Génération d'aléa dans les circuits électroniques numériques exploitant des cellules oscillantes

Ugo Mureddu

N° d'ordre NNT : 2019LYSES18

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON

opérée au sein de

Laboratoire Hubert Curien

École Doctorale N° 488

Sciences Ingénierie Santé

Spécialité de doctorat :

Microélectronique

Soutenue publiquement le 09/09/2019, par :

Ugo Mureddu

Génération d'aléa dans les circuits électroniques numériques exploitant des cellules oscillantes

Conception, caractérisation, modélisation et sécurisation

Devant le jury composé de :

Anghel, Lorena	PR	Grenoble INP	Rapporteur
Hély, David	MCF-HDR	Grenoble INP	Rapporteur
Dallet, Dominique	PR	IIP Bordeaux	Examinateur
Fischer, Viktor	PR	Université Jean Monnet	Examinateur
Haddad, Patrick	PhD	STMicroelectronics	Examinateur
Bossuet, Lilian	PR	Université Jean Monnet	Directeur de thèse

This work has received funding from the European Union's Horizon 2020 research and innovation programme in the framework of the project HECTOR (Hardware Enabled Crypto and Randomness) under grant agreement No 644052.

Remerciements

Table des matières

Liste des abréviations

ASIC	application-specific integrated circuit (en français : circuit intégré pour des applications spécifiques)
CLB	configuable logic bloc (en français : bloc logique configurable)
COSO-TRNG	ring oscillator coherent sampling based TRNG (en français : TRNG basé sur échantillonnage cohérent d'oscillateur en anneau)
DRNG	deterministic random number generator (en français : générateur de nombres pseudo-aléatoires)
EM	électromagnétique
ERO-TRNG	ring oscillator based elementary TRNG (en français : TRNG élémentaire à oscillateur en anneau)
Flip-flop-PUF	flip-flop based PUF (en français : PUF à base de bascule D)
FPGA	field-programmable gate array (en français : circuit logique programmable)
HD	Hamming distance (en français : distance de Hamming)
HRNG	hybrid random number generator (en français : générateur mixte de nombres aléatoires)
IID	indépendant et identiquement distribué
IoT	internet of things (en français : internet des objets)
LFSR	linear feedback shift register (en français : registre à décalage à rétroaction linéaire)
LUT	look-up table (en français : table de correspondance)
LVDS	low voltage differential signaling (en français : transmission de signal différentielle et basse tension)
MOSFET	metal oxide semiconductor field effect transistor (en français : transistor à effet de champ à structure métal-oxyde-semiconducteur)
MURO-TRNG	multi-ring oscillator based TRNG (en français : TRNG basé sur de multiples oscillateurs en anneaux)

NP-TRNG	non-physical true random number generator (en français : générateur non physique de nombres véritablement aléatoires)
PLL	phase-locked loop (en français : boucle à verrouillage de phase)
PLL-TRNG	coherent sampling based TRNG using PLL (en français : TRNG basé sur l'échantillonnage cohérent utilisant des boucles à verrouillage de phase)
PUF	physical unclonable function (en français : fonction physique non-clonable)
RO	ring oscillator (en français : oscillateur en anneau)
RO-PUF	ring oscillator based PUF (en français : PUF à base d'oscillateurs en anneau classiques)
SCADA	deterministic random number generator (en français : générateur de nombres pseudo-aléatoires)
SoC	system on chip (en français : système sur une puce)
SRAM	static random access memory (en français : mémoire volatile statique)
SRAM-PUF	static random access memory based PUF (en français : PUF à base de mémoires volatiles statiques)
SR-PUF	SR latch based PUF (en français : PUF à base de bascule RS)
STR	self-timed ring (en français : oscillateur en anneau auto séquencé)
STR-TRNG	self-timed ring based TRNG (en français : TRNG à oscillateur en anneau auto séquencé)
TERO	transient effect ring oscillator (en français : oscillateur en anneau à effet transitoire)
TERO-PUF	transient effect ring oscillator based PUF (en français : PUF à base d'oscillateurs en anneau à effet transitoire)
TERO-TRNG	transient effect ring oscillator based TRNG (en français : TRNG à oscillateur en anneau à effet transitoire)
TRNG	true random number generator (en français : générateur de nombres véritablement aléatoires)
XOR	Exclusive OR (en français : OU-EXCLUSIF logique)

Notations

a	Nombre d'évènements e (quand $e < (N/2)$) ou nombre de cellules de Muller vide (quand $e > (N/2)$)
A_i	Cellule numéro i du bloc A d'une PUF à base de cellules oscillantes
α	coefficient de pente de la partie linéaire de la sortie d'un inverseur
B_i	Cellule numéro i du bloc B d'une PUF à base de cellules oscillantes
β	Facteur de courant du transistor : $\mu C_{ox} \frac{W}{L}$
c	Célérité d'une onde dans le vide ($\approx 300\,000$ km /s)
C	Challenge/signal d'entrée d'une PUF
C_L	Capacité de charge d'une porte logique
C_{ox}	Capacité de l'oxyde de grille du transistor
D	Paramètre exprimant le rapport cyclique du signal de sortie d'une TERO normalisé entre 50 et 0 en %
d_{ET}	Délai moyen d'une porte ET logique
d_{INV}	Délai moyen d'une porte NON logique
d_m	Délai moyen d'une porte d'activation
d_{MULLER}	Délai moyen d'une cellule de Muller
d_n	Délai moyen d'une porte de délai
Δ_{e_1/e_2}	$\lvert t_{e_1} - t_{e_2} \rvert$
$\overline{\Delta f}$	$\lvert \overline{f_{pert} - f_{osc}} \rvert$
$\overline{\Delta_\varphi}$	$\lvert \overline{\varphi_{pert} - \varphi_{osc}} \rvert$
e	Nombre d'évènements à l'intérieur d'une cellule oscillante
e_1	Évènement numéro 1
e_2	Évènement numéro 2
φ_{osc}	Phase du signal de sortie d'une cellule oscillante

φ_{pert}	Phase d'un signal de perturbation
$\|FFTout(f)\|$	Transformée de Fourier rapide du signal de sortie d'une TERO
f_{osc}	Fréquence du signal de sortie d'une cellule oscillante
f_{pert}	Fréquence d'un signal de perturbation
F_{osc}	Variable aléatoire représentant la fréquence d'oscillations d'une cellule RO
HD_{ext}	Mesure d'unicité des réponses d'une PUF
HD_{int}	Mesure de stabilité des réponses d'une PUF
K	Nombre de bits d'une réponse de PUF
K_d	Facteur de division d'un compteur
K_D	Facteur de division d'une PLL
K_M	Facteur de multiplication d'une PLL
I	Nombre de circuits contenant une PUF
I_{ds}	Courant dans le canal de conduction d'un transistor
I_{lin}	Courant dans le canal de conduction d'un transistor en régime linéaire
I_{sat}	Courant dans le canal de conduction d'un transistor en régime saturé
L	Longueur de la grille du transistor
l	Longueur d'une ligne électrique
λ	Longueur d'onde
m	Nombre de cellules dans un bloc d'une PUF à base de cellules oscillantes
M	Nombre de portes d'activation d'une cellule oscillante
$mod(t)$	Signal de modulation
n	Nombre de bits
N	Nombre de portes de délai d'une cellule oscillante
N_{osc}	Nombre d'oscillations du signal de sortie d'une TERO
$PWM(t)$	Signal de modulation de largeur d'impulsions
$R_{i,y,k}$	Bit numéro k de la réponse numéro y d'une PUF sur un circuit i
$\overline{R_i}$	Réponse de référence du circuit i provenant de la moyenne des réponses $R_{i,y}$ aux conditions nominales de fonctionnement.
$\sigma_{T_{osc}}$	Écart type de T_{osc}
$\sigma_{T_{pert}}$	Écart type de T_{pert}

σ_{Δ_φ}	Écart type de Δ_φ
σ^2_β	Variance de β
$\sigma^2_{I_{ds}}$	Variance de I_{ds}
$\sigma^2_{V_{th}}$	Variance de V_{th}
t_d	Temps de descente de la sortie d'un inverseur
t_{e_1}	Délai de propagation de l'évènement 1 à travers une cellule oscillante
t_{e_2}	Délai de propagation de l'évènement 2 à travers une cellule oscillante
t_h	Temps à l'état logique haut ('1') d'un signal périodique
t_l	Temps à l'état logique bas ('0') d'un signal périodique
t_{li}	Délai de propagation à travers une ligne de longueur l
T_{lim}	Délai induit par le limiteur de pente
t_m	Temps de montée de la sortie d'un inverseur
T_d	Variable aléatoire gaussienne représentant le temps de descente d'un inverseur
T_{in}	Largeur d'impulsion à l'entrée d'un inverseur
T_{li}	Variable aléatoire gaussienne représentant le temps de propagation à travers une ligne de longueur l
T_m	Variable aléatoire gaussienne représentant le temps de montée d'un inverseur
T_{osc}	Période du signal de sortie d'une cellule oscillante
T_{out}	Largeur d'impulsion à la sortie d'un inverseur
T_{pert}	Période d'un signal de perturbation
T_r	Temps de retard
T_{th}	Temps au bout duquel le transistor passe du régime saturé au régime linéaire
$tri(t)$	Signal triangulaire
τ	Constante de temps
μ	Mobilité des porteurs de charges du transistor
v	Vitesse de propagation du signal à l'intérieur d'une ligne électrique
V_{DD}	Tension d'alimentation d'un circuit
V_{ds}	Tension entre le drain et la source d'un transistor
V_{gs}	Tension entre la grille et la source d'un transistor
V_{osc}	Tension du signal de sortie d'une cellule oscillante

V_{pert}	Tension d'un signal de perturbation
V_{th}	Tension de seuil d'un transistor
W	Largeur de la grille du transistor
X	Variable aléatoire discrète
Y	Nombre de réponses d'une PUF sur un même circuit

Introduction

Contexte actuel

Les avancées technologiques dans le monde de la microélectronique ont donné lieu à des transistors de plus en plus petits. Ils peuvent être intégrés dans d'innombrables systèmes et ce quelle que soit leur taille. En effet, avec des technologies de transistor inférieures à la dizaine de nanomètres, les possibilités d'intégration sont considérables. Ceci rend les systèmes embarqués totalement polyvalents, de plus en plus intelligents, pouvant prendre des décisions critiques, et ce tout en devenant de plus en plus connectés sur des réseaux ouverts sur le monde. L'émergence des systèmes embarqués a changé notre société. De jour comme de nuit, nous sommes entourés de systèmes embarqués dotés de capacités de communications sans fil et de capteurs intégrés. Tous ces appareils connectés forment l'internet des objets — en anglais internet of things (IoT). En février 2017, le nombre d'objets connectés dans le monde a été évalué à 8,4 milliards, ce qui représente une hausse de 31% par rapport à l'année précédente selon Gartner Inc [1]. Ce chiffre n'a pas fini de progresser, car en 2020 il est estimé à 20 milliards.

Dans ce monde où les objets connectés sont omniprésents (*ex.* véhicules, transports en commun, santé, domotique, smartphone, moyens de paiement ou encore industries connectées.), notre façon de vivre et de travailler a complètement changé. La connexion des appareils d'usage quotidien améliore considérablement notre confort, mais peut également engendrer des problèmes de sécurité sans précédent. Les problèmes de sécurités liés au vaste déploiement des systèmes embarqués et par extension de l'IoT sont doubles :

1. **La sécurité des données :** elle est compromise par l'accès d'une personne non autorisée aux données pour la lecture, la copie, l'écriture ou l'effacement complet. À titre d'exemple, nous citerons les deux vulnérabilités, Spectre et Meltdown, découvertes au début de l'année 2018 [1], [2] [2].
2. **La sécurité du système :** elle est compromise lors de l'utilisation du système pour une action non prévue par celui-ci, la mise hors service du système ou bien sa destruction. La

1. https://www.forbes.com/sites/louiscolumbus/2017/12/10/2017-roundup-of-internet-of-things-forecasts/
2. https://meltdownattack.com/

première attaque de ce genre remonte à juin 1982 lorsque l'ancienne Union soviétique a été victime d'une attaque sur son infrastructure de gazoduc provoquant une explosion en Sibérie [3]. Le célèbre virus Stuxnet est un autre exemple d'attaque touchant l'intégrité du système [4].

