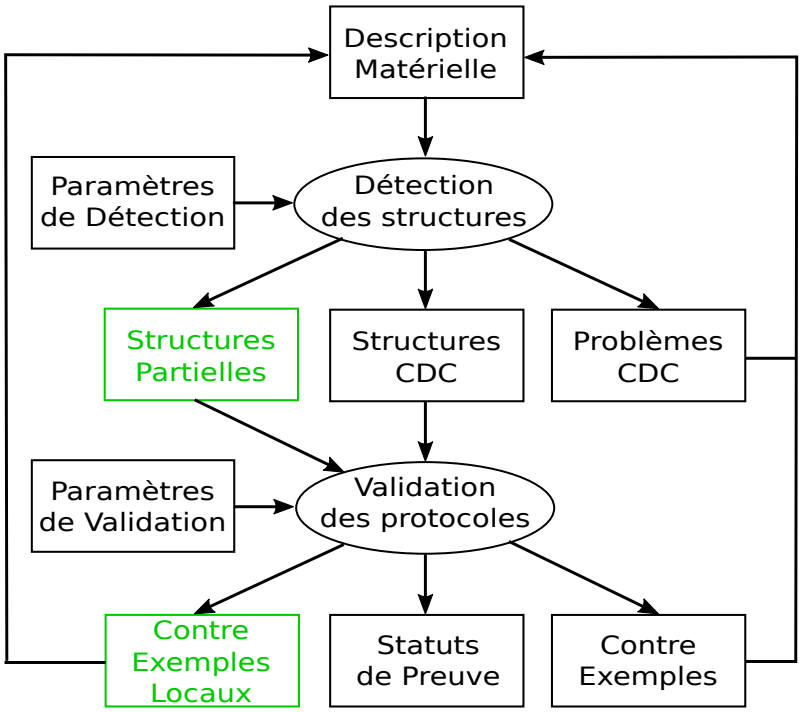


FIGURE 3.3 – Le flot de vérification des fournisseurs d'outils

2.1.2 Les limitations pratiques

La spécificité des configurations Pour un modèle donné, la configuration définie est fixe. L'approche consistant à adapter le moteur de détection est donc limitée aux circuits intégrant des structures de synchronisation uniformes. Si diverses structures spécifiques sont implémentées, plusieurs configurations doivent être définies. Pour une configuration donnée, certaines structures seront identifiées et d'autre non : la vérification CDC doit se faire de manière incrémentale. De plus, en considérant que chaque modèle intègre au moins trois niveaux de configuration, il existe plusieurs dizaines de cas possibles. Il n'est donc pas envisageable, en pratique, de tout tester afin de déterminer lesquels sont les plus adaptés au circuit à vérifier. La configuration est donc une étape manuelle, complexe, et ne répondant pas à tous les cas de figure. Cette approche est adaptée à une analyse locale des CDC mais présente des limitations lors de la vérification d'un circuit à haut niveau.

La détection de fausses structures La méthode de vérification basée sur l'utilisation de modèles génériques peut entraîner la détection de fausses structures. Pour

un outil, les modèles CDC sont soit spécifiques soit génériques. Lorsqu'ils sont génériques, la phase d'identification des structures est effectuée de manière partielle. Si plusieurs signaux potentiels répondent aux conditions de détection, l'outil rapporte le premier de la liste. Pour éviter le problème, il faut utiliser la configuration des modèles qui présente, elle aussi, des limitations [50]. De plus, une analyse CDC utilisant des modèles génériques nécessite une expertise outil car les cas rapportés ne correspondent à aucune structure CDC théorique.

Les explosions d'états Les modèles génériques étant partiels, les frontières de la structure ne sont pas identifiées. Lors de la validation des CDC à haut niveau, l'engin de preuve peut analyser des chemins à l'extérieur de la structure. La complexité du modèle est alors augmentée puisqu'il intègre, en plus de la structure CDC, des éléments de logique externes. La validation peut donc être sujette à des explosions d'états [45], augmentant le temps d'analyse et pouvant générer des statuts de preuve non-conclusifs.

La figure 3.4 présente les problématiques du flot de vérification des fournisseurs d'outils avec la nécessité d'adapter la configuration des moteurs de vérification lorsque les modèles utilisés sont trop génériques ou que les fonctionnalités de guidage rapportent des informations erronées.

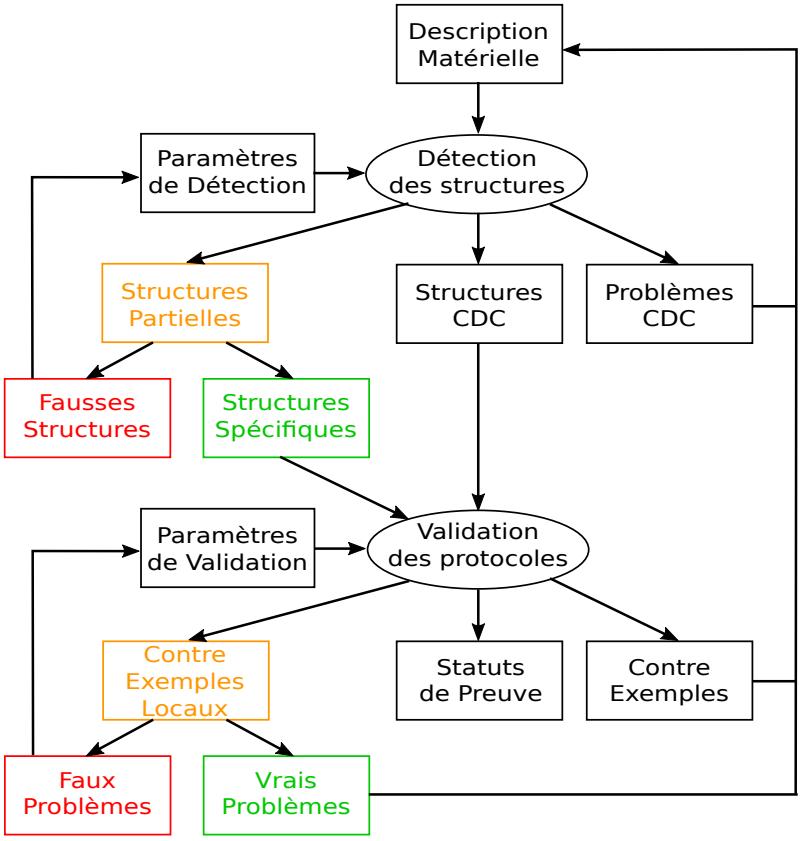


FIGURE 3.4 – Problèmes relatifs au flot de vérification des fournisseurs d'outils

2.1.3 L'injection de fautes en simulation

Le principe Quelles que soient les solutions développées par les fournisseurs d'outils, la vérification CDC n'est pas automatique. Un effort humain est requis, que ce soit pour la configuration des moteurs de détection, l'analyse de l'exhaustivité des modèles génériques détectés ou l'utilisation des fonctionnalités de guidage. Les experts du domaine ont alors proposé une solution permettant d'obtenir de manière automatique des résultats CDC : l'injection de fautes. L'idée est de générer des moniteurs de fautes en sortie des CDC et d'observer en simulation la robustesse du circuit à la métastabilité. La faute correspond à un retard (bonne résolution) ou à une perte de donnée (mauvaise résolution). Cette approche est très appréciée des utilisateurs de par sa simplicité de mise en œuvre. De nombreux travaux sur le sujet sont donc publiés chaque année [11, 64, 66, 67, 79].

Les limitations pratiques Bien qu'automatique dans l'utilisation des outils de vérification CDC, cette approche présente des limitations quant à la qualité des résultats obtenus.

L'analyse des CDC consistant à prouver [11, 67] un fonctionnement correct des interfaces asynchrones, la simulation d'un circuit avec des fautes ne suffit pas : des propriétés formelles doivent être vérifiées. La génération de propriétés CDC nécessitant l'identification des structures par les outils, les limitations liées à l'utilisation des engins de détection ne sont pas résolues par cette approche.

Par ailleurs, l'injection de fautes ne fait que déplacer le problème de mise en place de l'environnement de vérification à l'étape suivante. Les fautes étant injectées en simulation, la qualité des résultats obtenus est dépendante des scénarios et du temps de vérification. Par définition, un environnement de validation formelle est exhaustif. À l'inverse, un environnement de simulation exhaustif nécessite la définition de tous les scénarios possibles [19].

Pour finir les résultats rapportés par cette méthode de vérification ne sont pas suffisants car le comportement des fautes injectées est aléatoire. Afin d'obtenir une qualité de résultats correcte, les fautes injectées doivent dépendre de la structure à valider [101]. La définition de moniteurs déterministes est plus complexe que mettre en place un environnement de vérification formelle.

L'intérêt pratique Cette approche est donc simplement intéressante pour augmenter la couverture de vérification fonctionnelle mais n'augmente pas la qualité des résultats CDC. Elle peut être utilisée pour couvrir rapidement les cas difficiles à vérifier formellement et détecter certains problèmes réels. Cet aspect de la vérification CDC ne sera donc pas abordé dans la suite de ce document. La figure 3.5 présente le flot de vérification dynamique des CDC. Les outils de vérification CDC sont utilisés uniquement pour la génération de fautes et de propriétés formelles dédiées aux CDC. L'étape de vérification est réalisée avec un outil dédié à la simulation des circuits et requiert l'utilisation d'un environnement précis (banc de test).

2.1.4 Les nouveaux axes de développement

Le développement actuel des outils de vérification CDC est orienté sur deux axes. Le premier est la prise en compte de la norme UPF (Unified Power Format) et

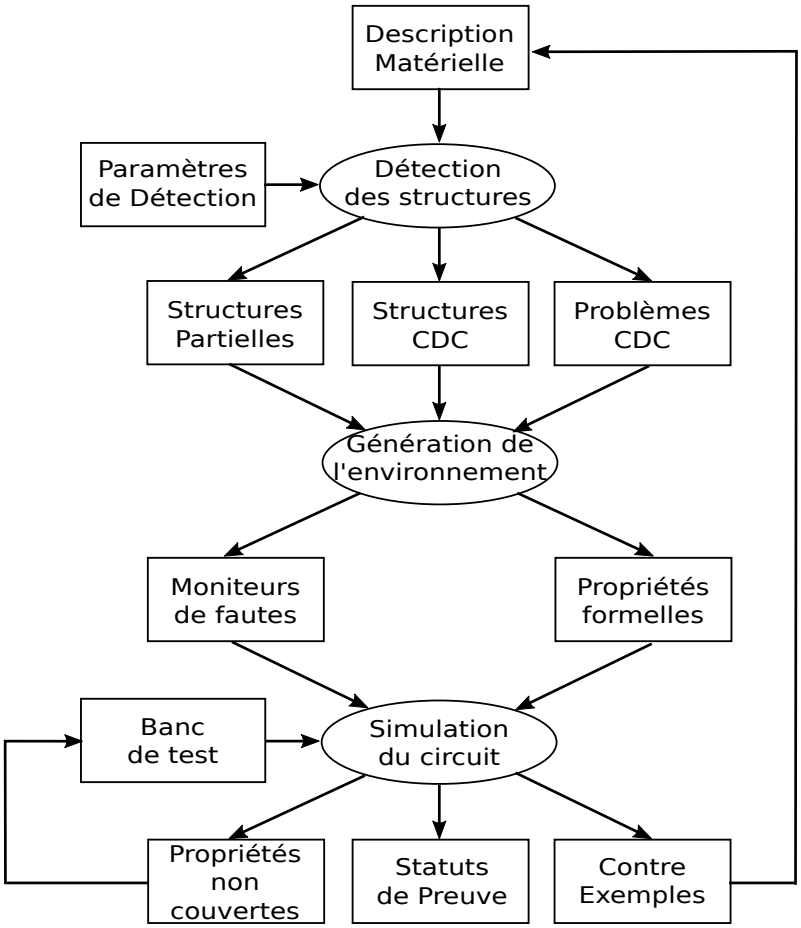


FIGURE 3.5 – Le flot de vérification dynamique des CDC

consiste à vérifier, dès le niveau RTL, les structures utilisées pour la réduction de la consommation d'énergie sur les signaux sujets à CDC [38,100,106]. L'idée est d'anticiper l'apparition de problèmes CDC dus à la synthèse de ces éléments au niveau GATE. Le deuxième est la vérification des RDC (Reset Domain Crossing) et permet de couvrir le cas des communications synchrones entre registres ne recevant pas le même signal de reset asynchrone [61,106]. L'idée est de couvrir des interfaces dont la problématique et les techniques de conception sont proches des CDC et qui ne sont pas pris en compte par les autres types de validation. Ces deux nouvelles fonctionnalités consistant à analyser des structures supplémentaires, les limitations relatives à la détection et la validation des modèles, citées précédemment, s'appliquent.

2.2 Etude comparative des différents outils du marché

Le marché de la conception assistée par ordinateur ou EDA (Electronic Design Automation), compte trois principaux fournisseurs d'outils : Cadence, Mentor Graphics et Synopsys. Chaque acteur propose une solution dédiée à la vérification statique des CDC [35,39,95]. Bien que l'approche et les fonctionnalités proposées soient similaires à ce qui a été présenté dans la sous-section précédente, les engins de filtrage et de preuve sont différents. Afin de déterminer les solutions les plus adaptées aux circuits CPUSS, plusieurs outils du marché doivent être comparés et étudiés.

2.2.1 La mise en place de l'étude

Le circuit Les essais sont réalisés sur un sous-système CPUSS (section 2, page 8) intégrant un processeur multi-cœur 64 bits de dernière génération présenté dans la partie "Préambule". En termes de CDC, les caractéristiques du circuit sont :

— 12 domaines d'horloge sujets à CDC ;

— 24 interfaces asynchrones fonctionnelles ;

— 14 interfaces synchronisées sur-mesure.

Les aspects étudiés L'analyse des CDC est découpée en plusieurs étapes afin de couvrir les phases de détection des structures et de validation des protocoles. Les fonctionnalités dédiées au guidage et à l'injection de fautes n'ayant pas d'impact réel sur la qualité des résultats, elles ne sont pas analysées. La reconnaissance structurelle est divisée en deux parties : les domaines d'horloge et les structures CDC. La validation des protocoles est étudiée sous deux aspects : les propriétés vérifiées et les résultats de vérification.

L'approche choisie Pour assurer des résultats comparables, l'approche de vérification est commune : les protocoles de séquençage sont détectés et validés à travers des modèles génériques. Les paramètres outils sont définis afin de permettre la reconnaissance de toutes les interfaces (classiques ou sur-mesure) de manière automatique et exhaustive. Les temps d'analyse ne sont pas pris en compte car l'étude porte sur la qualité des résultats obtenus. La figure 3.6 présente l'approche de vérification mise en place pour effectuer l'évaluation des outils.

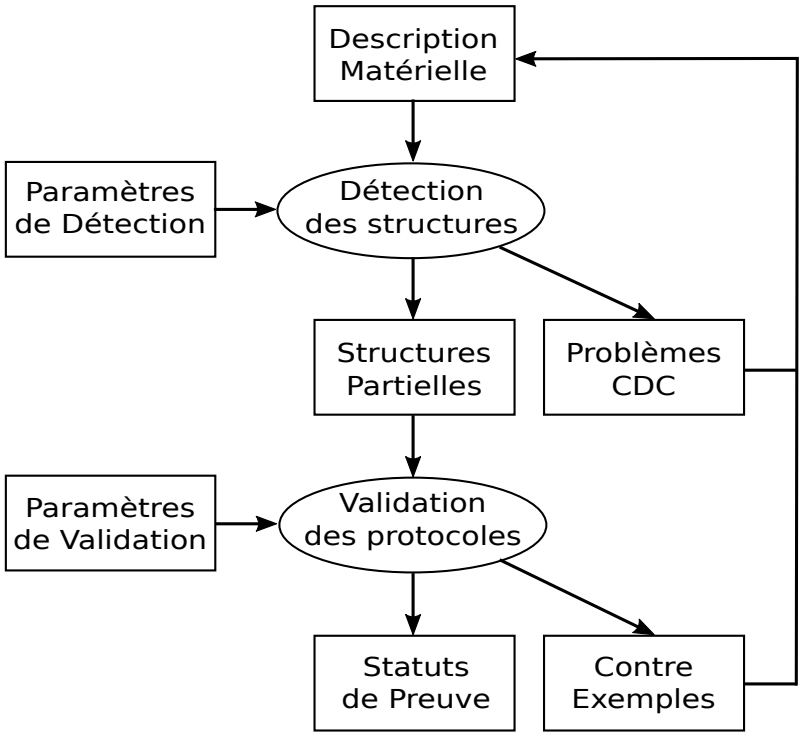


FIGURE 3.6 – L'approche de vérification pour l'évaluation des outils

2.2.2 Les résultats obtenus

Description du tableau Le tableau 3.1 présente les résultats obtenus. L'analyse des arbres d'horloges est notée CC (Clock Check) et la détection des structures est notée SC (Structural Check). Les propriétés vérifiées sont notées FP (Formal Properties) et les résultats de validation sont notés FC (Formal Check). Lorsque les résultats sont corrects OK est rapporté ; lorsqu'ils sont partiels POK est rapporté et lorsqu'ils sont incorrects NOK est rapporté. Enfin si la solution n'a pas été évaluée N.A. (Non Applicable) est renvoyée. Les colonnes 6 et 7, notées TP (Tools Parameters) et ES (Environment Setup), présentent respectivement le nombre de paramètre à configurer et le nombre de frontières à placer sur les modèles CDC rapportés pour obtenir des résultats avec un niveau de qualité maximal. Le total des paramètres à configurer est rapporté dans la dernière colonne. Afin que le comparatif reste confidentiel, le nom des outils n'est pas divulgué.

1	2	3	4	5	6	7	8
Solutions CDC	CC	SC	FP	FC	TP	ES	Total
Outil 1	POK	POK	POK	OK	51	58	109
Outil 2	POK	POK	POK	OK	39	65	104
Outil 3	NOK	NOK	POK	POK	20	63	83
Outil 4	NOK	POK	POK	N.A.	27	N.A.	27
Outil 5	NOK	NOK	POK	POK	17	63	80

TABLE 3.1 – Résultats de vérification CDC avec le flot outils

Analyse des résultats La première information visible est qu'il n'y a pas d'outil de vérification CDC permettant d'obtenir automatiquement une qualité de résultats optimale : toutes les solutions ont des POK à au moins deux étapes.

Les solutions 3, 4 et 5 présentent des limitations techniques majeures (NOK) sur l'analyse des arbres d'horloge (colonne CC). De plus, les outils 3 et 5 n'intègrent pas de modèles suffisamment configurables ou génériques pour permettre une reconnaissance automatique des structures spécifiques : la détection (SC) est incorrecte (NOK). Notons aussi que l'outil 4 permet de générer des propriétés (FP) mais ne propose pas de validation (FC) : cette étape n'est pas applicable (N.A.).

Seules les solutions 1 et 2 permettent d'effectuer une analyse CDC avec un niveau de qualité acceptable (pas de NOK). Cependant, ils présentent des résultats en détection (CC et SC) et en génération de propriétés (FP) partiels (POK).

La qualité de résultats rapportée pour chaque étape ne peut être obtenue qu'en effectuant une configuration précise des outils (TP) et en mettant en place un environnement de validation spécifique (ES). Bien que les outils 1 et 2 présentent les meilleurs résultats en termes d'analyse CDC, il y a plus d'une centaine de paramètres à configurer. Ce travail nécessite donc une connaissance des outils et un effort humain très important.

Conclusion Cette étude montre qu'aucun outil du marché ne présente des résultats de qualité pour la vérification CDC sans un effort humain. Ni l'approche de vérification académique, ni celles des outils n'est automatique. Le temps de développement des circuits est contraint par le marché des semi-conducteurs. Les industriels de la conception ont donc développé leur propre approche pour effectuer la vérification des CDC le plus rapidement possible.

3 La vérification pour les concepteurs de circuits

Du point de vue des concepteurs de circuits, la vérification CDC consiste à valider les structures implémentées avant l'envoi du circuit en fabrication. Ce travail est contraint par deux exigences : le temps de mise sur le marché et le flot de conception. L'analyse CDC doit donc se faire le plus rapidement possible et doit être exécutée sur les différents niveaux de modélisation du circuit. En pratique, il y a généralement un compromis à faire entre la qualité des résultats et le temps d'analyse. L'approche consiste à avoir une qualité maximale au niveau RTL et de faire des tests de non-régression aux niveaux GATE et IMPL : c'est le flot horizontal. Pour effectuer une non-régression rapide, une partie des informations est réutilisée entre deux versions du circuit. Ces informations intègrent la configuration des paramètres outil, l'environnement de vérification mais aussi des informations fournies directement par l'utilisateur à partir des spécifications techniques.

3.1 Etude théorique

3.1.1 Le flot de vérification manuel

La problématique outil Les méthodes de vérification des CDC, que ce soit du point de vue académique ou des fournisseurs d'outils, ont une approche de vérifica-

tion binaire : soit les structures CDC sont présentes et les protocoles sont validés, soit elles sont absentes et doivent être ajoutées par les concepteurs. En pratique, un troisième cas existe : les structures sont absentes car les CDC sont gérés à l'extérieur du circuit. De même, pour la validation, le comportement de certains signaux peut être assuré par le protocole d'utilisation du circuit : la logique est externe. Il y a donc deux types de structures : celles pouvant être automatiquement validées par un outil et celles dont une partie de la logique est à l'extérieur du circuit [26,57,60,73]. Pour ces dernières, même un outil parfait n'aurait pas assez d'informations pour effectuer une preuve : un problème est rapporté dans tous les cas. Au sein des circuits industriels plusieurs communications asynchrones sont gérées de l'extérieur, notamment celles dédiées au test des circuits après fabrication. La figure 3.7 présente les problèmes liés au flot de vérification des CDC pour l'utilisateur. Les faux problèmes identifiés par les outils sont indiqués en rouge. Ces cas sont dus à un manque d'informations sur le comportement de signaux gérés à l'extérieur (entrées primaires).

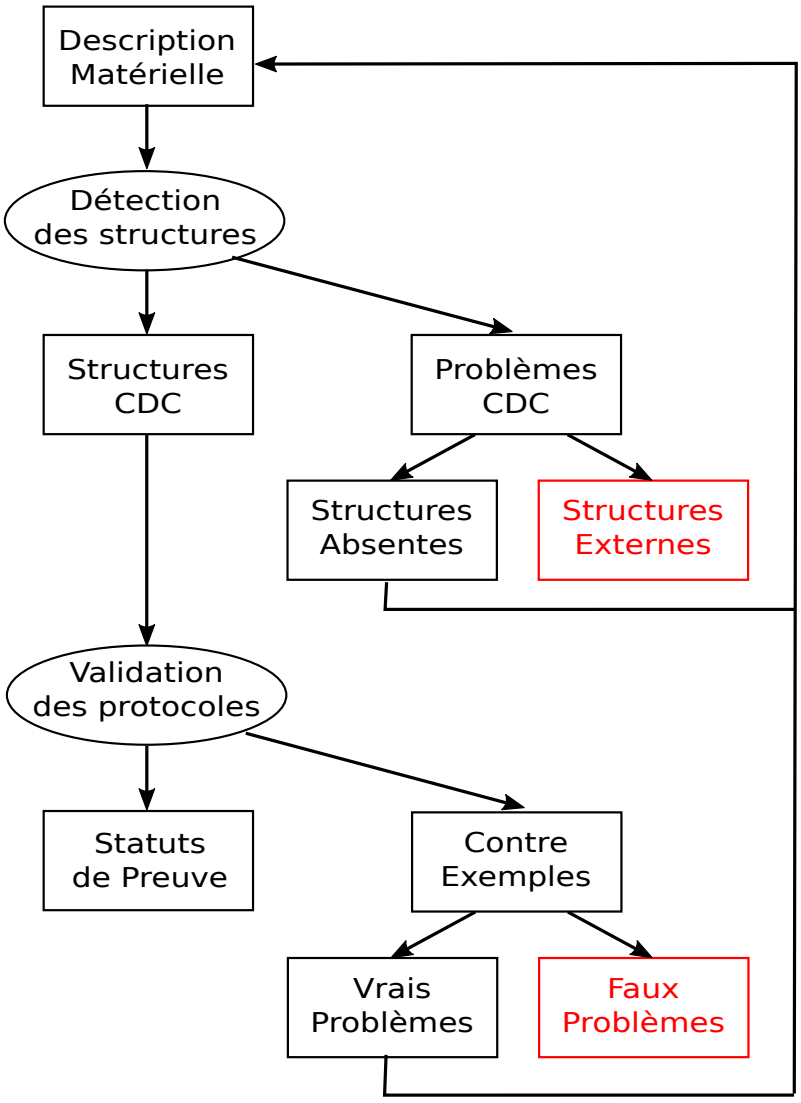


FIGURE 3.7 – Les problèmes de flot pour l'utilisateur

La définition de contraintes Les outils dédiés à la vérification CDC offrent, à l'utilisateur, la possibilité de se substituer aux engins d'analyse en donnant des informations, appelées contraintes. L'idée est d'éviter le rapport de faux problèmes dus à l'absence d'une structure. Lorsqu'elles sont appliquées à l'engin de filtrage, on parle de contraintes de détection et, lorsqu'elles sont appliquées à l'engin de preuve, on parle de contraintes de validation. Pour spécifier ces contraintes, trois approches sont possibles :

— L'exclusion des CDC [26, 73]. L'utilisateur indique que ces signaux ne doivent pas être analysés.

— La spécification du comportement des CDC [26, 37, 75]. L'utilisateur indique que ces signaux sont gérés à l'extérieur à travers un protocole de temporisation.

— La déclaration d'un signal de contrôle venant de l'extérieur. L'utilisateur indique que les signaux sont gérés à l'extérieur à travers un protocole de séquençage.

Les types de contraintes Ces trois approches n'ont pas le même impact et demandent un effort de spécification différent.

L'exclusion des CDC est une contrainte forte, l'outil n'analyse pas les chemins entre deux domaines d'horloge. Sa mise en place est simple, il suffit de spécifier les domaines d'envoi et de réception.

La spécification du comportement des CDC est une contrainte moyenne, l'utilisateur spécifie des sorties de registres ou des ports d'entrée primaires. S'ils sont sujets à CDC, l'outil vérifie que le comportement spécifié correspond à une structure manquante et rapporte les signaux comme contraints par l'utilisateur. Une analyse CDC est réalisée et les contraintes utilisateur sont triées. La définition du comportement se fait au niveau des registres ce qui est assez long.

La déclaration des signaux contrôlant des CDC est une contrainte faible. Sa mise en place requiert la définition des sorties de registres (ou ports d'entrée primaires) ainsi que les CDC qu'ils contrôlent. Les outils analysent ensuite si la logique présente entre le signal de contrôle et les CDC spécifiés respecte certaines conditions : si oui la contrainte est acceptée sinon elle est refusée. En termes d'effort de spécification, cette approche nécessite une analyse préalable de la structure et, si besoin, une configuration de l'outil, avant de définir les informations.

Il est donc recommandé d'utiliser des contraintes faibles pour les protocoles de séquençage et des contraintes moyennes pour les protocoles de temporisation plutôt que des contraintes fortes. Cependant, en fonction du temps alloué à la vérification des CDC, cela peut varier. Quelles que soient les contraintes spécifiées, elles peuvent être réutilisées d'une version à une autre du circuit et sont donc compatibles avec le flot horizontal.

3.1.2 Les limitations pratiques

Le temps de spécification La définition de contraintes utilisateur nécessite la présence de documentations techniques spécifiques aux interfaces asynchrones. Si ces dernières sont partielles ou inexistantes, l'effort requis peut être très important voire incompatible avec le temps alloué au développement des projets. À titre d'exemple, pour un CPUSS la mise en place de contraintes avec des spécifications techniques

prend plusieurs semaines et sans plusieurs mois. De plus, ces informations étant spécifiées par l'utilisateur, il y a un risque d'erreur humaine important, particulièrement en cas de contraintes fortes.

La mise en place des contraintes La spécification de contraintes faibles n'est possible que dans le cas de protocoles de séquençage avec un signal de contrôle à l'intérieur du circuit. Si ce n'est pas le cas, il faut mettre en place une contrainte moyenne ou forte. Pour établir les notions de contraintes fortes, moyennes et faibles, les informations utilisateur disponibles dans les outils doivent être formellement définies. En pratique, c'est rarement le cas, il est donc nécessaire de faire des tests pour connaître l'impact de chacune sur la vérification des CDC. Dès lors, de nombreux utilisateurs préfèrent utiliser des contraintes fortes malgré le risque de réduction de la couverture CDC.

La figure 3.8 présente le flot de vérification des CDC du point de vue d'un concepteur de circuits. Lorsque les outils détectent de faux problèmes, l'utilisateur doit ajouter des informations à travers des contraintes de détection et de validation.

3.2 Etude comparative des différents outils du marché

La vérification statique des CDC est réalisée avec des outils d'analyse automatique. Néanmoins la qualité des résultats rapportés n'est pas optimale et nécessite la définition de contraintes par l'utilisateur pour obtenir des informations conformes aux les spécifications techniques. Étant donné que l'analyse des interfaces asynchrones est limitée par le temps de mise sur le marché, les étapes manuelles doivent être minimisées. Afin de déterminer l'intérêt de cette approche, il est important d'évaluer les contraintes proposées par les outils et la qualité de résultats obtenus.

3.2.1 La mise en place de l'évaluation

Le circuit et les aspects étudiés Comme pour l'étude comparative du flot de vérification proposé par les fournisseurs outils, le circuit utilisé est le CPUSS (section 2, page 8) et l'analyse CDC est découpée en quatre parties :

- L'analyse structurelle des arbres d'horloge ;
- La détection des structures CDC ;
- Les propriétés vérifiées ;
- Les résultats de validation des protocoles.

L'approche choisie Pour assurer des résultats comparables, l'approche de définition manuelle des informations est commune. Les protocoles de séquençage gérés depuis l'extérieur sont contraints avec des informations faibles. Les protocoles de temporisation gérés depuis l'extérieur sont contraints avec des informations moyennes. Le comportement des entrées primaires ayant un impact sur la validation des protocoles est contraint avec des propriétés. La figure 3.9 présente l'approche de vérification mise en place pour effectuer l'évaluation des outils.

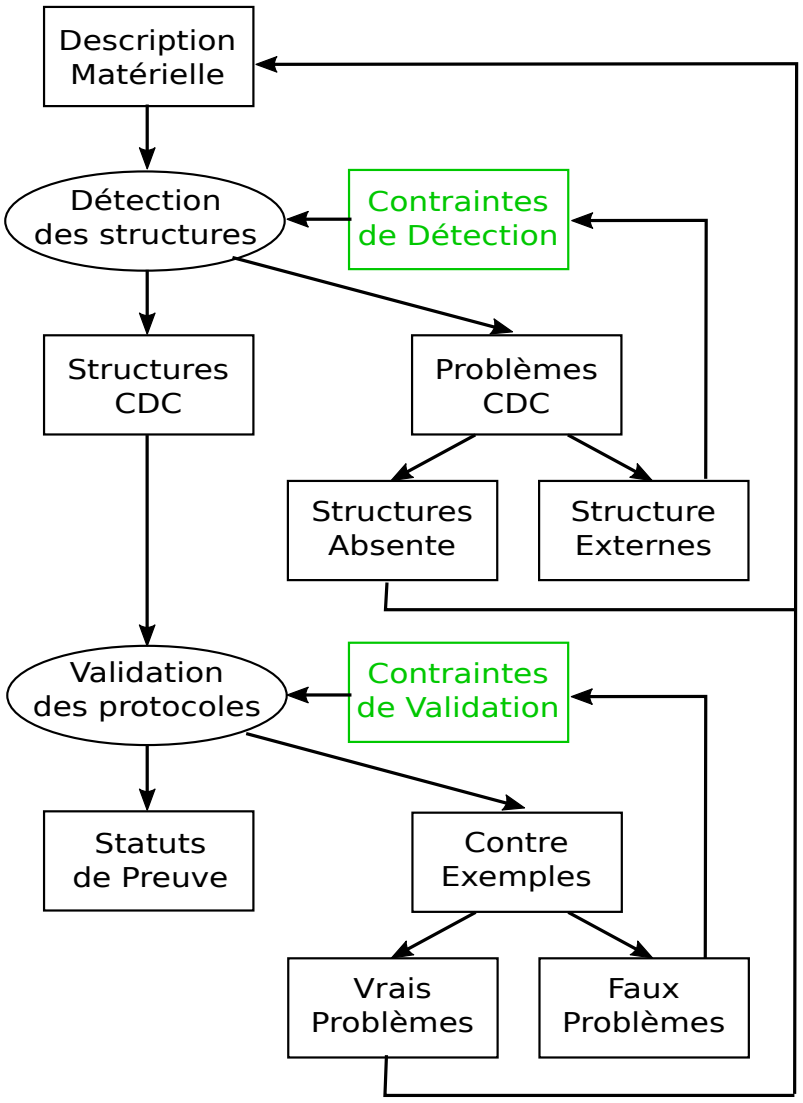


FIGURE 3.8 – Le flot de vérification du concepteur de circuits

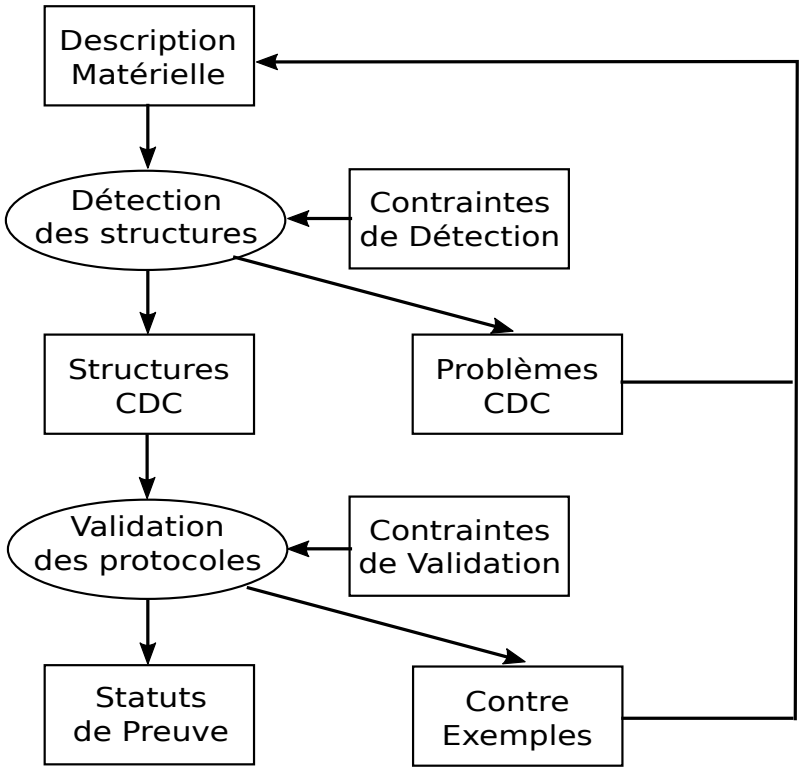


FIGURE 3.9 – L'approche de vérification pour l'évaluation des outils

3.2.2 Les résultats obtenus

Description du tableau Le tableau 3.2 présente les résultats obtenus. Les cinq premières colonnes sont les mêmes que celles du tableau 3.1. Les colonnes 6, 7 et 8, notées RC (Robust Constraints), DC (Définition Constraints) et EC (Exclusion Constraints) présentent respectivement le nombre de contraintes faibles, moyennes et fortes spécifiées. La dernière colonne donne la quantité totale de contraintes à ajouter pour obtenir des résultats optimaux.

1	2	3	4	5	6	7	8	9
Solutions CDC	CC	SC	FP	FC	RC	DC	EC	Total
Outil 1	OK	OK	OK	OK	62	183	2	247
Outil 2	OK	OK	OK	OK	N.A.	324	3	327
Outil 3	OK	OK	POK	POK	N.A.	324	21	345
Outil 4	OK	OK	OK	N.A.	N.A.	299	3	302
Outil 5	OK	OK	POK	POK	N.A.	324	21	345

TABLE 3.2 – Résultats de vérification CDC avec le flot utilisateur

Analyse des résultats La première information visible est que toutes les solutions nécessitent l'ajout de contraintes. Par ailleurs, tous les outils détectent de fausses structures nécessitant la spécification manuelle d'exclusions (EC).

Les solutions 3 et 5 présentent des limitations (POK) malgré la spécification de contraintes. Les structures spécifiques n'étant pas reconnues automatiquement,

elles doivent être spécifiées par l'utilisateur et ne peuvent donc pas être validées par l'outil. Par ailleurs elles ne proposent pas de contraintes faibles (RC).

Seules les solutions 1, 2 et 4 permettent d'effectuer une vérification des CDC complète avec une qualité de résultats optimale (OK) à l'aide d'un jeu de contraintes. Cependant, la solution 4 ne permettant pas d'effectuer une validation, elle ne couvre pas l'intégralité du flot de vérification CDC.

La solution 2 ne propose pas de spécifier des contraintes faibles (RC) car elle est en mesure de les détecter automatiquement et de générer les propriétés formelles permettant de les valider. Cependant elle présente des limitations dans la détection des structures nécessitant l'ajout de contraintes moyennes (DC) pour permettre leur reconnaissance.