À mesure que le monde de l'IoT augmente, nous aurons affaire à de plus en plus d'attaques de ce genre. Pour faire face à de tels risques, nous avons besoin de mécanismes de sécurité permettant de protéger les données sensibles, ainsi que d'une authentification et une autorisation d'accès pour chaque appareil de l'IoT. Ces mécanismes doivent être intégrés dès les premières étapes de la conception. Fort heureusement, les fonctions cryptographiques permettent de répondre à ces besoins puisqu'elles garantissent :

1. **Confidentialité :** garantir que les données ne sont accessibles que par les utilisateurs à qui elles sont destinées.
2. **Authenticité :** garantir que l'on communique avec la bonne personne.
3. **Intégrité :** garantir que les données n'ont pas été modifiées sans autorisation ou par erreur entre l'émetteur et le destinataire.
4. **Non-répudiation :** garantir qu'une action soit imputable à un utilisateur sans que cela puisse être contesté ou qu'un autre utilisateur puisse se l'attribuer.

Dans ce contexte, les générateurs physiques d'aléa sont essentiels puisqu'ils assurent le bon fonctionnement de certaines fonctions cryptographiques lorsque celles-ci sont réalisées dans les circuits intégrés qui sont au cœur des systèmes embarqués. En effet, ils exploitent des sources de bruit analogique présentes dans les circuits électroniques pour générer des clés secrètes permettant de chiffrer les données, des masques aléatoires ou vecteurs d'initialisation pour des protocoles cryptographiques, ou encore, des identifiants uniques permettant l'authentification des circuits.

Puisqu'ils font partie des éléments initiaux nécessaires à la mise en place de fonctions cryptographiques, les générateurs d'aléa sont classés dans la catégorie des primitives cryptographiques. La sécurité des fonctions cryptographiques repose sur la qualité des clés et identifiants générés par ces générateurs d'aléa. Les nombres produits par ces générateurs appliqués à la cryptographie doivent être imprévisibles. À défaut, les clés utilisées pour chiffrer les données pourraient être cassées et les identifiants recopiés.

Il est très important de réaliser que les systèmes embarqués qui nous entourent sont en majorité composés de circuits électroniques numériques. Or la génération d'aléa au sein de ces circuits va à l'encontre du type d'applications pour lesquelles ils ont été conçus. Un système numérique est un système dont les états parcourent un ensemble fini de possibilités. Les variations du système se font par palier et niveaux prédéfinis et sont toujours supposées être dans un état stable.

La génération d'aléa dans les circuits numériques, exploitant des variations aléatoires ou états instables, n'est donc pas triviale.

C'est pourquoi il est d'une extrême nécessité d'étudier les générateurs physiques d'aléa, les modéliser, s'assurer de leur efficacité une fois implantés sur un circuit électronique numérique et vérifier leur résistance aux attaques. La section suivante introduit les différents générateurs d'aléa et leurs utilisations.

Générateurs d'aléa

Il existe deux types de générateurs d'aléa : les générateurs déterministes étant les générateurs de nombres pseudo-aléatoires — en anglais deterministic random number generators (DRNG) — et les générateurs véritablement aléatoires incluant les générateurs de nombres véritablement aléatoires — en anglais true random number generators (TRNG) — et les fonctions physiques non-clonables — en anglais physical unclonable functions (PUF).

Générateurs de nombres pseudo-aléatoires

Les DRNG ont l'avantage d'être implémentables en logiciel, ce qui les rend peu coûteux. Ce sont des algorithmes capables de générer une suite de nombres avec un débit de sortie élevé et de bonnes propriétés statistiques. Cependant, la sortie est complètement dépendante de la valeur d'entrée de l'algorithme, appelée graine d'initialisation. En effet, une graine d'initialisation identique donne une suite de nombres aléatoires identiques. Pour cette raison, leur cas d'utilisation classiques sont, par exemple, la simulation stochastique pour laquelle les propriétés statistiques du DRNG sont utiles alors que l'imprévisibilité ne l'est pas.

Pour ne citer qu'un exemple de DRNG, nous prendrons le registre à décalage à rétroaction linéaire — en anglais linear feedback shift register (LFSR).

Les DRNG n'étant pas appropriés pour des applications telles que le chiffrement de données ou l'identification de circuits, ils ne seront pas traités dans cette thèse. Ainsi, ils ne seront pas détaillés outre mesure.

Générateurs physiques d'aléa

La figure 0.1 représente la structure générale d'un générateur physique d'aléa qui se décompose en deux parties :

1. **La source d'aléa**, un circuit électronique générant un signal analogique.
2. **Le numériseur**, responsable d'extraire et de convertir sous forme numérique (la plupart du temps une séquence binaire) la source d'aléa.

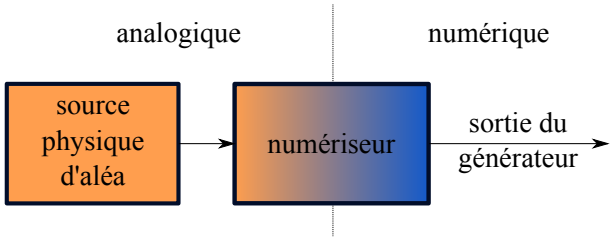


FIGURE 0.1: Structure d'un générateur physique d'aléa

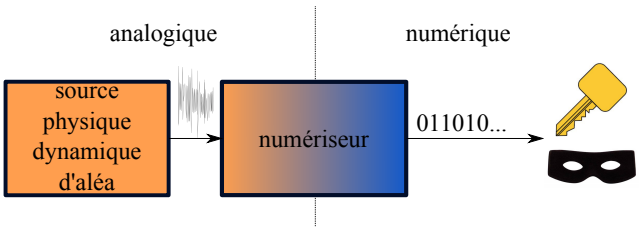
Ces deux entités représentent le cœur d'un générateur physique d'aléa. En revanche, bien souvent, la qualité statistique de la suite générée est moins bonne que celle d'un DRNG. Par conséquent, il n'est pas rare d'appliquer un post-traitement à la séquence brute produite par le générateur physique. Ce post-traitement se traduit par la mise en œuvre d'un algorithme augmentant la qualité statistique de la séquence.

Il existe deux grandes familles de générateurs d'aléa basés sur des phénomènes physiques :

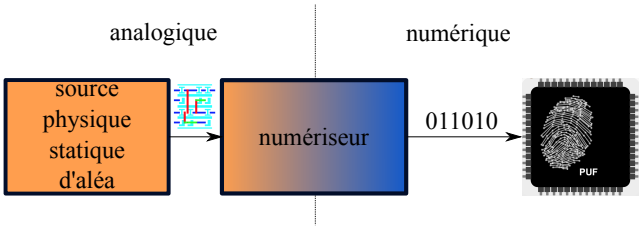
— **Les générateurs de nombres véritablement aléatoires** exploitant un phénomène physique aléatoire dynamique : les bruits dans les composants électroniques.

— **Les fonctions physiques non-clonables** exploitant un phénomène physique aléatoire statique : les variations de procédés de fabrication des composants électroniques.

Le TRNG génère un flux continu de bits aléatoires utilisés comme clés de chiffrement des données, masques aléatoires pour protéger contre les attaques matérielles ou vecteurs d'initialisation d'algorithme cryptographique. Le fonctionnement et ses utilisations sont imagés sur la figure 0.2a. La PUF génère un mot aléatoire constant de n bits utilisé pour identifier et authentifier un circuit électronique comme représenté sur la figure 0.2b.



(a) Principe du TRNG



(b) Principe de la PUF

FIGURE 0.2: Les deux types de générateurs physiques d'aléa

Générateurs de nombres véritablement aléatoires

Qu'est-ce qu'un générateur de nombres véritablement aléatoires ? Un TRNG idéal est un dispositif produisant une suite de nombres successifs pouvant prendre n'importe laquelle de ses valeurs admises avec la même probabilité et de manière totalement indépendante de ses prédécesseurs et successeurs. Avec une connaissance complète du dispositif et une puissance de calcul infinie, un attaquant ne doit pas avoir de meilleure stratégie qu'une attaque par force brute pour deviner la suite générée.

Naturellement, un TRNG idéal est une construction mathématique qui n'existe pas dans la réalité, car bien souvent la suite de nombres générés comporte un biais statistique ou une corrélation. Cependant, à l'inverse des générateurs de nombres pseudo-aléatoires, il n'existe aucun lien déterministe (connu) entre deux valeurs successives. Il est important de ne pas confondre TRNG et DRNG pour lesquelles les applications et propriétés sont différentes.

Pour certaines applications les TRNG peuvent être couplés avec les DRNG pour former des générateurs mixtes de nombres aléatoires — en anglais hybrid random number generators (HRNG). Dans ce cas, le TRNG génère périodiquement la graine d'initialisation du DRNG.

Par opposition aux DRNG, les TRNG ont l'avantage d'être imprévisibles, car ils utilisent une source d'aléa physique. Cependant, le débit de sortie est généralement plus faible, de même que la qualité de leurs propriétés statistiques (*ex.* biais statistique de la séquence générée).

Sources d'aléa dynamiques dans les circuits électroniques numériques. Les sources d'aléa liées à un processus physique, communément utilisées par les TRNG dans les circuits électroniques numériques, viennent de bruits électroniques. Les sources d'aléa les plus fréquemment utilisées sont les suivantes :

- le jitter de l'horloge : phénomène de fluctuation d'un signal périodique par rapport à sa période idéale. C'est la source d'aléa la plus utilisée pour les TRNG implantés dans les circuits électroniques numériques.
- la métastabilité analogique : aptitude d'un circuit électronique à rester en équilibre à un état indéfini pour une durée indéterminée.
- la métastabilité oscillatoire : aptitude d'un circuit électronique bistable à osciller pour une durée indéfinie.
- le chaos : comportement imprévisible d'un système déterministe, très sensible à ses conditions initiales.

Générateurs de nombres véritablement aléatoires non physiques. Il existe aussi des TRNG non physiques (NP-TRNG). Sont considérés comme NP-TRNG les générateurs exploitant, par exemple, les interactions d'un ordinateur avec un utilisateur (appuie sur une touche du clavier,

temps entre l'appui sur deux touches consécutives, mouvement de la souris, *etc.*) ou les données système (temps, données en mémoire volatile, nombres de tâches, *etc.*). Comme ils ne sont pas utilisables dans des circuits intégrés spécifiques, ce manuscrit ne traitera pas des NP-TRNG.

Fonctions physiques non-clonables

Qu'est-ce qu'une fonction physique non-clonable ? Une PUF est une structure numérique capable d'exploiter les variations aléatoires des procédés de fabrication des circuits intégrés pour générer un identifiant unique se présentant, la plupart du temps, sous un nombre binaire. Le principe des PUF repose sur le fait que les éléments composant chaque circuit électronique numérique sont, grâce aux variations de procédés de fabrication, impactés par des variations stochastiques locales. De ce fait, une même structure PUF implantée sur plusieurs circuits donne des résultats différents.

Dans la littérature, une PUF est très souvent définie comme étant l'empreinte digitale d'un circuit électronique numérique.

Sources d'aléa statiques dans les circuits électroniques numériques. La source d'aléa liée à un processus physique, utilisée par les PUF dans les circuits électroniques numériques, vient de la disparité entre les transistors du circuit produite par les variations de procédés de fabrication. Selon le type de PUF, les variations locales sont combinées, comparées ou lues directement pour générer une sortie binaire. Étant donné que les variations des composants ne peuvent pas être contrôlées de l'extérieur, une PUF ne peut pas être répliquée. Ce fait la rend, en théorie, «non-clonable». La sortie d'une PUF, aussi appelée réponse (notée R), dépend d'un signal d'entrée, aussi appelé challenge (noté C). Par conséquent, une PUF est une «fonction». Dans le terme PUF, «Physique» signifie une entité physique, par opposition à un algorithme.

La première version de PUF a été introduite par Pappu en 2002 [5]. Dans cette version, qui est basée sur des phénomènes optiques et non électroniques, la sortie est produite en évaluant le motif d'interférence d'un support optique transparent.

Dans le cadre des PUF appliquées aux circuits électroniques, les principales approches sont :

- Les PUF à base d'anneaux oscillants : ils seront détaillées et étudiées dans ce manuscrit.
- Les PUF à base d'arbitre : ils comparent le délai entre deux chemins numériques supposés identiques [6]. Chaque comparaison produit un bit de la réponse de la PUF.
- Les PUF à base de mémoire volatile : Après la mise sous tension du circuit, les cellules de mémoire se stabilisent dans un état défini par les disparités entre les transistors composant la cellule [7]. Chaque cellule mémoire produit un bit de la réponse de la PUF.

Comme nous venons de le voir, les générateurs physiques d'aléa sont essentiels pour la sécurité. Il existe une grande diversité de TRNG et PUF, mais leur conception et leur évaluation restent complexes. Une étude sérieuse de ces générateurs physiques d'aléa est nécessaire. C'est ce qu'ont proposé de faire les membres du projet H2020 HECTOR.

Le projet HECTOR

Les travaux présentés dans cette thèse ont été effectués dans le cadre du projet européen de recherche H2020 HECTOR [3] (Hardware Enabled CrypTo and Randomness). La mission du projet HECTOR, d'une durée de 3 ans, a été de faire le lien entre des algorithmes cryptographiques mathématiquement fiables, les TRNG, les PUF et leur efficacité ainsi que leur résistance aux attaques matérielles une fois implantées sur un circuit électronique numérique. Le consortium créé au sein de ce projet visait à renforcer le potentiel d'innovation, de compétitivité et de leadership de l'Europe en matière de sécurité par le biais d'une collaboration entre les principaux acteurs européens des technologies de la sécurité venant de différents horizons.

Ainsi, les membres du projet regroupaient des compétences variées en matière de sécurité comme : la conception d'algorithmes de chiffrement et d'authentification, la conception de primitives cryptographiques (TRNG et PUF), leur intégration dans un système complet et leur évaluation.

Le consortium HECTOR était composé des membres suivants :

— Technikon Forschungs–und Planungsgesellschaft mbH, Austria
— Katholieke Universiteit Leuven, Belgium
— Université Jean Monnet Saint-Etienne, France
— Thales Communications & Security SAS, France
— STMicroelectronics Rousset SAS, France
— STMicroelectronics SRL, Italy
— Micronic AS, Slovakia
— Technische Universität Graz, Austria
— Brightsight BV, Netherlands

Toutes les publications effectuées dans le cadre du projet HECTOR sont fournies en accès libre sur la plateforme Zenodo [4].

Durant cette thèse de doctorat, j'ai travaillé en étroite collaboration avec Oto Petura, un autre doctorant travaillant dans le cadre du projet HECTOR au sein du laboratoire Hubert Curien de l'Université Jean Monnet. Sa thèse traite de la mise en place d'un cadre de développement

3. https://hector-project.eu/
4. https://zenodo.org/communities/hector/

complet pour les TRNG. Cette thèse est une contribution supplémentaire se focalisant plus particulièrement sur les TRNG et les PUF à base de cellules oscillantes. Les contributions de cette thèse sont détaillées ci-après.

Contributions scientifiques

Dans ce travail de thèse, nous proposons tout d'abord une approche rigoureuse d'implémentation et de comparaison de TRNG et PUF sur les circuits électroniques numériques, suivie d'une intégration au sein d'un système complet de ces générateurs physiques d'aléa. Ensuite, nous amorçons une démarche de modélisation des PUF afin d'améliorer l'évaluation de leur imprévisibilité. Nous réalisons aussi une étude complète de l'impact du phénomène de verrouillage sur les cellules oscillantes et les conséquences sur les générateurs physiques d'aléa. Enfin, nous démontrons la sensibilité d'un type particulier de PUF à une attaque par analyse électromagnétique.

Organisation du mémoire

Ces contributions sont détaillées à travers ce manuscrit décliné en cinq chapitres. Leur contenu est le suivant :

Chapitre 1 : État de l'art. Ce chapitre dresse un état de l'art des différents principes TRNG et PUF implantés sur les circuits numériques ainsi que le résultat de leur caractérisation et de leur comparaison. Il présente aussi un historique comparatif entre la manière d'évaluer l'imprévisibilité des TRNG et celle des PUF. Le chapitre introduit ensuite deux types de menaces sur cellules oscillantes et les conséquences sur les TRNG et les PUF : leur sensibilité au phénomène de verrouillage et l'impact d'une attaque par analyse électromagnétique. Enfin, ce chapitre met en lumière les manques dans l'état de l'art avant de détailler la structure des chapitres suivants.

Chapitre 2 : Conception et caractérisation de TRNG et de PUF à base de cellules oscillantes. Ce chapitre présente des travaux de conception, caractérisation et comparaison de PUF à base de cellules oscillantes implantées sur circuits logiques configurables — en anglais field-programmable gate array (FPGA). Le même travail a été réalisé, en collaboration avec Oto Petura, sur les TRNG. Nous réalisons ensuite l'intégration dans un système complet de TRNG et PUF sur FPGA. L'implantation d'une PUF et d'un TRNG au sein de deux démonstrateurs (un système portable de stockage de données sécurisées et un système de messagerie sécurisée) a été réalisée en collaboration avec les membres du projet HECTOR. L'intégration d'une PUF au sein d'un démonstrateur pour la protection des données de conception, réalisé avec Brice Colombier qui

était un autre doctorant travaillant au laboratoire Hubert Curien de l'Université Jean Monnet, a aussi été réalisée.

Chapitre 3 : Modélisation des caractéristiques probabilistes des cellules oscillantes exploitées par les PUF. Ce chapitre étudie la possibilité d'une approche de modélisation des cellules oscillantes exploitées par les PUF, similaire à celle déjà en place pour les TRNG, dans un objectif de standardisation de la méthode d'évaluation de l'imprévisibilité de celle-ci. Ici, l'approche de modélisation est mise en œuvre sur deux types de cellules oscillantes : les oscillateurs en anneaux classiques — en anglais ring oscillator (RO) — et les oscillateurs en anneaux à effet transitoire — en anglais transient effect ring oscillator (TERO).

Chapitre 4 : Étude de l'impact du verrouillage sur les cellules oscillantes. Ce chapitre traite de la sensibilité des cellules oscillantes au phénomène de verrouillage et de son impact sur les TRNG et les PUF. Les trois principaux types de cellules oscillantes sont étudiés : l'oscillateur en anneau classique, l'oscillateur en anneau à effet transitoire et l'oscillateur en anneau auto séquencé — en anglais self-timed ring (STR).

Chapitre 5 : Analyse électromagnétique des cellules TERO. Dans ce chapitre, la sensibilité d'un type particulier de cellule oscillante, le TERO, aux analyses par rayonnement électromagnétique (EM) et les vulnérabilités induites sur une PUF à base de TERO — en anglais transient effect ring oscillator based PUF (TERO-PUF) — sont démontrées. Il propose ensuite des solutions pour réduire les risques d'une analyse EM sur une TERO-PUF.

Conclusion et perspectives. Résume les principales contributions de ce manuscrit et établit une liste de suggestions pour de potentiels travaux futurs sur le sujet.

Chapitre 1

État de l'art

Dans l'introduction, nous avons vu le contexte d'utilisation des générateurs physiques d'aléa en cryptographie. Le but est de pallier les problèmes de sécurité liés au vaste déploiement des systèmes embarqués et du monde de l'IoT. Intéressons-nous à présent à la génération d'aléa dans les circuits numériques. Dans ce chapitre, sera présenté un état de l'art des TRNG et des PUF dans les circuits numériques.

Dans un premier temps, la partie 1.1 dresse un historique des évolutions scientifiques de l'évaluation des TRNG et soulève le manque d'une démarche similaire pour les PUF. Nous tentons d'amorcer cette démarche dans ces travaux de thèse.

Dans un second temps, dans la partie 1.2, nous introduisons les différents TRNG et les PUF numériques existants.

Ensuite, la partie 1.3 étudie la possibilité d'implantation des TRNG et des PUF dans les circuits numériques. Elle démontre qu'il manque une approche équitable et rigoureuse dans la littérature pour concevoir, caractériser, comparer et intégrer au sein d'un système complet les différents PUF et TRNG.

La partie 1.4 répertorie certaines vulnérabilités affectant les PUF et les TRNG basés sur des cellules oscillantes avant de proposer des pistes d'exploration futures.

Enfin, la partie 1.5 résume les différentes possibilités d'améliorations soulevées au cours de cet état de l'art et introduit les contributions de ce manuscrit.

1.1 Vers la standardisation des générateurs d'aléa

Un des enjeux majeurs dans le domaine des générateurs d'aléa appliqués à la cryptographie est de pouvoir s'assurer de leur qualité principalement en ce qui concerne l'imprévisibilité. La figure 1.1 exprime de manière comique, mais claire, cette problématique.

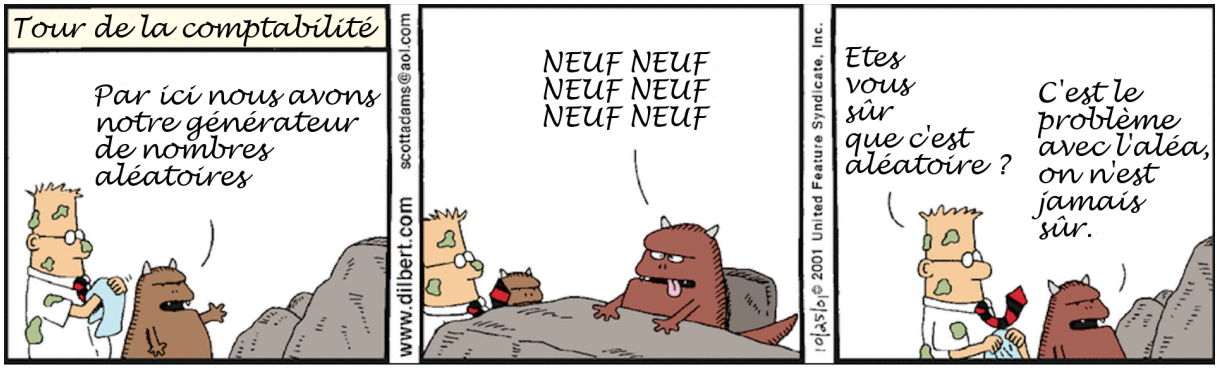


FIGURE 1.1: Comment assurer l'imprévisibilité d'un générateur d'aléa ? [1]

S'il est vrai qu'il n'est pas possible d'être absolument certain de l'imprévisibilité d'un générateur d'aléa, nous allons voir dans cette partie qu'il existe toutefois des moyens pour s'en approcher. De nombreuses études et avancées scientifiques sur le sujet ont été réalisées dans le domaine des TRNG. Les PUF, quant à elles, en raison de leur invention récente, ne jouissent pas encore de mêmes avancées. Intéressons-nous d'abord à l'évolution au cours du temps de la manière d'évaluer la sécurité des TRNG pour ensuite la comparer à celle des PUF.

1.1.1 Évaluation de la sécurité des TRNG

1.1.1.1 Méthode d'évaluation historique

Historiquement, lors du processus de développement, d'évaluation et de certification, le principe de TRNG et son implantation étaient évalués statistiquement. La suite de nombres générée était testée à l'aide d'une série de tests statistiques standards tels que les tests FIPS 140-1 [8], NIST SP 800-22 [9], DIEHARD [10] ou DIEHARDER [11]. Si le générateur ne passait pas avec succès la série de tests, le générateur n'était pas certifié. Ces tests statistiques sont des outils permettant de détecter si une suite est issue d'un générateur idéal, c'est-à-dire s'il s'agit d'une suite de nombres indépendants et uniformément distribués.

Les tests FIPS 140-1 nécessitent une séquence de 20 000 bits et sont composés de 4 tests (définis ci-après) : Monobit, Poker, Runs et Long runs. La norme NIST SP 800-22 en propose

1. https://dilbert.com/strip/2001-10-25

quant à elle 15 et nécessite 1 milliard de bits. Les 15 tests DIEHARD sont comparables à la norme NIST et nécessitent un minimum de 80 millions de bits.

Ainsi l'approche de conception historique d'un TRNG est comme représentée sur la figure 1.2. Le TRNG est conçu directement avec un algorithme de post-traitement chargé d'améliorer la qualité statistique des nombres générés et la séquence enregistrée est testée hors-ligne sur un nombre grand, mais limité de données. La qualité des nombres générés avant post-traitement ou encore l'impact des variations environnementales sur le système n'est pas pris en compte.

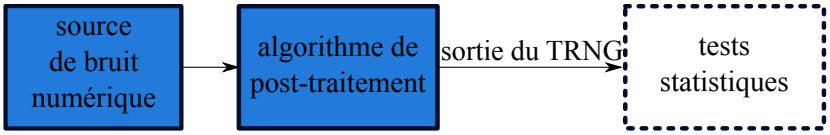


FIGURE 1.2: Approche de conception historique d'un TRNG

Bien que cette approche permette de détecter des faiblesses statistiques à un instant donné sur certains TRNG, les tests statistiques ne sont pas suffisants pour garantir la qualité d'un TRNG notamment en matière d'imprévisibilité. Les suites de nombres testées étant de taille finie, il est mathématiquement possible que les tests se trompent dans un sens comme dans l'autre. À savoir, un TRNG idéal ne réussi pas les tests ou un TRNG défectueux réussi les tests.