Pour obtenir des résultats conformes aux spécifications techniques (OK partout), il est nécessaire de définir des contraintes précises aux outils (RC, DC, EC). Bien que les outils 1 et 2 présentent les meilleurs résultats en termes d'analyse CDC, plusieurs centaines de contraintes doivent être définies. Pour éviter toute erreur humaine, ce travail nécessite, là encore, une connaissance des outils et un effort d'analyse très important.

4 Conclusion

L'analyse des approches utilisées pour effectuer la vérification des CDC selon les différents points de vue a montré que quel que soit le flot utilisé, il n'est pas possible d'obtenir automatiquement des résultats de qualité. Les modèles CDC internes aux outils doivent être configurés et les interfaces asynchrones gérées à l'extérieur doivent être contraintes.

Vérifier les CDC ne consiste donc pas uniquement à étudier le fonctionnement des communications asynchrones, l'utilisateur doit aussi être en mesure d'identifier la raison pour laquelle les problèmes sont rapportés par les outils et de spécifier si besoin une configuration précise ou des informations supplémentaire de manière manuelle. Dans ce contexte la vérification CDC ne demande pas simplement des compétences en conception des CDC mais nécessite une connaissance des outils, des méthodes de vérification et du fonctionnement du circuit.

La figure 3.10 présente le flot de vérification des CDC complet. Pour assurer des résultats exhaustifs, l'utilisateur doit à la fois configurer les moteurs de vérification (détection des structures et validation des protocoles) et donner des informations sur le comportement de certains signaux à travers des contraintes.

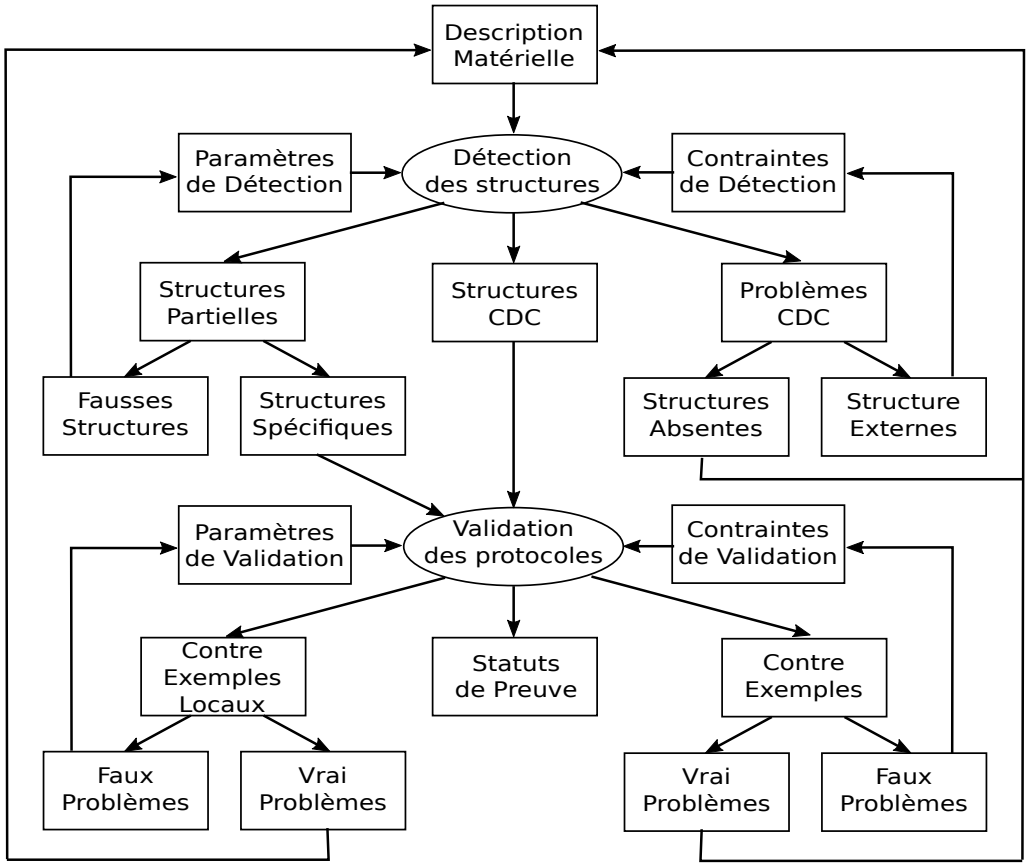


FIGURE 3.10 – Le flot de vérification complet des CDC

Chapitre 4

Analyse critiques des solutions développées et propositions

Les méthodes de vérification des CDC présentent des limitations pratiques (sous-sections 2.1.2 et 2.2.2, pages 46 et 51). Des interventions manuelles sont alors nécessaires pour obtenir des résultats de qualité. Le marché des circuits intégrés numériques étant extrêmement compétitif, il est indispensable de pouvoir effectuer l'analyse des interfaces asynchrones de manière automatique et avec un niveau de qualité optimal. Des solutions ont été développées par les trois acteurs du domaine mais, leurs points de vue étant différents, aucune n'est adaptée au contexte industriel. Il est donc essentiel de trouver de nouvelles solutions permettant d'automatiser la vérification des CDC.

1 Les solutions développées

1.1 Académique : la génération de structures CDC

L'objectif Le chercheur ayant pour but de détecter et de vérifier des cas théoriques, il ne considère pas les structures spécifiques. La vérification CDC du point de vue académique consiste à adapter les circuits aux modèles définis dans les outils.

L'axe de développement L'approche consiste à anticiper les étapes de vérification à travers la génération automatique de circuits au niveau RTL [12]. Pour les CDC, les structures générées sont des protocoles de séquençage corrects par construction. Dès lors, les structures implémentées sont connues et peuvent être identifiées et validées automatiquement par les outils de vérification.

La limitation La conception de structures spécifiques est généralement due à un contexte particulier. Les structures de départ sont donc classiques et, en fonction des besoins (performances, surface ou consommation d'énergie) des modifications sont apportées. Si une structure générée est validée automatiquement, elle devra l'être à nouveau après modification et ce problème n'est pas résolu. L'intérêt est donc simplement d'assurer l'implémentation d'une interface asynchrone fonctionnant correctement mais le problème de reconnaissance des structures spécifiques est toujours présent.

1.2 Outil : la vérification dynamique

L'objectif Le fournisseur d'outils a pour but de détecter et de vérifier toutes les structures possibles. Il développe donc des modèles génériques ou configurables. La vérification CDC du point de vue outils consiste à adapter les modèles aux circuits à vérifier.

L'axe de développement L'objectif recherché est d'obtenir automatiquement un statut de vérification sur toutes les structures quelle que soit l'approche. Étant donné le problème d'explosions d'états lors de la vérification formelle, les développements sont axés sur la vérification dynamique. L'approche consiste à générer, à partir d'une détection de modèles génériques, des fautes aléatoires et des propriétés relatives aux CDC destinés à être simulés [66, 83].

Les limitations Cependant, cette solution n'assure ni une couverture de tous les problèmes CDC ni une qualité de résultats optimale. De plus, elle nécessite un environnement de simulation adapté.

1.3 Concepteur : la génération de contraintes

L'objectif Le concepteur cherche à vérifier les structures CDC le plus rapidement possible. Puisque la validation doit être réalisée tout au long du flot de conception, l'approche consiste à spécifier des informations réutilisables entre deux vérifications.

L'axe de développement Les travaux d'automatisation cherchent à accélérer le processus de spécification d'informations. L'approche réside dans le développement de scripts permettant la génération automatique de contraintes à partir de spécifications sous format tabulaire ou de résultats de simulation [26].

La limitation La qualité des résultats CDC, dépend du type de contraintes spécifié. Générer des contraintes à partir de tableaux remplis par les personnes en charge de la conception est risqué. Si des contraintes fortes (exclusions) sont définies, l'analyse des structures n'est plus effectuée par les outils. De plus, les concepteurs ne tiennent pas forcément compte de tous les problèmes CDC, c'est la raison pour laquelle la vérification doit être réalisée. Cette méthode bien que réutilisable peut réduire la couverture de la vérification CDC. Il est préférable pour les ingénieurs en charge de la conception d'indiquer les protocoles d'utilisation des interfaces asynchrones.

1.4 Bilan

Le développement des approches de vérification des CDC tend vers une automatisation au détriment de la qualité des résultats. Les solutions proposées consistent simplement à se substituer aux étapes manuelles considérées par les différents points de vue. Pour le chercheur, une structure n'est pas validée parce qu'elle est incorrecte donc l'étape manuelle consiste à corriger la structure. Pour le fournisseur outil, une structure n'est pas validée parce que le circuit est trop complexe, l'étape manuelle

consiste à simuler le circuit. Pour le concepteur de circuit, une structure n'est pas validée parce qu'il manque des informations, l'étape manuelle consiste à les ajouter. Les travaux de développement cherchent donc à résoudre les problèmes selon leur point de vue, sans essayer d'en comprendre la cause.

2 Le problème des modèles CDC

Les solutions proposées pour éviter les étapes manuelles lors de la vérification CDC se limitent à les automatiser. Cependant, au regard des limitations des trois approches, les étapes manuelles sont dues à une divergence entre les modèles CDC internes aux outils et les structures implémentées dans les circuits.

les modèles académiques Concernant les modèles académiques, ils sont trop théoriques. Les conditions permettant leur reconnaissance sont trop restrictives. Ainsi, si des structures implémentées ne répondent pas à toutes les conditions de détection, elles ne sont pas reconnues : un problème est rapporté. Des modèles plus génériques doivent être développés.

les modèles outils Les outils industriels intègrent des modèles qui sont trop génériques. Les conditions permettant leur reconnaissance sont trop permissives. Si une structure n'est pas bien implémentée mais qu'elle respecte les conditions de détection, un faux modèle sera reconnu. De plus, les modèles ne correspondent qu'à des structures partielles, l'utilisateur doit donc connaître son circuit pour effectuer son analyse.

les modèles des concepteurs Les modèles associés aux structures des concepteurs sont généralement atypiques. De plus, les attentes du marché étant en évolution, les implémentations changent régulièrement en fonction de la demande. Il n'est donc pas approprié de développer des modèles dédiés car cette étape devra être répétée à chaque modification de structure.

Les composants implémentés dans les outils CDC sont imparfaits du fait que les modèles académiques sont trop fermés, et que les modèles outils sont trop ouverts. La solution serait de développer des modèles dont les conditions de détection sont situées entre les deux approches. Notons qu'aucun travail n'est fait pour corriger les modèles CDC. Cependant, comprendre les raisons du problème ne suffit pas à proposer des solutions intéressantes. Plus il y a de modèles dans les outils, plus il est difficile pour l'utilisateur de les configurer. Il faut donc proposer des solutions intégrées dans un flot de vérification des CDC.

3 Proposition de développement

Dans ce contexte, nous avons choisi de nous intéresser au problème majeur de la vérification CDC : les modèles outils. L'idée est de comprendre ce qui est fait afin de proposer des axes de développement assurant l'obtention d'une qualité de résultats optimale de manière automatique.

3.1 Les acteurs du projet

Au-delà des aspects théoriques, il faut pouvoir déterminer la faisabilité des travaux. L'idée est de proposer des axes d'amélioration des outils de vérification CDC pour un industriel de la conception : STMicroelectronics. Cependant, les outils utilisés sont développés par des entreprises privées de l'EDA et, il n'est pas possible de modifier ou ajouter des modèles. Une collaboration a donc été mise en place entre les trois points de vue afin de développer des outils adaptés :

— Académique : Laboratoire TIMA ;

— Outil : Synopsys ;

— Concepteur : STMicroelectronics.

Le laboratoire TIMA est en charge de proposer des solutions théoriques aux problèmes pratiques. L'entreprise Synopsys implémente des solutions au sein d'outils CDC industriels. L'entreprise STMicroelectronics a pour objectif de tester les solutions proposées et, si besoin, de proposer des améliorations.

3.2 Les objectifs du projet

Les travaux effectués traitent la problématique côté concepteur de circuits. L'objectif est donc de proposer des solutions permettant d'effectuer la vérification des CDC de manière complète et automatique tout en assurant des résultats conformes aux spécification techniques quelles que soient les structures implémentées. Cependant, il n'est pas possible d'accéder aux modèles des outils que ce soit pour les comprendre ou pour les modifier. L'approche va donc être de suivre le flot de vérification des outils et, pour chaque étape, de déterminer les motifs internes, de comprendre les limitations observées, et de proposer des solutions ni trop génériques ni trop spécifiques. Le flot de vérification des CDC dans les outils est décomposé en quatre étapes :

— L'analyse des arbres d'horloge ;

— L'analyse des CDC avec multi-flop ;

— L'analyse des protocoles de séquençage ;

— L'analyse des signaux non synchronisés.

4 Conclusion

Les solutions proposées respectivement par les chercheurs, les fournisseurs outils et les concepteurs de circuits pour vérifier automatiquement les CDC ne donnent pas des résultats satisfaisants. La raison principale est qu'elles ne tiennent pas compte du fond du problème : la précision des modèles. Pour répondre à cette problématique, nous avons mis en place une collaboration entre les trois acteurs. Les travaux présentés sont réalisés à travers le point de vue du concepteur. Ils vont donc consister à formaliser les modèles implémentés dans les outils, à détailler les principales limitations et à proposer de nouveaux modèles à vérifier. Ces modèles seront définis par des conditions de détection et des propriétés formelles associées.

Deuxième partie

Automatisation des techniques de vérification CDC

Chapitre 5

Description de la démarche

Comme expliqué dans le chapitre 4 (sous-section 3.2, page 64), nous étudions les problèmes outils et proposons des solutions en qualité d'industriel de la conception de circuits. Dès lors, l'objet de notre étude n'est pas le code interne aux solutions logicielles mais l'approche de vérification. Nous allons donc décrire, dans ce chapitre, le contexte de la mission, la démarche adoptée pour la mise en place des travaux et le périmètre des essais réalisés.

1 L'identification des structures CDC

Dans les outils de vérification statique, l'analyse du comportement des interfaces asynchrones requiert l'identification préalable de structures. Pour permettre la reconnaissance des structures, trois approches distinctes peuvent être mises en place :

— La définition manuelle à partir de spécifications techniques ;

— L'utilisation de fichiers existants dans le flot de conception ;

— La détection automatique par les engins de filtrage des outils.

La définition manuelle La définition manuelle des structures par la personne en charge de la vérification des CDC est risquée car si les spécifications techniques sont incomplètes, des cas peuvent être omis. Par ailleurs, lors de la spécification d'informations par l'utilisateur, les outils n'effectuent pas d'analyse structurelle. Or, l'objectif de la vérification statique est de détecter des problèmes d'implémentation aussi bien structurellement que fonctionnellement. Par exemple, si une structure de type multi-flop est spécifiée comme correcte mais que sa profondeur n'est pas conforme avec la spécification, l'augmentation du risque de propagation d'états métastables ne sera pas détectée. Notons par ailleurs qu'un circuit industriel peut contenir plusieurs milliers de structures CDC, la spécification manuelle de ces dernières est donc coûteuse en temps et présente un fort risque d'erreurs humaines.

L'utilisation de fichiers existants L'utilisation de fichiers existants dans le flot de conception présente, comme pour la définition manuelle, un risque lié à l'utilisation d'informations externes potentiellement erronées. Par ailleurs, certains fichiers peuvent être incompatibles avec l'objectif de la vérification CDC. L'exemple le plus

parlant est l'utilisation des contraintes SDC [30] donnant les signaux d'horloge pour la STA. Les interfaces quasi-synchrones pouvant être vérifiées soit en analyse CDC soit en STA, certains cas peuvent être omis. Si les contraintes pour la STA sont définies de manière pessimiste (couvrant ces interfaces), l'analyse CDC sera optimiste et les exclura. Pour finir, les CDC doivent être analysés le plus tôt possible dans le flot de conception, généralement dès les premières versions du circuit au niveau RTL. Or, à cette étape, les fichiers contenant la liste des horloges pour la STA ou encore les structures de synchronisation de type multi-flop pour la synthèse ne sont pas encore disponibles.

La détection automatique La détection automatique des structures par les outils de vérification est la solution la plus adaptée. Elle n'implique pas d'étape manuelle et ne requiert pas de points d'entrée (spécifications techniques ou fichiers) pour être effectuée. Cependant, comme vu dans le Chapitre 3 (sous-sections 2.1.2 et 2.2.2, pages 46 et 51), les outils présentent, en pratique, des limitations sur le sujet. Nos travaux se limiteront donc à l'approche de vérification basée sur la détection automatique de toutes les structures et informations nécessaires à la validation des communications entre domaines d'horloge.

2 L'analyse des problèmes outils

L'évaluation des outils (sous-section 2.2.2, page 51) a montré que la détection structurelle de certaines interfaces asynchrones est problématique. Par ailleurs, le détail du fonctionnement interne des solutions de vérification statique des CDC n'est pas accessible aux utilisateurs. De plus, aucune information sur le sujet n'a été publiée et, les fournisseurs d'outils restent évasifs quant au détail des modèles et propriétés utilisés. Nous allons donc essayer de formaliser l'existant et les problèmes associés.

Les modèles Pour chaque modèle nous définirons les conditions de détection permettant leur reconnaissance. Elles seront référencées par des acronymes en lettre majuscules. Nous illustrerons nos propos par des structures telles que celles présentées dans les documentations techniques des outils de vérification statique. Nous verrons par ailleurs, les différentes configurations proposées et les différences en termes de structure.

Les propriétés Pour chaque modèle pouvant être validés, les propriétés formelles associées seront définies et référencées par des acronymes en lettres minuscules. Bien qu'il existe différents langages permettant de spécifier des propriétés formelles tels que SVA [89] ou PSL [88], la plupart des outils intègrent uniquement des interpréteurs SVA. C'est donc ce langage qui sera utilisé pour la description des propriétés formelles (pour plus de détails, un bref aperçu du langage SVA est donné en annexe).

Les problèmes À partir des informations relatives à ce qui est implémenté dans les outils, nous présenterons les principales limitations pratiques. Pour chaque problème, un exemple pratique sera présenté. Nous donnerons les principales techniques

utilisées dans l'industrie pour les contourner de manière quasi-automatique. L'analyse des problèmes outils et les exemples associés seront le point de départ de nos travaux.

3 Les solutions proposées

Les travaux présentés dans ce manuscrit ne consistent pas à modifier les outils mais à proposer des solutions permettant d'éviter les problèmes pratiques avec un effort humain minimal. Il y a donc deux options possibles : automatiser l'approche de contournement des problèmes utilisée en pratique (approche utilisateurs) ou proposer des modèles avec des propriétés associées aux fournisseurs d'outils de vérification statique (approche outils).

L'approche utilisateurs La démarche consistant à une intervention de la part de l'utilisateur a déjà été étudiée [48] et, bien que les résultats soient intéressants, cette approche ne peut être mise en place que par des utilisateurs confirmés. Bien que sur le long terme le temps dédié à la vérification CDC diminue substantiellement, les ingénieurs doivent être formés à cette approche. De plus, la dépendance vis-à-vis des personnes en charge de la conception des circuits rend son intérêt dépendant du fonctionnement des équipes et du temps consacré à la prise en considération des interfaces asynchrones.

L'approche outils Cette démarche a une application limitée du point de vue utilisateur. Les code source des engins de filtrage et de preuve n'étant pas accessible, il n'est pas possible de mettre en place des optimisations directement dans les outils. Dès lors, les travaux se limitent à la définition de nouveaux modèles avec des conditions de détection et des propriétés formelles associées. Ces propositions d'améliorations sont ensuite soumises aux fournisseurs outils qui, s'ils le souhaitent, les intègrent. Cependant, le temps de développement des outils est assez lent et, comme expliqué dans le chapitre 4 (sous-section 1.2, page 62), la priorité des fournisseurs outils n'est pas de développer des approches de vérification statique.

4 Les essais pratiques

Les essais pratiques ont pour but de présenter l'intérêt des modèles proposés. Cependant, dans le contexte de travail, l'obtention de résultats est limitée par deux critères : l'accès à des circuits de test et l'accès aux outils.

Le circuit Comme expliqué dans le préambule (section 2, page 8), les essais sont réalisés sur un circuit CPUSS de STMicroelectronics. Ce circuit a été choisi car il présente une diversité de structures telle que la plupart des cas rapportés comme problématiques sont visibles. Nous présenterons donc dans un premier temps les spécifications techniques du circuit en question. Cependant la possibilité de tester les résultats sur d'autres systèmes est limitée par le fait que les interfaces asynchrones doivent être documentées et que les ingénieurs en charge de leur vérification doivent être conscients des problématiques. Dans le cas contraire, il est indispensable de

vérifier les circuits avec de multiples jeux de configuration afin de déterminer les problèmes potentiels et cette démarche est trop longue pour être mise en place dans le temps imparti aux travaux de thèse.

Les outils Comme expliqué dans le chapitre 4 (sous-section 3.2, page 64), les outils ne sont pas modifiables : il n'est pas possible de spécifier des modèles avec des propriétés associées. Il est donc nécessaire de développer des scripts décrivant l'analyse des modèles. Cependant, écrire un script qui ferait une nouvelle vérification CDC reviendrait à se substituer à l'outil et ce n'est pas l'objectif des travaux. Dès lors, il faut pouvoir identifier les outils permettant d'obtenir la meilleure qualité de résultats sur les circuits et pour les cas présentant des limitations, définir des scripts permettant d'appliquer des corrections. Comme présenté dans les résultats d'évaluation du chapitre 3 (sous-section 2.2.2, page 51), deux solutions logicielles ont été choisies : l'outil 1 qui permet de configurer les modèles avec précision et l'outil 2 qui permet à l'utilisateur de spécifier ses propres propriétés formelles.

Les résultats Les résultats seront présentés sous forme de tableaux divisés en trois groupes de colonnes : le premier rapportera les structures CDC présentes dans le circuit, le deuxième présentera les résultats de détection structurelle obtenus avec l'outil 1 et le dernier donner les résultats de validation des protocoles effectuée avec l'outil 2. Pour chacune des deux étapes de vérification des CDC, il y aura une comparaison avec les résultats obtenus à l'aide des scripts développés permettant de couvrir les modèles vérifiés. Les essais basés sur les scripts seront nommées respectivement détection robuste et validation robuste. Concernant les temps d'analyse, seules les étapes de validation seront rapportées car ceux associés à la détection des structures sont négligeables (inférieurs à deux heures pour l'intégralité des structures) dans le contexte des circuits CPUSS.

5 Conclusion

Tous les travaux seront réalisés à travers une approche commune. Les modèles et propriétés internes aux outils seront étudiés afin de formaliser les problèmes associés. De nouveaux modèles avec des propriétés seront ensuite proposés et leur intérêt sera présenté à travers des résultats obtenus sur un sous-système CPUSS. Les travaux seront basés sur l'utilisation des outils 1 et 2, de scripts permettant la détection de nouveaux modèles et de propriétés formelles décrites en SVA pour la validation des protocoles.

Chapitre 6

L'analyse des arbres d'horloge

L'analyse des arbres d'horloge est l'étape préalable à la vérification des CDC. Si un circuit ne contient qu'un seul domaine d'horloge, aucun CDC ne sera détecté. À l'inverse, si plusieurs domaines coexistent, il faut les définir afin d'identifier et de vérifier les CDC. Il s'agit de la mise en place de l'environnement de vérification.

La principale difficulté de cette étape est la notion de domaine d'horloge. Des horloges appartiennent à un même domaine si elles sont alignées en phase. L'analyse des arbres d'horloge est réalisée au niveau RTL avant la première vérification CDC. Or, l'alignement des horloges est effectué lors de l'implémentation physique, qui arrive généralement au niveau GATE. Les horloges ayant une source commune doivent donc être considérées comme appartenant à des domaines différents.

1 Formalisation des problèmes outils

1.1 Motifs et propriétés

Dans les outils, l'analyse des arbres d'horloge est découpée en deux parties : la détection des domaines d'horloge et la détection de la logique de contrôle [36,97,99]. Dans le flot de vérification des CDC, ces étapes interviennent lors de la phase de détection des structures (voir figure 3.1, page 43). Elles sont donc basées sur la détection de modèles à partir de conditions prédéfinies.

1.1.1 La détection des domaines d'horloge

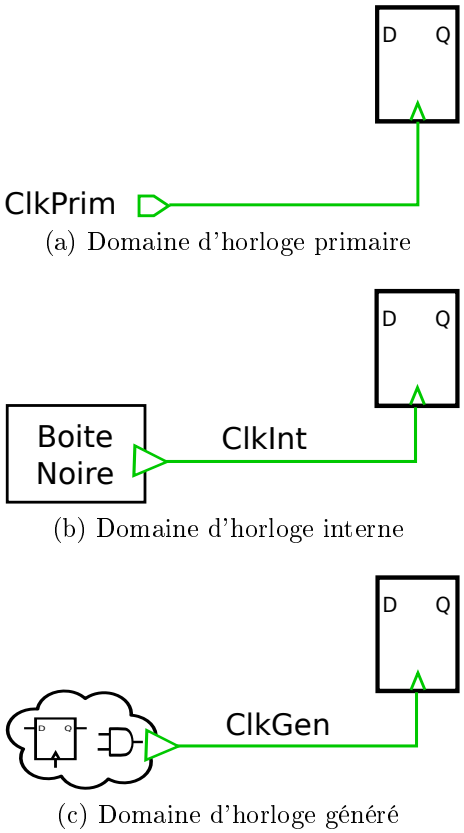
Les conditions L'analyse des arbres d'horloge est initiée par la remontée des entrées d'horloge d'éléments séquentiels. Elle se poursuit par l'identification des trois types de domaines d'horloge suivants : primaire, interne et généré. La reconnaissance de chacun de ces trois cas répond à des conditions de détection spécifiques :

CK1 Un signal (*ClkPrim*) provenant d'un port d'entrée primaire et directement connecté à une entrée d'horloge d'élément séquentiel ;

CK2 Un signal (*ClkInt*) provenant d'une sortie de boite noire et directement connecté à une entrée d'horloge d'élément séquentiel ;

CK3 Un signal (*ClkGen*) provenant d'un élément de logique combinatoire ou séquentielle et directement connecté à une entrée d'horloge d'élément séquentiel ;

La figure 6.1 présente respectivement les structures associées aux conditions de détection des domaines d'horloge primaires *ClkPrim* (figure 6.1a), internes *ClkInt* (figure 6.1b) et générés *ClkGen* (figure 6.1c).



(a) Domaine d'horloge primaire

(b) Domaine d'horloge interne

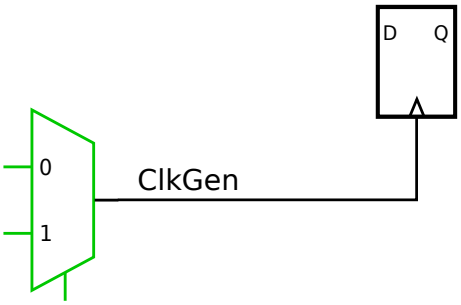
(c) Domaine d'horloge généré

FIGURE 6.1 – Les structures associées aux modèles de type "domaine d'horloge"

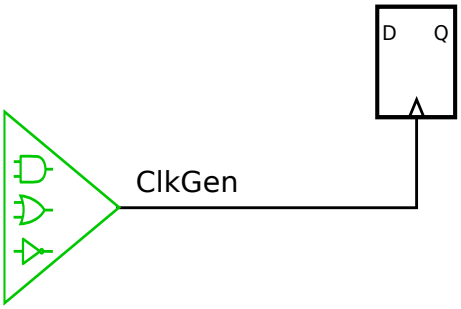
La configuration Afin de ne pas masquer la détection d'horloges présentes uniquement dans le cône d'entrée d'horloges générées, les outils permettent d'adapter l'identification selon quatre axes :

- La traversée d'éléments de logique combinatoire de type "Multiplexeur" (MUX) ;
- La traversée d'éléments de logique combinatoire de type "non-MUX" ;
- La traversée d'éléments séquentiels de type "Clock Gate" ;
- La traversée d'éléments séquentiels de type "Diviseur".

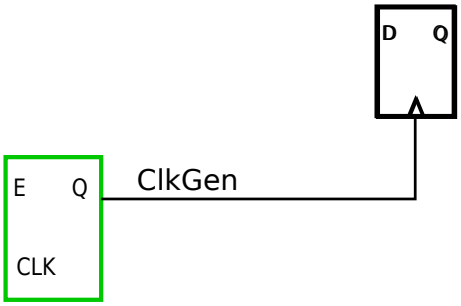
L'analyse Les outils identifient les différentes structures logiques possibles sur les arbres d'horloge. Parmi ces structures, certaines sont utilisées pour du contrôle. Étant donné que les outils sont capables d'identifier toutes les structures potentielles, ils sont en mesure de discriminer celles dédiées au contrôle des horloges. La détection des domaines d'horloge ne se limite donc pas à l'identification des trois types. Elle inclut par ailleurs la reconnaissance et la classification, en quatre catégories, de toutes les structures logiques présentes sur les arbres d'horloge. La figure 6.2 présente les quatre structures logiques identifiées par les outils sur un arbre d'horloge.



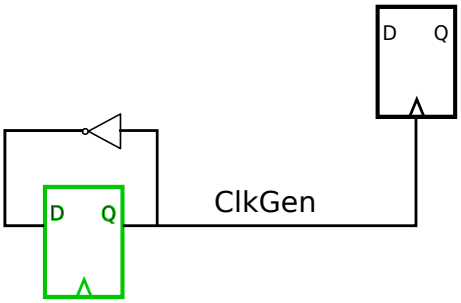
(a) Logique type Multiplexeur



(b) Logique type Combinatoire non-MUX



(c) Logique type Clock Gate



(d) Logique type Diviseur d'horloge

FIGURE 6.2 – Les modèles de logique détectés sur les arbres d'horloge

1.1.2 La détection de la logique de contrôle

Lorsque les domaines générés ont dans leur cône d'entrée plusieurs domaines d'horloge, les outils cherchent à détecter les structures de contrôle et les potentiels signaux de configuration. En reprenant les différentes structures logiques identifiées lors de la détection des domaines d'horloge, seules deux d'entre elles peuvent contenir plusieurs domaines dans leur cône d'entrée :

— MUX d'horloge ;

— Contrôle combinatoire autre que MUX.

MUX d'horloge Les conditions de détection des structures de type "MUX d'horloge" sont :

MX1 Une horloge générée (*ClkGen*) en sortie d'un MUX ;

MX2 Deux domaines d'horloge différents ($ClkIn[i]$) en entrée du MUX.

MX3 Un signal de contrôle (*ClkCtrl*) en entrée du MUX

Contrôle combinatoire autre que MUX Si la structure de type "MUX d'horloge" n'est pas identifiée mais qu'un signal de contrôle est reconnu, les outils rapportent la logique comme "Contrôle combinatoire autre que MUX". Les conditions de détection de ce modèle sont équivalentes à MX1, MX2 et MX3 en remplaçant la logique de type MUX par de la logique combinatoire autre que MUX :

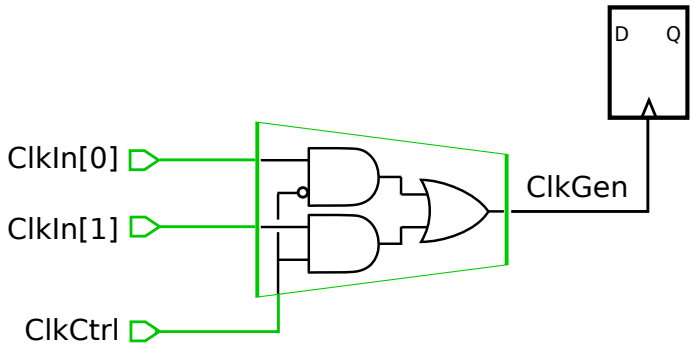
CB1 Une horloge générée (*ClkGen*) en sortie d'un bloc de logique combinatoire ;

CB2 Deux domaines d'horloge différents ($ClkIn[i]$) en entrée du bloc de logique combinatoire.

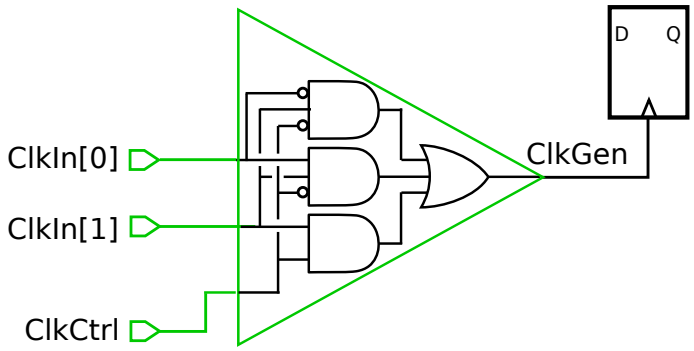
CB3 Un signal de contrôle (*ClkCtrl*) en entrée du bloc de logique combinatoire.

La figure 6.3 présente des exemples structures associées aux modèles de logique de contrôle sur les arbres d'horloge : "MUX d'horloge" (figure 6.3a) et "Contrôle combinatoire autre que MUX" (figure 6.3b). Les structures sont représentées avec en entrée deux domaines primaires ($ClkIn[0]$ et $ClkIn[1]$) et un signal de configuration (*ClkCtrl*) et, en sortie, un domaine généré (*ClkGen*). Notons que le "Contrôle combinatoire autre que MUX" à la fonctionnalité d'un multiplexeur mais la présence de logique redondante empêche sa reconnaissance.

L'analyse À partir de la liste des signaux de contrôle, les outils peuvent effectuer une analyse des arbres d'horloge. L'idée est de remonter les cônes d'entrée des signaux de contrôle (*ClkCtrl*) jusqu'aux ports d'entrée primaire et de définir les différentes configurations possibles. Ces configurations permettent de propager uniquement certains domaines d'horloge primaires et internes, et de ne pas considérer les horloges générées comme de nouveaux domaines. On parle alors de modes d'analyse des CDC. Si aucune structure de contrôle n'est identifiée, les outils considèrent tous les domaines d'horloge comme indépendants. Dans le cas où une structure respecte les conditions MX2 ou CB2, mais qu'aucun signal de contrôle n'est identifié (conditions MX3 ou CB3), les outils rapportent un problème de convergence d'horloges : le circuit doit être corrigé.



(a) Contrôle type "MUX d'horloge"



(b) Contrôle type "Combinatoire autre que MUX"

FIGURE 6.3 – Les modèles de logique de contrôle sur les arbres d'horloge

1.2 Limitations

Pour assurer une couverture maximale des CDC, tous les domaines d'horloge doivent être définis. Sur des circuits industriels, il peut y avoir plusieurs centaines de domaines détectés. La complexité de l'analyse CDC dépend à la fois du nombre de chemins entre des registres et de la quantité de domaines d'horloge définis. Plus il y a de domaines d'horloge, plus l'analyse des CDC risque d'être complexe. Afin de faire le meilleur compromis entre la couverture CDC et le temps d'exécution, la mise en place de l'analyse CDC peut se faire de deux manières :

— Analyse modale : propagation des domaines d'horloge primaires et internes ;

— Analyse multi-modale : définition de tous les domaines d'horloge (primaires, internes et générés).

1.2.1 L'analyse modale

Cette approche consiste à mettre en place l'environnement de vérification des CDC en déclarant uniquement les domaines d'horloge primaires et internes et en les propageant. L'idée est de détecter et d'analyser la logique de contrôle sur les arbres d'horloge afin de rapporter les différents modes d'analyse des CDC. La figure 6.4 présente le flot relatif à l'analyse modale. Cependant, la recherche et la définition des modes présente trois limitations majeures :

— Couverture partielle des CDC ;

— Détection incomplète des signaux de contrôle ;

— Génération non conclusive des modes de fonctionnement.