En effet, le post-traitement peut masquer de manière considérable les faiblesses de la source d'aléa. De plus, ces tests sont complexes et nécessitent un grand nombre de données ce qui les rend lents et coûteux. Par conséquent, ils ne sont exécutés qu'occasionnellement et sur un nombre limité de données.

L'imprévisibilité des nombres générés par un TRNG est essentielle pour les applications cryptographiques. Face à l'inefficacité des tests statistiques à garantir cette propriété, une autre approche d'évaluation a été mise en place par la communauté scientifique. Elle repose sur la compréhension et la modélisation du fonctionnement du TRNG et en particulier de sa source d'aléa.

Dans cette nouvelle approche, les tests statistiques sont toujours requis, mais ne sont pas suffisants à eux seuls.

1.1.1.2 Méthode d'évaluation moderne

Estimation de l'entropie Comme nous venons de le voir, la sécurité d'un TRNG visant des applications cryptographiques repose sur l'imprévisibilité des nombres générés. L'imprévisibilité est caractérisée par l'entropie. L'entropie est définie comme le degré de désorganisation. Pour garantir une imprévisibilité de la suite générée, le taux d'entropie d'un vecteur de n bits aléatoires doit être le plus proche possible de n.

Plusieurs définitions de l'entropie existent. La plus générale est l'entropie de Rényi [12]. Étant donné une variable aléatoire discrète X à n valeurs possibles $(x_1, x_2, \ldots x_n)$, elle est définie comme suit :

$$H_\alpha(X) = \frac{1}{1-\alpha} \log_2 \left(\sum_{i=1}^{n} p_i^\alpha \right) \tag{1.1}$$

avec α un paramètre réel ≥ 0 et $\neq 1$ et $p_i = Pr[X = x_i]$ (la probabilité que $X = x_i$).

L'entropie de Rényi est généralement difficile à estimer en pratique. À la place, deux cas particuliers de l'entropie de Rényi sont utilisés : la min-entropie et l'entropie de Shannon.

La min-entropie est la mesure la plus conservatrice. Elle est basée sur le fait que l'entropie de Rényi décroît de manière monotone avec α. Par conséquent, elle atteint sa valeur minimale (sa min-entropie) pour $\alpha \to \infty$. La min-entropie est donc définie comme suit :

$$H_\infty(X) = \inf_{i=1\ldots n} (-\log_2(p_i)) = -\log_2 \sup_{i=1\ldots n} p_i \tag{1.2}$$

Le conservatisme de la min-entropie peut être utile pour garantir la sécurité, mais sa définition présentée dans l'équation 1.2 n'est valide que pour des variables indépendantes.

La définition de l'entropie la plus communément admise est l'entropie de Shannon. Elle est obtenue grâce à l'entropie de Rényi pour $\alpha \to 1$:

$$H_1(X) = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{1.3}$$

Tout comme la min-entropie, l'équation 1.3 n'est valide que si les nombres générés sont indépendants. À défaut, l'entropie conditionnelle basée sur l'entropie de Shannon peut être utilisée [13].

La probabilité d'un vecteur de n bits ayant une distribution uniforme est proche de $Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = 1/2^n$. Ainsi, comme évoqué précédemment, l'entropie de ce vecteur est proche de n. Par conséquent, l'entropie par bit du TRNG est proche de 1 (selon la norme AIS31 [13], $H_1(X)$ doit être supérieure à 0,997). Un haut taux d'entropie garantit que les bits précédents et suivants ne peuvent être devinés avec une probabilité autre que 0,5.

Cependant, l'entropie est une propriété des variables aléatoires et non des valeurs prises par ces variables aléatoires. L'entropie ne peut pas être directement mesurée sur une suite de nombres générés. D'un point de vue strictement mathématique, l'entropie doit être estimée à l'aide d'un modèle stochastique du générateur.

Il existe deux normes qui suivent cette approche. La première est la norme américaine SP 800-90B [14] développée par le NIST [2]. Elle indique que pour la certification d'un TRNG, une documentation doit justifier de l'entropie revendiquée et peut inclure un modèle stochastique. La

2. National Institute of Standards and Technology

seconde est une norme allemande AIS31 [13] développée par la BSI [3]. Elle impose aux concepteurs de construire un modèle stochastique et de l'utiliser pour estimer l'entropie.

En France, l'ANSSI [4] applique la norme AIS31 pour la certification des TRNG. Pour cette raison, dans la suite de ce manuscrit, nous nous focaliserons sur la norme AIS31.

Standard AIS31

Modélisation du TRNG Un modèle stochastique définit une famille de lois de probabilité qui contient toutes les distributions possibles des nombres aléatoires générés. Il peut inclure ou non la fonction de post-traitement. L'objectif principal du modèle stochastique est de caractériser la probabilité qu'un bit de sortie soit égal à 1 ($Pr(X = 1)$), et/ou la probabilité qu'un vecteur de n bits comporte un motif ($Pr(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)$) et ainsi en déduire une estimation de l'entropie (si les variables sont indépendantes et identiquement distribuées [IID]) ou l'entropie conditionnelle (si les variables ne sont pas IID) des valeurs de sortie.

À la construction ou vérification d'un modèle stochastique, le concepteur ou l'évaluateur doit :

— vérifier si les nombres aléatoires sont stationnaires (c'est-à-dire leurs paramètres statistiques ne changent pas dans le temps) ;
— vérifier la distribution des nombres aléatoires dans différentes conditions limites (tension, température, *etc.*) pour s'assurer que le TRNG est résistant aux variations de conditions opérationnelles ;
— vérifier si les variables aléatoires sont IID ou non ;
— caractériser la distribution des nombres aléatoires générés.

L'utilité du modèle stochastique est double. Il sert premièrement à caractériser l'imprévisibilité des nombres générés (estimation de l'entropie). Si les nombres générés sont IID, la min-entropie ou l'entropie de Shannon peuvent être utilisées. Sinon, l'entropie conditionnelle doit être utilisée. Il sert aussi de base pour la construction de tests spécifiques au TRNG.

Tests statistiques La norme AIS31 propose une série de 9 tests permettant de vérifier la qualité statistique des nombres générés. Ils sont résumés dans le tableau 1.1. Pour chacun d'entre eux, la dernière colonne explique brièvement la propriété statistique vérifiée.

La norme AIS31 recommande de réaliser ces tests dans des conditions environnementales cohérentes à l'utilisation du circuit.

3. Bundesamt für Sicherheit in der Informationstechnik
4. Agence Nationale de la Sécurité des Systèmes d'Information

Test	Fonction
Test d'indépendance	
T0 —Disjointness test	2^{16} blocs de 48 bits doivent être différents
T1 —Monobit test	Test le biais sur la séquence générée
T2 – Poker test	Test l'indépendance de la séquence
T3 – Runs test	Compte le nombre de séries courtes de 0 successifs (et de 1)
T4 – Long runs test	Une série de 34 bits ne doit pas être composée seulement de 0 (ou 1)
T5 – Correlation test	Test la corrélation entre les bits successifs
Test de distribution uniforme	
T6 – Uniform distribution test	Test l'uniformité de la distribution
T7 – Test for homogeneity	Test l'homogénéité de la séquence
Estimation de l'entropie	
T8 – Coron's entropy test	Estime l'entropie de la séquence à l'aide du test de Maurer

TABLE 1.1: Suite de tests de la norme AIS31

Tests embarqués La norme AIS31 inclut aussi des tests dits "en ligne" avec le TRNG pour détecter les changements dynamiques hors des bornes acceptables en matière de qualité des nombres générés. Les tests "en ligne" sont des tests statistiques embarqués, spécifiques au principe de TRNG, basés sur le modèle stochastique. Trois types de tests embarqués dédiés sont définis par la norme :

— Test(s) d'échec total ;
— Test(s) en ligne ;
— Test(s) au démarrage.

Le test d'échec total est un test grossier qui doit être en mesure de détecter rapidement une déviation importante du TRNG de son fonctionnement normal (*ex.* détecter un défaut d'entropie). S'il détecte ce genre de défaut, le TRNG doit immédiatement arrêter de générer des nombres. Étant donné que chaque déclenchement d'une alarme par le test d'échec total va entrainer le redémarrage du TRNG, la probabilité pour ce test de déclencher une fausse alarme doit être faible. À défaut, dans le pire des cas, c'est-à-dire quand l'alarme se déclenche trop souvent, le TRNG pourrait être désactivé en permanence. La meilleure façon d'assurer la rapidité et la fiabilité de ce test est qu'il soit basé sur le modèle stochastique afin d'évaluer les propriétés statistiques les plus importantes du TRNG.

Le test en ligne est plus précis que le test d'échec total et sert à détecter des défaillances statistiques. Son temps d'exécution est plus long que le test d'échec total. Par conséquent, l'intervalle de temps entre l'apparition réelle de la défaillance et le déclenchement d'une alarme est aussi plus long. Pendant cet intervalle de temps, le fonctionnement correct du TRNG doit être assuré par une solution de secours (*ex.* le TRNG a accumulé assez d'entropie au préalable pour la redistribuer depuis un registre interne). Tout comme pour le test d'échec total, le test en ligne

sera plus efficace s'il est basé sur le modèle stochastique.

Le test au démarrage, comme son nom l'indique, est lancé à l'initialisation du TRNG et après chaque déclenchement d'alarme par les autres tests. Il est responsable de tester le bon fonctionnement des deux autres tests, puis il vérifie le comportement de la source d'aléa et l'algorithme de post-traitement.

Classes Nous venons de présenter les exigences requises par la norme AIS31 pour qu'un TRNG soit certifié. Il existe plusieurs classes de certification d'un TRNG définies par L'AIS31. Selon le niveau de satisfaction du TRNG à ces exigences, une classe de certification lui sera attribuée. Les trois classes de la norme AIS31 sont :

— PTG-1 : Test d'échec total de la source d'aléa et test en ligne de la sortie du TRNG ;
— PTG-2 : PTG-1 + test en ligne avant post-traitement et modèle stochastique du TRNG ;
— PTG-3 : PTG-2 + post-traitement de la sortie du TRNG par un algorithme cryptographique.

Elles conditionnent le type d'application cryptographique du TRNG. La classe PTG-1 est la moins exigeante. Les TRNG de cette classe pourront être utilisés pour des applications cryptographiques où l'imprévisibilité de la séquence générée n'est pas requise. Pour cette raison, elle ne requiert pas d'estimation d'entropie ni de modèle stochastique. Seule la qualité statistique des nombres générés sera vérifiée à l'aide des tests définis dans la section 1.1.1.2. La classe PTG-2 certifie des TRNG qui pourront être utilisés pour des applications cryptographiques nécessitant l'imprévisibilité des nombres générés (*ex.* générer des clés de chiffrement). Ainsi, elle requiert un modèle stochastique et une estimation de l'entropie. La classe PTG-3 certifie les TRNG pour n'importe quelle application cryptographique. Elle requiert, en plus de la classe PTG-2, un algorithme de post-traitement de type cryptographique permettant d'assurer pendant un certain temps la qualité statistique de la suite générée en cas de mise en défaut du TRNG.

Pour les niveaux de certification PTG-2 et PTG-3, pour lesquels l'imprévisibilité est essentielle, le concepteur doit justifier d'une entropie par bit en sortie du TRNG d'un minimum de 0,997.

Approche de sécurité étendue Dans une approche plus étendue, conforme à la norme AIS31, Fischer dans [15], propose une surveillance de la source d'aléa avant numérisation.

Pour résumer, les TRNG visant des applications cryptographiques doivent se conformer au standard AIS31, ce qui inclut, dans sa version la plus approfondie, les exigences suivantes :

— leur design doit être compréhensible et simple avec une source d'aléa clairement définie et identifiée ;

— le processus aléatoire sous-jacent doit être stationnaire et un modèle stochastique visant à estimer l'entropie doit être réalisable ;

— la suite binaire en sortie du TRNG sans post-traitement doit être disponible pour des tests temps réel embarqués ainsi que des tests hors-ligne ;

— le rôle d'un algorithme de post-traitement est redéfini. Il devrait corriger de légères imperfections statistiques occasionnelles sur la suite binaire en sortie du TRNG et éventuellement augmenter l'entropie par bit généré par méthode de compression ;

— un algorithme de post-traitement de type cryptographique est nécessaire pour assurer l'imprévisibilité des nombres générés en cas de défaillance de la source d'aléa.

La figure 1.3 représente l'approche moderne de conception d'un TRNG.