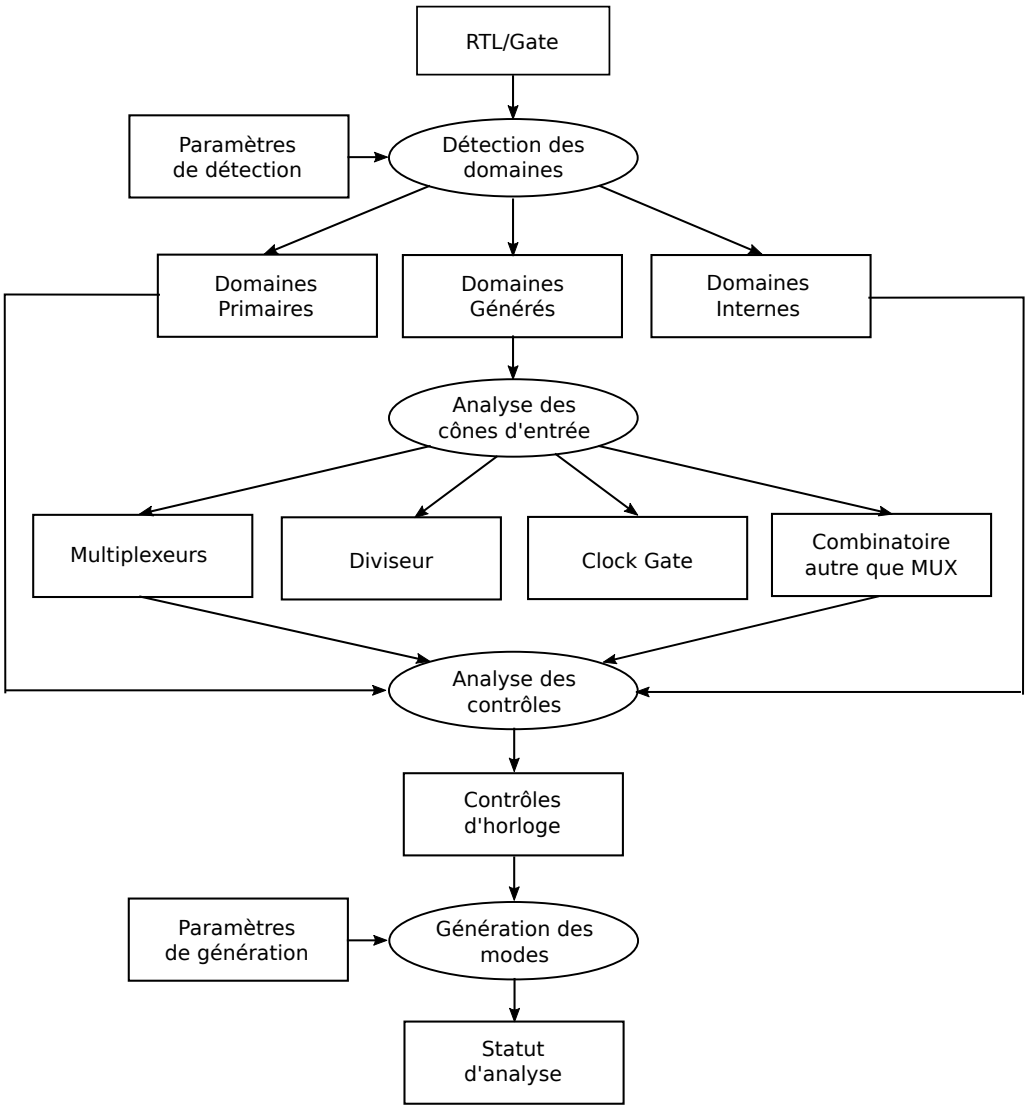


FIGURE 6.4 – Le flot d'analyse modale des CDC

Couverture partielle des CDC La couverture partielle des CDC est due au fait que l'approche considère que toutes les horloges ayant une source commune sont alignées en phase. Si les horloges utilisées pour ces communications sont considérées comme déphasées par l'implémentation physique, des CDC ne seront pas couverts. Pour remédier à ce problème, il est possible d'ajouter manuellement, à partir des contraintes définies en STA, des domaines d'horloge générés. La figure 6.5 présente la problématique des interfaces non couvertes lors d'une analyse modale avec une horloge primaire *Clk* qui se propage à travers deux nuages de logique différents avant d'aller vers deux registres qui communiquent entre eux de façon asynchrone. Du fait de la propagation de l'horloge primaire, l'interface est considérée comme synchrone par les outils.

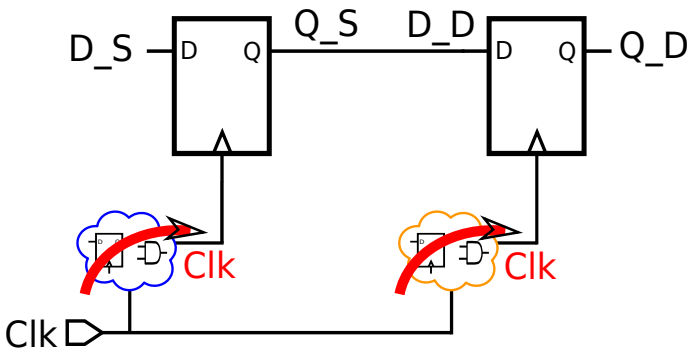


FIGURE 6.5 – Les interfaces non couvertes par l'analyse modale

Détection incomplète des signaux de contrôle Les modèles associés à la logique de contrôle étant combinatoires, des structures peuvent être omises. Lors de la conception des arbres d'horloge, il est d'usage de remplacer la logique combinatoire par des cellules de type "Clock Gate" pour éviter les courses de signaux [68]. Comme illustré par la figure 6.6, ces structures intègrent un verrou, qui est un élément séquentiel. Or, si la logique de contrôle intègre des éléments séquentiels, les structures ne sont pas identifiées. Les signaux de contrôle ne sont donc pas vus et un problème est rapporté. Pour pallier cette limitation, il est possible d'analyser manuellement le cône d'entrée des cas de convergence d'horloge et, si un signal de contrôle est trouvé, de le configurer à une valeur constante.

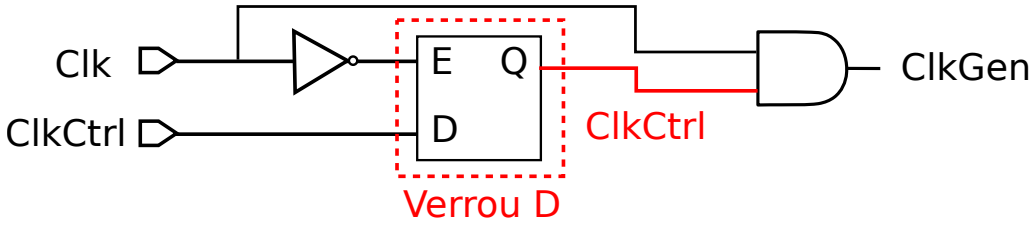


FIGURE 6.6 – La structure associée au modèle "Clock Gate"

Génération non-conclusive des modes Dans les circuits industriels, la logique présente entre les points de détection des signaux de contrôle et les ports d'entrée primaires peut être très importante. Cette complexité, conduit à des explosions d'états lors de la remontée des cônes d'entrée. Dès lors la définition des contrôles sur les entrées primaires est non-conclusive : aucun mode n'est rapporté. Pour empêcher les explosions d'états lors de l'analyse modale, il est possible de limiter la remontée. Par exemple, les outils peuvent être configurés afin qu'un point d'arrêt soit défini à la sortie du premier élément séquentiel présent dans le cône d'entrée d'un signal de contrôle. Cependant le nombre de modes rapportés ne sera pas factorisé au maximum et de faux modes peuvent être proposés. La figure 6.7 présente une structure où le signal de sélection d'un MUX d'horloge possède un cône d'entrée important. Lors de l'étape de génération des modes, les outils doivent analyser l'intégralité de la logique de contrôle permettant la propagation des horloges en sortie du MUX et des problèmes d'explosions d'états peuvent survenir.

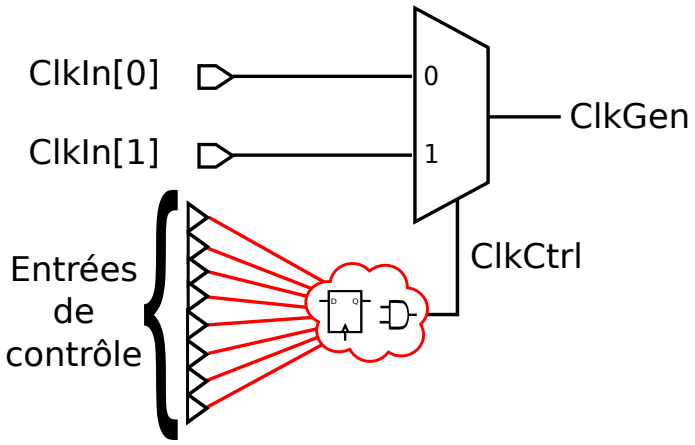


FIGURE 6.7 – Problème d'explosions d'états lors de l'analyse modale des CDC

La figure 6.8 présente le flot d'analyse modale des CDC avec les principaux problèmes pratiques associés.

1.2.2 L'analyse multi-modale

Cette approche consiste à mettre en place l'environnement de vérification des CDC en déclarant uniquement les domaines d'horloge primaires, internes et générés. La détection et l'analyse des signaux contrôle n'étant pas effectuée, l'intégralité des domaines doit être définie. La figure 6.9 présente le flot d'analyse multi-modale des CDC qui se limite à la génération de toutes les horloges d'un circuit en les considérant dans des domaines différents.

La principale limitation liée à l'analyse multi-modale est la couverture de faux CDC. Ce problème peut être dû à deux raisons :

— L'absence des contraintes utilisées pour la STA ;
— La non-analyse des structures de contrôle.

L'absence des contraintes STA Lors d'une analyse multi-modale, toutes les horloges d'un circuit sont définies dans des domaines différents, même celles ayant une source commune. Dès lors, tous les CDC sont couverts de manière pessimiste. Si deux horloges sont considérées comme alignées en phase par l'implémentation physique, il est nécessaire de les définir comme appartenant au même domaine pour l'analyse CDC. Cependant, dans les circuits industriels, il peut y avoir des centaines de domaines d'horloge générés. Il est alors indispensable de limiter l'analyse de faux CDC en redéfinissant manuellement les domaines. La figure 6.10 illustre la problématique pratique d'utilisation de l'analyse multi-modale avec le cas d'une horloge *ClkGen* générée en sortie d'un MUX et dont le cône de sortie est composé de nombreux éléments de logique avant d'aller vers des éléments séquentiels. Les outils vont alors déclarer à la sortie de chaque élément de logique de nouvelles horloges générées appartenant à des domaines différents.

La non-analyse des structures de contrôle Lorsque tous les domaines d'horloge sont correctement définis, de faux CDC peuvent encore être vus car la logique

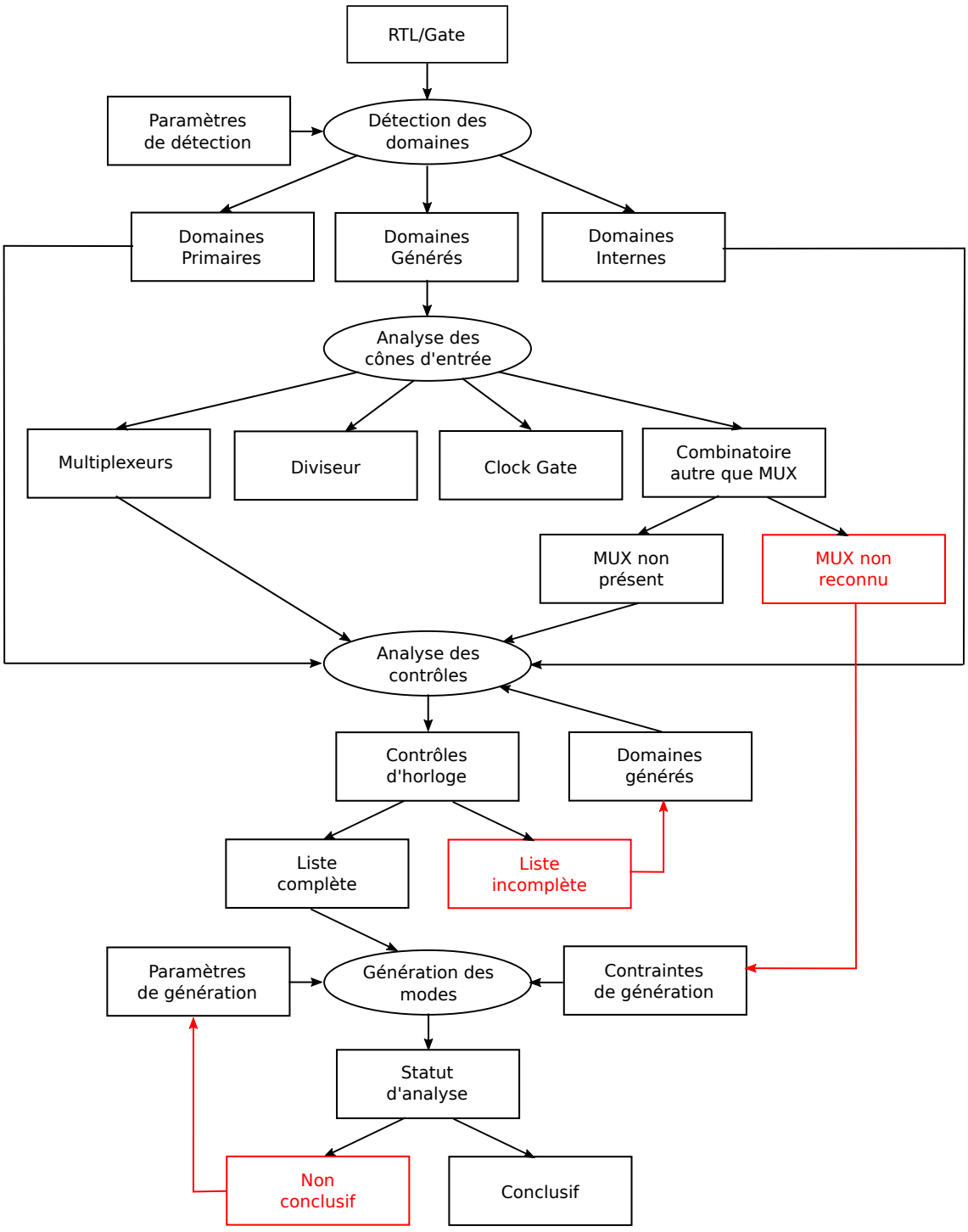


FIGURE 6.8 – Problématiques liées au flot d'analyse modale des CDC

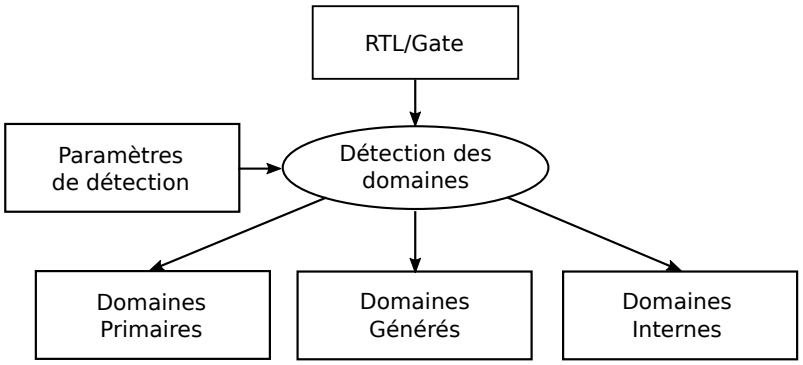


FIGURE 6.9 – Le flot d'analyse multi-modale des CDC

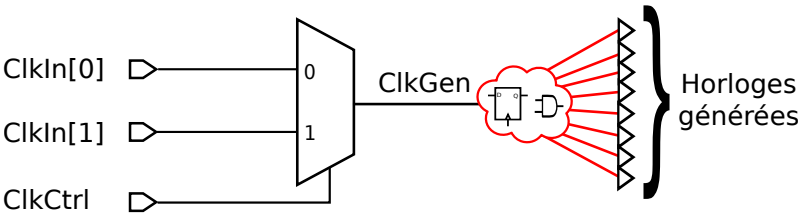


FIGURE 6.10 – Le problème pratique lié à l'analyse multi-modale des CDC

de contrôle n'est pas analysée. Des domaines peuvent coexister structurellement mais être fonctionnellement exclusifs. En d'autres termes, ils ne peuvent jamais être actifs en même temps. On parle alors de domaines physiquement exclusifs. Le cas le plus simple est celui de deux domaines d'horloge traversant respectivement une structure de type "Clock Gate" avec un signal de contrôle identique et des conditions d'activation inversées. Lorsque le signal de contrôle est à un état donné (0 ou 1), un des domaines est actif et l'autre non. Les CDC entre ces deux domaines sont inexistants fonctionnellement. Cependant, l'analyse multi-modale ne se basant pas sur la détection des signaux de contrôle, les CDC entre domaines physiquement exclusifs sont analysés. Par ailleurs, si l'analyse de la logique de contrôle était effectuée, les résultats seraient sujets au problème de détection incomplète des signaux de contrôle. La solution consiste alors à analyser manuellement les cônes d'entrée des horloges générées en sortie de "Clock Gate" afin de définir des exclusions entre domaines d'horloge. La figure 6.11 présente un exemple de faux CDC, analysé lors d'une analyse multi-modale. Les domaines en sortie de "Clock Gate" ($ClkGen[0]$ et $ClkGen[1]$) sont identifiés et le CDC (Q_S/D_D) est analysé. Cependant le signal $ClkCtrl$ assure que les deux horloges ne peuvent jamais être actives simultanément.

.

La figure 6.12 présente les problèmes pratiques liés au flot d'analyse multi-modale des CDC.

1.2.3 Bilan

Les deux approches de mise en place de l'analyse CDC présentent des problèmes. L'analyse modale est optimiste : certains CDC peuvent être manqués. À l'inverse, l'analyse multi-modale est pessimiste : de faux CDC peuvent être analysés. La spécification manuelle d'informations (contraintes) est requise pour obtenir une couverture CDC correspondant à ce qui est implémenté dans le circuit. À ce sujet,

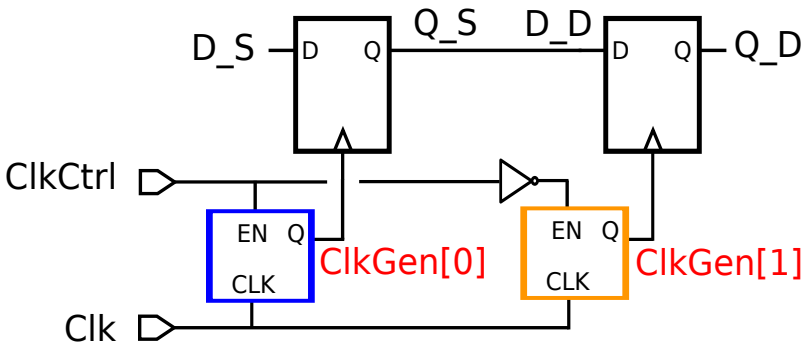


FIGURE 6.11 – Exemple de faux CDC analysé avec l'approche multi-modale

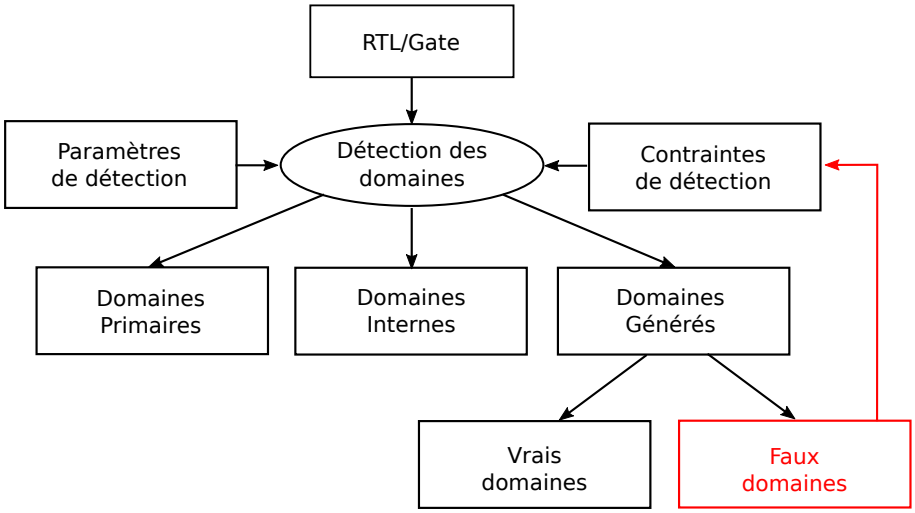


FIGURE 6.12 – Problèmes associés au flot d'analyse multi-modale des CDC

les outils se limitent au calcul du pourcentage d'éléments séquentiels recevant un signal d'horloge. Il n'y a pas de notion de CDC dans les informations rapportées. Le plan de vérification doit donc être mis en place manuellement. Par ailleurs, les relations entre les horloges est incomplète : les domaines pouvant être alignés ne sont pas rapportés comme tel. Ces problèmes motivent l'élaboration de nouveaux modèles afin d'assurer une mise en place de la vérification CDC de manière optimale et automatique.

2 Proposition de modèles et du plan de vérification

Parmi les deux approches proposées pour la mise en place de la vérification CDC, l'analyse multi-modale est la seule permettant d'assurer une couverture complète des CDC. Nous avons donc choisi de concentrer nos travaux sur la résolution des problèmes liés à cette méthode d'analyse des arbres d'horloge. Tout d'abord nous décidons d'ajouter à cette approche l'identification des structures de contrôle décrites dans la sous-section 1.1.2. Par ailleurs, nous proposons trois nouveaux modèles de logique de contrôle sur les arbres d'horloge afin de détecter de manière exhaustive tous les domaines d'horloge et de classifier les types de communication entre les domaines :

— Le "MUX anti-courses" (Glitch Free MUX) ;
— L'"Exclusivité physique" ;
— L'"Exclusivité logique".

2.1 Le "MUX anti-courses"

Cette structure logique est utilisée pour permettre une sélection d'horloge sans courses de signaux. La structure associée au modèle "MUX anti-courses" est similaire à celle des MUX combinatoire (figure 6.3a) en remplaçant les porte "ET" (ou "NON-OU") d'entrée par des "Clock Gate". Les structures de type "Clock Gate" étant détectées par les outils, les "MUX anti-courses" peuvent facilement être identifiés en considérant les verrous internes comme transparents. Les conditions de détection du modèle "MUX anti-courses" sont :

GFM1 Une horloge générée ($ClkGen$) en sortie d'une porte OU ;
GFM2 Les entrée de la porte "OU" (ClkInt[i]) venant de structures "Clock Gate" ;
GFM3 Deux domaines d'horloge différents ($ClkIn[i]$) en entrée des "Clock Gate" ;
GFM4 Un signal de contrôle ($ClkCtrl$) commun en entrée des "Clock Gate", et inversé sur une des deux.

La figure 6.13 présente la structure associée au modèle "MUX anti-courses". Le modèle intègre quatre groupes de signaux : l'horloge de sortie ($ClkOut$), les horloges d'entrée ($ClkIn[i]$), les horloges internes ($ClkInt[i]$) et le signal de contrôle ($ClkCtrl$). Le signal $ClkCtrl$ permet la propagation d'une des horloges d'entrée, en sortie du MUX. Ce modèle permet de reconnaître toutes les structures de type MUX sur les arbres d'horloge.

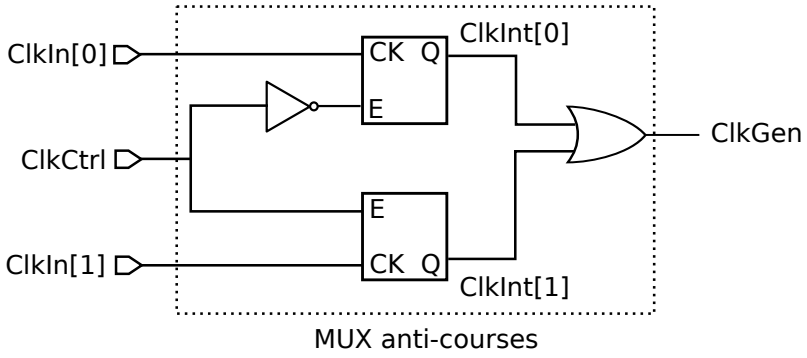


FIGURE 6.13 – La structure associée au modèle "MUX anti-courses"

2.2 L"'Exclusivité physique"

La détection Les domaines physiquement exclusifs, en CDC, sont des signaux d'horloge ne pouvant être actifs simultanément. La structure associée au modèle "Exclusivité physique" est composée de deux structures de type "Clock Gate" avec un signal de contrôle commun et des conditions d'activations inversées. Elle est donc similaire à celle des "MUX anti-courses" sans la porte "OU" de sortie. Le modèle se limite au cas des structures de type "Clock Gate" car dans le cas de logique combinatoire de type "ET" (ou "NON-OU"), il n'est pas possible de différencier l'entrée d'horloge et celle dédiée à la configuration. Les conditions de détection du modèle "Exclusivité physique" sont :

PE1 Deux horloges générées ($ClkGen[i]$) en sortie de structures "Clock Gate" ;

PE2 Un signal de contrôle commun ($ClkCtrl$) à l'entrée des "Clock Gate", inversé sur un des deux chemins.

La figure 6.14 présente la structure associée au modèle "Exclusivité physique". Le modèle intègre deux groupes de signaux : les horloges de sortie ($ClkGen[i]$) et le signal de contrôle ($ClkCtrl$). Le signal $ClkCtrl$ permet d'assurer l'exclusivité physique entre les horloges. Ce modèle permet de reconnaître tous les domaines d'horloge exclusifs entre eux.

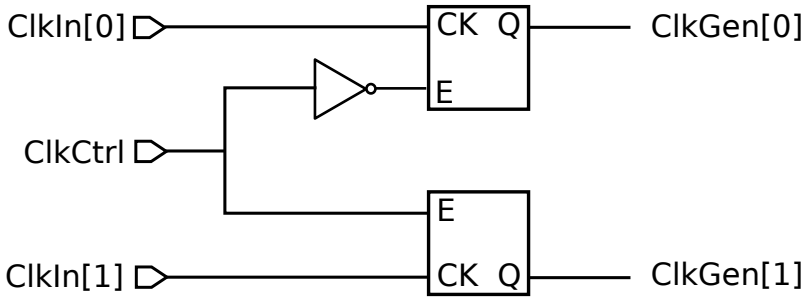


FIGURE 6.14 – La structure associée au modèle "Exclusivité physique'

La validation La validation de l'exclusivité physique peut se faire de trois manières différentes :

— Détection et analyse structurelle de la logique de contrôle ;

— Simulation de valeurs statiques sur la logique de contrôle ;
— Vérification formelle de l'exclusivité.

N'ayant pas la possibilité de modifier les moteurs de détection ou de simuler des valeurs, nous proposons une propriété formelle définissant l'exclusivité physique entre les horloges. Cette propriété, décrite en SVA, assure que lorsque le signal de contrôle *ClkCtrl* est stable, une seule sortie de "Clock Gate" peut changer d'état (au moins une des deux est stable).

```
property Exclusivite_physique(ClkIn0,ClkIn1,ClkCtrl,ClkGen0,ClkGen1);
(@(posedge ClkIn0 or posedge ClkIn1) disable iff (Reset)
$stable(ClkCtrl) |=> $stable(ClkGen0) | $stable(ClkGen1));
endproperty

pe:assert property(
Exclusivite_physique(ClkIn[0],ClkIn[1],ClkCtrl,ClkGen[0],ClkGen[1]));
```

2.3 L"'Exclusivité logique"

La détection Des domaines logiquement exclusifs sont des horloges générées, ayant dans leur cône d'entrée plusieurs domaines, et pouvant être alignées en phase. Les interfaces entre horloges logiquement exclusives sont quasi-synchrones. La structure associée au modèle "Exclusivité logique" est composée de structures de type MUX (purement combinatoire ou anti-courses) ayant les mêmes signaux d'horloge et le même contrôle en entrée. Les conditions de détection du modèle "Exclusivité logique" sont :

LE1 Deux horloges générées (*ClkGen*) en sortie de structures MUX ;
LE2 Les horloge d'entrée ($ClkIn[i]$) des MUX sont les mêmes ;
LE3 Un signal de contrôle commun (*ClkCtrl*) à l'entrée des MUX.

La figure 6.15 présente la structure associée au modèle "Exclusivité logique". Le modèle intègre trois groupes de signaux : les horloges de sortie ($ClkGen[i]$), les horloges d'entrée ($ClkIn[i]$) et le signal de contrôle (*ClkCtrl*). Le signal *ClkCtrl* permet d'assurer l'exclusivité logique entre les horloges. Ce modèle permet de reconnaître tous les domaines d'horloge générés en sortie de MUX et pouvant être alignés en phase.

La validation Comme pour les horloges physiquement exclusives, le modèle "Exclusivité logique" peut être validé de trois manières. La propriété formelle de type SVA associée vérifie que lorsque le signal de sélection des multiplexeurs est à un état donné (stable), les horloges générées en sortie sont identiques.

```
property Exclusivite_logique(ClkIn0,Clk1,ClkCtrl,ClkGen0,ClkGen1);
(@(posedge ClkIn0 or posedge ClkIn1) disable iff (Reset)
$stable(ClkCtrl) |-> (ClkGen0 == ClkGen1));
endproperty

le:assert property(
Exclusivite_logique(ClkIn[0],ClkIn[1],ClkCtrl,ClkGen[0],ClkGen[1]));
```

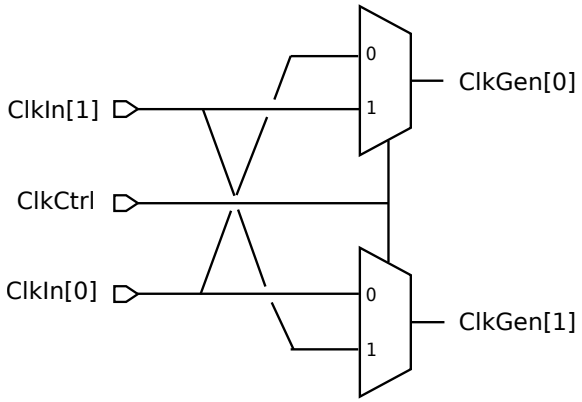


FIGURE 6.15 – La structure associée au modèle "Exclusivité logique'

2.4 Le plan de vérification

En considérant les modèles "MUX anti-courses", "Exclusivité physique" et "Exclusivité logique", toutes les structures de contrôle sur les arbres d'horloge peuvent être identifiées. Dès lors, les relations entre domaines d'horloge peuvent être définies. Dans le cas d'une horloge générée à partir d'une seule source, les phases peuvent être alignées. À l'inverse, lors de la recombinaison de plusieurs horloges à travers de la logique de contrôle, la relation entre les différentes sources et l'horloge générée dépend de la relation entre les horloges d'entrée.

Si les domaines d'entrée sont alignés ou peuvent l'être, le domaine généré en sortie peut être aligné avec ses sources. Cependant, si les entrées sont déphasées, alors l'horloge générée sera déphasée. Pour finir, la relation entre deux horloges générées dépend de la structure en entrée. Si deux domaines sont générés en sortie d'un contrôleur d'exclusivité physique, ils sont exclusifs. Par contre, s'ils sont générés en sortie de contrôleurs d'exclusivité logique, ils ne sont pas exclusifs mais peuvent être alignés. S'il n'y a pas de contrôleurs d'exclusivité, la relation entre les générées dépend de la relation entre les horloges d'entrée.

Le tableau 6.1 récapitule les différentes relations entre domaines d'horloge en fonction des cas. Lorsqu'une relation n'est pas applicable, "N.A." (Non Applicable) est indiqué. Si les domaines sont en phase alors ils sont notés comme "Alignés", s'ils peuvent être alignés, "Ajustables" est rapporté et sinon ils sont considérés comme "Déphasés".

À partir de ces informations, il est possible de couvrir l'intégralité des CDC d'un circuit, de classifier les interfaces et d'exclure certains CDC. Les CDC entre horloges déphasées doivent être analysés en priorité. Les domaines identifiés comme pouvant être alignés doivent être rapportés aux personnes en charge de l'implémentation physique pour savoir s'ils doivent être considérés comme déphasés. En effectuant une vérification CDC intégrant ces différentes informations, le résultat de l'analyse correspond à une couverture égale à 100%. Il est alors possible de donner une information de couverture et de générer un plan de vérification à l'utilisateur.

Type d'arbre d'horloge	Structure logique	Relation entre domaines source	Relation entre domaines source(s) et généré(s)	Relation entre domaines générés
Domaine source unique et domaine généré unique	Combinatoire	N.A.	Ajustables	N.A.
	Diviseur			
	Clock Gate			
	PLL			
Domaines sources multiples et domaine généré unique	Contrôle d'horloge	Alignés	Ajustables	N.A.
		Ajustables		
		Déphasés	Déphasés	
	Combinaison d'horloge	Toutes	Déphasés	
Domaines sources multiples et domaines générés multiples	Exclusivité physique	Toutes	Toutes	Exclusifs
	Exclusivité logique	Toutes	Toutes	Ajustables
	Autre	Alignés	Ajustables	Ajustables
		Ajustables		
		Déphasés	Déphasés	Déphasés

TABLE 6.1 – Relation entre domaines d'horloge en fonction des cas

La figure 6.16 présente le flot proposé pour la mise en place de l'environnement de vérification CDC. Il se compose du flot d'analyse multi-modale auquel s'ajoute une étape d'analyse de la logique de contrôle (comme dans le flot d'analyse modale) incluant les modèles proposés, et une génération des relations entre les différents signaux d'horloge détectés.

3 Mise en application

3.1 La mise en place des tests

Les caractéristiques du circuit Le circuit utilisé pour les résultats pratiques est le sous-système à base de processeurs (CPUSS) présenté dans le préambule (sous-section 2, page 8). Ses caractéristiques en termes d'arbres d'horloge sont :

— 16 domaines d'horloge primaires ;

— 22 domaines d'horloge internes ;

— 780 domaines d'horloge générés.

Les domaines primaires et internes sont envoyés vers des blocs de sélection en fonction du mode de fonctionnement du circuit. La logique dédiée au contrôle des horloges dépend du contexte dans lequel le circuit est utilisé : fonctionnement ou test. Pour les horloges dédiées aux différents modes de test du circuit après fabrication,

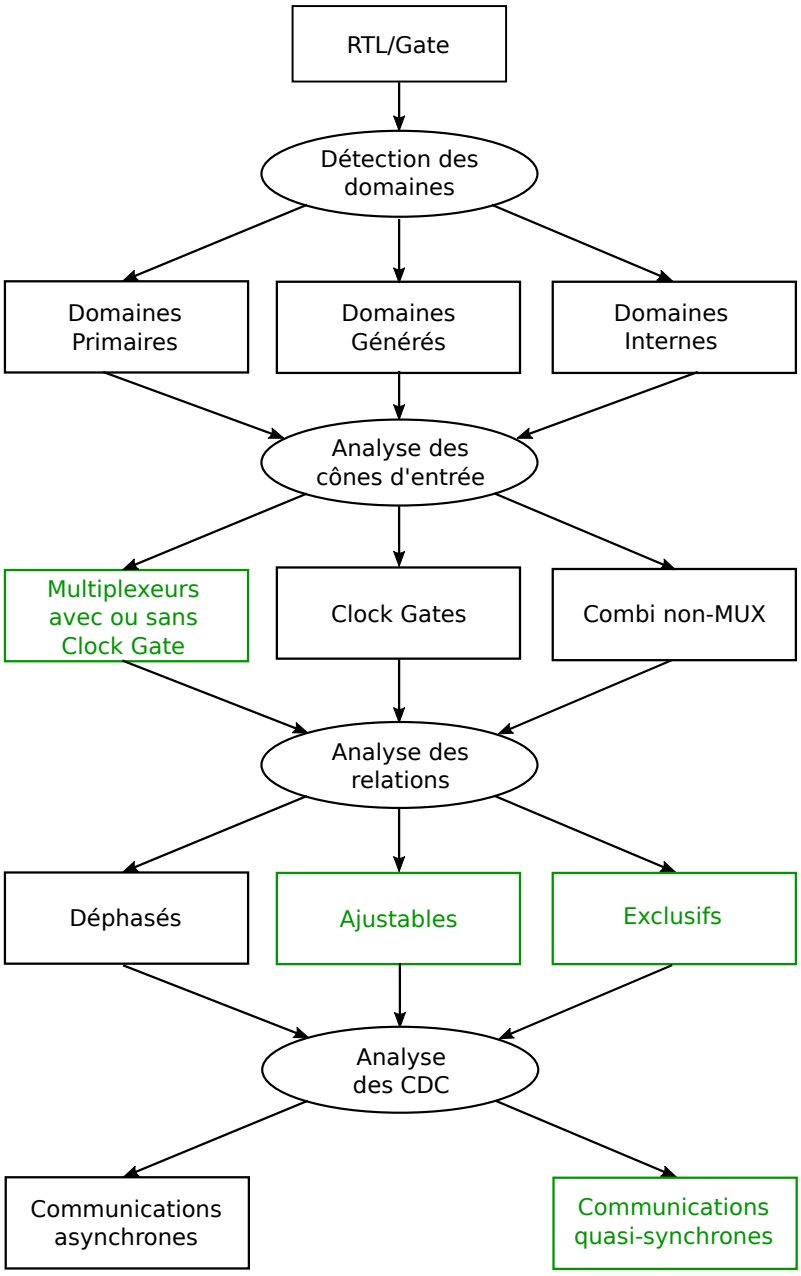


FIGURE 6.16 – Le flot proposé pour l'analyse des arbres d'horloge

des multiplexeurs combinatoires sont utilisés. Cependant, lorsque le circuit est en cours de fonctionnement, la sélection des horloges se fait à travers des "MUX anti-courses". De plus, afin de limiter la consommation d'énergie, des centaines de "Clock Gate" sont implémentées à travers le circuit. En considérant par ailleurs les diviseurs d'horloge et la logique combinatoire autre que MUX, la définition des 780 domaines d'horloge générés est faite en sortie des éléments suivants :

— 73 MUX combinatoires ;
— 10 structures combinatoires autre que MUX ;
— 23 MUX anti-courses ;
— 15 diviseurs d'horloge ;
— 659 clock gate.

L'approche choisie La méthodologie proposée pour l'analyse des arbres d'horloge consiste à identifier les relations entre domaines d'horloge à l'aide des modèles "MUX anti-courses", "Exclusivité physique" et "Exclusivité logique". Bien que les structures de type "Diviseur d'horloge", "MUX d'horloge", "Clock Gate" et "Logique combinatoire autre que MUX" soient rapportés, les outils n'intègrent aucun des trois modèles développés et il n'est pas possible d'en ajouter ou d'en modifier. En outre, les sorties des "MUX anti-courses" sont vues comme des recombinaisons d'horloges à travers des éléments de logique combinatoire autre que MUX. À partir de ces informations, il est possible d'analyser, à l'aide de scripts, les cônes d'entrée afin de déterminer s'il répondent aux conditions de détection définies pour chacun des trois modèles.

Après l'identification de toutes les structures logiques sur les arbres d'horloge du circuit, la seconde étape consiste à établir les relation entre les domaines horloges. L'approche consiste à définir les relations en fonction des cas tel que présenté dans le tableau 6.1. Concernant le calcul de couverture, une analyse CDC avec les différentes informations donne la liste correspondant à 100% des chemins.

3.2 Les résultats obtenus

Description des tests Les essais sont effectués avec les deux seuls outils CDC du marché permettant une analyse précise des arbres d'horloge : outil 1 et outil 2. Les résultats sont rapportés dans le tableau 6.2. Les deux premières colonnes donnent respectivement le type de structure sur l'arbre d'horloge et le nombre de cas présents dans le circuit. Les colonnes 3 et 4 rapportent le nombre de cas détectés par les outils. La colonne 5 présente le nombre de cas détectés avec la méthode proposée. Lorsque les outils n'ont pas de modèle permettant de détecter les structures implémentées, "N.A." (Non applicable) est rapporté. À l'inverse si les outils ou l'approche prennent en compte les modèles et détectent l'intégralité des structures, "OK" est indiqué. Si les modèles existent mais que seule une partie des structures est identifiée, "NOK(x)" est rapporté (avec x le nombre de structures identifiées).

Analyse des résultats Les tests effectués mettent en évidence que les modèles proposés permettent de détecter l'intégralité des structures implémentées dans le circuit. À l'inverse, les outils de vérification statique, ne reconnaissent ni les "MUX

1	2	3	4	5
Types de structure	Nombre de cas	Résultats Outil 1	Resultats Outil 2	Résultats Robustes
MUX combinatoire	73	NOK (72)	NOK (57)	OK
MUX anti-courses	23	N.A.	N.A.	OK
PLL multi-horloge	7	OK	NOK (1)	OK
Diviseur d'horloge	3	OK	OK	OK
Combinatoire non MUX	10	NOK (32)	NOK (33)	OK
Exclusivité physique	0	N.A.	N.A.	N.A.
Exclusivité logique	8	N.A.	N.A.	OK
Toutes	124	114 (92%)	94 (76%)	OK (100%)

TABLE 6.2 – Résultats d'analyse des arbres d'horloge

Types de communication	Nombre de cas	Résultats Outil 1	Resultats Outil 2	Résultats Robustes
Asynchrone	125956	OK (100%)	OK (100%)	OK (100%)
Quasi-synchrone	44117	42496 (96%)	OK (100%)	OK (100%)
Toutes	170073	168452 (99%)	OK (100%)	OK (100%)

TABLE 6.3 – Résultats de couverture des CDC

anti-courses", ni les cas d'"Exclusivité logique". Par ailleurs, les interfaces quasi-synchrones ne sont pas identifiées comme telles. Les approches outils nécessitent donc l'analyse manuelle des arbres d'horloge pour pouvoir différencier les interfaces quasi-synchrones et asynchrones. Le tableau 6.3 présente la couverture CDC analysée automatiquement par les outils à travers une analyse multi-modale. Bien que l'outil 2 présente une couverture maximale, il ne différencie pas les interfaces asynchrones des interfaces quasi-synchrones. Or, dans le circuit CPUSS utilisé pour les essais pratiques, seules 3% des communications quasi-synchrones (4639) ne sont pas considérées comme synchrone par la STA, il est donc primordial de différencier les types de communication afin d'analyser en priorité les interfaces asynchrones. Concernant l'outil 1, la couverture CDC est incomplète car il considère que si des horloges en entrée d'un MUX appartiennent au même domaine, l'horloge générée en sortie du MUX est alignée en phase. L'approche proposée permet de couvrir de manière automatique la totalité des CDC, de classifier les interfaces et de générer un plan de vérification.

4 Conclusion

L'analyse des arbres d'horloge permet de mettre en place l'analyse CDC, c'est une étape indispensable. Cependant, sur les circuits industriels, les outils n'identifient que partiellement la logique de contrôle et les relations entre les domaines d'horloge sont pessimistes. L'intégration des modèles proposés permettrait d'extraire toutes les structures possibles sur les arbres d'horloge, d'en déduire les relations entre les domaines et de générer un plan de vérification CDC. Dans le contexte des circuits

CPUSS, l'approche proposée est la plus adaptée car elle est automatique et en ligne avec la méthodologie utilisée pour la STA. Néanmoins, cette solution n'est, pour l'instant pas optimisée puisqu'elle augmente de 40% le nombre de modèles à rechercher pour l'identification de la logique de contrôle (sept modèles au lieu de quatre), avec potentiellement des paramètres associés. Par ailleurs, la génération du plan de vérification requiert une analyse des CDC ce qui, dans le cas des circuits CPUSS, multiplie par deux le temps d'execution de l'étape de mise en place par l'outil.

Chapitre 7

L'analyse des structures de type multi-flop

La détection et l'analyse des interfaces synchronisées par multi-flop (MF) correspondent à la première étape de la vérification des CDC. Si aucune structure n'est détectée, tous les CDC sont rapportés comme problématiques. De plus, les protocoles de séquençage, les courses de signaux et les convergences ne sont pas analysés. À l'inverse, la détection de toutes les MF d'un circuit garantit que tous les protocoles de temporisation sont vérifiés. Cette étape est donc la base de la qualité de détection des structures CDC.

1 Formalisation des problèmes outils

1.1 Motifs et propriétés

L'analyse des multi-flops, comme la vérification des CDC à haut niveau, se divise dans les outils, en deux parties : la détection des structures et la validation des protocoles [36, 97, 99]. Lors de la première étape, trois modèles sont recherchés :

— La structure MF ;

— La recombinaison de signaux avant MF ;

— La recombinaison de signaux après MF.

Une fois les modèles identifiés, la phase de validation peut être effectuée. L'idée est de vérifier que les structures respectent les protocoles de temporisation. Les outils analysent donc respectivement pour chaque modèle un des trois comportements suivants :

— Le maintien de la donnée (MF) ;

— Le comportement anti-courses (recombinaison avant MF) ;

— L'exclusivité des signaux recombinés (recombinaison après MF).

1.1.1 La structure MF

Les conditions Cette technique de conception des CDC est basée sur un empilement d'au moins un élément séquentiel en cascade après le registre de destination

(sous-section 1.2.2, page 27). La reconnaissance des MF se fait à travers le respect des deux conditions suivantes :

MF1 Une donnée (Q_S/D_D) émise par un domaine d'horloge source et réceptionnée par un domaine d'horloge différent ;

MF2 Un chemin direct et unique entre la sortie du CDC (Q_D) et un élément séquentiel dans le domaine de destination.

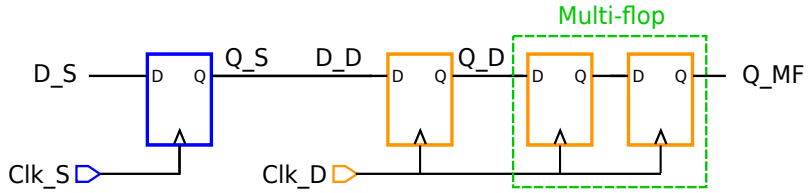
La configuration Cependant, certaines structures peuvent présenter des éléments supplémentaires tels que des registres en cascade ou de la logique de contrôle en entrée. Afin d'éviter que des implémentations correctes ne soient pas identifiées, les outils permettent d'adapter la détection des MF selon trois axes :

— La profondeur de la multi-flop (supérieure à 2) ;

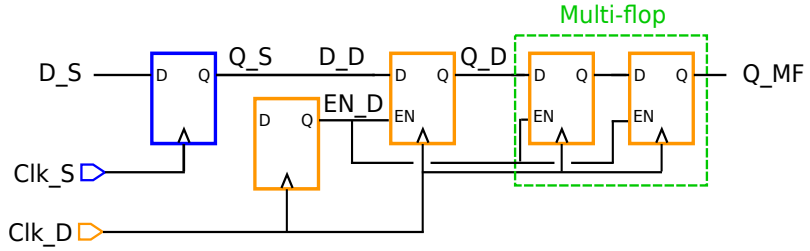
— La présence de signaux de contrôle en entrée des registres de la MF ;

— La présence de logique sur les horloges d'entrée des registres de la MF.

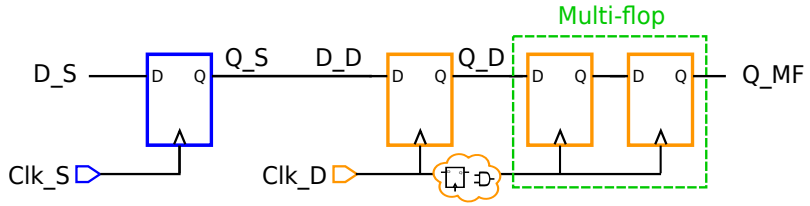
La figure 7.1 présente les trois modèles associés à la détection des structures de type multi-flop selon l'axe de configuration.



(a) Structure classique



(b) Structure avec contrôle



(c) Structure multi-horloges

FIGURE 7.1 – Les structures associées au modèle de type multi-flop

La validation La présence d'une MF en sortie n'empêche pas la perte des données car ce problème intervient avant la capture du CDC par le registre de destination. Ce phénomène résulte d'une des deux raisons suivantes : le passage d'une horloge

rapide à une horloge lente ou la mauvaise résolution de la métastabilité (sous-section 2.3, page 32). La propriété associée au modèle "Multi-flop" spécifie qu'après un changement d'état, la donnée transférée Q_S/D_D reste stable pendant au moins deux cycles de l'horloge de destination Clk_D. L'idée est d'assurer que la donnée transférée ne peut pas être perdue, quelle que soit la raison.

```
property Maintien_donnee(Clk, Data);
(@(posedge Clk) disable iff (Reset)
!$stable (Data) |=> $stable (Data)[*2]);
endproperty

mf:assert property(Maintien_donnee(Clk_S, Q_S));
```

1.1.2 La recombinaison de signaux avant MF

Les conditions Contrairement à la MF, la recombinaison de signaux avant MF n'est pas une technique de conception des CDC. Elle correspond à un problème potentiel : la capture d'états transitoires dus aux courses de signaux (section 1, page 17). Pour permettre la reconnaissance de recombinaisons de signaux avant MF, deux conditions doivent être respectées :

GC1 Plusieurs signaux ($Q\_S[i]$) provenant d'un domaine d'horloge source et capturés par un même registre de destination (D_D) ;

GC2 Un chemin direct et unique entre la sortie du CDC (Q_D) et un élément séquentiel dans le domaine de destination (Multi-flop).

La configuration Cependant, le capture de courses de signaux intervient au niveau du registre de destination. Ce phénomène peut donc se produire indépendamment de la présence de structures MF. La configuration outil permet de désactiver la condition GC2 afin que le modèle soit reconnu avec ou sans MF dans le domaine de destination. La figure 7.2 présente la structure associée au modèle des courses de signaux avec deux signaux sujets à CDC ($Q\_S[0]$ et $Q\_S[1]$) et recombinés à travers des éléments de logique combinatoire avant d'arriver en entrée (D_D) du registre de destination.

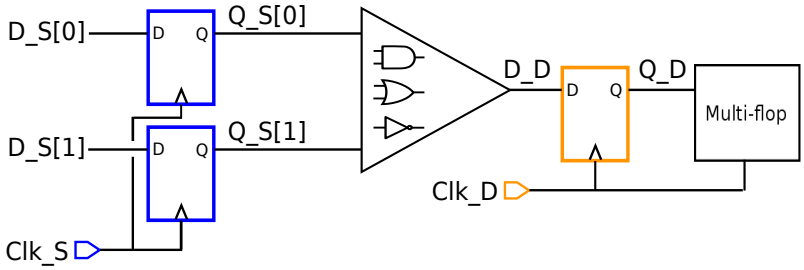


FIGURE 7.2 – La structure associée au modèle "Recombinaison avant MF"

Les mauvaises structures La condition GC1 présente une restriction forte : les signaux recombinés doivent provenir du même domaine d'horloge. Si ce n'est pas le cas, il n'est pas possible de mettre en place un protocole empêchant les courses : le circuit doit être corrigé. Ces structures problématiques sont détectées à travers un

modèle dédié indiquant que le circuit doit être corrigé. La figure 7.3 illustre ce cas à travers la recombinaison avant MF de deux signaux ($Q\_S[0]$ et $Q\_S[1]$) provenant de deux domaines d'horloge différents (Clk_S0 et Clk_S1).

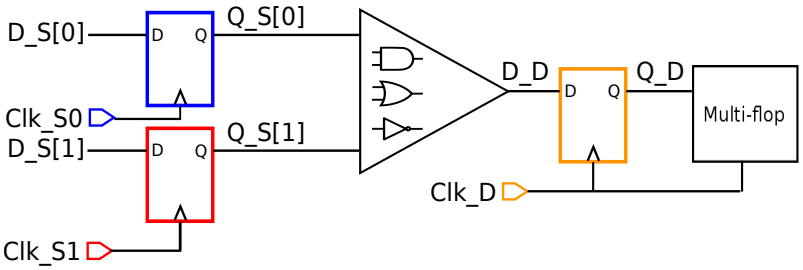


FIGURE 7.3 – Exemple de structure sujette aux courses de signaux

La validation La capture d'états transitoires est due à la dynamique des signaux recombinés. La validation des structures consiste à vérifier que le comportement des signaux est géré par un protocole de temporisation empêchant la génération d'états transitoires. La propriété associée au modèle "Recombinaison avant MF" assure que les données transférées sont encodées en Gray (sous-section 2.1, page 29), c'est-à-dire qu'un seul signal peut changer d'état par cycle de l'horloge source. L'idée est de vérifier qu'aucune course ne peut survenir et donc qu'aucun état transitoire ne peut être capturée par la destination.

```
property Codage_Gray(Clk, Data);
(@(posedge Clk) disable iff(Reset)
!$stable(Data) |-> $onehot(Data ^ $past(Data));
endproperty

gc:assert property(Codage_Gray(Clk_S, Q_S));
```

1.1.3 La recombinaison de signaux après MF

Les conditions Comme pour la recombinaison de signaux avant MF, la recombinaison de signaux après MF correspond à un problème CDC potentiel (sous-section 3.1, page 19). Les conditions de détection sont triples : les deux premières correspondent à la détection de plusieurs CDC entre deux domaines uniques, synchronisés par MF et la troisième à la recombinaison :

CV1 Plusieurs signaux ($Q\_S[i]/D\_D[i]$) provenant d'un domaine d'horloge source et capturés dans un domaine différent de destination ;

CV2 Un chemin direct et unique entre chaque sortie des CDC ($Q\_D[i]$) et un élément séquentiel dans le domaine de destination (Multi-flop).

CV3 Un point de convergence commun (C_D) entre les différentes sorties de multi-flop dans le domaine d'horloge de destination.

La configuration Le problème de recombinaison de signaux après MF est la propagation de données incohérentes. Ce phénomène est dû aux différents scénarios de résolution de la métastabilité pour les signaux recombinés. En cas de mauvaise résolution, le signal sera retardé d'un cycle d'horloge (le maintien de la donnée est

assuré par les propriétés relatives à la validation des multi-flops). Un délai d'au moins un cycle d'horloge se propage aussi bien à travers les éléments combinatoires que séquentiels. Afin de couvrir tous les problèmes potentiels dus à la recombinaison de signaux après MF, les outils permettent d'adapter la détection selon le nombre d'éléments séquentiels traversés entre les multi-flops et le point de convergence. La figure 7.4 présente la structure associée au modèle des convergences de signaux.

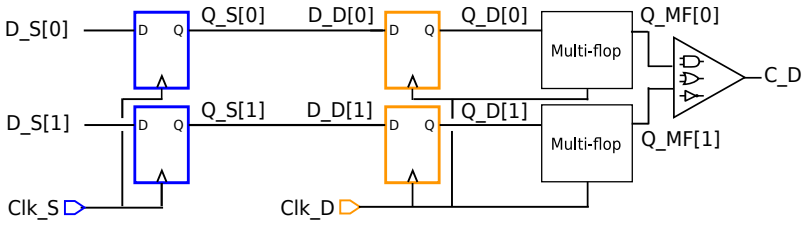


FIGURE 7.4 – La structures associée au modèle "Convergence de signaux"

Les mauvaises structures Pour les mêmes raisons que lors de la recombinaison de signaux avant MF, si les signaux recombinés ne sont pas issus de CDC entre des domaines identiques, la structure est rapportée comme problématique. La figure 7.5 illustre un cas de recombinaison de signaux après MF sujet à la propagation de données incohérentes.

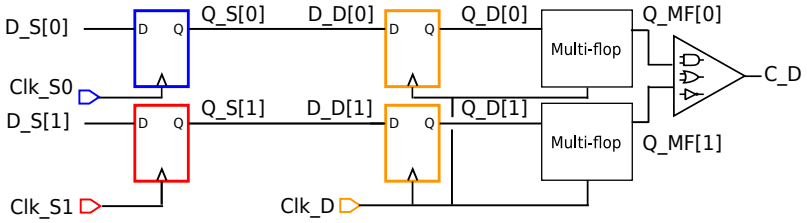


FIGURE 7.5 – Exemple de structure sujette aux incohérences

La validation Le comportement aléatoire de la métastabilité rend les temps de propagations entre les signaux recombinés imprévisibles. La présence de multi-flops n'empêche donc pas le problème de données incohérentes. Le seul moyen pour que la structure fonctionne correctement est d'assurer l'encodage des données en Gray. La propriété associée au modèle "Recombinaison après MF" est la même que pour les structures sujettes aux courses de signaux : les données recombinées sont exclusives.

```
cv:assert property(Codage_Gray(Clk_S, Q_S));
```

La figure 7.6 présente le flot d'analyse des multi-flops implémenté dans les outils de vérification des CDC. L'étape de détection des structures démarre par une identification des multi-flops et se poursuit par une analyse des recombinaisons de signaux avant et après MF. L'étape de validation des protocoles consiste à vérifier des propriétés pour chacun des trois modèles. Pour la validation du maintien de la donnée, les outils se limitent aux CDC allant d'un domaine rapide vers un domaine lent.

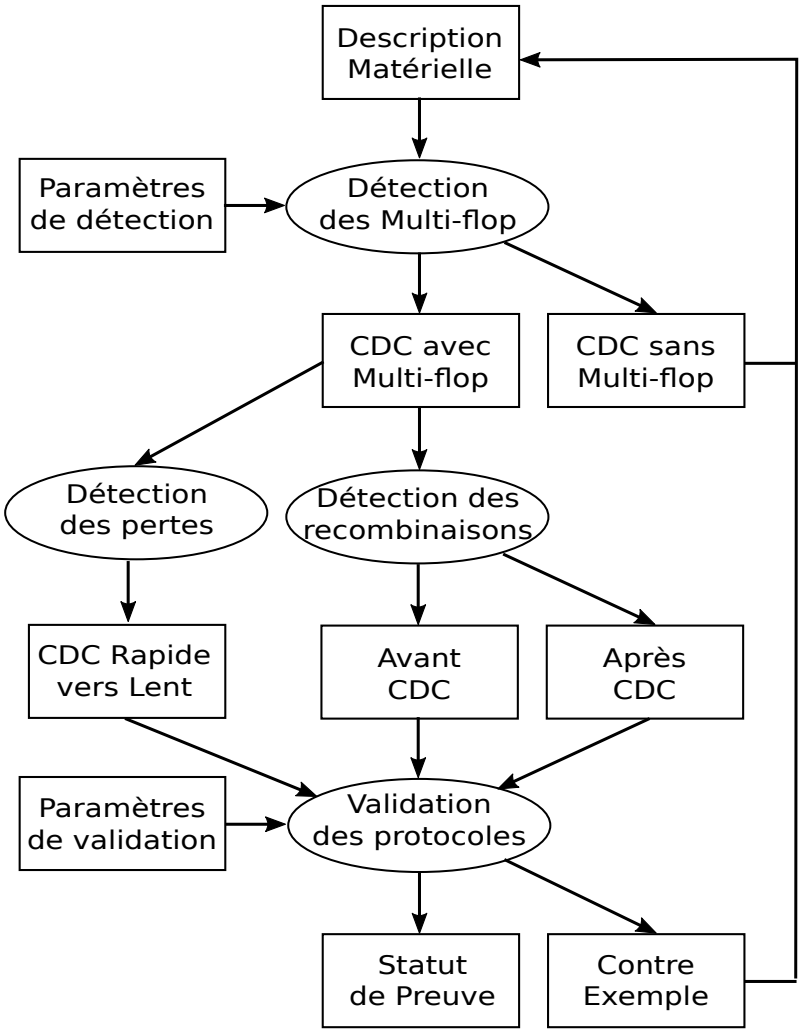


FIGURE 7.6 – Le flot de validation des "Multi-flop"

1.2 Problèmes industriels

Pour assurer une couverture maximale des structures CDC, toutes les MF d'un circuit doivent être identifiées. Cependant, les circuits industriels peuvent contenir des architectures incompatibles avec les modèles CDC des outils. Par ailleurs, les propriétés associées aux modèles ne couvrent pas toutes les techniques de conception possibles.

1.2.1 Les MF à profondeur variable

Dans les circuits, les composants peuvent être alimentés à différents niveaux de tension afin de réduire l'énergie électrique consommée. Or, l'alimentation ayant un impact sur le MTBF, la profondeur des MF doit être adaptée. Afin de faire le meilleur compromis entre performances et robustesse à la métastabilité, des multi-flops avec une profondeur variable sont implémentées [93, 110]. Ces structures peuvent être conçues de deux manières en fonction des besoins :

— Plusieurs MF de profondeurs différentes en entrée d'un MUX ;

— Une MF de profondeur fixe avec un signal retardant la traversée.

Plusieurs MF en entrée de MUX Dans le cas d'une structure avec multiplexeur, il y a une augmentation du nombre de registres, et donc de la place occupée sur le circuit. Afin de factoriser la quantité d'éléments séquentiels, il est possible de définir une profondeur minimale commune et de faire diverger la sortie vers les entrées d'un MUX à travers des profondeurs séquentielles différentes. Cependant, si la profondeur commune décrite par le concepteur est définie à un unique registre, la structure n'est pas identifiable. En effet, la condition MF2 indique que chaque chemin entre les éléments cascadés doit être direct et unique.

Bien que la valeur du MTBF puisse être réduite par la divergence de chemins entre les éléments cascadés, ces structures doivent être détectées. En pratique, la détection de ces MF spécifiques nécessite de définir manuellement une valeur constante sur le signal de configuration du MUX pour n'activer qu'un seul des chemins d'entrée. La figure 7.7 présente un exemple de multi-flop spécifique fonctionnant correctement mais dont la structure n'est pas reconnue par les outils de vérification CDC car le cône de sortie du registre de destination est double. Dans ce cas, le signal de contrôle Sel_MF doit être défini à un état constant afin qu'un seul des deux chemins soit actif.

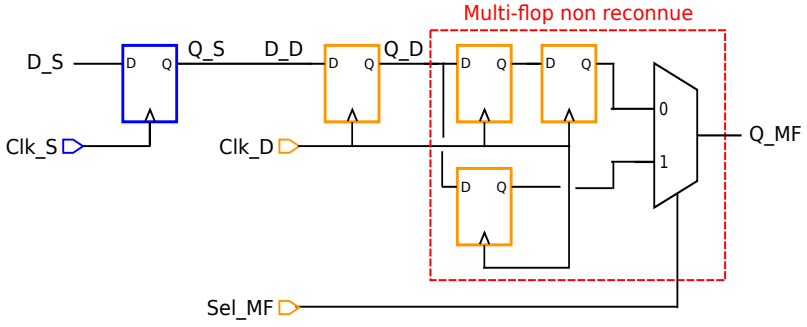


FIGURE 7.7 – Exemple de structure de type "Multi-flop" non reconnue

Un signal émulant la profondeur Pour détecter les structures avec un signal retardant la traversée de la MF, il faut configurer le modèle en autorisant la présence d'un signal de contrôle en entrée des éléments cascadés. Cependant, lors de l'implémentation physique, le contrôle peut être synthétisé de deux façons différentes : une structure de type "Clock Gate" ou des "Mux à recirculation" [9]. La structure de type "Clock Gate" ne pose pas de problème : la multi-flop correspond au modèle classique. À l'inverse, substituer les entrées de contrôle par des "MUX à recirculation" est problématique car la présence de logique combinatoire entre les éléments de la MF n'est pas compatible avec la condition de détection MF2. Comme pour le cas de plusieurs MF en entrée de MUX, il faut configurer le signal retardant la propagation. La figure 7.7 présente les deux types d'implémentation possibles pour les multi-flops intégrant un signal de contrôle. La structure avec Clock Gate ne pose pas de problème d'identification pour les outils car la multi-flop est classique. Cependant, lorsque la multi-flop est implémentée avec des "MUX à recirculation", les outils ne les détectent pas car il y a de la logique combinatoire entre les éléments cascadés.

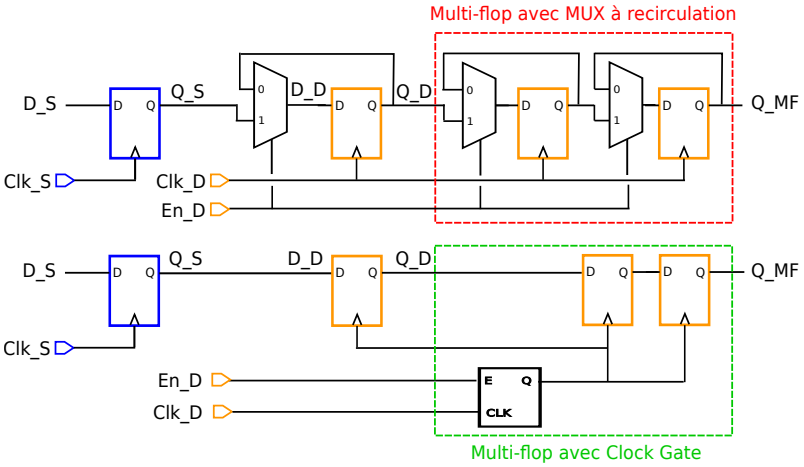


FIGURE 7.8 – Implémentations possibles d'une MF avec contrôle

1.2.2 La recombinaison après MF

Empêcher la propagation de données incohérentes requiert l'exclusivité des signaux recombinés après MF. Si l'approche la plus simple consiste à encoder en Gray dans le domaine source les données transférées, ce n'est pas le seul moyen. Une autre approche consiste assurer l'exclusivité mutuelle entre les chemins recombinés dans le domaine de destination mais ce cas n'est pas couvert par les outils. Par ailleurs, lors de la détection des structures sujettes à la propagation de données incohérentes, les outils s'efforcent de réduire le nombre de cas rapportés. Si deux signaux convergent directement puis convergent avec un troisième, après traversée d'éléments séquentiels, un seul cas sera rapporté. Dès lors, les structures identifiées peuvent inclure des dizaines de CDC et d'étages de logique séquentielle, ce qui complexifie l'analyse. En pratique, afin d'être en mesure d'étudier les différentes structures, la vérification est effectuée de manière itérative en configurant plusieurs niveaux de profondeur séquentielle maximale au moteur de détection. Cependant, il est difficile de définir à l'avance la profondeur souhaitable. Cela demande un effort d'analyse très important,

si bien que les recombinaisons séquentielles après MF sont rarement analysées. La figure 7.9 présente un exemple de convergence après traversée de plusieurs étages de logique combinatoire et séquentielle demandant un effort d'analyse important du fait de la quantité de signaux recombinés. En ne contraignant pas la profondeur séquentielle maximale, cette structure est rapportée par les outils à travers un seul cas de recombinaison de signaux après MF.

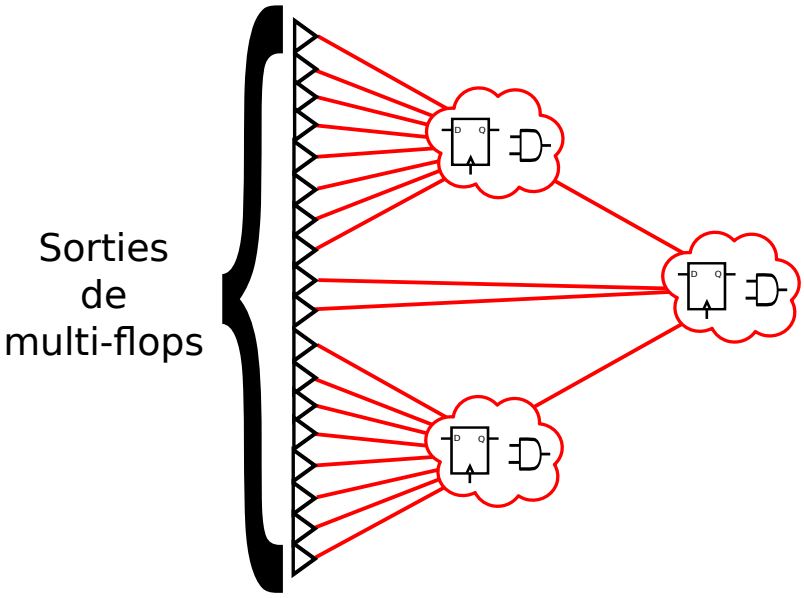


FIGURE 7.9 – Exemple de structure de type "Recombinaison après MF" complexe

1.2.3 La vérification des modèles

Même si toutes les structures liées à l'analyse des MF d'un circuit sont bien détectées, des limitations subsistent. Lors de la validation, les outils cherchent à prouver les propriétés formelles associées aux modèles détectés. Or, ces propriétés ne couvrent pas toujours tous les problèmes potentiels.

La mauvaise résolution de la métastabilité La propriété `mf1` n'est définie que si la fréquence de l'horloge source est supérieure à celle de l'horloge de destination. Cependant, si deux horloges ont la même fréquence, la mauvaise résolution de la métastabilité peut générer une perte de données. Pour couvrir ces cas, l'utilisateur doit définir des fréquences d'horloges permettant la validation de ces interfaces.

La vérification des courses de signaux La propriété assurant qu'aucune course de signaux ne peut survenir définit uniquement l'exclusivité des signaux recombinés. Or, comme expliqué dans le chapitre 2 (sous-section 2.2, page 30), il est possible d'assurer un changement d'état des signaux au même cycle d'horloge sans générer d'états transitoires. Lors de la validation d'un protocole de type "Comportement anti-course", un contre-exemple sera rapporté. Il est alors nécessaire pour l'utilisateur de définir ses propres propriétés pour valider le modèle.

La complexité des convergences séquentielles Lors des convergences après traversée de plusieurs éléments séquentiels, le modèle à vérifier peut être très complexe. L'engin de preuve vérifiant l'intégralité du cône d'influence de la propriété, le temps d'analyse peut être très long, voir non-conclusif. Il est alors nécessaire de simplifier le modèle à vérifier en spécifiant des contraintes de validation. La figure 7.10 présente le flot de détection et validation des structures de type multi-flop avec les limitations associées en rouge.

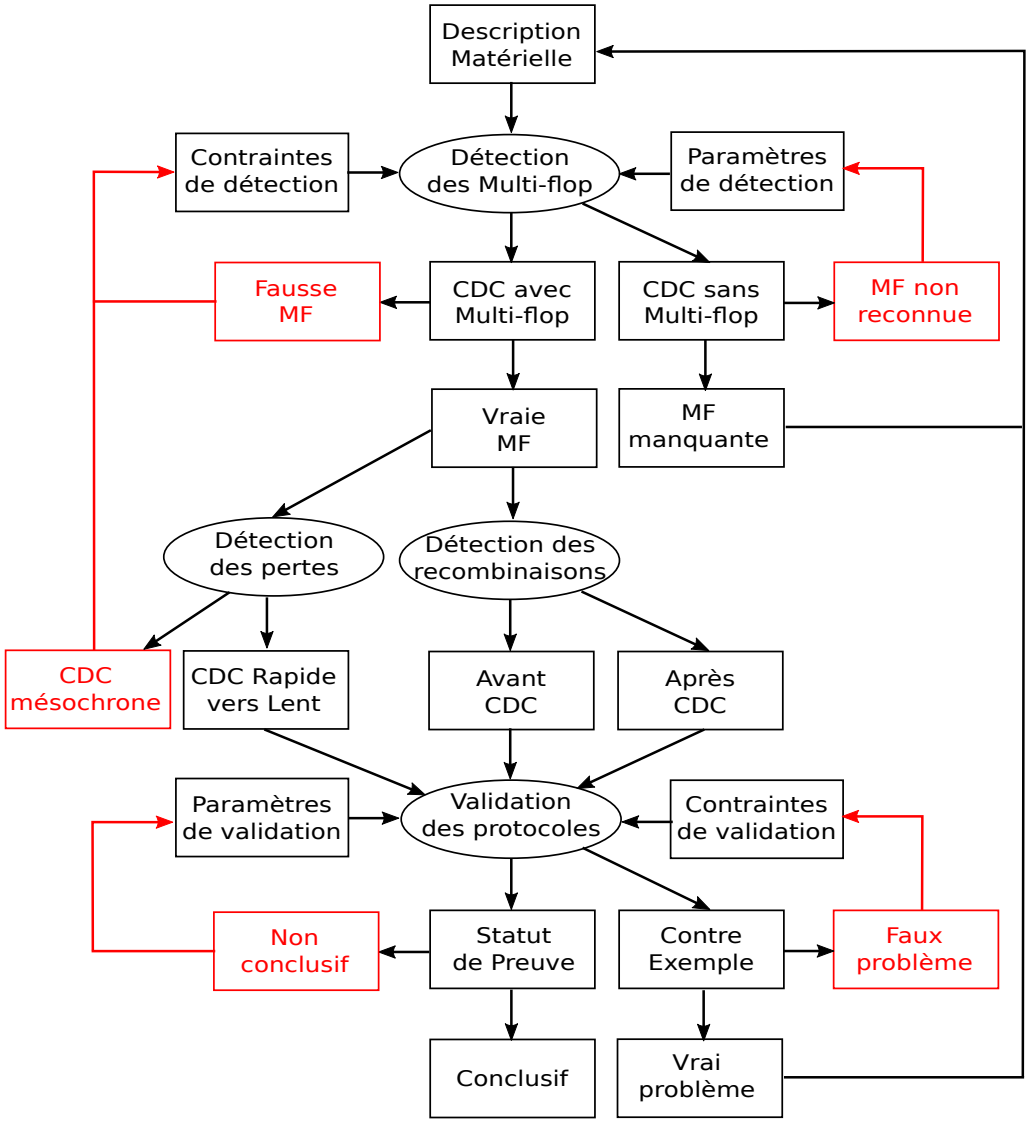


FIGURE 7.10 – Les problèmes associés au flot de validation des "Multi-flops"

1.2.4 Bilan

Les modèles et les propriétés liés à l'analyse des structures de type MF présentent des limitations importantes : certains modèles ne sont pas vérifiés et la validation rapporte des statuts de preuve non-conclusifs voire faux. Afin d'obtenir une qualité de résultats optimale en terme de détection des problèmes CDC, des informations doivent être ajoutées par l'utilisateur. Cependant cette approche requiert un effort

humain trop important pour répondre aux exigences de temps de mise sur le marché. Par ailleurs, la configuration des modèles et la définition de contraintes sont dépendantes des outils et des circuits.

2 Définition des modèles optimisés

Les modèles implémentés dans les outils, pour l'analyse des structures de type MF, ne permettent pas de couvrir l'ensemble des méthodes de conception. Nous proposons donc trois nouveaux modèles permettant d'obtenir automatiquement des résultats exhaustifs, en termes de détection et de validation des structures liées aux MF :

— La "Multi-flop à recirculation" ;

— La "Multi-flop avec divergence" ;

— La "Recombinaison contrôlée après MF".

Les deux premiers modèles permettent de reconnaître les MF avec une profondeur variable. Ces structures correspondent à des synchronisations potentielles car, bien qu'elles intègrent une succession de registres en cascade, leurs architectures respectives réduisent la valeur du MTBF. Dès lors, l'intérêt de ces méthodes de conception en terme de temps de résolution de la métastabilité n'est que potentiel. Le troisième modèle permet d'identifier des cas de signaux recombinés après MF dont l'exclusivité n'est pas assurée par le comportement des signaux mais par un signal de contrôle (protocole).

2.1 Multi-flop à recirculation

La structure de type "Multi-flop à recirculation" inclut de la logique combinatoire entre les registres de la MF. La partie combinatoire a pour but de retarder, sous certaines conditions, le transfert des données entre les éléments cascadés. La technique la plus connue est le MUX à recirculation. La structure est donc la même que pour une MF classique en considérant que les éléments cascadés peuvent être des registres avec un MUX à recirculation à l'entrée. Afin d'éviter la génération d'états métastables, le signal de contrôle du MUX doit être échantillonné dans le domaine de destination. La détection du modèle "Multi-flop à recirculation" se fait via les trois conditions suivantes :

RMF1 Une donnée (Q_S) émise par un domaine d'horloge source et réceptionnée par un domaine d'horloge différent ;

RMF2 Un chemin entre la sortie du CDC (Q_D) et un élément séquentiel et/ou un MUX à recirculation (D_D) dans le domaine de destination.

RMF3 Le signal de contrôle de chaque MUX à recirculation (Sel_MF) est commun et échantillonné dans le domaine de destination.

La figure 7.11 présente la structure associée au modèle "Multi-flop à recirculation". Le modèle intègre trois groupes de signaux : les données transférées (Q_S/D_D), les données capturées (Q_D) et le signal de contrôle (Sel_MF). Le signal Sel_MF

peut être considéré comme le contrôle permettant la propagation de la donnée dans le domaine de destination.