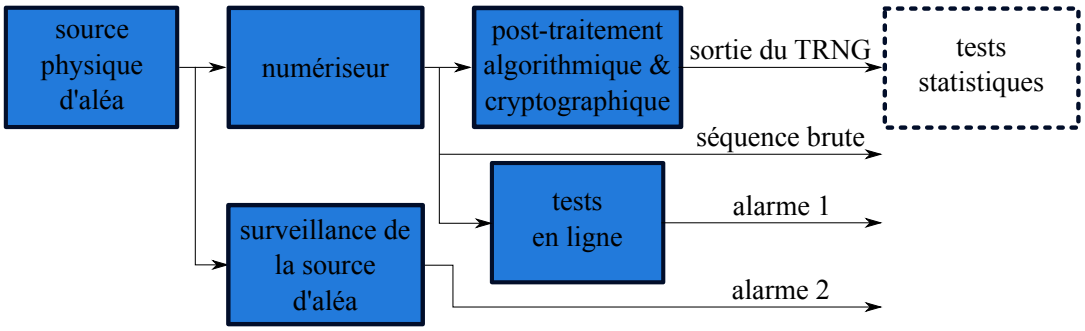


FIGURE 1.3: Approche de conception moderne d'un TRNG

1.1.2 Évaluation de la sécurité des PUF

Nous venons de voir avec les TRNG que deux propriétés importantes des générateurs utilisés pour des applications cryptographiques devaient être vérifiées : la qualité statistique et l'imprévisibilité des nombres générés. Intéressons-nous à présent à la façon d'évaluer la qualité d'une PUF. Pour rappel, la réponse d'une PUF va servir d'identifiant unique à un circuit. Ainsi, elle doit être :

— Stable : la réponse R_i à un challenge C d'une PUF sur un circuit i doit toujours être la même ;

— Unique : les réponses R_i et R_j des circuits i et j au même challenge C doivent être différentes ;

— Imprévisible : il ne doit pas être possible de prédire la réponse R_i d'une PUF sur un circuit i à un challenge C, et ce même en ayant connaissance des réponses de cette PUF sur d'autres circuits à ce même challenge C.

Ainsi pour caractériser ces propriétés, la méthode communément admise, à l'instar de l'approche historique d'évaluation des TRNG, est de mesurer les propriétés statistiques des PUF à

l'aide de plusieurs paramètres de mesure présentés ci-après. La figure 1.4 représente l'approche d'évaluation classique des PUF.

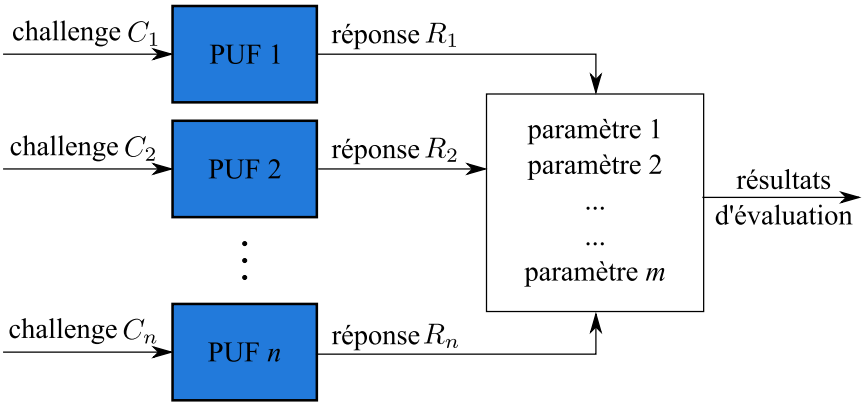


FIGURE 1.4: Méthode d'évaluation classique de PUF

Les différents paramètres d'évaluation sont décrits dans les paragraphes suivants.

1.1.2.1 Stabilité

La réponse d'une PUF doit être stable pour une zone de fonctionnement prédéfinie. Une instabilité se produit si certains bits de la réponse changent entre deux appels successifs au même challenge sur un même circuit. Cette instabilité est mesurée à l'aide de la distance de Hamming — en anglais Hamming distance (HD).

La distance de Hamming tire son nom du mathématicien Richard W. Hamming. C'est un paramètre de la théorie de l'information qui donne le nombre d'éléments différents entre deux vecteurs binaires de même taille. Elle est déterminée en faisant un OU-EXCLUSIF logique — en anglais Exclusive OR (XOR) — entre les deux vecteurs puis la somme des éléments du résultat (c'est-à-dire la somme des bits composant le vecteur de résultat). La table 1.2 montre un exemple de HD sur deux vecteurs binaires de 6 bits :

vecteur1 :	**0**	0	1	**0**	1	**1**	$\oplus$
vecteur2 :	**1**	0	1	**1**	1	**0**	
XOR :	**1**	0	0	**1**	0	**1**	$\rightarrow \sum = 3$

TABLE 1.2: Exemple de distance de Hamming

Ainsi, nous noterons :

$$HD(\text{vecteur1, vecteur2}) = \sum_{k=1}^{6} (\text{vecteur1}_k \oplus \text{vecteur2}_k) = 3 \rightarrow HD(\text{vecteur1, vecteur2})_{\%} = 50\%$$

Pour déterminer l'instabilité, on calcule la distance de Hamming entre la réponse d'une PUF et sa réponse de référence (une moyenne de ses réponses successives au même challenge). Nous

l'appellerons HD_{int}, signifiant la distance de Hamming interne au circuit. La mesure de l'instabilité définie dans l'équation 1.4 quantifie les changements à la sortie d'une PUF sur de nombreuses mesures.

$$HD_{int} = \frac{1}{Y} \sum_{y=1}^{Y} \frac{HD(R_{i,y}, \overline{R_i})}{K} \times 100\% \tag{1.4}$$

où Y est le nombre total de réponses, K est le nombre de bits d'une réponse, $R_{i,y}$ est la réponse y du circuit i et $\overline{R_i}$ est la réponse de référence du circuit i provenant de la moyenne des échantillons $R_{i,y}$ aux conditions nominales de fonctionnement. Une implantation de PUF parfaitement stable aura une HD_{int} de 0 %. Cependant, comme tous les circuits électroniques, une PUF est affectée par le bruit, les variations de température ou encore de tension et sa HD_{int} sera supérieure à 0 %. Bien qu'il soit assez rare que les PUF aient une bonne stabilité en pratique, c'est aussi le paramètre le plus facile à quantifier statistiquement de manière précise. En effet, il est possible de générer autant de fois que nécessaire la réponse d'une PUF à un challenge.

1.1.2.2 Unicité

Si le design de la PUF n'est pas fait avec attention, il peut arriver que les bits de la réponse dépendent de leur position sur le circuit plutôt que des variations de procédés de fabrication. Lorsque c'est le cas, certains bits de la réponse vont tendre vers une certaine valeur et ceci de la même façon pour tous les circuits. Ce phénomène pose des problèmes de sécurité puisque connaitre la réponse d'un circuit aidera un attaquant à deviner la réponse des autres circuits. Comme pour la mesure de l'instabilité, la corrélation entre les réponses de différents circuits est mesurée avec la distance de Hamming. Nous l'appellerons HD_{ext}, signifiant la distance de Hamming entre les circuits. Si la valeur moyenne de HD_{ext} entre tous les circuits est de 50 %, cela signifie qu'aucune corrélation entre les circuits n'existe.

La mesure de l'unicité définie dans l'équation 1.5 montre à quel point les réponses générées par la PUF sur différents circuits sont uniques. Cette mesure a été introduite pour la première fois par Lee *et al.* dans [6]. Ensuite, elle est mise en équation par Maiti *et al.* dans [16]. Soit deux circuits i et j, et leurs réponses respectives $R_{i,y}$ et $R_{j,y}$, la valeur moyenne de HD_{ext} pour un ensemble de I circuits générant Y réponses est définie comme suit :

$$HD_{ext} = \frac{1}{I(I-1)Y} \sum_{i=1}^{I} \sum_{\substack{j=1 \\ j \neq i}}^{I} \sum_{y=1}^{Y} \frac{HD(R_{i,y}, \overline{R_j})}{K} \times 100\% \tag{1.5}$$

où $\overline{R_j}$ est la réponse de référence du circuit j provenant de la moyenne des échantillons $R_{j,y}$ aux conditions nominales de fonctionnement.

Avec une quantité de données suffisante, la distribution d'une HD_{ext} parfaite converge vers une distribution binomiale de paramètres n le nombre d'expériences et de probabilité $p = 0,5$ où son espérance est $\mu = np = 50\%$ et sa variance $\sigma^2 = np(1-p) = 25\%$. Si μ est différent de 50 %, cela signifie soit qu'il y a une corrélation entre les réponses de différents circuits, soit que certaines réponses comportent un biais statistique. Si σ^2 est supérieur à 25 %, cela indique aussi une corrélation entre les réponses de différents circuits. C'est pourquoi il est important de regarder toute la distribution de HD_{ext} et pas seulement sa valeur moyenne.

1.1.2.3 Imprévisibilité

Biais Initialement, l'imprévisibilité des PUF n'était évaluée qu'avec la mesure du biais statistique, également introduite dans [16]. Cette approche repose sur le fait que la probabilité des bits de réponses des PUF de prendre la valeur '1' logique ou '0' logique doit être égale. À défaut, les bits de la réponse peuvent être devinés avec une probabilité autre que 0,5. Cette mesure de biais est définie dans l'équation 1.6 :

$$\overline{b} = \frac{1}{K}\sum_{k=1}^{K} b_k \tag{1.6}$$

où $\overline{b}$ représente le biais, K le nombre de bits évalués et b_k le bit numéro k. $\overline{b}$ est compris entre 0 et 1. Un $\overline{b}$ idéal sera égal à 0,5 signifiant que les bits ont la même probabilité de prendre la valeur '1' logique ou '0' logique. À l'inverse, un $\overline{b}$ égal à 0 (ou 1) signifie que tous les bits des réponses sont à 0 (ou 1) sur tous les circuits. Cette mesure de biais est effectuée à la fois entre les bits d'une même réponse, et entre les réponses de plusieurs circuits. Il arrive parfois, comme dans [16], qu'elle soit divisée en deux mesures distinctes : biais entre les bits de la réponse et biais du bit numéro k entre plusieurs réponses de plusieurs circuits.

D'autres études ([17], [18], [19]) présentent des paramètres de caractérisation très similaires à ceux présentés ci-avant.

En 2013, Maiti *et al.* proposent une étude regroupant tous les paramètres extraits de ces différentes études pour les uniformiser et amorcer une méthode ordonnée d'évaluation et de comparaison des PUF [20]. Dans cette étude, une PUF à base d'arbitre est comparée à une PUF à base de RO.

Par la suite, dans une étude sur les métriques pour quantifier le caractère unique des PUF, Feiten *et al.* démontrent la redondance entre la mesure d'unicité (HD_{ext}) et la mesure du biais entre plusieurs réponses de plusieurs circuits [21].

Corrélation En plus d'être identiquement distribués, les bits générés par une PUF doivent, tout comme les TRNG, être indépendants (IID). En effet, afin d'éviter des problèmes de sécurité, deux

cellules voisines composant une PUF ne doivent pas s'influencer mutuellement. Une corrélation entre ces cellules réduirait le nombre de combinaisons possibles et ainsi augmenterait le risque d'une attaque par force brute. Afin de vérifier s'il existe une corrélation entre les bits générés Böhm *et al.* définissent dans [22] une mesure d'autocorrélation comme suit :

$$R_{bb}(j) = \sum_{k=1}^{K} corr(b_k b_{k-j}) \tag{1.7}$$

où R_{bb} est évalué avec un décalage j, avec

$$corr(b_k b_{k-j}) = \begin{cases} 1, & si\ b_k = b_{k-j} \\ -1, & sinon \end{cases} \tag{1.8}$$

Ainsi, si $R_{bb}(j) = 1$ ou $R_{bb}(j) = -1$ les bits sont complètement corrélés. À l'inverse, pour $R_{bb}(j) = 0$, les bits ne sont pas du tout corrélés.

Récemment, Wilde *et al.* ont réalisé une étude approfondie sur la corrélation des PUF et proposent une nouvelle méthode d'analyse d'autocorrélation spatiale [23].

Estimation de l'entropie À l'image de Maiti *et al.*, Pehl *et al.* dans [24] initient une approche ordonnée pour évaluer la qualité des PUF. Leur étude regroupe les différentes mesures de biais et de corrélation et propose une estimation de l'entropie à l'aide de l'entropie de Shannon (définie dans l'équation 1.3). Ils démontrent également que des défauts de conception peuvent justifier de mauvais résultats statistiques.