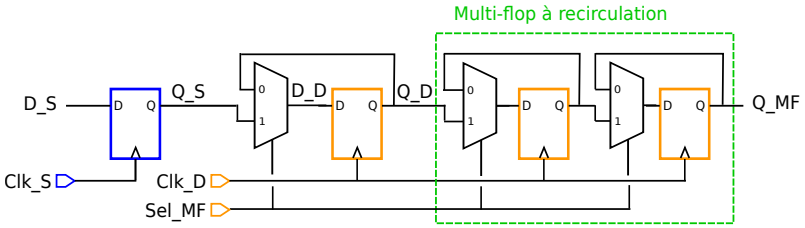


FIGURE 7.11 – La structure associée au modèle "Multi-flop à recirculation"

2.2 Multi-flop avec divergence

La structure de type "Multi-flop avec divergence" est composée, comme son nom l'indique, de chemins divergents en sortie du registre de destination. Étant donné que les éléments cascadés sont utilisés pour éviter la propagation de la métastabilité, les chemins divergents doivent être exclusifs. Autrement dit, à un cycle d'horloge donné, les registres de la MF ont un cône de sortie unique. Pour cela, un signal de contrôle commun doit être présent sur chaque chemin divergent.

Les conditions Le modèle intègre trois groupes de signaux : les données transférées (Q_S/D_D), les données capturées (Q_D) et le/les signal/signaux de contrôle (En_MF). Le signal En_MF peut être considéré comme le contrôle assurant un cône de sortie unique entre les registres de la MF. Le bon fonctionnement de ce modèle repose sur l'exclusion mutuelle entre les chemins divergents. Les conditions de détection du modèle "Multi-flop avec divergence" sont :

DMF1 Une donnée (Q_S/D_D) émise par un domaine d'horloge source et réceptionnée par un domaine d'horloge différent ;

DMF2 Un cône multiple en sortie du registre de destination (Q_D).

DMF3 Un signal de contrôle commun à chaque chemin (En_MF) et échantillonné dans le domaine de destination.

La figure 7.12 présente la structure associée au modèle "Multi-flop avec divergence".

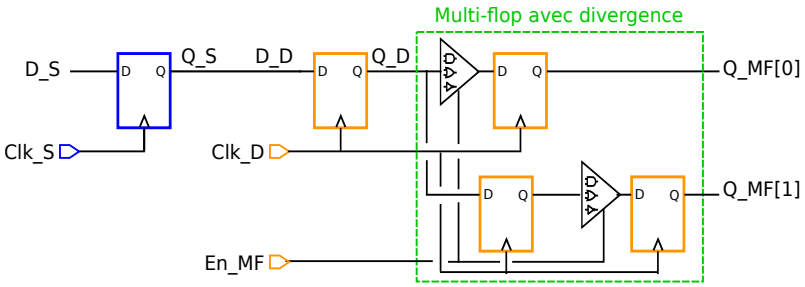


FIGURE 7.12 – La structure associée au modèle "Multi-flop avec divergence"

La validation Ce n'est pas parce qu'un signal de contrôle commun En_MF existe que la structure "Multi-flop avec divergence" correspond à une multi-flop à profondeur variable. Le modèle doit être validé à travers l'analyse de l'exclusivité des chemins divergents. La propriété associée, "Chemins_exclusifs" définit donc que lorsque le signal de contrôle En_MF est stable, un seul des chemins divergents peut être sujet à des changements d'états (les autre sont stables) : ce comportement est très proche de la propriété "Exclusivite_physique" utilisé pour les horloges qui ne sont jamais actives simultanément. Cependant, les profondeurs des multi-flops étant différentes, pour éviter toute propagation de données incohérentes, En_MF ne doit jamais changer d'état lorsque l'horloge de destination est active. La seconde propriété "Stabilite", vérifie donc la stabilité du signal de contrôle En_MF lorsque l'horloge de destination est active.

```
property Chemins_exclusifs(Clk, Ctrl, Data0, Data1);
(@(posedge Clk) disable iff(Reset)
$stable(Ctrl) |=> $stable(Data0) | $stable(Data1));
endproperty

property Stabilite(Clk, Data);
(@(posedge Clk) disable iff(Reset)
$stable(Data);
endproperty

dmf1:assert property(Chemins_exclusifs(Clk_D, En_MF, Q_MF[0], Q_MF[1]));
dmf2:assert property(Stabilite(Clk_D, En_MF));
```

2.3 La "Recombinaison contrôlée après MF"

Les conditions Le principe d'exclusion mutuelle entre les chemins peut s'appliquer aux cas des recombinaisons de signaux après MF. L'idée est de rechercher des signaux de contrôle communs aux différents chemins avant recombinaison. Si des cas sont détectés alors la structure est rapportée à travers le modèle "Recombinaison contrôlée après MF". Les contrôles sont échantillonnés dans le domaine de destination et assurent l'exclusivité entre les chemins. Les conditions de détection du modèle sont celles de la structure "Recombinaison après MF" classique auxquelles s'ajoute la condition suivante :

OCV1 Un signal de contrôle commun à plusieurs chemins (En_C) et échantillonné dans le domaine de destination.

La figure 7.13 présente la structure associée au modèle "Recombinaison contrôlée après MF".

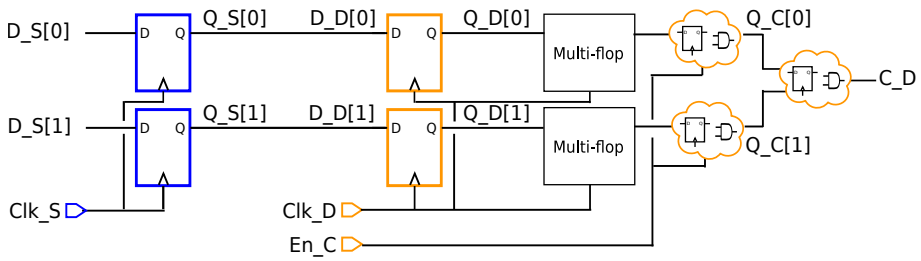


FIGURE 7.13 – La structure associée au modèle "Recombinaison contrôlée après MF"

La validation Les propriétés associées à ce modèle sont les même que pour les cas de "Multi-flop avec divergence" en configurant la propriété $Chemins\_exclusifs$ avec En_C et $Q\_C[i]$ et la propriété $Stabilite$ avec En_C. Elles assurent que le signal de contrôle ne change jamais d'état lorsque l'horloge de destination est active et contrôle l'exclusivité des signaux recombinés après MF.

```
ocv1: assert property(Chemins_exclusifs(Clk_D, En_C, Q_C[0], Q_C[1]));
ocv2: assert property(Stabilite(Clk_D, En_C));
```

2.4 La validation des potentiels problèmes CDC

Si les modèles permettant la détection des structures de type multi-flop et des potentiels problèmes associés présentent des limitations, les étapes de validation présentent elles aussi des disfonctionnements.

La perte de données Dans les outils, la propriété "Maintien_donnee" n'est vérifiée que pour les CDC allant d'une horloge rapide vers une horloge lente. Or, si les domaines d'horloge source et destination fonctionnent à des fréquences équivalentes, des données peuvent être perdues lorsque qu'un état métastable survient et se résout au mauvais état. Nous proposons donc de vérifier le maintien de la donnée pour toutes les interfaces asynchrones avec une horloge d'émission dont la fréquence est supérieure ou égale à celle de l'horloge de réception. Notons que les modèles et les propriétés proposés pour les multi-flops à profondeur variable n'empêchent pas la perte de données : la propriété "Maintien_donnee" doit être vérifié si besoin est.

Les données incohérentes La validation des recombinaisons après MF assure qu'aucune donnée incohérente ne peut être propagée. Si un modèle de type "Recombinaison contrôlée après MF" est identifié, les propriétés ocv1 et ocv2 sont vérifiées. Si un contre-exemple est trouvé, cela ne veut pas dire qu'un problème peut survenir. En effet, les signaux recombinés peuvent être encodés en Gray, il faut donc, dans ce cas, vérifier la propriété "Encodage_gray". Nous proposons donc de discriminer le cas des convergences contrôlées après MF et de vérifier les propriétés associées. Si elles sont prouvées, la structure est validée, à l'inverse l'exclusivité des signaux recombinés doit être vérifiée.

Les courses de signaux Concernant la propriété associée recombinaisons avant MF elle doit être modifiée. Même s'il est vrai que des signaux exclusifs (encodés en Gray) sont immunisés aux courses de signaux, c'est une condition forte. Sur une porte logique classique de type ET ou OU à deux entrées, une sortie ne peut générer un état transitoire que si les signaux recombinés changent d'état au même cycle et dans des directions opposées. Dès lors, des signaux qui changent au même cycle mais dans la même direction ne sont ni encodés en Gray ni sujets aux courses de signaux. Nous proposons donc de vérifier tous les comportements permettant d'assurer un transfert correct des données lors de la recombinaison de signaux avant MF sur des portes simples type "ET" et "OU". La propriété vérifiée assure que si des signaux recombinés changent d'état au même cycle d'horloge, leur état est le même.

```
property Anti_course(Clk, Data0, Data1);
(@(posedge Clk) disable iff(Reset)
```

```
!$stable(Data0) && !$stable(Data1) |-> Data0 == Data1;
endproperty

ng1:assert property(Anti_course(Clk_S, Q_S[0], Q_S[1]));
```

Bilan Les deux modèles "Multi-flop avec divergence" et "Multi-flop à recirculation" permettant la détection d'une structure MF à profondeur variable assurent une reconnaissance automatique de toutes les architectures relatives à l'analyse des MF. Cependant, l'identification de ces dernières ne prend pas en compte leur performance en terme de résolution de la métastabilité. Dès lors, ces cas doivent être rapportés aux personnes en charge de la conception du circuit afin de confirmer l'intégrité du MTBF. La figure 7.14 présente le flot proposé pour la validation des structures de type multi-flop. Il intègre les modèles proposés en vert ainsi que les optimisations en termes de validation des protocoles.

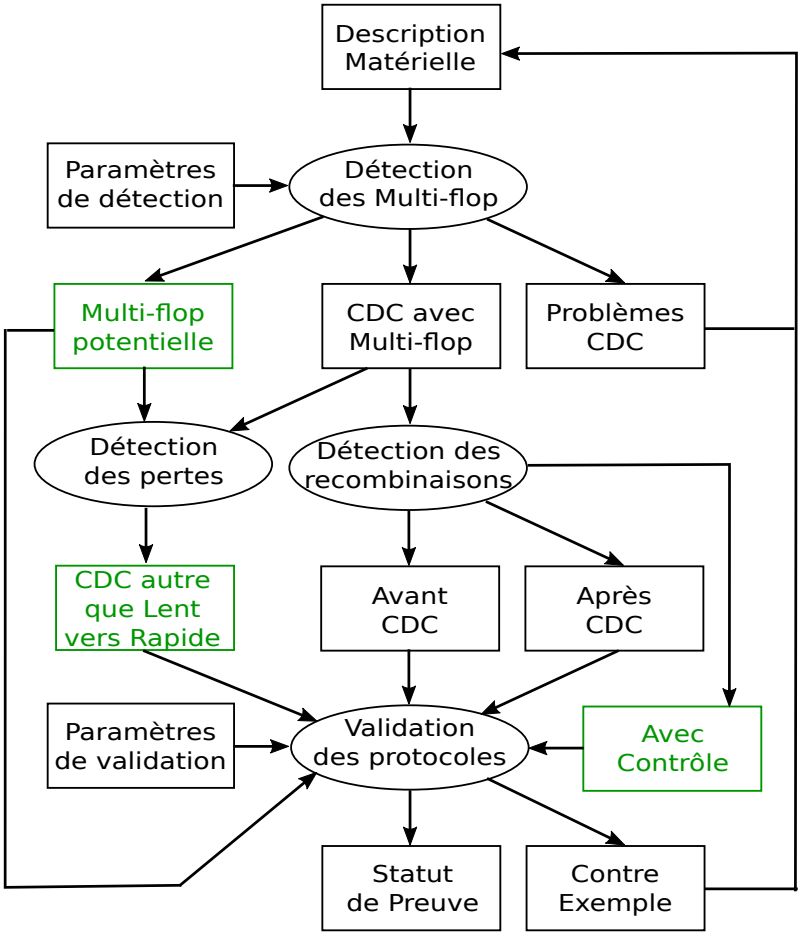


FIGURE 7.14 – Le flot proposé pour la validation des multi-flops

3 Mise en application

3.1 La mise en place des tests

Les caractéristiques du circuit En termes d'analyse des multi-flops, le circuit CPUSS (section 2, page 8) présente les caractéristiques suivantes :

— 1548 structures de type multi-flop ;

— 153 recombinaisons avant MF ;

— 243 recombinaisons après MF.

Dans le circuit, l'implémentation des multi-flops est dépendante de la stratégie de réduction de la consommation d'énergie. Si la destination est toujours active, une structure classique est utilisée. Cependant, si le bloc intégrant la multi-flop peut être éteint, la structure intègre un signal de contrôle. De plus, si la destination peut être alimentée à différents niveaux de tension, les multi-flops ont une profondeur variable et correspondent aux modèles potentiels "Multi-flop à recirculation" ou "Multi-flop avec divergence". Pour résumer, les structures multi-flop du circuit présentent trois architectures possibles :

— 906 structures MF classiques ;

— 553 multi-flops avec contrôle ;

— 84 cas de MF à profondeur variable.

L'approche choisie L'approche proposée pour l'analyse des multi-flops se base sur la détection des structures à profondeur variable à travers les modèles "Multi-flop à recirculation" et "Multi-flop avec divergence". Toutes les structures classiques et avec un unique signal de contrôle sont identifiées correctement par les outils. Concernant les structures à profondeur variable, elles sont identifiées par les outils comme non synchronisées avec une raison particulière : soit la présence de logique combinatoire entre les éléments séquentiels, soit parce que la sortie du registre de destination diverge. À partir de ces informations, il est possible de vérifier, à l'aide d'un script, si un signal de contrôle commun existe et, si c'est le cas, spécifier la structure comme multi-flop aux outils. Pour les cas de convergence, le signal de contrôle est identifié avec la même approche.

Après l'identification de toutes les structures de type MF du circuit, la seconde étape est la validation des modèles. Les premiers modèles à valider concernent les multi-flops à profondeur variable et les cas de "Recombinaison contrôlée après MF". Une fois les structures validées, les cas de perte de données sont vérifiées, notamment pour les interfaces mésochrones. Concernant la validation des recombinaisons de signaux avant MF, le comportement anti-courses mais non-exclusif est spécifié, à travers des contraintes de validation, sur les cas dont les sources proviennent de ports d'entrée primaires.

3.2 Les résultats obtenus

Description des tests Les expérimentations en termes de détection des multi-flops ont été réalisées avec l'outil 1 car il est le seul du marché à proposer différents

1	2	3	4
Types de Structure	Nombre de cas	Détection Outil 1	Détection Robuste
MF Classique	906	OK	OK
MF avec contrôle	558	OK	OK
MF à profondeur variable	84	N.A.	OK
Recombinaison avant MF	152	OK	OK
Recombinaison après MF	135	NOK (6)	OK
Recombinaison après Seq	124	NOK (52)	OK
Toutes	1959	1522 (77%)	OK

TABLE 7.1 – Résultats de détection des structures associées aux MF

modèles. Les résultats sont rapportés dans le tableau 7.1. Les deux premières colonnes donnent respectivement le type de structure CDC et le nombre de cas présents dans le circuit. La colonne 3 donne le nombre de cas détectés par l'outil. La colonne 4 donne le nombre de cas détectés avec la méthode proposée.

La validation est effectuée avec l'outil 2 car il est le seul permettant à un utilisateur de spécifier ses propres propriétés formelles. Les résultats sont rapportés dans le tableau 7.2. Les deux premières colonnes donnent respectivement les propriétés vérifiées et leur nombre. La colonne 3 donne le nombre de cas validés avec l'outil. La colonne 4 donne le nombre de cas validés avec la méthode proposée. La dernière colonne donne le temps d'analyse nécessaire pour obtenir un statut de preuve conclusif et cohérent avec la méthode proposée. L'outil 2 a été configuré avec un effort illimité en terme de temps d'analyse afin d'obtenir un résultat conclusif (preuve ou contre-exemple) dans tous les cas de figure. Lorsque les outils n'ont pas de modèle (ou la propriété) permettant de détecter (ou valider) les structures implémentées, "N.A." (Non applicable) est rapporté. À l'inverse si les outils ou l'approche prennent en compte les modèles (ou les propriétés) et détectent (ou vérifient) l'intégralité des structures, "OK" est indiqué. Si les modèles (ou les propriétés) existent mais que seule une partie des structures est rapportée (ou validé), "NOK(x)" est rapporté (avec x le nombre de structures identifiées ou de propriétés validées).

Analyse des résultats Les résultats rapportés en termes de détection (tableau 7.1) montrent que les modèles associés aux structures MF à profondeur variable et "Recombinaison contrôlée" permettent de détecter automatiquement l'intégralité des structures du circuit. À l'inverse, les outils de vérification statique, ne reconnaissent pas les multi-flops à profondeur variable et, par implication, peu de recombinaisons après MF sont identifiées.

Concernant la validation, les outils ne couvrent pas tous les cas de perte de données potentielles (interfaces mésochrones manquantes), rapportent de faux problèmes de courses de signaux et de données incohérentes lorsque les convergences sont contrôlées. Notre approche permet en plus d'assurer des résultats de qualité en termes d'analyse des MF, de confirmer si les structures détectées comme "MF à recirculation" et "MF avec divergence" fonctionnent correctement. Notons par ailleurs que l'intégrité du MTBF n'est pas couvert par l'analyse CDC.

1	2	3	4	5
Type de propriétés	Nombre de cas	Validation Outil 2	Validation Robuste	Temps d'analyse
Maintien_donnee (MF classique)	56	NOK (54)	OK	50 min
Maintien_donnee (MF contrôle)	177	OK	OK	2.5 h
Maintien_donnee (profondeur variable)	42	OK	OK	40 min
Codage_Gray (avant MF)	19	OK	OK	3 min
Anti_course (avant MF)	133	N.A.	OK	30 min
Codage_Gray (après MF)	135	OK	OK	20 min
Codage_Gray (après seq)	111	OK	OK	4 h
Chemins_exclusifs (après seq)	13	N.A.	OK	5 min
Toutes	686	538 (78%)	OK	>8 h

TABLE 7.2 – Résultats de validation des propriétés associées aux MF

Concernant le temps d'analyse, notons que l'intégralité des signaux synchronisés par MF et sujets à de potentielles pertes de données sont utilisés pour le contrôle de protocoles de séquençage. Le maintien des états est donc assuré par le protocole. En considérant, que le protocole est vérifié pour le transfert des données, il est inutile de chercher à valider le protocole pour la perte de données. Dès lors, le temps d'analyse peut être divisé par deux.

4 Conclusion

L'analyse des multi-flops est la base de la vérification CDC. Cependant, sur des circuits industriels, les outils ne tiennent pas compte de certaines structures lors de la détection. En outre, les propriétés permettant de valider les cas de signaux recombinés en amont ou en aval d'un CDC sont trop spécifiques, générant le rapport de faux contre-exemples lors de l'étape de validation. La prise en compte des MF à profondeur variable permettrait l'extraction de toutes les structures possibles de manière automatique. De même, le modèle "Recombinaison contrôlée après MF" permet de réduire la complexité des modèles et de valider certains cas. Pour finir les signaux synchronisés par MF et utilisés comme contrôle dans les protocoles de séquençage doivent être exclus de l'analyse du maintien de la donnée. Pour les sous-systèmes CPUSS, l'approche proposée présente une qualité de résultats, en termes de détection et de validation, conforme aux spécifications techniques. Par ailleurs en considérant que l'analyse du maintien de la donnée peut être exclue si les signaux

synchronisés par MF sont utilisés pour des protocoles de séquençage, le temps d'analyse peut être réduit de moitié. Néanmoins, il convient de noter que la reconnaissance des structures "Multi-flop à divergence" peut fortement détériorer la qualité des résultats de détection structurelle du fait de l'identification de fausses structures. Par ailleurs, les différentes solutions proposées ne présentent un intérêt que dans le cas où l'étape de validation des protocoles est effectuée.

Chapitre 8

L'analyse des synchronisations de données

L'analyse des structures dédiées à la synchronisation des données correspond à la deuxième étape de la vérification des CDC. Si aucun cas n'est identifié, tous les bus de données, sujets à CDC, sont rapportés comme problématiques. À l'inverse, la présence de ces structures assure que les problèmes telles que la capture de courses de signaux, la propagation de données incohérentes ou la perte de données ne peuvent survenir. En effet, à partir du moment où les structures respectent un protocole de séquençage, la cohérence et la stabilité des données transférées est assurée.

Le principal problème est que les techniques dédiées au transfert de bus de données sujets à CDC sont complexes à implémenter. De plus, diverses structures de synchronisation ont été développées en fonction des besoins en termes de performances. Les protocoles de séquençage doivent donc être validés formellement afin d'éviter tout dysfonctionnement. Notons que l'analyse des synchronisations de données requiert la détection préalable de signaux de contrôles synchronisés par multi-flop.

1 Formalisation des problèmes outils

1.1 Motifs et propriétés

L'analyse des structures de synchronisation de données est découpée, dans les outils, en deux parties. La première est la détection des modèles et, la seconde est la validation des protocoles de séquençage [36, 97, 99]. Afin d'éviter les différents problèmes liés aux CDC, le transfert des informations peut être géré de trois manières différentes :

— "Poignée de main" ou Handshake ;

— "Contrôle unique" ou Control-based ;

— "Premier arrivé premier sorti" ou FIFO.

1.1.1 La structure "Poignée de main"

Les conditions Cette structure assure un séquençage des étapes d'envoi et de réception des données à l'aide de deux signaux de contrôle : la requête et l'accusé de

réception (sous-section 3.1, page 33). Les conditions permettant la reconnaissance des structures de type "Poignée de main" sont :

H1 Un signal de donnée, $(Q\_S[N-1:0]/D\_D[N-1:0])$, allant du domaine d'horloge source au domaine de destination sans être synchronisé par multi-flop ;

H2 Un signal de contrôle "aller", (Req_CDC), émis par le domaine d'horloge source et allant vers le registre de donnée de destination à travers une multi-flop ;

H3 Un signal de contrôle "retour" (Ack_CDC), émis par le domaine d'horloge destination et allant vers le registre de donnée source à travers une multi-flop.

La figure 8.1 présente la structure associée au modèle de type "Poignée de main". Il intègre la détection des deux signaux de contrôle synchronisés par multi-flop : Req_CDC et Ack_CDC.

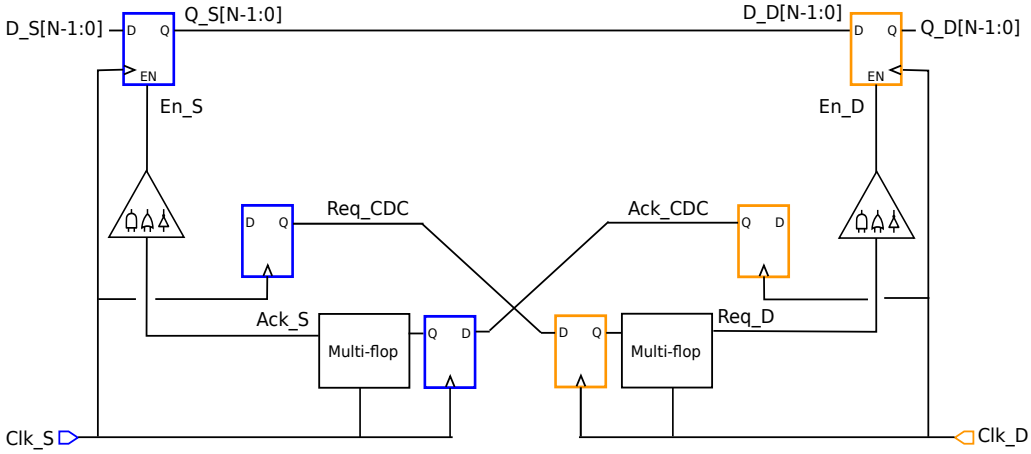


FIGURE 8.1 – La structure associée au modèle "Poignée de main"

Les propriétés formelles La première propriété "Stabilite_et_coherence" associée à la technique de synchronisation de type "Poignée de main" exprime la stabilité des données lors de leur capture. Les quatre autres propriétés sont basées sur Req_CDC, Req_D, Ack_CDC et Ack_S : elles définissent le protocole de séquençage : dans le domaine source, la requête transmise correspond à l'inverse de l'accusé de réception retardé d'un cycle et, côté destination, l'accusé de réception transmis correspond à la requête retardée d'un cycle.

```
property Stabilite_et_coherence(En_D, D_D);
(@(posedge Clk_S or posedge Clk_D) disable iff(Reset)
En_D |-> $stable(D_D));
endproperty

property Protocole_HS(Clk, FSM_In, FSM_Out);
(@(posedge Clk) disable iff(Reset)
FSM_In |=> FSM_Out;
endproperty
```

```
h1: assert property(Stabilite_et_coherence(Req_D && !Ack_CDC, D_D));
h2: assert property(Protocole_HS(Clk_S,!Ack_S,Req_CDC));
h3: assert property(Protocole_HS(Clk_S,Ack_S,!Req_CDC));
h4: assert property(Protocole_HS(Clk_D,Req_D,Ack_CDC));
h5: assert property(Protocole_HS(Clk_D,!Req_D,!Ack_CDC));
```

1.1.2 La structure "Contrôle unique"

Les conditions Cette technique, comme son nom l'indique, est basée sur l'utilisation d'un seul signal de contrôle. En pratique, elle ne s'applique qu'aux interfaces quasi-synchrones, c'est un protocole de séquençage partiel (sous-section 3.3, page 37). Sa structure correspond à l'architecture minimale commune à tous les types de synchronisation de données : c'est un modèle générique. Pour permettre la détection des structures de type "Contrôle unique" (protocole de séquençage partiel), les conditions sont limitées à H1 et H2. Une "Poignée de main" intègre donc une structure de type "Contrôle unique".

La figure 8.2 présente la structure associée au modèle de type "Contrôle unique". Il intègre la détection d'un unique signal de contrôle synchronisé par multi-flop : En_CDC.

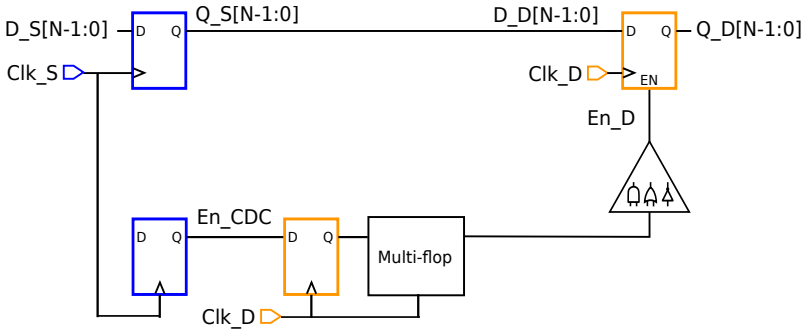


FIGURE 8.2 – La structure associée au modèle "Contrôle unique"

La propriété Pour la synchronisation par un seul signal de contrôle, seule la propriété "Stabilite_et_coherence" est vérifiée, paramétrée avec En_D. Par contre, étant donné que la structure ne correspond pas à un protocole de séquençage, la propriété "Maintien_donnee" (sous-section 1.1.1, page 91) doit être vérifiée sur le chemin de contrôle En_CDC.

```
c1: assert property(Stabilite_et_coherence(En_D,D_D));
c2: assert property(Maintien_donnee(Clk_S, En_CDC))
```

1.1.3 La structure "Premier Arrivé Premier Sorti"

Les conditions Le protocole de séquençage associé à cette technique garantit à la fois robustesse aux problèmes CDC et vitesse de transfert des données (sous-section 3.2, page 35). Sa structure est composée d'une mémoire (FIFO), d'un contrôleur d'envoi et d'un contrôleur de réception. Le contrôleur d'envoi gère les adresses d'écriture et celui de réception gère les adresses de lecture. Chaque élément de la mémoire correspond au registre de donnée source d'une synchronisation de type "Poignée de

main" : le modèle "Premier Arrivé Premier Sorti" intègre donc plusieurs "Poignée de Main" (et donc plusieurs "Contrôle unique"). La condition F1 est la même que H1. Afin de pouvoir différencier les deux structures, le modèle est basé sur la présence d'une mémoire et de bus de contrôle :

F2 Un bus de contrôle ($WrAdd\_CDC$) venant du domaine d'horloge source et allant vers le registre de données de destination à travers une multi-flop ($WrAdd\_D$).

F3 Un bus de contrôle retour ($RdAdd\_CDC$) venant du domaine d'horloge de destination et allant à la fois vers le registre de données source, à travers une multi-flop ($RdAdd\_S$), et vers le registre de données de destination à travers la logique de la mémoire.

La figure 8.3 présente la structure associée au modèle de type "Premier arrivé premier sorti". Il intègre la détection de deux bus de contrôles synchronisés par multi-flop : $WrAdd\_CDC$ et $RdAdd\_CDC$.

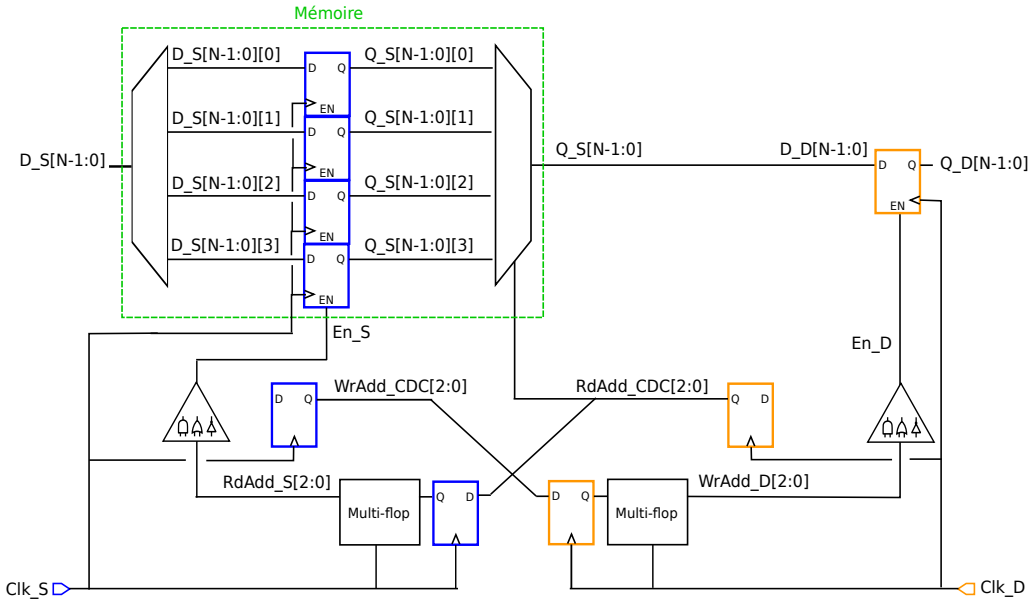


FIGURE 8.3 – La structure associée au modèle "Premier arrivé premier sorti"

Les propriétés Les propriétés formelles associées sont plus complexes et sont basées sur $Q\_S[N-1:0]/D\_D[N-1:0]$, $WrAdd\_CDC$, $WrAdd\_S$, $RdAdd\_CDC$ et $RdAdd\_S$:

```
property Memoire_FIFO(Data_In, Data_Out, WrAdd, RdAdd, full, empty)
(@(posedge Clk_S) disable iff (Reset)
(!full && (WrAdd == N) && (Data_In[N] == data)) |=>
@(posedge Clk_D)(
first_match (##[0:$] (!empty && (RdAdd == N)))
|->(Data_Out == data));
endproperty

property Full_Empty(Clk, Ptr_CDC, Flag);
(@(posedge Clk) disable iff(Reset)
```

```
    !$stable(Ptr_CDC) |-> !Flag;
endproperty

f1:assert property(Memoire_FIFO(D_S,Q_S,WrAdd_CDC,RdAdd_CDC,full,empty));
f2:assert property(Codage_Gray(Clk_S, WrAdd_CDC));
f3:assert property(Codage_Gray(Clk_D, RdAdd_CDC));
f4:assert property(Full_Empty(Clk_S, WrAdd_CDC, full));
f5:assert property(Full_Empty(Clk_D, RdAdd_CDC, empty));
```

Les propriétés "Stabilite_et_coherence" et "Protocole_HS" sont remplacées par la propriété "Memoire_FIFO" qui assure que la FIFO préserve l'ordre et la valeur des données stockées dans la mémoire. Elle définit que si une donnée est écrite à une adresse N alors, lorsque l'adresse de lecture sera égale à N, la donnée lue sera équivalente à celle qui avait été écrite. Les propriétés `f2` et `f3` vérifient que les bus de contrôle sont encodés en Gray. Les propriétés `f4` et `f5` définissent respectivement qu'il n'y a pas d'écriture quand la mémoire est pleine et pas de lecture lorsqu'elle est vide. Pour faciliter la compréhension, les termes "full" et "empty" dans les propriétés remplacent les expressions logiques basés sur les adresses d'écriture et de lecture. La FIFO est pleine lorsque $RdAdd\_S = WrAdd\_CDC + 1$ et vide lorsque $RdAdd\_CDC = WrAdd\_D$.

1.2 Bilan

L'analyse des structures de synchronisation de données est basée sur trois modèles ayant chacun des conditions de détection permettant de les distinguer. Les propriétés associées permettent de vérifier, en plus des comportements relatifs aux CDC, le bon fonctionnement des protocoles de séquençage. La figure 8.4 présente le flot outils pour la validation des synchronisations de données.

1.3 Problèmes industriels

Des modèles trop classiques Lors de la conception de circuits industriels, les structures de synchronisation classiques ne suffisent pas toujours. Pour des raisons de performances ou de consommation d'énergie, des techniques spécifiques peuvent être développées (section 2, page 8) [49,90]. Les modèles implémentés dans les outils n'intégrant pas tous les cas possibles, certaines structures ne sont pas reconnues. La figure 8.5 présente un exemple de structure de type "Premier arrivé premier sorti" spécifique intégrant un séquenceur sur le chemin entre la sortie du registre d'envoi de l'adresse de lecture $RdAdd\_CDC$ et l'entrée registre de données de destination $D\_D[N-1:0]$. Or, la conditions F3 n'est respectée que si le chemin entre le registre d'envoi de l'adresse de lecture (dans le domaine de destination) et l'entrée de donnée du registre de capture $D\_D[N-1:0]$ est purement combinatoire. Le séquenceur étant composé d'éléments séquentiels, la structure n'est pas identifiée.

La détection de fausses structures Afin de permettre la validation des synchronisations spécifiques, sans concevoir des modèles dépendants du circuit à vérifier, la plupart des outils proposent de les détecter à travers le modèle "Contrôle unique".