Tests statistiques D'autres études proposent d'utiliser les mêmes tests statistiques standards que les TRNG (c'est-à-dire NIST SP 800-22 ou AIS31) pour évaluer la qualité des PUF [25], [26].

1.1.2.4 Conclusion

Nous venons de voir qu'un nombre important de travaux pour quantifier la qualité des PUF ont été réalisés. Dans toutes ces études, seule la qualité statistique est évaluée. Face à cette approche, plusieurs problèmes se posent. Tout d'abord, l'évaluation de la qualité statistique d'une PUF est bien plus complexe que celle d'un TRNG. Alors qu'un TRNG peut générer un nombre important de données, l'évaluation des PUF nécessite un nombre important d'implantations de PUF. Autrement dit, l'évaluation des PUF nécessite un nombre important de circuits. Si nous prenons par exemple le cas du test NIST SP 800-22, il faudrait un milliard d'implantations de PUF. C'est une chose qui ne semble pas réalisable en pratique.

De plus, nous avons montré dans la section 1.1.1.1 l'inefficacité des tests statistiques à garantir l'imprévisibilité. Bien que la qualité statistique des PUF soit une condition nécessaire pour l'utilisation de celles-ci dans des applications cryptographiques, elle n'est pas suffisante. C'est

pourquoi les méthodes d'évaluation présentées ci-avant doivent servir à identifier des faiblesses dans l'implantation ou le principe de la PUF, mais ne permettent pas de garantir l'imprévisibilité.

Enfin, comme l'a démontré la communauté scientifique des TRNG ([13], [15]), la meilleure manière de garantir l'imprévisibilité d'un générateur physique d'aléa est d'estimer son entropie en se basant sur un modèle stochastique. Les efforts vers une évaluation et une certification standardisée des PUF ne font que commencer et prennent du temps. Ceci est dû au fait que, tout comme les TRNG, les PUF ne reposent pas seulement sur leur principe, mais aussi sur leur implantation physique propre à la technologie. Pour le moment, aucun standard semblable à l'AIS31 n'existe pour les PUF.

Nous pensons qu'une démarche scientifique dans ce sens est essentielle pour la pérennité des PUF et leur intégration solide dans le monde de l'industrie. C'est ce que nous tentons d'amorcer dans le chapitre 3.

À travers cette section, nous avons vu les enjeux et l'importance d'une approche d'évaluation standardisée des TRNG et des PUF. Nous avons aussi vu que la complexité d'une telle approche repose sur le fait que, contrairement aux autres primitives cryptographiques, les TRNG et les PUF ne dépendent pas seulement de leur principe, mais aussi de la façon dont ils sont implantés. Intéressons-nous donc à présent aux différents principes de générateurs physiques d'aléa pour les circuits numériques, puis à leurs implantations existantes dans la littérature.

1.2 Générateurs physiques d'aléa dans les circuits électroniques numériques

Le besoin de nombres aléatoires n'est pas récent. Toutes les civilisations au cours de l'histoire disposaient d'une méthode pour générer des nombres aléatoires pour diverses raisons (jeu, prise de décision, *etc.*). À tel point que l'homme inventa les générateurs de nombres aléatoires avant les symboles représentant les nombres eux-mêmes [27]. En effet, des dés d'environ 5000 ans ont été retrouvés en Irak et en Iran. Ils étaient également populaires en Inde et en Chine il y a au moins 4000 ans. De nos jours, lancer des dés et noter le résultat comme le faisait Galton en 1890 serait évidemment beaucoup trop lent [28]. Heureusement, l'apparition de l'électronique a rendu la génération d'aléa beaucoup plus rapide. Depuis, un nombre considérable, que nous nous abstiendrons de citer dans ce manuscrit, de générateurs ont été inventés. Étant donné le contexte de cette étude, nous nous focaliserons exclusivement sur les générateurs d'aléa dans les circuits électroniques numériques. Dans la partie suivante, nous réalisons un état de l'art des TRNG et des PUF numériques classés selon leurs différentes approches, listées dans l'introduction.

Un nombre important de ces générateurs physiques d'aléa exploitent les cellules oscillantes comme source d'aléa. Ainsi, avant de réaliser un état de l'art des principes de TRNG et PUF, le fonctionnement détaillé de ces cellules est décrit.

1.2.1 Cellules oscillantes pour la génération d'aléa dans les circuits numériques

Toutes les cellules oscillantes étudiées dans ce manuscrit pour des applications TRNG ou PUF sont composées de deux éléments logiques de base : M portes d'activation utilisées pour déclencher les oscillations et N portes de délai utilisées pour régler la fréquence d'oscillation. Les éléments composant la cellule sont connectés en série et la sortie du dernier élément est rebouclée à l'entrée du premier élément pour former un anneau. Le nombre de portes d'activation et de portes de délai dépend du type de cellule et de sa configuration.

Un changement d'état logique (un front montant) à l'entrée de la/des porte(s) d'activation démarre les oscillations. Selon le type de cellule, un ou plusieurs événement(s), noté e et produit par le changement d'état logique à l'entrée, se propagent à travers l'anneau. Une fois la cellule déclenchée, les événements e peuvent se propager de manière permanente, si aucune collision entre eux ne se produit, ou de manière temporaire, si un événement en rattrape un autre, provoquant une collision qui va arrêter les oscillations.

La période d'oscillation du signal produit en sortie de la cellule est égale à deux fois la somme des délais de chaque élément de la cellule divisée par le nombre d'événements à l'intérieur de l'anneau :

$$T_{osc} = \frac{2 \times (M \times d_m + N \times d_n)}{e} \tag{1.9}$$

où d_m représente le délai moyen des portes d'activation et d_n le délai moyen des portes de délai, M et N représentent respectivement le nombre de portes d'activation et le nombre de portes de délai.

1.2.1.1 Oscillateur en anneau

Le RO est une cellule oscillante à un seul événement. Il est composé d'un nombre impair N de portes inverseurs logiques (portes de délai) et d'une porte ET-logique utilisée comme porte d'activation ($M = 1$). La figure 1.5 montre l'architecture d'un RO.

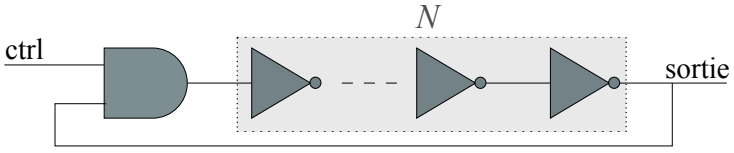


FIGURE 1.5: Architecture d'un oscillateur en anneau classique (RO)

Lorsque le signal de contrôle, dénoté *ctrl* sur la figure, passe d'un état logique bas à un état logique haut (front montant), les oscillations commencent. À tout moment, un seul événement électrique (front montant ou front descendant) se propage à travers la cellule ($e = 1$). En traversant un inverseur logique, un front montant est transformé en front descendant et vice-versa.

Ainsi, la fréquence d'oscillation en sortie d'un RO est :

$$f_{osc} = \frac{1}{2 \times (d_{ET} + N \times d_{INV})} \tag{1.10}$$

où d_{ET} représente le délai de la porte ET-logique et d_{INV} le délai moyen des inverseurs logiques.

1.2.1.2 Oscillateur en anneau à effet transitoire

Le TERO est une cellule oscillante à événements multiples avec collision. Elle a deux états distincts : un état transitoire d'oscillation et un état stable de non-oscillation.

Comme représenté dans la figure 1.6, la cellule est composée de deux chaines de N portes inverseurs logiques (portes de délai), avec N impair, et deux portes ET-logique utilisées comme portes d'activation ($M = 2$).

La cellule TERO correspond ainsi à une configuration particulière d'une bascule RS caractérisée par des délais supplémentaires dans les deux branches de la bascule [29].

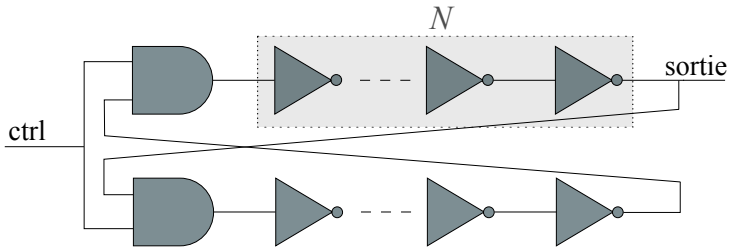


FIGURE 1.6: Architecture d'un oscillateur en anneau à effet transitoire (TERO)

Lorsque le signal de contrôle, dénoté *ctrl* sur la figure 1.6, produit un front montant, deux événements électriques vont se propager au sein de la cellule ($e = 2$). À cause des disparités entre les transistors CMOS composant la cellule, produites par les variations de procédés de fabrication, un événement est plus rapide que l'autre. Par conséquent, alors que la fréquence d'oscillation du signal en sortie de la cellule reste constante, son rapport cyclique évolue en direction de 0 % ou 100 % jusqu'à l'arrêt des oscillations. La figure 1.7 montre un exemple d'oscillation temporaire d'un TERO.

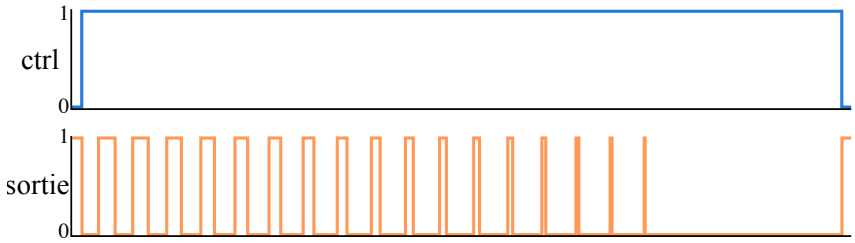


FIGURE 1.7: Comportement du signal de sortie d'un TERO (*sortie*) après déclenchement grâce au signal de contrôle (*ctrl*)

La fréquence d'oscillation en sortie d'un TERO est :

$$f_{osc} = \frac{1}{2 \times (d_{ET} + N \times d_{INV})} \tag{1.11}$$

où d_{ET} représente le délai moyen des portes ET-logique et d_{INV} le délai moyen des portes inverseurs logiques.

1.2.1.3 Oscillateur en anneau auto séquencé

Le STR est une cellule oscillante à événements multiples sans collision. La figure 1.8 représente une cellule STR. Chaque étage du STR est composé d'une porte de Muller à deux entrées [30] et d'une porte inverseur logique. Chacun de ses étages sert à la fois de porte d'activation et de porte de délai. Les étages du STR communiquent entre eux à l'aide d'un protocole de négociation en deux phases.

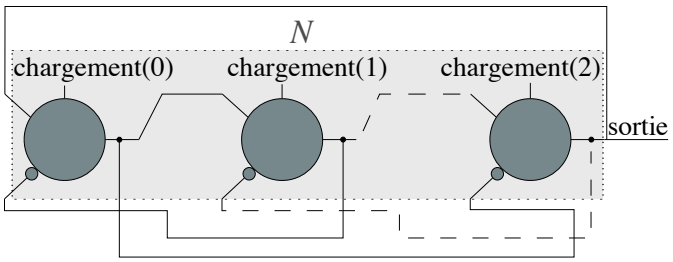


FIGURE 1.8: Architecture d'un oscillateur en anneau auto séquencé (STR)

Plusieurs événements peuvent se propager au sein de la cellule sans collision grâce à ce protocole de négociation. La cellule est initialisée avec e événements électriques. Un événement présent à un étage ira au prochain étage si, et seulement si, l'étage suivant est vide (c'est-à-dire s'il ne contient pas d'événement). Pour cette raison, le STR doit avoir au minimum un étage vide pour pouvoir osciller. Cela signifie que jusqu'à $N - 1$ événements peuvent se propager à travers une cellule composée de N étages. Indépendamment de leurs positions initiales et grâce à des mécanismes analogiques influençant la cellule, les événements se retrouvent soit en un groupe qui se propage dans l'anneau (en mode rafale), soit réparties autour de l'anneau de manière équidistante et se propagent avec un espacement temporel constant (en mode régulièrement espacé).

S'il y a plus d'événements que d'étages vides dans la cellule ($e > N/2$), la fréquence d'oscillation en sortie du STR est limitée par le nombre d'étages vides.

La fréquence d'oscillation en sortie d'un STR est :

$$f_{osc} = \frac{a}{N \times d_{MULLER}} \tag{1.12}$$

où a représente le nombre d'événements (quand $e < N/2$) ou le nombre d'étages vides (quand $e > N/2$), et d_{MULLER} le délai moyen d'un étage du STR.

Intéressons-nous à présent aux différents principes de TRNG existants.