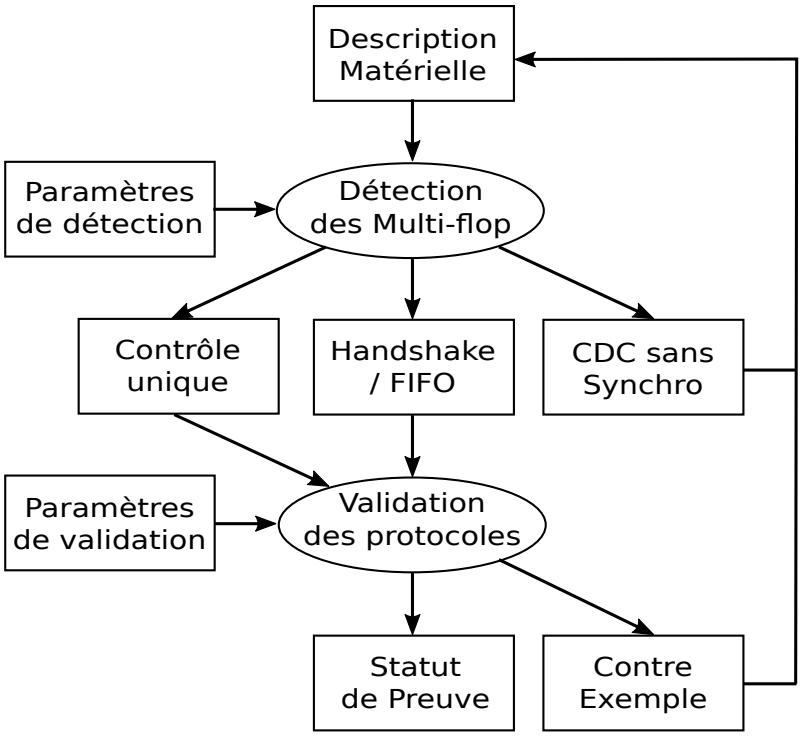


FIGURE 8.4 – Le flot de validation des synchronisations de données

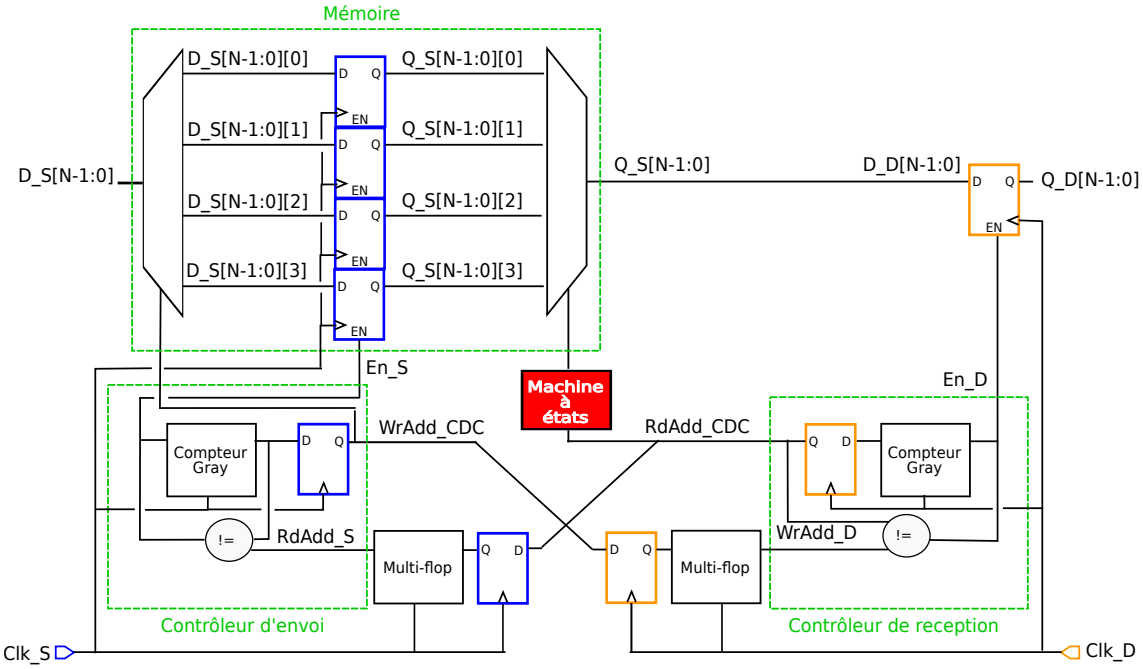


FIGURE 8.5 – Exemple de structure type "Premier arrivé premier sorti" spécifique

En effet, ce modèle est partagé par toutes les structures de synchronisation de données. Cette technique assure la stabilité et la cohérence des données transférées, propriétés communes à tous les protocoles de séquençages. L'idée est d'identifier uniquement le signal qui contrôle la stabilité des données transférées. Cependant, le modèle implémenté dans les outils correspond à l'architecture minimale commune à tous les types de synchronisation. Ni la machine d'état source ni le signal d'activation source (En_S) ne sont inclus. Cela conduit à la détection des signaux respectant la condition H2 mais qui ne sont pas utilisés pour contrôler le transfert de données. La solution est la spécification de contraintes de conception pour guider la détection des synchronisations de données. La plupart des contraintes sont des directives d'exclusion des signaux de contrôle non utilisés pour le transfert des données. La figure 8.6 illustre le problème lié au modèle type contrôle unique avec la présence de deux signaux respectant les conditions associées au modèle mais dont un seul permet aussi de contrôler l'envoi des données. Dans ce cas les outils rapportent le premier signal trouvé et, si ce n'est pas le bon, l'utilisateur doit le spécifier à travers une contrainte d'exclusion.

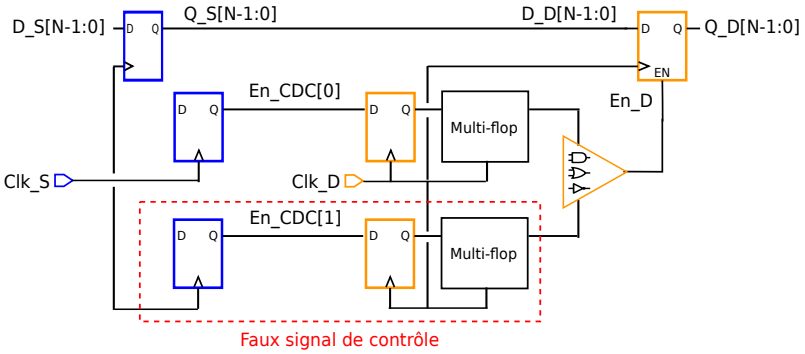


FIGURE 8.6 – Exemple de fausse structure de type contrôle unique

La validation non-conclusive Même si toutes les synchronisations de données sont bien détectées à travers des modèles "Contrôle unique", un problème subsiste. Lors de la validation des structures, l'engin de preuve analyse l'intégralité du cône d'influence des propriétés formelles. Or la propriété `h1` a dans son cône d'influence aussi bien la logique de données que de contrôle. Si le modèle intègre toute la logique de contrôle, les frontières de la structure sont définies et une abstraction du reste du circuit est réalisée. À l'inverse, si la logique de contrôle est partiellement modélisée, l'engin de preuve considère l'intégralité du cône d'entrée de la structure et le temps dédié à la validation peut être très long. Il est alors nécessaire de définir aux outils les frontières du modèle à vérifier, ce sont les contraintes de validation.

Bilan Ce flot requiert un effort humain trop important pour répondre aux exigences de temps de mise sur le marché. La définition des contraintes de conception et de vérification est dépendante des outils, des circuits et nécessite une connaissance poussée des CDC. En outre, la spécification manuelle d'informations augmente le risque d'erreurs humaines, certaines pouvant masquer de vrais problèmes CDC. Cela motive l'élaboration d'un nouveau modèle de synchronisation de donnée, ap-

pelé "Méta-modèle" avec un jeu de propriétés associé. La figure 8.7 présente le flot de validation de protocoles de séquençage avec les problèmes associés en rouge.

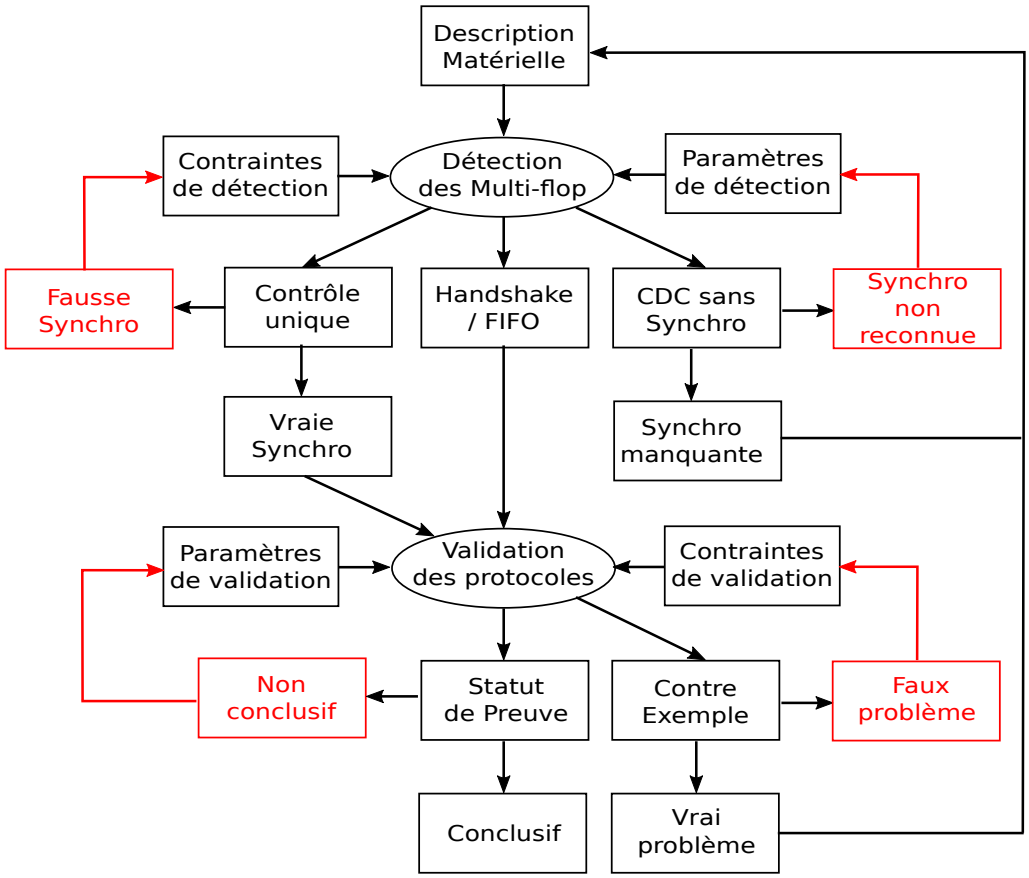


FIGURE 8.7 – Problèmes associés au flot de validation des synchronisations de données

2 Le "Méta-modèle"

2.1 Définition du "Méta-modèle"

Les modèles et les propriétés implémentés dans les outils, pour l'analyse des synchronisations de données, ne permettent pas d'obtenir une qualité de résultats optimale sans un effort humain (contraintes de détection et de validation). Afin d'assurer une validation automatique de ces méthodes de conception, nous proposons un modèle générique : le "Méta-modèle" [49, 51]. Il permet d'obtenir une qualité de détection maximale et une validation locale des protocoles de séquençage sans intervention manuelle.

La structure La structure associée au "Méta-modèle" est très proche de celle du type "Poignée de Main". Elle est composée d'un signal de contrôle unique allant à la fois sur le registre de donnée source et celui de destination (voir figure 8.1). Deux groupes de signaux sont sujets à CDC : les données transférées ($Q\_S[N-1:0]/D\_D[N-1:0]$) et le signal de contrôle (En_CDC). En_CDC est dans le

cône de sortie de En_S, le signal de contrôle du registre de données source, et dans le cône d'entrée de En_D (à travers une multi-flop), le signal de contrôle du registre de données destination. Le signal En_S peut être considéré comme le signal permettant l'émission de données et En_D comme le signal d'activation de la capture. Le signal En_CDC correspond, dans la structure "Poignée de main" à la requête de lecture $ReqCDC$. Les conditions de détection du "Méta-modèle" de synchronisation sont triples : Les deux premières sont H1 et H2, la troisème est :

M3 Un signal d'activation source (En_S) allant vers le registre de transfert du signal de contrôle et vers le registre de données source ;

Le protocole Le signal d'activation de la destination En_D assure que les données transférées sont stables lors de leur capture par la destination. Le signal de contrôle transféré En_CDC correspond au signal de contrôle du "Contrôle unique", à la requête de la "Poignée de main" à l'adresse d'écriture pour la FIFO. Le signal d'activation de la source En_S évite la perte de données en assurant que les données ne changent pas d'état tant qu'elles n'ont pas été capturées par le registre de données de destination. La figure 8.8 présente la structure associée au méta-modèle de structures dédiées à la synchronisation de données.

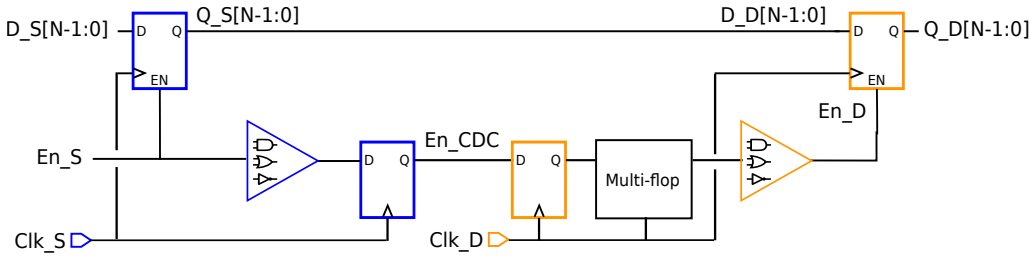


FIGURE 8.8 – La structure associée au "Méta-modèle"

Le bon fonctionnement de ce modèle est basé sur l'exclusion mutuelle des opérations d'envoi et de réception des données. L'absence de perte de données est assurée par un protocole de synchronisation en quatre-phases :

(1) Pas de capture quand la donnée est en train d'être transmise (phase d'écriture : $En\_D =' 0'$ et $En\_S =' 1'$),

(2) Pas de capture de données tant que la donnée n'est pas stabilisée (phase de stabilisation $En\_D =' 0'$ et $En\_S =' 0'$).

(3) Pas d'envoi quand la donnée est en train d'être capturée (phase de lecture : $En\_D =' 1'$ et $En\_S =' 0'$),

(4) Pas de nouvel envoi tant que la donnée n'a pas été capturée (phase d'attente : $En\_D =' 0'$ et $En\_S =' 0'$),

2.2 Ajustement du "Méta-modèle"

En considérant la structure associée au "Méta-modèle", les propriétés formelles doivent être modifiées. La validation est basée uniquement sur les signaux détectés lors de la détection structurelle, à savoir ceux dédiés à l'activation des phases d'envoi (En_S) et de capture (En_D).

2.3 Description des propriétés formelles

Le protocole de synchronisation quatre phases indique que les signaux d'activation En_S et En_D sont mutuellement exclusifs (phases d'émission et de réception pour éviter la métastabilité et la propagation de données incohérentes), et s'alternent (phases de stabilisation et d'attente pour éviter la perte de données). Ces quatre phases peuvent facilement être réécrites en un jeu de propriétés formelles de type SVA.

```
property Exclusivite(En1, En2);
(@(posedge Clk_S or posedge Clk_D) disable iff(Reset)
En1 |-> !En2);
endproperty

property Alternance(En1, En2);
(@(posedge Clk_S or posedge Clk_D) disable iff(Reset)
$fell(En1)|->!En1 until En2);
endproperty

m1: assert property(Exclusivite(En_S, En_D));
m2: assert property(Alternance(En_S, En_D));
m3: assert property(Exclusivite(En_D, En_S));
m4: assert property(Alternance(En_D, En_S));
```

Chaque propriété correspond à une phase du protocole de synchronisation. La propriété `m1` représente la phase d'envoi : quand une donnée est transférée, il n'y a pas de capture, réciproquement, la propriété `m3` représente la phase de capture : quand une donnée est réceptionnée, il n'y a d'envoi. La combinaison de `m1` et `m3` assure l'exclusion mutuelle de En_D et En_S. Il convient de noter que ces deux propriétés sont équivalentes et que la distinction est employée uniquement pour différencier les quatre phases du protocole de synchronisation. Par ailleurs, il est important de remarquer que `m3` est équivalent à la propriété `stability_and_consistency`.

Les propriétés `m2` et `m4` assurent que l'activation de En_D et En_S est alternée : quand En_D se désactive, il doit le rester jusqu'à l'activation de En_S et inversement lorsque En_S se désactive.

Pour assurer que le "Méta-modèle" permet de remplacer les modèles "Poignée de main" et "Premier arrivé premier sorti", il faut vérifier que ces deux stuctures peuvent être instanciées par le "Méta-modèle", et que les propriétés du "Méta-modèle" peuvent être prouvées. Ces structures ont été décrites en VHDL et les propriétés ont été prouvés par l'outil 2. Nous esquisserons simplement la preuve ici.

2.3.1 Application au modèle "Poignée de main"

Si nous appliquons le "Méta-modèle" au modèle de type "Poignée de main", les signaux d'activation En_D et En_S peuvent être instanciés (voir figure 8.11). Le signal En_S est à l'état '1' lorsque la donnée est envoyée ; cela indique que lors de cette phase, la requête et l'accusé de réception sont inactifs ($Req\_CDC =' 0'$ et $Ack\_S =' 0'$). De même, le signal En_D est à l'état '1' lorsque la donnée est capturée ; cela indique que lors de cette phase, la requête est active mais pas l'accusé de réception ($Req\_D =' 1'$ et $Ack\_CDC =' 0'$).

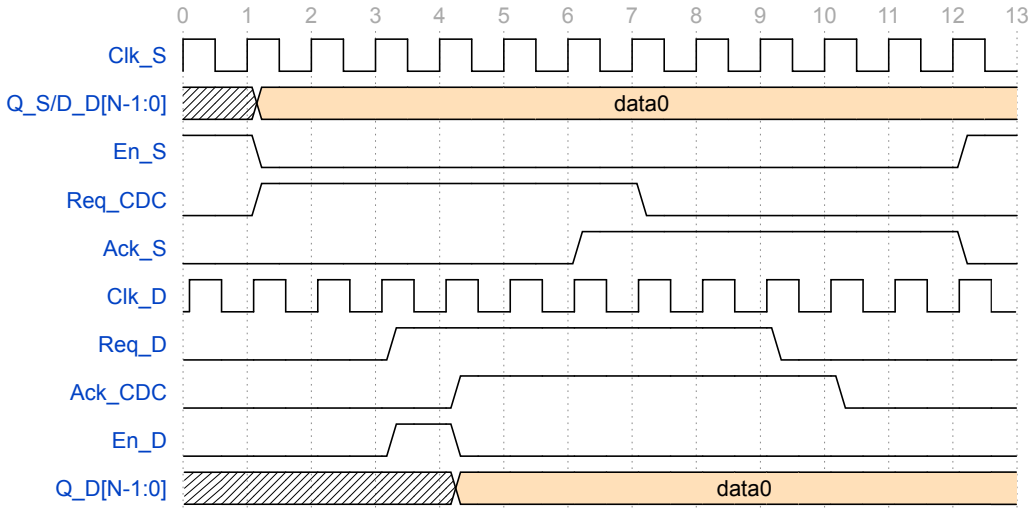


FIGURE 8.9 – Chronogramme associé à la "Poignée de main"

$$En\_D = Req\_D \wedge \neg Ack\_CDC \quad (8.1)$$
$$En\_S = \neg Req\_CDC \wedge \neg Ack\_S \quad (8.2)$$

Par ailleurs, remarquons à partir des propriétés `h2` et `h3`, que l'état du signal Req_CDC est égal à l'inverse de Ack_S retardé d'un cycle de l'horloge source. De la même manière, à partir de `h4` et `h5`, nous pouvons déduire que le signal Ack_CDC est égal à Req_D retardé d'un cycle de l'horloge de destination.

$$Req\_CDC = \text{prev}(\neg Ack\_S) \quad (8.3)$$
$$Ack\_CDC = \text{prev}(Req\_D) \quad (8.4)$$

Ces égalités sont illustrées par le chronogramme présenté figure 8.9 : au cycle ♯7, il y a un front montant sur Ack_S et, un cycle plus tard, il y a un front descendant sur Req_CDC. Le même comportement peut s'appliquer sur le front montant de Ack_CDC, au cycle ♯5, correspondant à un front montant sur Req_D retardé d'un cycle. En remplaçant, dans les équations (8.1) et (8.2), Req_CDC et Ack_CDC par les expressions décrites par les égalités (8.3) et (8.4), il apparaît que l'activation de En_S correspond à un front descendant sur Ack_S et que l'activation de En_D correspond à un front montant sur Req_D. Prouver les propriétés m1, m2, m3 et m4 revient alors à prouver l'exclusivité et l'alternance entre un front descendant sur Ack_S et un front montant sur Req_D.

Cependant, les deux signaux ne sont pas synchronisés sur la même horloge. Nous illustrons donc nos propos à travers les chronogrammes de la figure 8.9. Avec la condition de détection H2, nous savons que le front montant de Req_D au cycle ♯4 correspond au front montant de Req_CDC retardé d'un nombre de cycles d'horloge de destination égal à la profondeur de la multi-flop (= 2). En reprenant l'équation (8.3), nous déduisons que le front montant de Req_CDC (= 2) correspond à un

front descendant sur Ack_S (♯1) retardée d'un cycle d'horloge source. Il y a donc une exclusivité entre l'apparition d'un front descendant sur Ack_S et la présence d'un front montant sur Req_S.

Concernant l'alternance, nous pouvons montrer qu'entre les cycles ♯1 et ♯4 il n'y a pas de nouveaux front descendant sur Ack_S. La preuve vient du fait que le prochain front montant sur Ack_S au cycle ♯7 correspond à un front montant sur Ack_CDC retardé d'un nombre de cycles d'horloge source égal à la profondeur de la multi-flop au cycle ♯5 (condition H3). Or, un front montant sur Ack_CDC correspond précisément à un front montant sur Req_D au cycle ♯4 (equation (8.4)).

2.3.2 Application au modèle "Premier arrivé premier sorti"

Structurellement, le "Méta-modèle" est instancié pour chaque adresse de la FIFO.

$$En\_D(x) = \neg Empty \wedge RdAdd\_CDC = x \quad (8.5)$$

$$En\_S(x) = \neg Full \wedge WrAdd\_CDC = x \quad (8.6)$$

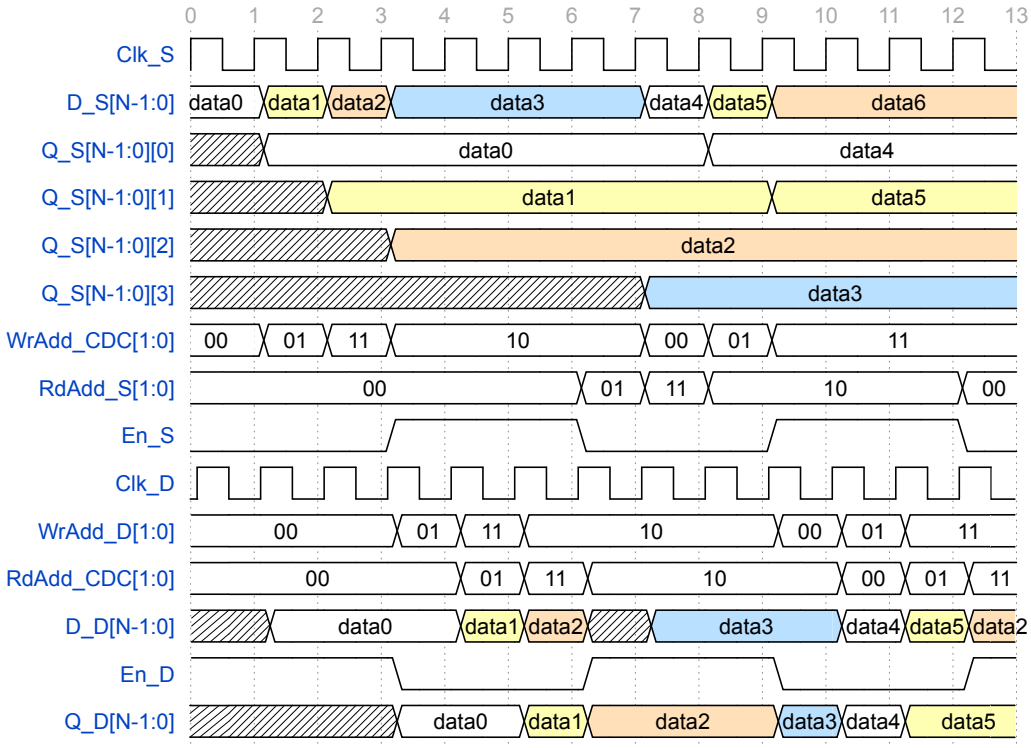


FIGURE 8.10 – Chronogramme associé à la structure "Premier arrivé premier sorti"

Nous allons montrer que les propriétés `f1`, `f2`, `f3`, `f4` et `f5` impliquent les propriétés `m1`, `m2`, `m3` et `m4`. L'exclusivité et l'alternance entre En_S et En_D correspondent au fait qu'il n'est pas possible d'écrire à une adresse donnée x ($WrAdd\_CDC = x$) tant que la FIFO est pleine ($Full = 1$) ou que la donnée stockée à l'adresse x n'a pas encore été lue. Le cas de la FIFO pleine est trivial puisque le signal En_S est

désactivé et que les propriétés `m1` et `m3` sont satisfaites. Nous illustrerons nos propos avec le chronogrammes de la figure 8.10.

Au cycle ♯1, la FIFO n'est pas pleine, une donnée est écrite à l'adresse "00" ($WrAdd\_CDC = 00$ et le signal d'activation $En\_S(00) = 1$). L'adresse d'écriture va s'incrémenter à chaque cycle de l'horloge source tant que la FIFO n'est pas pleine ($Full = 0$). Du fait de la condition F2 (traversée d'une multi-flop), l'adresse d'écriture émise par le domaine source n'est réceptionnée par le domaine de destination que deux cycles d'horloge plus tard (♯3). $WrAdd\_D$ est alors égal à "00" et, puisque les valeurs respectives des adresses d'écriture et de lecture sont égales, la FIFO est vide ($Empty = 1$) : les données écrites au cycle ♯1 ne peuvent pas encore être capturées. Ce n'est qu'à partir du cycle ♯4 que La FIFO n'est plus vide car l'adresse d'écriture côté destination $WrAdd\_D$ a été incrémentée. Ce changement indique qu'une nouvelle donnée a été écrite dans la FIFO mais à une adresse différente. À ce cycle, la donnée $Q\_S[N-1:0][0]$ est lue et le signal $En\_S(00)$ est à l'état bas et le restera jusqu'a ce que $WrAdd\_CDC$ repasse à l'état "00" (equation (8.6)). Il y a donc un délai (dû à la traversée de multi-flop) entre la phase d'écriture ($En\_S(x) = 1$) et la phase de lecture ($En\_D(x) = 1$).

À partir du cycle ♯5, et en accord avec la propriété `f5` l'adresse de lecture ($RdAdd\_CDC$) est incrementée à chaque cycle de l'horloge de destination jusqu'à ce que la FIFO soit vide ($RdAdd\_CDC = WrAdd\_D$). Côté source, l'adresse de lecture $RdAdd\_S$ est encore égale à "00" (du fait du temps de traversée de la multi-flop), et plusieurs données peuvent avoir été écrites générant un état de FIFO pleine (ce qui est le cas à partir du cycle ♯4). Si la FIFO est pleine, cela indique une différence de 1 entre les valeurs respectives des adresses de lecture et d'écriture (en codage binaire). La valeur de l'adresse d'écriture est celle précédant "00″ soit "11″ en binaire et "10″ en Gray. Le signal $En\_S(00)$ est désactivé car la FIFO est pleine et, à partir de la propriété `f4` l'adresse d'écriture doit rester stable. Lorsque l'adresse de lecture $RdAdd\_CDC$ est incrémentée au cycle ♯5, le signal $En\_D(00)$ est désactivé car l'adresse de lecture vaut "01". $En\_S(00)$ reste alors désactivé pendant deux cycles (jusqu'au cycle ♯7) du fait de la condition `F3` (traversée de la multi-flop). Clairement, les signaux indiquant l'état de la FIFO (full et empty) assurent l'exclusivité et l'alternance entre les phase d'envoi ($En\_S(x) = 1$) et de capture des données ($En\_D(x) = 1$).

2.3.3 Bilan

La "Méta-modèle" que nous proposons pour la détection des structures de synchronisation de données peut donc se substituer aux modèles de type "Poignée de main" et "Premier arrivé premier sorti". Les conditions de détection étant moins précises que celles associées aux structures classiques, sa structure est adaptée à la reconnaissance de synchronisations de données spécifiques. Par ailleurs, contrairement au modèle "Contrôle unique", le "Méta-modèle" inclut une partie de la logique de contrôle dans le domaine d'horloge source avec le signal En_S et est immunisé à la détection de faux signaux de contrôle. La figure 8.11 présente le flot proposé pour la validation des protocoles de séquençage. Il est conservatif puisqu'il intègre

les modèles existants tout en considérant que celui de type "Contrôle unique" ne doit pas être validé (excepté si c'est un protocole de séquençage pour une interface quasi-synchrone comme illustré sur la figure 2.11, page 38). En effet, si des structures spécifiques sont implémentées, elles doivent être reconnues à travers le "Méta-modèle" et, si ce n'est pas le cas, c'est que leur fonctionnement ne peut pas être validé de manière automatique (une partie de la logique de contrôle est gérée à l'extérieur du circuit).

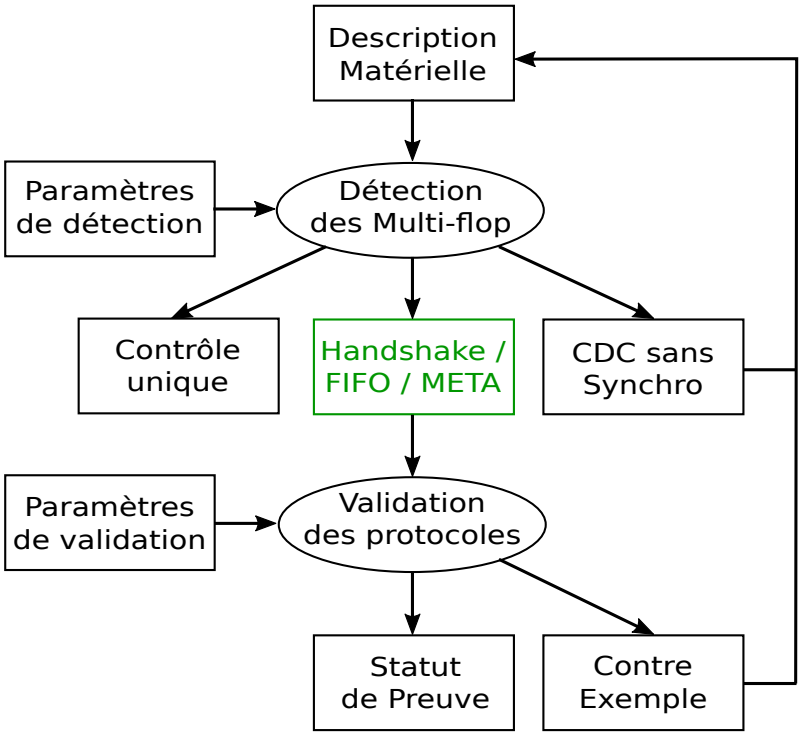


FIGURE 8.11 – Le flot proposé pour la validation des protocoles de séquençage

3 Mise en application

3.1 La mise en place des tests

Le circuit Les travaux associés à l'analyse des structures de synchronisation de données sont effectués sur le sous-système CPUSS à base de processeurs multi-coeurs (section 2, page 8). Le circuit est composé de différentes techniques de synchronisation :

— 21 techniques de type "Premier arrivé premier sorti" ;

— 2 structures "Poignée de main" ;

— 3 synchronisation de type "Contrôle unique".

Comme pour les multi-flops, les structures de synchronisation de données dépendent de la stratégie de réduction de la consommation d'énergie. De plus, les

architectures permettant de concevoir des circuits avec différents niveaux d'alimentation, impactent les performances. Ainsi, des optimisations sont réalisées sur certaines structures de synchronisation et, au sein du circuit, on retrouve quatre types de techniques :

— 3 "Premier arrivé premier sorti" classiques ;

— 18 "Premier arrivé premier sorti" optimisées ;

— 2 "Poignée de main" optimisées ;

— 3 "Contrôle unique" ;

L'approche choisie La procédure proposée pour l'analyse des synchronisations de données se base sur la détection générique de toutes les structures du circuit à travers le "Méta-modèle". Cependant, il n'est pas possible de modifier la bibliothèque de motifs intégrée aux outils, de fait toutes les structures sont identifiées à travers le modèle "Contrôle unique". Dès lors, 133 cas sont identifiés par les outils dont trois ne sont pas vérifiables automatiquement (les "Contrôle unique"). À partir de ces informations, il est possible de rechercher, à l'aide d'un script, s'il existe, dans le domaine d'horloge source, un signal d'activation présent dans les cône d'entrée respectifs des registres de données et des registres de contrôle. Après l'identification de toutes les structures de synchronisation du circuit, la seconde étape de la vérification est la validation des modèles. Pour chaque structure détectée autre que contrôle unique, qui n'est pas validable, le protocole associé au "Méta-modèle" est vérifié.

3.2 Les résultats obtenus

Description des tests Les travaux ont été réalisés avec un outil spécialisé dans la vérification des CDC : l'outil 2. Les résultats sont rapportés dans le tableau 8.1. Les trois premières colonnes donnent respectivement les nom des structures de synchronisation, leur type et leur nombre. Les colonnes 4 et 5 donnent respectivement la logique dans les cônes d'entrée et de sortie. Lorsque le cônes viennent (ou vont) directement d'entrées (ou vers des sorties) primaires, "Ext." est rapporté. Sinon, cela indique qu'il y a des IP (notés SSC ou CPU) autour de l'interface asynchrone. La colonne 6 donne les temps de preuve obtenus avec les propriétés associées au "Méta-modèle". La colonne 7 donne les temps de preuve obtenus avec les propriétés associées au modèle basé sur un signal de contrôle. Comme pour l'analyse des MF, l'outil a été configuré avec un effort illimité afin qu'un résultat conclusif (preuve ou contre-exemple) soit toujours obtenu.

Analyse des résultats En comparant les résultats obtenus, il apparaît que les propriétés associées au "Méta-modèle" sont vérifiées entre 1,07 et 20000 fois plus rapidement que celles associées au modèle partiel et intégrées aux outils de vérification CDC. De plus, pour une interface donnée, une preuve est toujours obtenue avec le même effort, ce qui n'est pas le cas pour celles intégrées dans les outils. Globalement, sur un circuit industriel, le temps de vérification a été substantiellement diminué (divisé par 66). Les résultats montrent que, en utilisant un outil de vérification CDC, les propriétés proposées sont plus faciles à vérifier que celles implémentées dans les outils.

1	2	3	4	5	6	7
Nom de l'interface	Type de synchro	Nombre de canaux	Logique source	Logique dest.	Résultats robuste	Résultats outil 2
AXI_1_B	FIFO	2	SSC	Ext	1 min	5 min
AXI_1_R	FIFO	2	SSC	Ext	1 min	5 min
AXI_1_W	FIFO	2	Ext	SSC	1 min	1.2 min
AXI_1_AW	FIFO	2	Ext	SSC	1 min	1.2 min
AXI_1_AR	FIFO	2	Ext	SSC	1 min	1.2 min
AXI_2_B	FIFO	8	CPU	Ext	1.4 min	1.5 h
AXI_2_R	FIFO	8	CPU	Ext	1.4 min	1.5 h
AXI_2_W	FIFO	8	Ext	CPU	1.4 min	1.5 min
AXI_2_AW	FIFO	8	Ext	CPU	1.4 min	1.5 min
AXI_2_AR	FIFO	8	Ext	CPU	1.4 min	1.5 min
ACE_B	FIFO	8	Ext	CPU	1.4 min	1.5 min
ACE_R	FIFO	8	Ext	CPU	1.4 min	1.5 min
ACE_AC	FIFO	8	Ext	Ext	1.4 min	1.5 min
ACE_W	FIFO	8	CPU	Ext	1.4 min	8 h
ACE_AW	FIFO	8	CPU	Ext	1.4 min	8 h
ACE_AR	FIFO	8	CPU	Ext	1.4 min	8h
ACE_CD	FIFO	8	CPU	Ext	1.4 min	8h
ACE_CR	FIFO	8	CPU	Ext	1.4 min	8h
ATB	FIFO	4	CPU	Ext	1.1 min	30 min
APB_R	HS	1	CPU	Ext	40 s	15 min
APB_W	HS	1	Ext	CPU	40 s	50 s
TS	FIFO	4	Ext	CPU	6 min	20 min
CNT	FIFO	6	Ext	CPU	6 min	20 min
Toutes	Toutes	130	N.A.	N.A.	$< 40min$	$> 44h$

TABLE 8.1 – Résultats de validation des protocoles de séquençage

4 Conclusion

L'analyse des structures de synchronisation de données est la principale étape de la vérification CDC. Cependant, sur les circuits industriels, les outils n'identifient que partiellement les techniques utilisées et les propriétés associées ne sont pas assez précises, générant des résultats non-conclusifs. L'adoption du "Méta-modèle" permettrait l'extraction d'une grande variété de synchronisations et une validation locale du fonctionnement des structures de manière automatique. Concernant les circuits CPUSS, l'approche proposée présente des résultats meilleurs que n'importe quel outil du marché évalué, aussi bien en termes de détection que de validation des structures. Néanmoins le modèle proposé ne se substitue pas au modèle "Contrôle unique", qui correspond à une structure de synchronisation de données partielle ne pouvant être validée automatiquement (la logique de contrôle source étant externe, l'utilisateur doit décrire son fonctionnement à travers des contraintes de validation).

Chapitre 9

L'analyse des signaux non synchronisés

L'analyse des signaux non synchronisés correspond à la dernière étape de la vérification des CDC : la détection des problèmes. Si un cas est détecté, aucune analyse supplémentaire n'est effectuée par les outils : le circuit doit être corrigé. À l'inverse si les outils n'en rapportent pas, cela indique que tous les CDC analysés sont gérés afin d'empêcher la propagation d'états métastables. Cette étape est donc la plus importante car l'objectif final de la vérification est que toutes les interfaces asynchrones fonctionnent correctement.

En considérant que toutes les synchronisations d'un circuit sont identifiées par les outils, la principale difficulté de cette étape est de déterminer si les cas rapportés sont problématiques. En effet, les méthodes de conception des CDC ont pour but d'empêcher la propagation d'états métastables. Or la génération de ces états requiert que le signal transféré change d'état lorsqu'un front d'horloge de destination arrive. Si un CDC n'est jamais sujet à ce scénario, l'implémentation d'une structure n'est pas nécessaire. En pratique, bien qu'une quantité très importante de CDC non synchronisés soit rapportée par les outils, peu de cas correspondent à des problèmes réels.

1 Formalisation des problèmes outils

1.1 Motifs et propriétés

Dans les outils, l'analyse des CDC non synchronisés se divise en deux parties. La première est la détection de tous les signaux non synchronisés et la deuxième effectue le rapport des signaux potentiellement immunisés à la métastabilité [36,97]. Afin d'éviter le signalement de faux problèmes, les outils autorisent trois catégories de signaux ne nécessitant pas de structures de synchronisation :

— Les signaux constants ;

— Les signaux bloqués ;

— Les signaux statiques.