1.2.2 Principes de TRNG dans les circuits électroniques numériques

1.2.2.1 TRNG exploitant le jitter d'horloge

Le jitter (anglicisme désignant la gigue) d'horloge correspond à de courtes fluctuations d'un signal numérique dues au bruit dans les composants électroniques. Dans le domaine temporel, il se traduit par la fluctuation de la période d'un signal numérique. Son extraction se base uniquement sur des composants numériques (bascules D, compteurs, *etc.*). Ce phénomène est par conséquent très utilisé pour les TRNG dans les circuits numériques. La figure 1.9 représente un exemple de signal périodique affecté par le jitter.

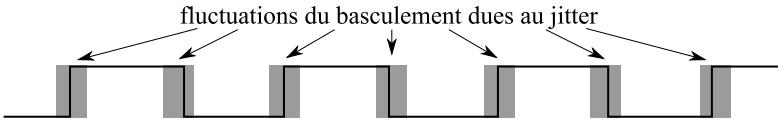


FIGURE 1.9: Exemple de signal périodique affecté par le jitter

Extraction du jitter Il existe deux méthodes utilisées par les TRNG pour extraire l'aléa du jitter :

- **Le comptage :** compter le nombre de fronts d'un signal périodique soumis au jitter pendant un temps donné suffisamment long pour accumuler le jitter. En effet, les fluctuations temporelles créées par le jitter sont additives dans le temps et peuvent influencer le nombre de fronts si la fenêtre de comptage est suffisamment longue.
- **L'échantillonnage cohérent :** échantillonner un signal périodique au moment de sa commutation d'un état logique bas vers un état logique haut ou inversement. Ainsi, la valeur échantillonnée sera aléatoire due au jitter, comme représenté sur la figure. 1.10

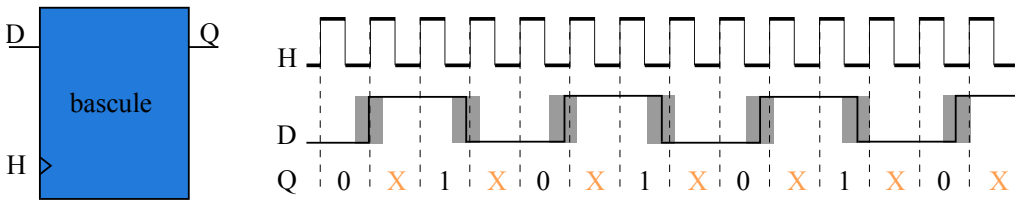


FIGURE 1.10: Exemple d'échantillonnage d'un signal soumis au jitter

L'entrée H est le signal d'horloge permettant l'échantillonnage, l'entrée D est le signal soumis au jitter et la sortie Q est le résultat de l'échantillonnage. Les *X* de la figure 1.10 représentent les valeurs aléatoires extraites de l'échantillonnage du signal à un moment proche de sa commutation.

TRNG de Fairfield *et al.* Le premier dispositif TRNG implanté dans un circuit numérique exploitant le principe d'extraction du jitter fut introduit par Fairfield *et al.* [31]. Le principe

repose sur l'échantillonnage cohérent d'un signal soumis au jitter venant d'un oscillateur embarqué de 8MHz à l'aide d'une bascule D. Les travaux introduits dans cet article proposent aussi une analyse mathématique du générateur réalisé ainsi qu'une calibration de la fréquence du signal d'échantillonnage par rapport au signal échantillonné pour maximiser la probabilité d'échantillonner un bit aléatoire. Ce travail sera par la suite très largement repris et amélioré. Il constitue la base des TRNG numériques exploitant le jitter d'horloge.

PLL-TRNG La première variante du générateur de Fairfield *et al.* est le TRNG à base de boucle à verrouillage de phase — en anglais phase-locked loop based TRNG (PLL-TRNG) — présenté par Fischer *et al.* [32]. Les PLL sont en partie constituées de composants analogiques. Ce générateur n'est donc pas réalisable uniquement avec des éléments numériques. Cependant, les PLL sont très utilisées en électronique numérique pour réaliser des horloges à des fréquences configurables. À tel point qu'il y a peu de famille de circuit logique programmable — en anglais field programmable gate array (FPGA) — récente n'incluant pas de PLL. En effet, elles sont capables d'asservir une fréquence de sortie sur un multiple de la fréquence d'entrée et ainsi produire un signal périodique à une fréquence choisie. Le signal en sortie de la PLL, comme tout signal périodique, est affecté par le jitter. C'est donc le jitter de la PLL qu'exploite ce TRNG. Comme le générateur de Fairfield *et al.*, le PLL-TRNG est constitué d'une bascule D et de deux signaux oscillants comme représentés sur la figure 1.11.

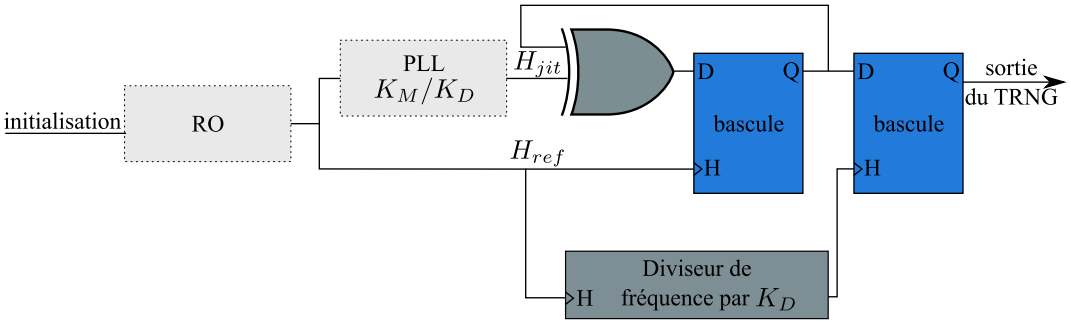


FIGURE 1.11: Structure PLL-TRNG

Le signal d'entrée de la PLL (H_{ref} sur la figure), qui est aussi le signal échantillonneur du signal de sortie de la PLL soumis au jitter (H_{jit} sur la figure), est fourni par un oscillateur en anneau (RO). Les fréquences des deux signaux H_{ref} et H_{jit}, respectivement f_{ref} et f_{jit}, sont liées par la relation suivante :

$$f_{jit} = f_{ref} \times \frac{K_M}{K_D} \tag{1.13}$$

avec K_M le coefficient de multiplication de la PLL et K_D le coefficient de division. En connaissant précisément le rapport entre les fréquences, il est possible de déterminer la position des bits aléatoires afin de les extraire avec un décimateur. Ainsi, un décimateur réalisant la fonction OU-

EXCLUSIF logique entre K_D échantillons consécutifs de la bascule D est placé après à la sortie de la PLL afin d'obtenir un bit aléatoire.

Ce générateur est amélioré dans [33] grâce à l'utilisation de deux PLL comme sources d'aléa (voir figure 1.12). Cette fois le signal échantillonneur est issu de la seconde PLL, ce qui permet d'augmenter la sensibilité au jitter.

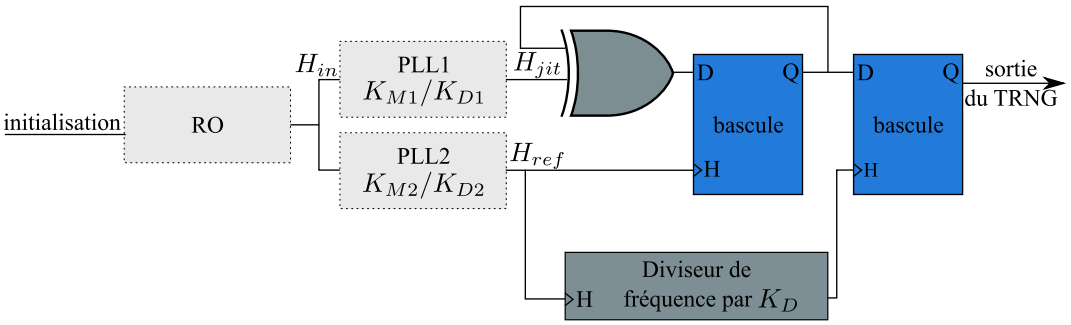


FIGURE 1.12: Structure PLL-TRNG à deux PLL

La relation précédente devient :

$$f_{jit} = f_{ref} \times \frac{K_{M1}}{K_{D1}} \times \frac{K_{D2}}{K_{M2}} \tag{1.14}$$

COSO-TRNG Le TRNG basé sur l'échantillonnage cohérent — en anglais coherent sampling ring oscillator based TRNG (COSO-TRNG) — est proposé par Kohlbrenner *et al.* en 2004 [34]. La structure de ce TRNG est représentée sur la figure 1.13. Cette architecture reprend le même principe que celui de [32] mais offre une alternative à l'utilisation de la PLL. Kohlebrenner *et al.* proposent d'utiliser deux RO ayant les mêmes caractéristiques afin de générer deux fréquences extrêmement proches. Le signal H_{jit}, de fréquence f_{jit}, est échantillonné par le signal H_{ref}, de fréquence f_{ref} (avec $f_{jit} \simeq f_{ref}$). Le flux d'échantillons consiste en une série d'échantillons à '1' logique puis une série d'échantillons à '0' logique avec une série d'échantillons aléatoires intercalés entre elles. L'aléa est extrait en comptant ces deux séries de bits à '1' puis à '0' durant une fenêtre de comptage donnée par H_{out}.

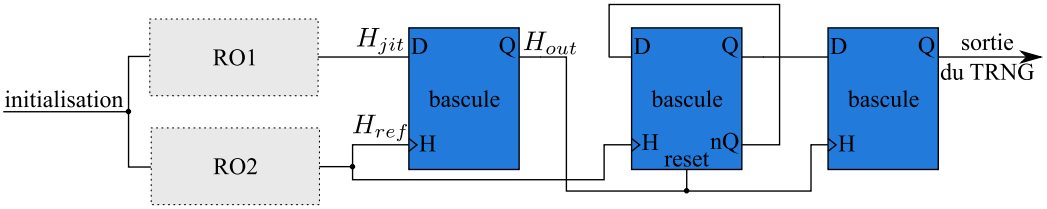


FIGURE 1.13: Structure COSO-TRNG

MURO-TRNG Un autre principe de TRNG exploitant le jitter d'horloge est le TRNG basé sur de multiples RO — en anglais multiple ring oscillator based TRNG (MURO-TRNG) — introduit par Sunar *et al.* [35]. Sa structure est représentée sur la figure 1.14. Le générateur utilise un nombre m de RO comme source d'aléa. Si les RO sont considérés comme indépendants, leurs phases sont distribuées de manière uniforme. Partant de cette hypothèse, la probabilité que la bascule D échantillonne un front (parmi les m fronts théoriquement disponibles à la sortie du OU-EXCLUSIF logique) est aussi uniformément distribuée.

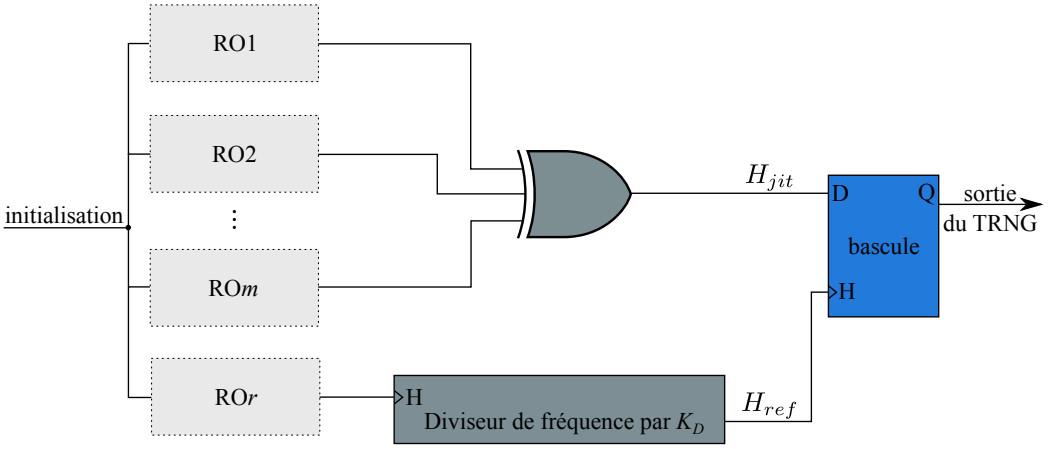


FIGURE 1.14: Structure MURO-TRNG

Cependant, les auteurs de [36] ont montré que la sortie d'une unique porte OU-EXCLUSIF logique à m entrées ne peut pas suivre des signaux d'entrée trop rapides. Ils ont proposé d'utiliser des bascules D supplémentaires devant chaque entrée du OU-EXCLUSIF logique pour résoudre ce problème.