1.1.1 Les signaux constants

Les conditions La particularité de ces CDC est que leur état ne change jamais. Dès lors, aucune transition ne peut survenir à l'entrée de l'élément séquentiel de destination : l'interface est immunisée au phénomène de métastabilité. Les conditions permettant la reconnaissance des signaux constants sont :

CST1 Un signal sujet à CDC (Q_S/D_D) et non synchronisé (ni multi-flop ni synchronisation de données) ;

CST2 Le cône d'entrée du signal (D_S) est constant lorsque le signal de reset source (Rst_S) est désactivé.

La figure 9.1 présente la structure associée au modèle de type "CDC constant" avec un signal Q_S en sortie du registre source dont l'état est fixe (0 ou 1).

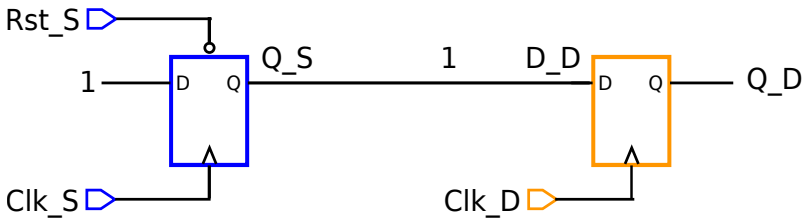


FIGURE 9.1 – La structure associée au modèle "CDC constant"

La propriété La valeur constante sur le cône d'entrée du signal peut être définie soit par le concepteur, via la description matérielle du circuit, soit par l'ingénieur en charge de la vérification CDC, à travers les contraintes de détection. Lorsque la constante est définie directement par le concepteur, aucun problème CDC ne peut survenir. Cependant, lorsque l'information est définie à travers des contraintes, elle doit être validée. La propriété formelle associée spécifie que le CDC est toujours à l'état défini lorsque le signal de reset source est désactivé.

```
property Constante(Clk, Sig, Constant);
(@(posedge Clk) disable iff(Reset)
Sig == Constant;
endproperty

cst1:assert property(Constante(Clk_S, Q_S, 1'b0));
cst2:assert property(Constante(Clk_S, Q_S, 1'b1));
```

1.1.2 Les signaux bloqués

Les conditions Ces CDC sont particuliers car ils peuvent générer des états métastables. Néanmoins, la métastabilité n'a pas d'impact sur le circuit car le cône de sortie du CDC empêche sa propagation. Les conditions permettant la reconnaissance des signaux bloqués sont :

B1 Un signal sujet à CDC (Q_S/D_D) et non synchronisé (ni multi-flop ni synchronisation de données) ;

B2 Le cône de sortie du signal (Q_D) est non connecté ou constant lorsque le signal de reset destination (Rst_D) est désactivé.

La figure 9.2 présente la structure associée au modèle de type "CDC bloqué" avec un signal Q_D, en sortie du registre de destination, dont le cône de sortie est à valeur constante (0).

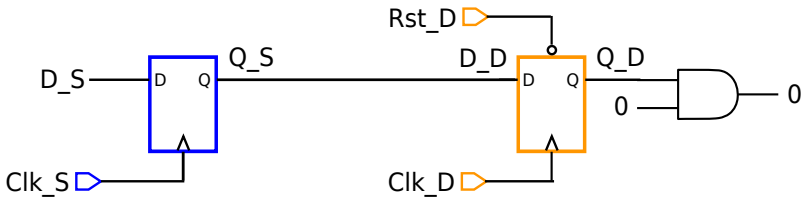


FIGURE 9.2 – La structure associée au modèle "CDC bloqué"

La propriété Comme pour les signaux constants, le blocage peut être effectué soit par le concepteur soit par l'ingénieur en charge de la vérification CDC et, lorsque l'information est définie à travers des contraintes, elle doit être vérifiée. La propriété associée à cette structure assure que le cône de sortie du CDC est constant lorsque le reset de destination est désactivé.

```
stop1:assert property(Constante(Clk_D, C_Out, 1'b0));
stop2:assert property(Constante(Clk_D, C_Out, 1'b1));
```

1.1.3 Les signaux statiques

Les conditions Ces CDC peuvent changer d'état sans générer d'états métastables. Autrement dit, leur transition s'effectue lorsque l'horloge de destination est désactivée. Dès lors, il existe un signal de contrôle permettant d'assurer une exclusivité entre l'envoi de la donnée et sa capture par le registre de destination. Cette technique peut être faite à travers un MUX à recirculation sur la donnée ou un signal de contrôle de l'horloge (sous-section 1.2.1, page 97). En d'autres termes, ces signaux sont gérés par un protocole de séquençage. Cependant, si les signaux sont détectés comme non synchronisés, c'est qu'aucune structure de type "Multi-flop" n'est implémentée. S'il existe un signal de contrôle au sein du circuit, il est possible d'identifier les signaux potentiellement statiques à travers les conditions suivantes :

S1 Un signal sujet à CDC (Q_S/D_D) et non synchronisé (ni multi-flop ni synchronisation de données) ;

S2 Un signal d'activation de la capture (En_D) échantillonné dans le domaine de destination.

La propriété La vérification de la structure "CDC statique" est similaire à celle de la synchronisation de type "Contrôle unique". Seule la propriété *Stabilite_et_coherence* (sous-section 1.1.1, page 111) est vérifiée, paramétrée avec En_D. Elle exprime la stabilité et la cohérence des données lors de leur capture.

```
c1:assert property(Stabilite_et_coherence(En_D,D_D));
c2:assert property(Maintien_donnee(Clk_S, En_CDC))
```

La figure 9.3 présente la structure associée au modèle de type "CDC statique" avec un signal de contrôle En_D, en entrée du registre de destination échantillonné dans le domaine de destination mais non synchronisé par multi-flop.

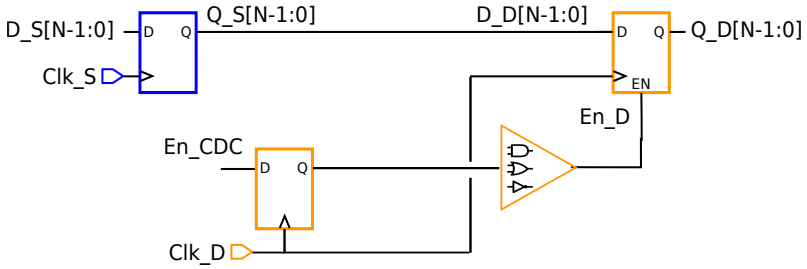


FIGURE 9.3 – La structure associée au modèle "CDC statique"

La configuration Lorsqu'aucun signal de contrôle n'est détecté, certains outils proposent d'identifier les CDC avec une source et plusieurs destinations sans structure de synchronisation [81]. Dès lors, la reconnaissance de ces structures se fait à travers une unique condition :

— Un signal sujet à CDC (Q_S), non synchronisé (ni multi-flop ni synchronisation de données), et divergent vers plusieurs registres de destination ($D\_D[i]$) ;

La figure 9.4 présente la structure associée au modèle de type "CDC potentiellement statique" avec un signal Q_S, en sortie du registre source, transféré vers trois registres de destination $D\_D[2:0]$ en parallèle. Étant donné qu'aucun signal de contrôle n'est identifié, la propriété à vérifier est "Stabilite" (sous-section 2.2, page 102). Elle définit simplement que la donnée transférée est stable lorsque l'horloge de destination est active.

```
qs1: assert property(Stabilite(Clk_D, Q_S));
```

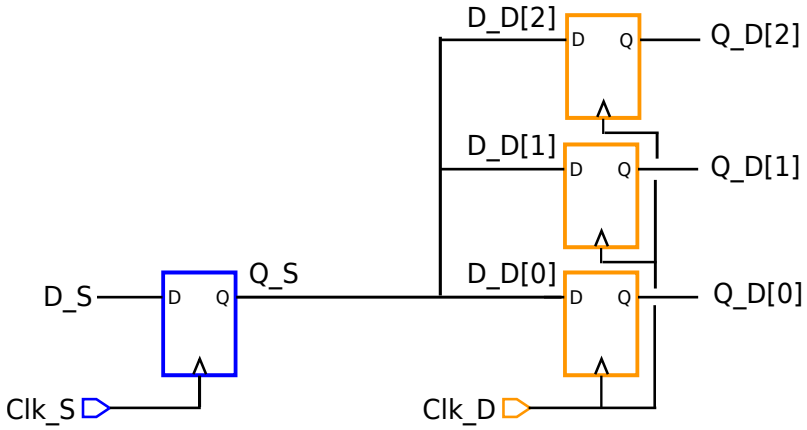


FIGURE 9.4 – La structure associée au modèle "CDC potentiellement statique"

Les outils sont donc en mesure d'identifier les différents CDC, sans structure, mais potentiellement immunisés à la métastabilité. La figure 9.5 présente le flot d'analyse des structures de type "CDC statique". La détection des signaux potentiellements immunisés à la métastabilité se fait en trois étapes : identification des CDC sans structures, extraction des cas avec un signal de contrôle et rapport des cas sans signal de contrôle mais avec des destinations multiples.

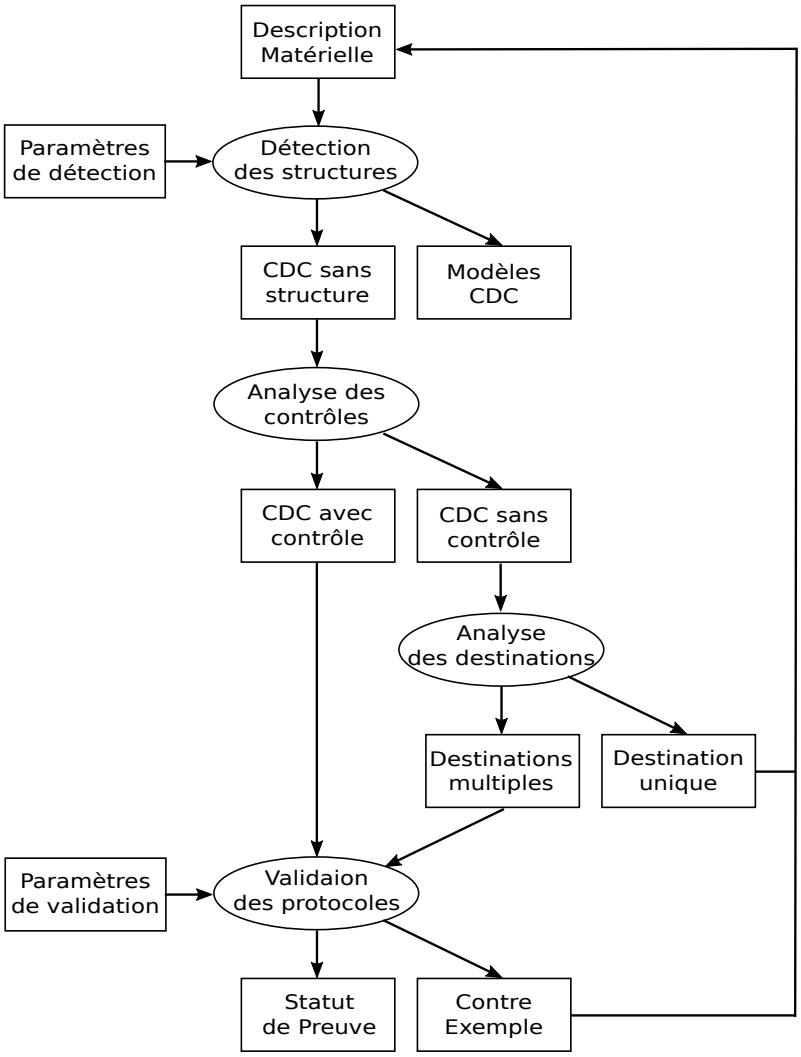


FIGURE 9.5 – Le flot d'analyse des signaux statiques

1.2 Problèmes industriels

Les cas manquants Lors de la conception de circuits industriels, le contrôle des signaux non synchronisés peut s'opérer autrement qu'à travers le chemin des données ou celui de l'horloge. Pour des raisons de consommation d'énergie, la robustesse à la métastabilité peut se faire à travers le signal de reset de destination. Les modèles implémentés dans les outils ne considérant pas l'arbre de reset de destination, certaines structures ne sont pas détectées. L'utilisateur doit alors spécifier manuellement les CDC comme statiques. La figure 9.6 présente une structure de type "CDC

statique" non détectée du fait que le signal de contrôle est sur le chemin du signal de reset de destination Rst_D.

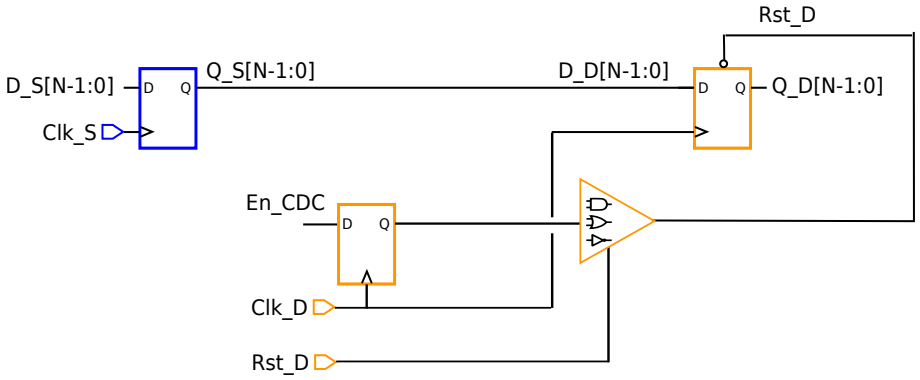


FIGURE 9.6 – Structure de type "CDC statique" non couverte

Les faux cas Afin de permettre la validation des signaux immunisés aux problèmes CDC sans concevoir des modèles potentiellement faux, la plupart des outils de vérification proposent de définir des contraintes de détection. Comme décrit dans le chapitre 3 (sous-section 3.1.1, page 52, ces informations sont des directives spécifiant, s'ils sont présents, les signaux de contrôle (contraintes faibles), sinon les CDC statiques (contraintes moyennes). Cependant, le modèle associé à la reconnaissance des signaux statiques étant générique, de faux cas peuvent être détectés. Prenons l'exemple d'un signal de contrôle provenant du domaine de destination (En_D) mais dont le cône d'entrée est sujet à CDC (En_CDC). Si l'utilisateur déclare le signal de contrôle non synchronisé comme statique, l'interface pourra être détectée comme contrôlée par ce signal. Or, si le signal de contrôle est métastable, la sortie du registre de donnée (Q_D) l'est aussi. À l'inverse, si le signal de contrôle et l'interface sont statiques, il existe nécessairement un autre signal assurant la stabilité de ces deux CDC lors de leur capture par le registre de destination. Dès lors, la généricité du modèle autorise la détection de fausses techniques de synchronisation. L'utilisateur doit alors spécifier à l'outil le signal de contrôle commun aux signaux statiques cascadés ou, s'il n'y en pas spécifier les deux CDC non synchronisés comme statiques. La figure 9.7 présente le problème associé au modèle de type "CDC statique" avec un CDC sans structure Q_S dont le registre de destination possède un signal de contrôle En_D lui-même sujet à CDC (En_CDC) et sans structure.

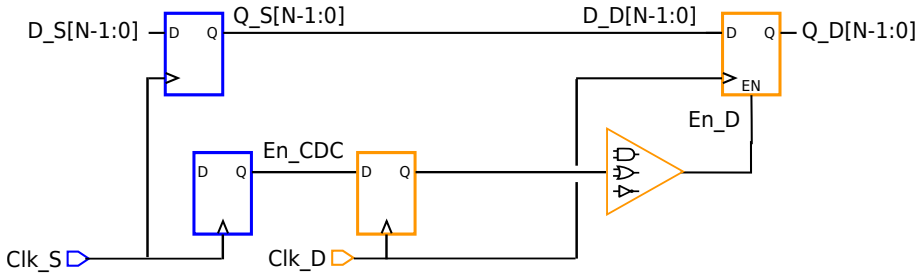


FIGURE 9.7 – Le problème associé au modèle "CDC statique"

La validation inutile En considérant que tous les CDC ne nécessitant pas de synchronisation sont bien détectés, un problème subsiste : les structures de synchronisation sont externes au circuit. Dès lors, les propriétés ne peuvent être vérifiées et l'engin de preuve rapportera pour chacune un contre-exemple. De plus, la détection des signaux d'activation de la capture des données est locale. Une propriété est générée pour chaque CDC non synchronisé et comportant un signal d'activation dans le domaine de destination. La vérification prend donc du temps, sans compter que le problème d'explosion d'état associé au modèle "Contrôle unique" s'applique aussi à cette structure. La vérification formelle des propriétés associés aux signaux sans synchronisation est donc à la fois inutile et très coûteuse en temps. La figure 9.8 présente les problèmes associés au flot d'analyse des structures de type "CDC statique" en rouge.

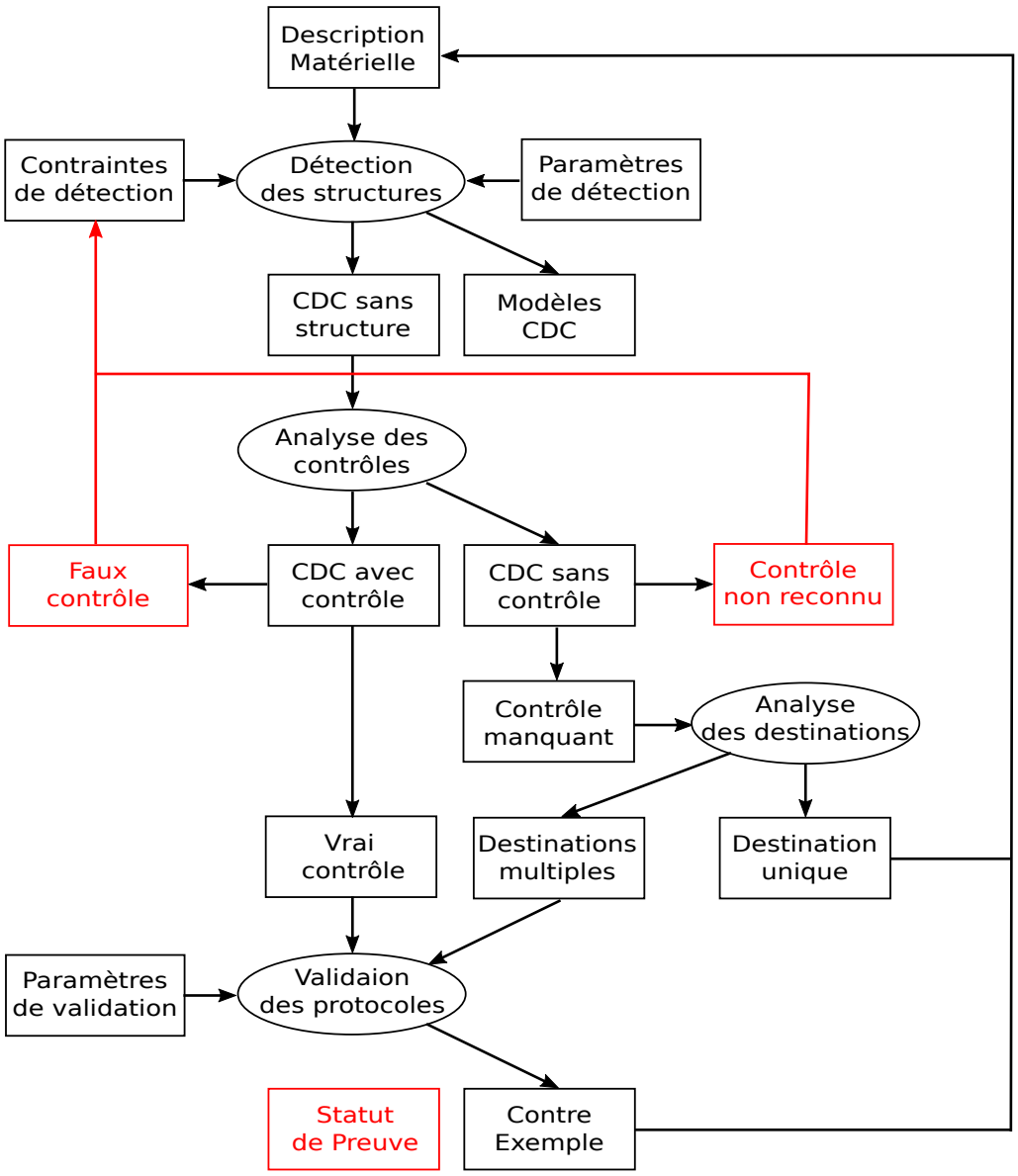


FIGURE 9.8 – Le flot de détection des signaux statiques et les problèmes associés

Bilan Les modèles associés aux signaux statiques sont donc incomplets, trop génériques et vérifiés inutilement. Cela motive l'élaboration d'un nouveau modèle, appelé "Synchronisation externe", et remplaçant les trois modèles associés aux signaux non synchronisés et robustes à la métastabilité. Ce modèle ne pouvant être vérifié, une approche de tri des cas rapportés est proposée.

2 La "Synchronisation externe"

2.1 Definition du modèle'

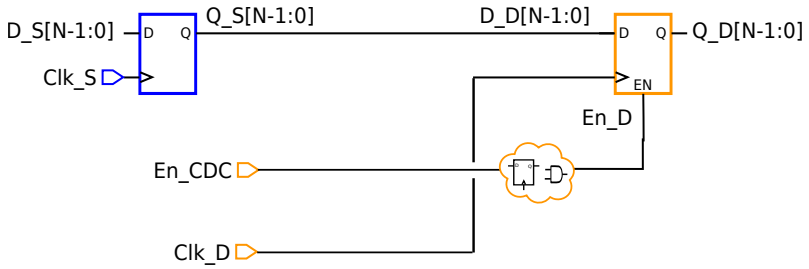
Nous proposons un modèle de "Synchronisation externe" permettant d'assurer une détection automatique des CDC immunisés au problème de métastabilité. Ce modèle permet d'obtenir une quantité maximale de problèmes résolus et de générer un jeu de propriétés formelles pouvant être vérifié en simulation.

La structure La structure associée à la "Synchronisation externe" est très proche de celle du type "CDC statique". Elle est composée d'un signal de contrôle unique sur le registre de données destination, En_D, échantillonné dans le domaine destination et dont le cône d'entrée jusqu'aux ports primaires l'est aussi. Il n'est donc ni sujet à CDC ni synchronisé par multi-flop : ce n'est pas une structure de synchronisation de données. Dans le cas de signaux de données statiques, En_D est connecté au registre de données de destination et, dans le cas de signaux de données bloquées, En_D est connecté au reset de destination ou à une porte logique dans le cône de sortie de la destination (Q_D). La première condition de détection du modèle est S1 et la seconde est :

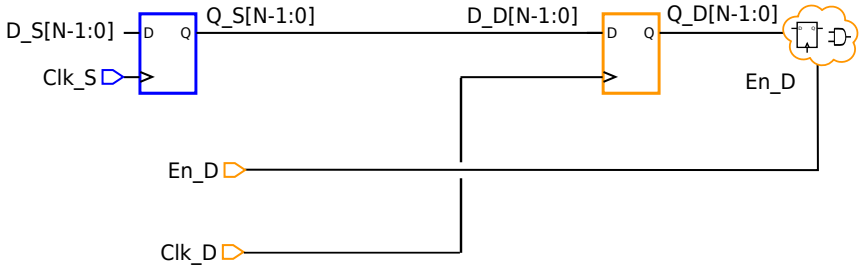
PS1 Un signal unique (En_D) d'activation de la capture ou de blocage des données en sortie dont le cône d'entrée jusqu'aux ports primaires est déclaré dans le domaine de destination.

La figure 9.9 présente les structures associées au modèle "Synchronisation externe" avec un signal Q_S sujet à CDC mais sans structure de synchronisation (ni multi-flop ni synchronisation de données). La figure 9.9a présente le cas d'un signal statique avec un signal contrôle En_D allant directement à l'entrée du registre de destination. La figure 9.9b présente un exemple de données bloquée avec un signal contrôle En_D allant dans le cône de sortie du registre de destination. Quelle que soit la structure associée au modèle "Synchronisation externe", le cône d'entrée du signal de contrôle jusqu'au port primaire En_CDC est échantillonné uniquement dans le domaine de destination Clk_D.

Le protocole Le signal d'activation de la destination En_D assure que les données transférées ne peuvent pas générer d'état métastable. Ce signal ne peut avoir dans son cône d'entrée jusqu'aux ports primaires que des signaux échantillonnés dans le domaine de destination. Si ce n'est pas le cas, son cône d'entrée est sujet à CDC et donc, soit il est synchronisé et une structure de type "Meta-modèle" ou "Contrôle unique" (sous-section 2.1, page 118) est identifiée, soit il n'est pas synchronisé et il peut être métastable : le circuit doit être corrigé. Le bon fonctionnement de ce



(a) Structure de type signal statique



(b) Structure de type signal bloqué

FIGURE 9.9 – La structure associée au modèle "Synchronisation externe"

modèle est basé sur la présence, à l'extérieur du circuit, d'une logique de contrôle et de structures multi-flops adéquates. La stabilité, la cohérence et le maintien des données ne peuvent donc pas être vérifiés formellement sur ce type de structure puisque la logique de synchronisation est externe. La propriété associée est la même que pour la synchronisation "Contrôle unique" mais sa validation ne peut être réalisée qu'en simulation ou par l'équipe en charge de l'intégration du circuit (au niveau SoC par exemple).

2.2 Application de la structure

Le modèle "Synchronisation externe" a pour but de détecter toutes les interfaces immunisées à la métastabilité sans distinction entre signaux mis à valeur constante, signaux bloqués et signaux statiques. Notons que pour les signaux constants et bloqués, si l'état est déclaré dans la description du circuit, il n'y a pas besoin de rechercher une structure ou de vérifier une propriété. Nous nous intéresserons uniquement aux signaux définis comme constants par la personne en charge de la vérification (à travers des contraintes). La structure étant incomplète, la vérification formelle n'est pas réalisable au niveau du circuit. Il faut donc vérifier que les conditions de détection de ce modèle prennent en compte tous les cas de figure.

2.2.1 Application au modèle "CDC constant"

En reprenant le modèle associé au cas des signaux constants, le registre de destination n'est pas analysé. Dès lors, le signal d'activation En_D ne peut pas être instanciés. Il existe deux cas pour cette contrainte :

— La contrainte est définie sur un port primaire ;

— La contrainte est définie sur un signal interne.

La définition de contraintes sur des ports primaires revient à une déclaration par le concepteur car les informations externes doivent être spécifiées dans des documentations techniques. À l'inverse, la définition d'un état constant sur un signal interne est réalisé lorsqu'il existe un protocole assurant qu'un signal de contrôle dans le domaine d'horloge source rend les données transférées constantes. Bien que le modèle "Synchronisation externe" ne prenne pas en compte le cône d'entrée du CDC, la propriété "Stabilite_et_coherence" des protocoles de séquençage indique que l'immunité à la métastabilité se fait nécessairement au niveau du registre de destination. Dès lors, s'il n'y a pas de signal de contrôle côté destination, un état métastable peut survenir lorsque le CDC passe à l'état constant. Le modèle "Synchronisation externe" est donc applicable au cas des signaux constants.

2.2.2 Application aux signaux bloqués et statiques

Au regard du modèles correspondant aux signaux statiques tel que définis dans les outils, le signal En_D du modèle "Synchronisation externe" ne couvre pas autant de cas. D'après la condition $S2$, le signal de contrôle peut être sujet à CDC à partir du moment où il est lui-même défini comme statique par l'utilisateur. Néanmoins, si le signal de contrôle assurant la capture des données est statique, il est lui-même contrôlé par un signal dont le cône d'entrée jusqu'aux ports primaire est dans le domaine de destination : c'est le cas de signaux statiques cascadés. Le modèle "Synchronisation externe" est donc applicable au modèle "CDC statique".

Pour le cas des signaux bloqués, le signal défini comme constant par l'utilisateur est le signal de contrôle En_D. Le modèle "Synchronisation externe" peut donc être directement instancié. Notons que pour le blocage des données, le modèle "Synchronisation externe" prend en compte l'arbre de reset. Lors d'un blocage par l'activation d'un reset asynchrone, il n'y a pas de considération d'horloge ou de CDC : le signal de contrôle n'a pas besoin d'être synchronisé. Cependant, lorsque le signal de reset est désactivé, et donc que le signal de contrôle est inactif, la synchronisation du signal de reset et du contrôle est nécessaire pour éviter de générer un état métastable en sortie. Dès lors la structure peut être reconnue à travers un "Contrôle unique". Si les structures sont absentes, le signal de contrôle doit avoir un cône d'entrée jusqu'aux ports primaires déclaré dans le domaine de destination. Notons que les structures de synchronisation des signaux de reset sont spécifiques [24], elles peuvent donc être distinguées des structures MF, de manière à éviter la détection de tous les reset synchronisés comme des signaux de contrôle de structures type "Contrôle unique".

2.2.3 Bilan

Le modèle proposé permet donc de couvrir tous les cas de CDC potentiellement immunisés à la métastabilité à partir du moment où un signal de contrôle est présent. S'il n'y en a pas, il n'est pas possible d'émettre une hypothèse quant au comportement des signaux (contraintes moyennes) et, les conditions relatives au nombre de destinations, telles que proposées par les outils de vérification est la solution la plus appropriée. De plus, quelle que soit la structure identifiée, elles ne sont que partielles, il est donc indispensable de valider le comportement des signaux, si possible lors de l'intégration du circuit, et au pire en simulation.

La figure 9.10 présente le flot proposé pour la détection des CDC gérés à l'extérieur. Il intègre les modèles proposés à la place de celui de type "CDC statique" afin d'éviter la détection de fausses structures.

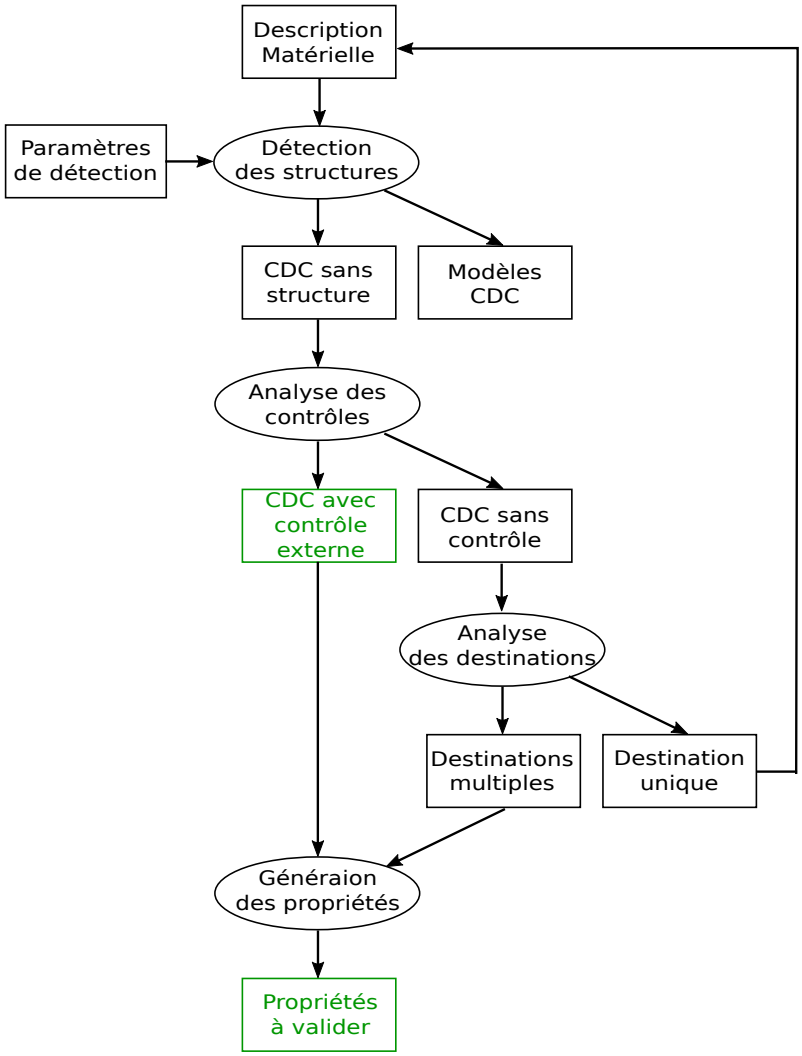


FIGURE 9.10 – le flot proposé pour la détection des CDC gérés à l'extérieur

3 Mise en application

3.1 La mise en place des tests

Le circuit Les résultats expérimentaux sont effectués sur le circuit CPUSS section 2, page 8). Certaines communications sont effectuées à travers les protocoles d'utilisation du circuit. Ses caractéristiques en termes de signaux immunisés à la métastabilité sont les suivantes :

— 9339 CDC avec des contrôles externes ;

— 38868 CDC sans contrôle externe ;

— 5032 CDC avec des structures de blocage.

L'approche choisie Le flot proposé pour l'analyse des signaux non synchronisés se base sur l'identification des structures à travers le modèle "Synchronisation externe". Cependant, il n'est pas possible de modifier la bibliothèque de motifs intégrée aux outils. Toutes les structures sont identifiées comme non synchronisées et il n'est pas possible de différencier les cas. Néanmoins, les outils sont en mesure de rapporter les signaux de contrôle de destination pour toutes les interfaces. À partir de ces informations, un script vient analyser le cône d'entrée des contrôles jusqu'aux ports primaires afin de déterminer ceux définis dans le domaine de destination. Après l'identification des modèles, la seconde étape consiste à générer des propriétés devant être vérifiées en simulation.

3.2 Les résultats obtenus

Description des tests Les tests ont été effectués avec deux outils spécialisés dans la vérification statique des CDC : l'outil 1 et l'outil 2. Les résultats sont rapportés dans le tableau 9.1. Les deux premières colonnes donnent respectivement les noms des interfaces immunisées aux problèmes CDC et le nombre de CDC. La colonne 3 indique le type de technique utilisée pour gérer les signaux non synchronisés. Les colonnes 4 et 5 donnent le nombre de signaux et le type de modèles détectés respectivement avec l'outil 1 et l'outil 2. La dernière colonne rapporte le nombre de signaux et le type de modèles détectés avec le script décrivant le modèle "Synchronisation externe". Lorsque les outils n'ont pas de modèle permettant de détecter les structures implémentées, "N.A." (Non applicable) est rapporté. À l'inverse si les outils ou l'approche prennent en compte les modèles et détectent l'intégralité des structures, "OK(x)" est indiqué avec x le type de structure identifiée. Si les modèles existent mais que seule une partie des structures est rapportée, "NOK(x)" est rapporté (avec x le nombre de structures identifiées).

Analyse des résultats Au regard des essais réalisés, les modèles associées aux "Synchronisation externe" a permis de détecter toutes les structures existantes. Cependant pour les CDC n'ayant pas de signal de contrôle respectant la condition PS1, la couverture s'est limitée aux cas qui vont vers plusieurs registres de destination. Afin de couvrir tous les cas, nous avons essayé de généraliser la spécification de signaux statiques entre deux domaines d'horloge à partir du moment où un signal allant vers plusieurs destinations est identifié. Cependant les résultats ne sont pas concluants : des signaux synchronisés sont identifiés comme statiques.

Concernant les résultats de l'outil 2, bien que toutes les interfaces soient couvertes, les modèles reconnus correspondent à des signaux avec contrôle bien que certains soient faux : la qualité de résultats n'est que de 27%. De plus, les outils identifiant ce type de modèle cherchent à vérifier des propriétés qui ne sont pas prouvables formellement. À titre d'exemple, pour ce circuit CPUSS, l'outil 2 a généré 8749 propriétés et il lui a fallu plus de 10 jours d'analyse pour rapporter un contre-exemple pour chacune.

Concernant les résultats de l'outil 1, la couverture des signaux est faible : 65%. L'outil s'est contenté de rapporter les CDC avec plusieurs registres de destination (modèle "CDC potentiellement statique"). Par ailleurs, cet outil ne différencie pas les signaux synchronisés des signaux sans structure. Les cas rapportés dans la table ne

1	2	3	4	5	6
Nom de l'interface	Nombre de CDC	Type de structure	Détection outil 1	Détection outil 2	Détection robuste
BIST_C	14347	Statique	NOK (646)	OK (Contrôle)	NOK (646)
BIST_O	8624	Contrôle	OK (Statique)	OK (Contrôle)	OK (Contrôle)
JESS_C	1786	Statique	OK (Statique)	OK (Contrôle)	OK (Statique)
JESS_O	1792	Bloqueur	N.A.	OK (Contrôle)	OK (Bloqueur)
PMB_C	3240	Bloqueur	OK (Statique)	OK (Contrôle)	OK (Bloqueur)
PMB_O	715	Contrôle	N.A.	OK (Contrôle)	OK (Contrôle)
VCPU_C	15744	Statique	NOK (14502)	OK (Contrôle)	NOK (14502)
VSOC_C	2929	Statique	NOK (2102)	OK (Contrôle)	NOK (2102)
VAPE_C	4062	Statique	NOK (3969)	OK (Contrôle)	NOK (3969)
Total	53239	N.A.	34869 (65%)	53239 (100%)	37376 (70%)

TABLE 9.1 – Résultats de détection des signaux immunisés aux problèmes CDC

correspondent qu'à 6% de la totalité des CDC vus comme potentiellement statiques par l'outil.

Pour finir, les outils de vérification statiques détectent uniquement les signaux d'activation dans le domaine de destination et ne tiennent pas compte des structures de blocages. Dès lors, des structures problématiques sont vues comme potentiellement synchronisées et de vraies structures sont mal identifiées.

4 Conclusion

L'analyse des CDC non synchronisés est l'étape la plus complexe de la vérification. Cependant, sur les circuits industriels, les outils n'identifient que partiellement les interfaces potentiellement immunisées à la métastabilité et cherchent inutilement à vérifier leur fonctionnement. L'implémentation du modèle "Synchronisation externe" permettrait l'extraction d'une grande variété de cas et la génération d'un jeu de propriété permettant de les valider en simulation. Dans le contexte des circuits type CPUSS, l'approche proposée présente des résultats encourageants. Néanmoins, la recherche de structure de blocage peut générer une augmentation du temps de détection des structures très importante car les cônes de sortie des CDC peuvent être à la fois importants et complexes. Par ailleurs, la détection du modèle proposé

ne doit pas être considéré comme une technique de conception correcte mais comme une information permettant de trier les CDC non synchronisés.

Conclusion

Après trois années de collaboration avec la division CPU-GPU de STMicroelectronics et l'équipe de recherche AMfoRS du laboratoire TIMA, le bilan des différentes missions qui m'ont été confiées est positif. Nous avons montré, à travers deux contributions majeures, que les techniques actuelles de vérification statique des CDC ne répondent pas aux attentes des industriels de la conception et qu'il est possible d'améliorer la qualité des résultats obtenus à moindre coût.

Les principaux outils du marché ont été évalués sous trois aspects : l'approche de vérification, les modèles de détection et les propriétés formelles utilisées pour vérifier le comportement des interfaces. Nous avons montré que les solutions logicielles existantes ne sont pas efficaces sur des circuits intégrant des techniques de synchronisation spécifiques. Les structures sont, au mieux, partiellement reconnues et les propriétés vérifiées ne sont pas assez précises, générant des statuts de preuve non-conclusifs ou, au pire, faux. Néanmoins, avec un effort en termes de configuration et de spécification de contraintes, deux outils permettent d'obtenir des résultats avec un niveau de qualité optimal.

Nous avons donc analysé et formalisé les différents modèles (avec les propriétés associées) et les approches implémentés dans les outils pour mettre en évidence les raisons des limitations pratiques. Ce travail représente un apport essentiel pour la communauté de la vérification des CDC puisqu'il n'existe aucune autre publication de ce type à ce jour. Par ailleurs, il a permis de sensibiliser les personnes en charge de la validation des CDC chez STMicroelectronics à la nécessité de configurer les modèles et de spécifier des contraintes pour obtenir des résultats d'analyse cohérents.

À partir de ces informations, nous avons proposé un jeu de nouveaux modèles permettant l'extraction d'une grande variété de structures avec leurs propriétés. L'adoption de nos modèles a permis une vérification la plus complète possible de manière automatique ce qui est essentiel dans un environnement industriel compétitif. En effet, le nombre de paramètres à configurer et la quantité de contraintes à spécifier pour obtenir des résultats avec un niveau de qualité équivalent ont été réduits de moitié pour l'outil 1 (55 paramètres et 116 contraintes) et presque divisés par trois pour l'outil 2 (40 paramètres et 117 contraintes). De plus, le temps dédié à la validation des structures a été divisé par neuf en passant de plus de 50h à moins de 6h.

Cependant, tous les essais ont été réalisés sur un unique circuit de type sous-système à base de processeurs (CPUSS). Afin d'assurer qu'aucun cas n'a été omis et

que les solutions proposées ne sont pas dédiées à la vérification d'architectures CDC marginales, les travaux doivent être reproduits sur d'autres circuits industriels. Nous avons donc mis en place une collaboration avec d'autres divisions de STMicroelectronics. Les travaux en cours consistent à configurer les outils en fonction circuits et à spécifier des contraintes. L'idée étant de déterminer si les modèles proposés sont nécessaires pour les circuits à vérifier. Bien que les résultats soient encourageants, le temps dédié à l'analyse des cas rapportés est tributaire des délais imposés par le marché. Par manque de temps, le test des modèles n'a pas encore été mis en application.

Par ailleurs, les outils ne permettent pas la définition manuelle de nouveaux modèles par l'utilisateur. En outre, un seul outil CDC du marché permet à la fois la détection de modèles configurables et la spécification de propriétés formelles. Les résultats obtenus se basent donc sur l'utilisation des deux outils du marché développés par des fournisseurs EDA différents. L'approche a consisté à utiliser des fonctionalités dédiées à l'obtention d'informations spécifiques et à développer des scripts permettant d'exploiter ces informations pour détecter des structures correspondant aux modèles. Cependant, l'approche est dépendante des outils, du circuit et n'est donc pas facilement réutilisable. De plus, la compatibilité entre les outils n'est pas automatique et la mise en application des travaux requiert une expertise dans l'utilisation des deux solutions. Par ailleurs, un seul des deux outils est actuellement utilisé, sur projets, par STMicroelectronics. La solution serait donc d'avoir un outil qui intègre les avantages de l'autre ou que les modèles proposés soient intégrables.

Pour finir, les interfaces non synchronisées et ne nécessitant pas l'implémentation de structures dédiées ont présenté des limitations. Il n'est pas possible, avec un outil de vérification statique d'assurer que les structures identifiées sont correctes. Il est donc nécessaire de valider le comportement en simulation ou formellement chez le client qui intègre le circuit. Nous avons donc préféré sensibiliser les ingénieurs en charge de la conception des circuits afin d'obtenir une documentation détaillée des différentes interfaces gérées à l'extérieur du circuit. En effet, sur projet, le manque d'informations relatives aux interfaces asynchrones est un problème courant. Les ingénieurs en charge de la conception des circuits doivent donc être sensibilisés aux problématiques CDC. Nous avons donc défini des fichiers de spécification sous forme tabulaire afin que les personnes responsables des interfaces asynchrones fournissent les informations relatives au comportement des signaux considérés comme immunisés aux problèmes CDC.

Comme expliqué dans ce manuscrit les travaux actuels en termes d'évolution des outils de vérification CDC se concentre sur l'analyse dynamique et non sur le développement de nouveaux modèles CDC. De plus, les solutions proposées n'ont pas encore été adoptées par les fournisseurs d'outils EDA. L'analyse des communications entre domaines d'horloge requiert donc, pour l'instant, un effort humain conséquent, quelle que soit l'approche choisie.

Publications et conférences

Chapitre de livre :

[1] Kebaili M., Morin-Allory K., Brignonne J. C., Borrione D., Enabler-based synchronizer model for clock domain crossing static verification, in Languages, Design Methods, and Tools for Electronic System Design : Selected Contributions from FDL 2015, pages 103-124, Springer International Publishing, 2016.

Actes de Rencontres Internationales (avec comité de sélection) :

[2] Kebaili M., Morin-Allory K., Brignone J. C., Borrione D., Enabler-Based Synchronizer Model for Clock Domain Crossing static Verification, Forum on specification & Design Languages (FDL'15), Barcelona, Spain, September 2015.

[3] Kebaili M., Plassan G., Brignone J. C., Binois J. P., Conclusive formal verification of clock domain crossings using SpyGlass-CDC, Synopsys User Group (SNUG 2016), Best Paper Award, Grenoble, France, June 2016.

[4] Kebaili M., Brignonne J. C., Morin-Allory K., Clock Domain Crossing Formal Verification : A Meta Model, 18th IEEE International High-Level Design Validation and Test Workshop, Santa Cruz, California, October 2016.

[5] Breme A., Brignone J. C., Kebaili M., Detection & management of unwanted logic in a multi-flop synchronizer at gate level using Spyglass-CDC, Synopsys User Group (SNUG 2017), Grenoble, France, June 2017.

Rencontres Internationales sans acte :

[6] Kebaili M., Brignone J. C., Sarwary S., Mal Jain P., Efficient clock domain crossing (cdc) verification methodology for MP-Core asynchronous interfaces, in IEEE Design Automation Conference (52nd DAC), Designer and IP Track Poster session, San Francisco, California, June 2015.

[7] Kebaili M., Brignone J. C., Clock domain crossing verification with Questa-CDC for ST MP-Core subsystems, Mentor Graphics Forum, Grenoble, France, October 2015.

[8] Kebaili M., Clock domain crossings (cdc) checks flow development dedicated to ST MP-Core Subsystems, FETCH 2016, Villard-de-Lans, France, January 2016.

[9] Kebaili M., Generic Synchronizer Model for Clock Domain Crossing Static Verification, 11ème GDR SoC-SiP, Nantes, France, June 2016

[10] Kebaili M., Conclusive formal verification of clock domain crossings using SpyGlass-CDC, Synopsys User Group (SNUG 2016), London, UK, June 2016.

Bibliographie

[1] M. Alshaikh, D. J. Kinniment, and A. Yakovlev. A synchronizer design based on wagging. In *2010 International Conference on Microelectronics*, pages 415–418, Dec. 2010.

[2] I. Beer, S. Ben-David, C. Eisner, and A. Landver. Rulebase : An industry-oriented formal verification tool. In *Proceedings of the 33rd Annual Design Automation Conference*, DAC '96, pages 655–660, 1996. `http://doi.acm.org/10.1145/240518.240642`.

[3] S. Beer, J. Cox, T. Chaney, and D. M. Zar. Mtbf bounds for multistage synchronizers. In *2013 IEEE 19th International Symposium on Asynchronous Circuits and Systems*, pages 158–165, May 2013.

[4] S. Beer and R. Ginosar. An extended metastability simulation method ; extended node short simulation (enss). In *2012 IEEE 27th Convention of Electrical and Electronics Engineers in Israel*, pages 1–4, Nov. 2012.

[5] S. Beer and R. Ginosar. Eleven ways to boost your synchronizer. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 23(6) :1040–1049, June 2015.

[6] S. Beer, R. Ginosar, R. Dobkin, and Y. Weizman. Mtbf estimation in coherent clock domains. In *2013 IEEE 19th International Symposium on Asynchronous Circuits and Systems*, pages 166–173, May 2013.

[7] S. Beer, R. Ginosar, M. Priel, et al. The devolution of synchronizers. In *2010 IEEE Symposium on Asynchronous Circuits and Systems*, pages 94–103, May 2010.

[8] F. Behrens and W. Encinas. *Clock Domain Crossing Check Based on Assertions-A Case Study in IP Design.* Freescale Semiconductor, 2011. `http://www.lbd.dcc.ufmg.br/colecoes/wcas/2011/0017.pdf`.

[9] A. Breme, J. C. Brignone, and M. Kebaili. Detection & management of unwanted logic in a multi-flop synchronizer at gate level using spyglass-cdc. *Synopsys User Group (SNUG) France*, June 2017.

[10] G. M. Brown and L. Pike. Easy parameterized verificaton of cross clock domain protocols. In *Seventh International Workshop on Designing Correct Circuits DCC : Participants Proceedings*, 2006.

[11] J. Bulone and R. Sabbagh. A pragmatic approach to metastability-aware simulation. In *Design and Verification Conference and Exhibition Europe*, 2014.

[12] F. Burns, D. Sokolov, and A. Yakovlev. Gals synthesis and verification for xmas models. In *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1419–1424, Mar. 2015.

[13] M. Cannizzaro, S. Beer, J. Cortadella, et al. Saferazor : Metastability-robust adaptive clocking in resilient circuits. *IEEE TCSI 2015*, 62(9) :2238–2247, Sept. 2015.

[14] T. Chaney. My work on all things metastable or : (me and my glitch), 2012. `http://blendics.com/wp-content/uploads/2016/08/My-Work-on-All-Things-Metastable-OR-Me-and-My-Glitch.pdf`.

[15] T. J. Chaney and C. E. Molnar. Anomalous behavior of synchronizer and arbiter circuits. *IEEE Transactions on Computers*, C-22(4) :421–422, Apr. 1973.

[16] S. Chaturvedi. Static analysis of asynchronous clock domain crossings. In *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1122–1125, Mar. 2012.

[17] S. Churiwala. Tackling multiple clocks in socs. *EE Times*, 2004. `http://www.eetimes.com/document.asp?doc_id=1148839`.

[18] S. Churiwala, S. Garg, C. Gupta, and P. Joshi. Verification of clock domain crossing in socs. *Chip design magazine*, 2009. `http://chipdesignmag.com/display.php?articleId=3303`.

[19] E. Clarke, O. Grumberg, S. Jha, et al. Counterexample-guided abstraction refinement for symbolic model checking. *J. ACM*, 50(5) :752–794, 2003. `http://doi.acm.org/10.1145/876638.876643`.

[20] R. L. Cline. Method and circuit for metastable resolving time in low power multistate devices. *Patent US 5,789,945*, Aug. 1998.

[21] J. Cox, T. Chaney, and D. Zar. Simulating the behavior of synchronizers, 2011. `http://blendics.com/wp-content/uploads/2016/08/Blendics_Synchronizer13Feb2011.pdf`.

[22] J. Cox, G. Engel, D. Zar, et al. Metastability and fatal system errors, 2013. `http://blendics.com/wp-content/uploads/2016/08/Metastability-and-Fatal-System-Errors-rev-16-Sept-2013.pdf`.

[23] C. E. Cummings. Clock domain crossing (cdc) design & verification techniques using systemverilog. *SNUG Boston*, 2008.

[24] C. E. Cummings, D. Mills, and S. Golson. Asynchronous & synchronous reset design techniques - part deux. *SNUG Boston*, 2003.

[25] A. Cunningham. Characterization of metastability in at&t's orca fpgas. Master's thesis, Lehigh University, Jan. 1995.

[26] A. Cunningham and I. Sobanski. Implementation of a closed loop cdc verification methodology. In *Design and Verification Conference and Exhibition Europe*, 2014.

[27] W. J. Dally and J. W. Poulton. *Digital Systems Engineering*. Cambridge University Press, 1998.

[28] C. Dike and E. Burton. Miller and noise effects in a synchronizing flip-flop. *IEEE Journal of Solid-State Circuits*, 34(6) :849–855, Jun. 1999.

[29] R. Dobkin, R. Ginosar, and C. P. Sotiriou. Data synchronization issues in gals socs. In *Proceedings of the 2004 Tenth International Symposium on Asynchronous Circuits and Systems*, 2004.

[30] S. Gangadharan and S. Churiwala. *Constraining Designs for Synthesis and Timing Analysis : A Practical Guide to Synopsys Design Constraints (SDC).* Springer Publishing Company, 2015.

[31] R. Ginosar. Fourteen ways to fool your synchronizer. In *Proceedings of the 2003 Ninth International Symposium on Asynchronous Circuits and Systems*, pages 89–96, May 2003.

[32] R. Ginosar. Metastability and synchronizers : A tutorial. *IEEE Design Test of Computers*, 28(5) :23–35, Sept. 2011.

[33] R. Ginosar and R. Kol. Adaptive synchronization. In *Proceedings IEEE International Conference ICCD'98*, pages 188–189, Oct. 1998.

[34] S. Golson. Synchronization and metastability, 2014. `http://blendics.com/wp-content/uploads/2016/08/golson_snug14.pdf`.

[35] Mentor Graphics. *Questa Clock-Domain Crossing.* `http://www.mentor.com`.

[36] Mentor Graphics. *Questa CDC User Guide*, 2017.

[37] S. Gupta, A. Gupta, and A. Sethi. Efficient analysis of cdc violations in a million gate soc. *EDN*, 2014. `http://www.edn.com/design/integrated-circuit-design/4427206/Efficient-analysis-of-CDC-violations-in-a-million-gate-SoC--part-1`.

[38] M. Handover, J. Lovett, and K. Takara. Power aware cdc verification of dynamic frequency and voltage scaling (dvfs) artifacts. In *Design and Verification Conference and Exhibition Europe*, 2015.

[39] P. Hardee. *Choosing the Right Verification Technology for CDC-Clean RTL Signoff, White Paper.* `http://www.cadence.com`.

[40] M. Hurtado and D. L. Elliott. Ambiguous behavior of logic bistable systems. pages 605–611, Oct. 1975.

[41] H. Jackobson. Asynchronous circuit design : A case study of a framework called ack. Master's thesis, Lulea University, May 1996.

[42] I. W. Jones, S. Yang, and M. Greenstreet. Synchronizer behavior and analysis. In *2009 15th IEEE Symposium on Asynchronous Circuits and Systems*, pages 117–126, May 2009.

[43] P. L. Kapitza. Dynamic stability of a pendulum when its point of suspension vibrates. *Soviet Phys. JETP*, 21 :588–592, 1951.

[44] T. Kapschitz and R. Ginosar. Formal verification of synchronizers. In *Advanced Research Working Conference on Correct Hardware Design and Verification Methods*, pages 359–362. Springer, 2005.

[45] T. Kapschitz, R. Ginosar, and R. Newton. Verifying synchronization in multi-clock domain soc. *Proceedings of DVCon*, 4, 2004.

[46] N. Karimi and K. Chakrabarty. Detection, diagnosis, and recovery from clock-domain crossing failures in multiclock socs. *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, 32(9) :1395–1408, Sept. 2013. `http://dx.doi.org/10.1109/TCAD.2013.2255127`.

[47] M. Kayam, R. Ginosar, and C. E. Dike. Symmetric boost synchronizer for robust low voltage, low temperature operation, 2007.

[48] M. Kebaili. Vérification des "clock domain crossings" sur des systèmes hautes-performances et faible consommation d'énergie à base de processeurs. Master's thesis, Grenoble INP, 2014.

[49] M. Kebaili, J. C. Brignone, K. Morin-Allory, and D. Borrione. Enabler-based synchronizer model for clock domain crossing static verification. In *2015 Forum on Specification and Design Languages (FDL)*, pages 1–7, Sept. 2015.

[50] M. Kebaili, J. C. Brignone, S. Sarwary, and P. Mal Jain. Efficient clock domain crossing (cdc) verification methodology for mp-core asynchronous interfaces. In *IEEE Design Automation Conference*, June 2015. Designer and IP Track.

[51] M. Kebaili and d K. Morin-Allory J. C. Brignone an. Clock domain crossing formal verification : a meta-model. In *2016 IEEE International High Level Design Validation and Test Workshop (HLDVT)*, pages 136–141, OctN 2016.

[52] L. S. Kim and R. W. Dutton. Metastability of cmos latch/flip-flop. *IEEE Journal of Solid-State Circuits*, 25(4) :942–951, Aug. 1990.

[53] D. Kinniment, K. Heron, and G. Russell. Measuring deep metastability. In *Proceedings of ASYNC'06*, 2006.

[54] D. J. Kinniment, A. Bystrov, and A. V. Yakovlev. Synchronization circuit performance. *IEEE Journal of Solid-State Circuits*, 37(2) :202–209, Feb. 2002.

[55] D. J. Kinniment, C. E. Dike, et al. Measuring deep metastability and its effect on synchronizer performance. *IEEE Transactions on Very Large Scale Integration Systems*, 15(9) :1028–1039, Sept. 2007.

[56] D. J. Kinniment and J. V. Woods. Synchronisation and arbitration circuits in digital systems. *Proceedings of the Institution of Electrical Engineers*, 123(10) :961–966, Oct. 1976.

[57] S. Kirihennedige. Clock-domain crossing verification : Essentials to achieve soc success. *ChipEstimate Website*, 2014. `https://www.chipestimate.com/tech-talks/2014/10/28/Real-Intent-Clock-Domain-Crossing-Verification-Essentials-To-Achieve-SoC-Success`.

[58] L. Kleeman and A. Cantoni. Metastable behavior in digital systems. *IEEE Design Test of Computers*, 4(6) :4–19, Dec. 1987.

[59] C. Kwok, V. Gupta, and T. Ly. Using assertion-based verification to verify clock domain crossing signals. In *Proc. Design and Verification Conf*, pages 654–659, 2003.

[60] C. Kwok, J. Y. Larguier, Wesley Park, et al. Bridging block-level to top-level cdc verification : Hierarchical cdc verification. In *International Engineering Consortium DesignCon 2008*, volume 1 of *DesignCon 2008*, pages 245–261, 2008.

[61] C. K. Kwok, P. Viswanathan, and P. Yeung. Addressing the challenges of reset verification in soc designs. In *Design and Verification Conference and Exhibition United States*, 2015.

[62] C. Leong, P. Machado, V. Bexiga, et al. Built-in clock domain crossing (cdc) test and diagnosis in gals systems. In *13th IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems*, pages 72–77, 2010.

[63] B. Li and C. K. Kwok. Automatic formal verification of clock domain crossing signals. In *Proceedings of the 2009 Asia and South Pacific Design Automation Conference*, pages 654–659, 2009.

[64] M. Litterick et al. Pragmatic simulation-based verification of clock domain crossing signals and jitter using systemverilog assertions. *Proceedings of DV-Con*, 6, 2006.

[65] T. Ly, N. Hand, and C. K. Kwok. Formally verifying clock domain crossing jitter using assertion-based verification. In *Design & Verification Conference*, 2004.

[66] T. Ly, C. K. Kwok, V. Gupta, et al. Metastability effects simulation for a circuit description. *Patent US 7,356,789*, 2010.

[67] T. Ly, C. K. Kwok, and L. C. Widdoes Jr. Four steps to verifying unpredictable failure. *Chip design magazine*, 2006. `http://chipdesignmag.com/display.php?articleId=363`.

[68] Rafey Mahmud. Techniques to make clock switching glitch free. *EE Times*, 2003. `http://www.eetimes.com/document.asp?doc_id=1202359`.

[69] T. Mak. Truncation error analysis of mtbf computation for multi-latch synchronizers. *Microelectron. J.*, 43(2) :160–163, Feb. 2012. `http://dx.doi.org/10.1016/j.mejo.2011.09.011`.

[70] K. L. McMillan. *Symbolic Model Checking*. 1993.

[71] R. Melchiorre. A study of metastability in cmos latches. Master's thesis, Lehigh University, May 1992.

[72] D. G. Messerschmitt. Synchronization in digital system design. *IEEE Journal on Selected Areas in Communications*, 8(8) :1404–1419, Oct. 1990.

[73] B. Murphy. Reducing false errors in clock-domain crossing analysis. *EE Times*, 2005. `http://www.eetimes.com/document.asp?doc_id=1217953`.

[74] P. Narain and C. E. Cummings. Clock domain crossing demystified : The second generation solution for cdc verification. *Real Intent CDC White Paper*, 2008. `http://www.realintent.com/products/whitepapers`.

[75] A. Nordstrom. Don't over-constrain in formal property verification (fpv) flows. *EDN*, 2016. `http://www.edn.com/design/integrated-circuit-design/4441345/Don-t-over-constrain-in-formal-property-verification--FPV--flows`.

[76] K. P. Pandey, R. K. Singh, and S. K. Tadi. Clock domain crossing verification in a system on chip. *International Journal of Emerging Technology and Advanced Engineering*, 4 :617–622, 2014.

[77] G. Plassan, H. J. Peter, K. Morin-Allory, et al. Conclusively verifying clock-domain crossings in very large hardware designs. In *2016 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 1–6, Sept. 2016.

[78] R. Rajagopalan and A. Bhandari. Systematic method for capturing "design intent" of clock domain crossing (cdc) logic in constraints. In *IEEE Design Automation Conference*, June 2013. User Track.

[79] J. Rennert and D. Hafeman. Clock domain modeling is essential in high density soc design. *EE Times*, 2003. http://www.eetimes.com/document.asp?doc_id=1226543.

[80] R. Saleh, S. Wilton, S. Mirabbasi, et al. System-on-chip : Reuse and integration. *Proceedings of the IEEE*, 94(6) :1050–1069, June 2006.

[81] M. S. Sarwary, M. Mneimneh, P. Mal Jain, et al. System and method for filtration of error reports respective of static and quasi-static signals within an integrated circuit design. *Patent US 20,140,282,322*, Sept. 2014.

[82] S. Sarwary. Solving the toughest problems in cdc analysis. *EE Times*, 2006. http://www.eetimes.com/document.asp?doc_id=1276024.

[83] S. Sarwary, M. Mneimneh, and M. H. Movahed-Ezazi. System and method for a hybrid clock domain crossing verification, 2015. https://www.google.es/patents/US8984457.

[84] S. Sarwary and S. Verma. Critical clock-domain-crossing bugs, 2008. http://www.edn.com/design/integrated-circuit-design/4325224/Critical-clock-domain-crossing-bugs.

[85] S. Sarwary and S. Verma. Critical clock-domain-crossing bugs. *EDN*, 53(7) :55–64, 2008.

[86] J. Schmaltz. A formal model of clock domain crossing and automated verification of time-triggered hardware. In *Formal Methods in Computer Aided Design (FMCAD)*, pages 223–230, Nov. 2007.

[87] Y. Semiat and R. Ginosar. Timing measurements of synchronization circuits. In *Proceedings of the 2003 Ninth International Symposium on Asynchronous Circuits and Systems*, pages 68–77, May 2003.

[88] IEEE Computer Society and Working Group. *IEEE Standard for Property Specification Language (PSL)*. pub-IEEE-STD-1850, Apr. 2010.

[89] IEEE Computer Society and Working Group. *IEEE Standard for SystemVerilog–Unified Hardware Design, Specification, and Verification Language*. pub-IEEE-STD-1800, Feb. 2013.

[90] STMicroelectronics. *ST Custom asynchronous bridge behavior and customization description for CPU subsytems*, Feb. 2012.

[91] STMicroelectronics. *Clock Domain Crossing (CDC) Check Flow for Multi-Processor Sub-Systems*, Nov. 2014.

[92] STMicroelectronics. *MontBlanc CPU Subsystem Design Specification*, July 2014.

[93] STMicroelectronics. *MTBF and synchronizers in CPU subsystems*, Jan. 2014.

[94] STMicroelectronics. *Multi-Stage Synchronizer : Concepts & MTBF computation Methodology*, July 2014.

[95] Synopsys. *SpyGlass CDC : Comprehensive, Low-Noise Clock Domain Crossing Verification*. http://www.synopsys.com.

[96] Synopsys. *Leda CDC Documentation*, 2009. https://filebox.ece.vt.edu/\~athanas/4514/ledadoc/html/pol_cdc.html.

[97] Synopsys. *SpyGlass CDC Guideware*, 2017.

[98] Cadence Design System. *Closing the Loop on Clock Domain Functional Implementation Problems*, 2004. `http://www.cadence.com`.

[99] Cadence Design System. *Clock Domain Crossing Verification App User Guide*, 2017.

[100] K. Takara, C. K. Kwok, N. Jain, and A. Hari. Next-generation power aware cdc verification - what have we learned ? In *Design and Verification Conference and Exhibition United States*, 2015.

[101] G. Tarawneh, A. Mokhov, and A. Yakovlev. Formal verification of clock domain crossing using gate-level models of metastable flip-flops. In *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2016*, pages 1060–1065. IEEE, 2016.

[102] P. Teehan, M. Greenstreet, and G. Lemieux. A survey and taxonomy of gals design styles. *IEEE Design Test of Computers*, 24(5) :418–428, Sept. 2007.

[103] W. M. K. Trochim. Outcome pattern matching and program theory. *Evaluation and Program Planning*, 12(4) :355–366, 1989. `http://EconPapers.repec.org/RePEc:eee:epplan:v:12:y:1989:i:4:p:355-366`.

[104] H. J. M. Veendrick. The behavior of flip-flops used as synchronizers and prediction of their failure rate. *IEEE Journal of Solid-State Circuit*, 15(2) :169–176, Apr. 1980.

[105] S. Verma and A. S. Dabare. Understanding clock domain crossing issues. *EE Times*, 2007. `http://www.eetimes.com/document.asp?doc_id=1276114`.

[106] P. Yeung, E. Marschner, and K. Liu. Multi-domain verification : When clock, power and reset domains collide. In *Design and Verification Conference and Exhibition United States*, 2015.

[107] J. Zhou. *Design and Measurement of Synchronizers*. PhD thesis, Newcastle University, Nov. 2008.

[108] J. Zhou, M. Ashouei, D. J. Kinniment, et al. Extending synchronization from super-threshold to sub-threshold region. In *2010 IEEE Symposium on Asynchronous Circuits and Systems*, pages 85–93, May 2010.

[109] J. Zhou, D. J. Kinniment, C. E. Dike, et al. On-chip measurement of deep metastability in synchronizers. *IEEE Journal of Solid-State Circuits*, 43(2) :550–557, Feb. 2008.

[110] J. Zhou, D. J. Kinniment, G. Russell, and A. Yakovlev. Adapting synchronizers to the effects of on chip variability. In *2008 14th IEEE International Symposium on Asynchronous Circuits and Systems*, pages 39–47, Apr.l 2008.

[111] Jun Zhou, D. Kinniment, G. Russell, and A. Yakovlev. A robust synchronizer. In *IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures*, Mar. 2006.

Annexe : Le langage SVA

Les assertions sont principalement utilisées pour valider le comportement d'un circuit. Elles peuvent être vérifiées dynamiquement en simulation (assertions immédiates) ou statiquement par des outils de vérification formelle (assertions concurrentes). Lorsqu'une assertion concurente est définie, l'outil cherche à savoir si le circuit respecte ou non le comportement spécifié. Les assertions concurrentes sont basées sur la sémantique des horloges : elles sont évaluées à chaque front d'activation de l'horloge, on parle de coup d'horloge (clock tick). Elles décrivent donc le comportement du circuit de manière temporelle à travers des séquences (par rapport à une ou plusieurs horloges). La structure des assertions peut être décomposées en quatre niveaux :

— Le niveau expressions booléennes ;

— Le niveau séquences ;

— Le niveau propriétés ;

— Le niveau directives de vérification.

Le niveau expressions booléennes C'est le niveau le plus bas : les expressions booléennes expriment l'état de signaux à un coup d'horloge donnée. Elles sont donc composées de la définition du signal d'horloge et du front (montant ou descendant) sur lequel la propriété est évaluée. Par ailleurs, il est possible de définir que l'évaluation de la propriété est vraie lorsqu'un signal de reset asynchrone est actif. En effet, les signaux de reset asynchrones peuvent changer d'état indépendamment d'un signal d'horloge pour forcer la sortie des registres à un état donné et donc peuvent rendre pessimiste la vérification. Concernant l'état des signaux évalués, ils peuvent être décrits avec les opérateurs classiques des langages de description matérielle (arithmétique, binaire, logique, relationnel..) Il est possible par ailleurs d'affecter l'état de signaux ou d'expressions booléennes dans une variable afin de rendre les propriétés génériques.

Le niveau séquence Une séquence est une liste d'expressions booléennes se suivant linéairement dans le temps. La séquence est vraie tout le temps si les expressions booléennes sont vraies à des fronts spécifiques. Il existe trois principaux opérateurs permettant d'exprimer des séquences :

— "##N" pour définir l'exécution d'un délai correspondant au nombre de coups d'horloge spécifié par N. ;

— "∗N" pour spécifier le maintien de l'expression booléenne pendant N coups d'horloge consécutifs ;

— "$" pour exprimer le maintien de l'expression booléenne pendant au moins N coups d'horloge consécutifs ;

Notons qu'il est par ailleurs possible d'utiliser des fonctions qui intègrent des séquences. Le langage SVA, intègre six fonctions principales :

— "$rose" pour définir un front montant sur un signal ou une expression entre deux coups d'horloge consécutifs ;

— "$fell" pour définir un front descendant sur un signal ou une expression entre deux coups d'horloge consécutifs ;

— "$stable" pour exprimer le maintien de l'état d'un signal ou d'une expression entre deux coups d'horloge consécutifs ;

— "$past" pour rapporter l'état d'un signal ou d'une expression au coup d'horloge précédent ;

— "$onehot" pour spécifier que pour un bus données, un seul bit est à 1 par coup d'horloge ;

— "$onehot0" pour spécifier que pour un bus données, un seul bit peut être à 1 par coup d'horloge.

Le niveau propriétés Le niveau propriété permet de décrire des enchaînements de séquences à l'aide des opérateurs d'implication. L'opérande à gauche de l'implication est appelé antécédent, et celui de droite est la conséquence. Si l'antécédent n'est jamais vrai, l'implication est rapportée vraie mais vide de sens puisque la propriété n'a pas été évaluée. Si l'antécédent survient, c'est à dire qu'il est vrai alors la conséquence est évaluée, à partir du point d'implication. Il existe deux types d'implication : les chevauchantes "$|->$" et les non-chevauchantes "$|=>$". Comme leurs noms l'indique, pour les chevauchantes, la conséquence est évaluée au coup d'horloge où l'antécédent survient et pour les non-chevauchantes, l'évaluation se fait coup d'horloge suivant.

Le niveau directives de vérification Une propriété correspond à la définition d'un comportement, elle ne donne pas d'information sur la manière dont elle doit être interpretée. Le niveau directive permet donc de définir à l'outil comment la propriété doit être prise en compte. Elle peut être définie comme devant être vérifiée (assert) ou comme dévant être considérée comme vraie pour contraindre certains signaux (assume).

La déclaration des assertions Les assertions peuvent être directement définies dans la description du module sur laquelle elles s'appliquent ou être déclarées dans un module à part entière et instanciées par la suite. Elles peuvent aussi être écrites dans un programme séparé puis connectées à des signaux à travers des modules spécifiques. Dans les outils de vérification statique, après la détection des modèles, il y a une création automatique de modules spécifiques dédiés.

Exemple de propriété La figure ci-dessous présente un exemple d'assertion décrite en SVA, spécifiant à l'outil de vérifier que le signal Data doit rester stable pendant trois cycles d'horloge Clk lorsque le signal enable est à l'état '1'.

```
assert property                // Il faut vérifier
  (@(posedge Clk)              // à chaque front montant de l'horloge Clk
    disable iff (Reset)        // hormis quand le signal Reset est actif
      enable                   // que si le signal enable est à l'état '1'
        |=>                    // alors au coup d'horloge suivant
          $stable(data)        // le signal data reste au même état
                [*3]           // pendant trois coups d'horloge consécutifs
```

FIGURE 11 – Exemple d'assertion en SVA

Réflexions autour de la méthodologie de vérification des circuits multi-horloges : analyse qualitative et automatisation

Résumé - Depuis plusieurs années, le marché des circuits intégrés numériques requiert des systèmes de plus en plus complexes dans un temps toujours plus réduit. Afin de répondre à ses deux exigences, les industriels de la conception font appel à des fournisseurs externes proposant des circuits fonctionnant sur des signaux d'horloge dédiés. Lorsque ces derniers communiquent entre eux, les horloges d'émission et de réception ne sont pas les mêmes, on parle de « Clock Domain Crossing » (CDC). Les CDC correspondent à des communications asynchrones et peuvent provoquer des dysfonctionnements critiques. Avec l'augmentation du nombre de CDC dans les circuits, les fournisseurs d'outils (EDA) ont proposé des solutions logicielles spécialisées dans la vérification statique des CDC. Cependant, les circuits développés étant en constante évolution, les outils ne sont pas en mesure de s'adapter. Pour pallier ces problèmes, la vérification industrielle des CDC est basée sur la spécification de contraintes et d'exclusions par l'utilisateur. Ces actions, qui se substituent aux outils, peuvent masquer des bugs. Les travaux de recherche ont consisté à analyser les différentes techniques de conception et de vérification des CDC à travers l'évaluation des principaux outils du marché. A partir des résultats obtenus, nous avons formalisé les problèmes pratiques et proposé des modèles permettant d'obtenir des résultats exhaustifs automatiquement. Les essais ont été réalisés sur un sous-système à base de processeurs (CPUSS) développé chez STMicroelectronics. L'adoption de nos modèles permet une vérification complète des CPUSS de manière automatique ce qui est essentiel dans un environnement industriel compétitif. Une collaboration ayant été mise en place avec les principaux fournisseurs outils, certaines solutions seront probablement intégrées aux outils dans les années à venir.

Mots clés : CDC, Interface asynchrone, Flot de vérification, Métastabilité, Propriétés formelles, GALS, Structures de synchronisation, Outils de vérification statique, Vérification de modèles.

"Reflections on the methodology for verifying multi-clock design: qualitative analysis and automation"

Abstract - For several years now, the digital IC market has been requiring both more complex systems and reduced production times. In this context, the semiconductor chip maker companies call on external IP providers offering components working on dedicated clock signals. When these IPs communicate between them, the source and destination clocks are not the same, we talk about "Clock Domain Crossing" (CDC). CDC correspond to asynchronous communications and can cause critical failures. With the increase of CDC in the digital designs, EDA tools providers have developed software solutions dedicated to CDC static verification. Whereas, the designs are subject to continuous change, the verification tools are not able to be up to date. To resolve these practical issues, the CDC industrial verification is based on the specification of constraints and exclusions by the user. This manual flow, which replaces the tools, can mask bugs. Our goal has been to automate the verification submitting solutions based on formal properties. The work consisted in the analysis of the different CDC design and verification approaches through the evaluation of main CDC checker tools. From the results obtained, we have formalized the practical problems and proposed models to obtain automatically exhaustive results. The tests have been performed on a processor-based subsystem (CPUSS) developed at STMicroelectronics. Adopting our models enables a complete checking of CPUSS in an automatic way, which is essential within a competitive industrial environment. Since a collaboration has been established with the main tool providers some solutions would probably be included into the tools over the coming years.

Keywords: CDC, Asynchronous interface, Verification flow, Metastability, Formal properties, GALS, Synchronizers, Static checker tools, Model checking.

Thèse préparée au laboratoire TIMA (Techniques de l'Informatique et de la Microélectronique pour l'Architecture des ordinateurs), 46 avenue Félix Viallet, 38031 Grenoble Cédex, France

ISBN : 978-2-11-129230-7