ERO-TRNG Le TRNG de Baudet *et al.* à base de RO — en anglais elementary ring oscillator based TRNG (ERO-TRNG) — présenté dans [37] en 2011 est probablement le TRNG numérique qui se rapproche le plus de celui de Fairfield *et al.*. Il est composé de deux RO identiques. La sortie d'un RO est échantillonnée à l'aide d'une bascule D après un temps suffisamment long pour accumuler le jitter. Le temps d'accumulation est obtenu à partir du deuxième RO à l'aide d'un diviseur de fréquence par K_d. La structure du ERO-TRNG est représentée sur la figure 1.15.

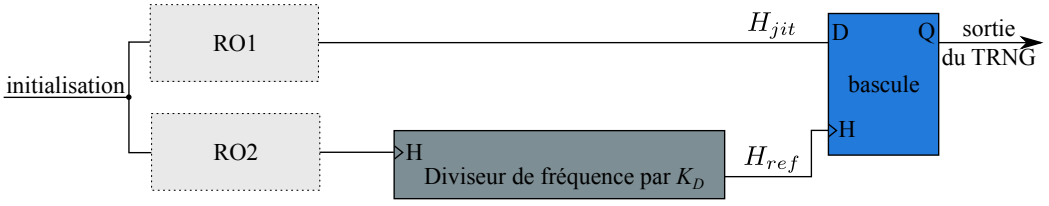


FIGURE 1.15: Structure ERO-TRNG

STR-TRNG La dernière variante du générateur de Fairfield *et al.* présentée dans ce manuscrit est proposée par Cherkaoui *et al.* [38] en 2013. Le principe est basé sur un oscillateur en anneau auto séquencé à N étages — en anglais self-timed ring based TRNG (STR-TRNG). Le STR est un oscillateur à événements multiples sans collision de signal. Son fonctionnement détaillé est présenté dans la partie 1.2.1. L'aléa est extrait en échantillonnant les sorties des N étages du STR en même temps, puis en appliquant une fonction OU-EXCLUSIF logique aux échantillons obtenus comme cela est représenté sur la figure 1.16.

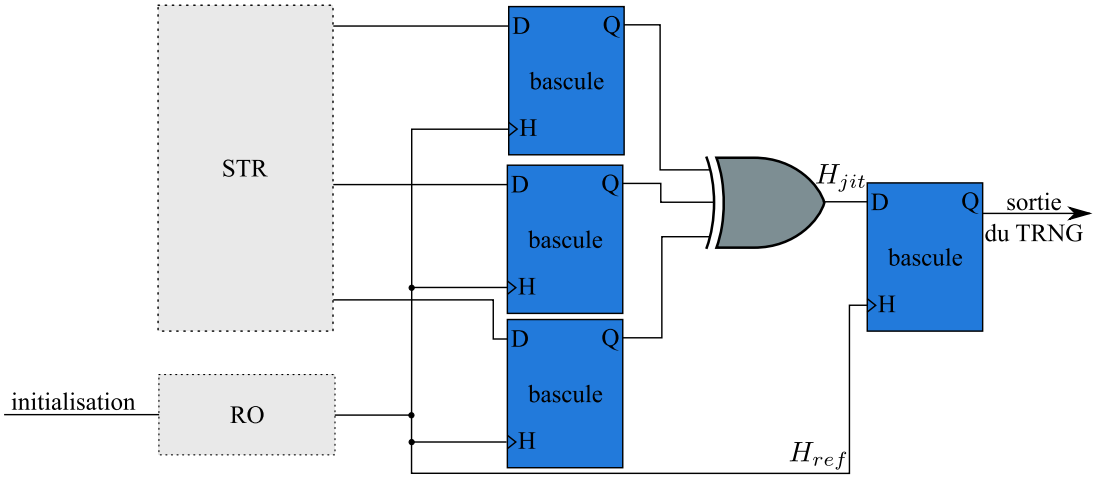


FIGURE 1.16: Structure STR-TRNG

Ce principe sera dérivé en remplaçant le STR par un RO [39]. Les sorties des N inverseurs logiques qui composent le RO sont échantillonnées au même moment.

1.2.2.2 TRNG exploitant la métastabilité oscillatoire

La métastabilité oscillatoire désigne la capacité d'un circuit bistable (*ex.* une bascule RS) à osciller pour une période indéfinie. Ce phénomène a été étudié par Reyneri *et al.* [29]. La métastabilité peut très facilement être transformée en nombres aléatoires en comptant le nombre d'oscillations. Ce nombre est directement influencé par le bruit.

TERO-TRNG Le TRNG à oscillateur en anneau à effet transitoire — en anglais transient effect ring oscillator based TRNG (TERO-TRNG) — fut présenté pour la première fois par Varchola *et al.* [40]. Le TERO est un oscillateur à événements multiples avec collision de signal qui à chaque activation va osciller un certain temps avant de se stabiliser à '1' ou '0' logique. Tout comme pour le STR, le fonctionnement détaillé de la cellule TERO est décrit dans la partie 1.2.1. La structure du TERO-TRNG est représentée sur la figure 1.17. Le RO sert de signal d'activation pour démarrer de manière périodique la cellule TERO. Une bascule T est placée en sortie du TERO pour capter le bit de poids faible du nombre d'oscillations du TERO. Cette valeur sera

échantillonnée par une bascule D avant chaque redémarrage de la cellule TERO et représente un bit aléatoire du TRNG.

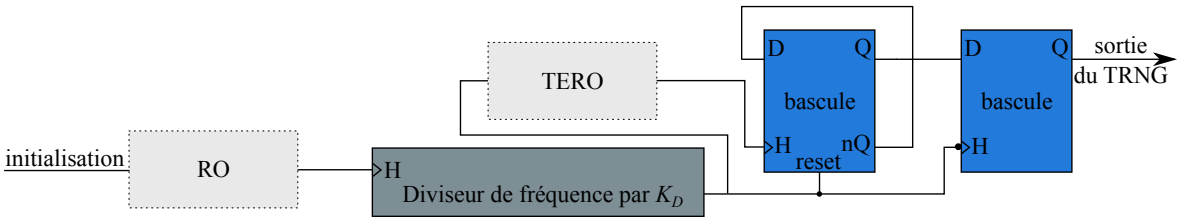


FIGURE 1.17: Structure TERO-TRNG

1.2.2.3 TRNG exploitant la métastabilité analogique

La métastabilité analogique représente l'état d'équilibre instable d'un système électronique pendant une durée indéterminée. Cet état est souvent atteint en ne respectant pas les conditions normales d'utilisation d'un système.

Par exemple, il est possible de reboucler la sortie d'un inverseur logique sur son entrée. La tension de sortie va ainsi se fixer à une valeur indéterminée entre le niveau haut logique et le niveau bas logique pendant un certain temps avant de se stabiliser de manière aléatoire (en fonction du bruit) à un de ces deux niveaux. Entrer dans un état métastable provoque deux phénomènes aléatoires :

— la durée de l'état métastable (le temps de stabilisation).
— l'état final dans lequel se stabilise le circuit.

Générateur de Epstein *et al.* Un des premiers TRNG numériques exploitant la métastabilité analogique est celui de Epstein *et al.* publié en 2003 [41]. L'état métastable est provoqué en rebouclant les inverseurs d'une cellule bistable sur eux-mêmes à l'aide de multiplexeurs. Les deux multiplexeurs permettent de basculer entre deux modes de fonctionnement. Si l'entrée de contrôle des multiplexeurs est à l'état logique '0', les inverseurs sont bouclés sur eux-mêmes. Si elle est à '1', le système complet forme un circuit bistable. La structure de ce TRNG est schématisée sur la figure 1.18. Les sorties de plusieurs structures de ce type sont combinées à l'aide d'un OU-EXCLUSIF logique pour fournir un bit aléatoire.

Cette structure sera reprise par Vasyltsov *et al.* qui proposeront en 2008 une variante de ce TRNG [42]. La figure 1.19 montre le schéma bloc du TRNG de Vasyltsov *et al.*. Ce TRNG fonctionne en deux phases contrôlées par un RO. Soit les inverseurs sont rebouclés sur eux-mêmes, soit ils sont connectés entre eux pour former un anneau.

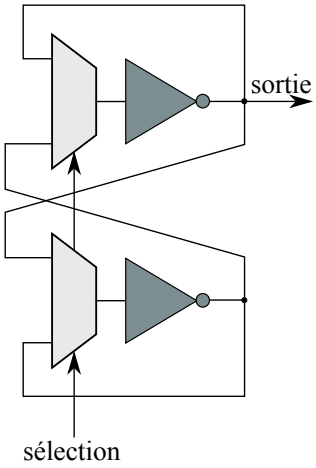


FIGURE 1.18: Structure du TRNG de Epstein *et al.*

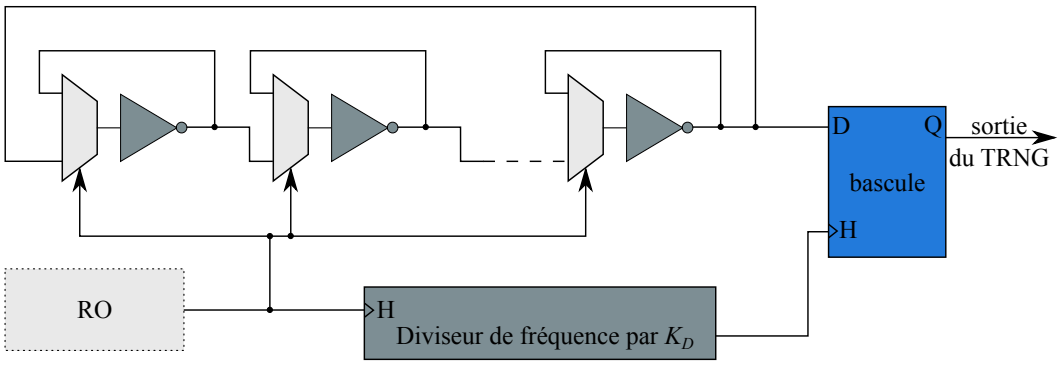


FIGURE 1.19: Structure du TRNG de Vasyltsov *et al.*

Générateur de Majzoobi *et al.* Pour qu'une bascule fonctionne correctement, il est nécessaire de respecter certaines contraintes temporelles : les temps de *setup* et *hold*, c'est-à-dire que le signal d'entrée d'une bascule ne doit pas être enregistré directement après un changement d'état (pour qu'il ait le temps de s'établir correctement) ou qu'il ne change pas d'état directement après avoir été enregistré (pour que la valeur soit enregistrée correctement). À défaut, la bascule entre dans un état métastable.

La figure 1.20 montre un exemple de ce type de métastabilité sur une bascule D pour quatre situations possibles : le signal D bascule à un niveau logique '1' suffisamment longtemps avant l'enregistrement, le signal D bascule pendant le temps de *setup*, le signal D bascule pendant le temps de *hold* et le signal D bascule après l'enregistrement.

C'est cette propriété que Majzoobi *et al.* proposent d'utiliser pour leur TRNG [43]. Leur principe repose sur deux lignes à retard programmables et une bascule D. Les lignes de délai servent à faire en sorte que le signal échantillonné (entrée D) change d'état en même temps que le signal échantillonneur (entrée H). Ainsi le bit généré en sortie (Q) est aléatoire.

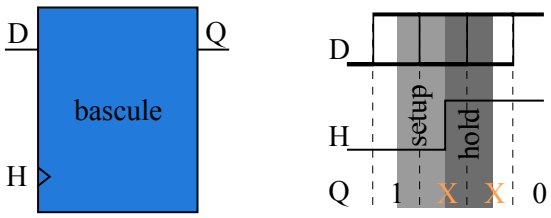


FIGURE 1.20: Métastabilité d'une bascule D

Générateur de Danger *et al.* Le TRNG de Danger *et al.* présenté dans [44] est constitué de m bascules D. Le même signal d'horloge est utilisé comme échantillonneur et signal échantillonné à la différence qu'avant d'être échantillonné il passe par une série de délais d très courts placés entre chaque bascule. Ainsi, le signal échantillonné par la bascule m sera décalé de $(m-1)\times d$ par rapport au signal échantillonné par la bascule 1. De cette manière, Danger *et al.* affirment qu'une bascule D au minimum sera en métastabilité. L'ensemble des sorties des bascules D passe par un OU-EXCLUSIF logique pour générer un bit aléatoire. La structure de ce TRNG est représentée sur la figure 1.21.

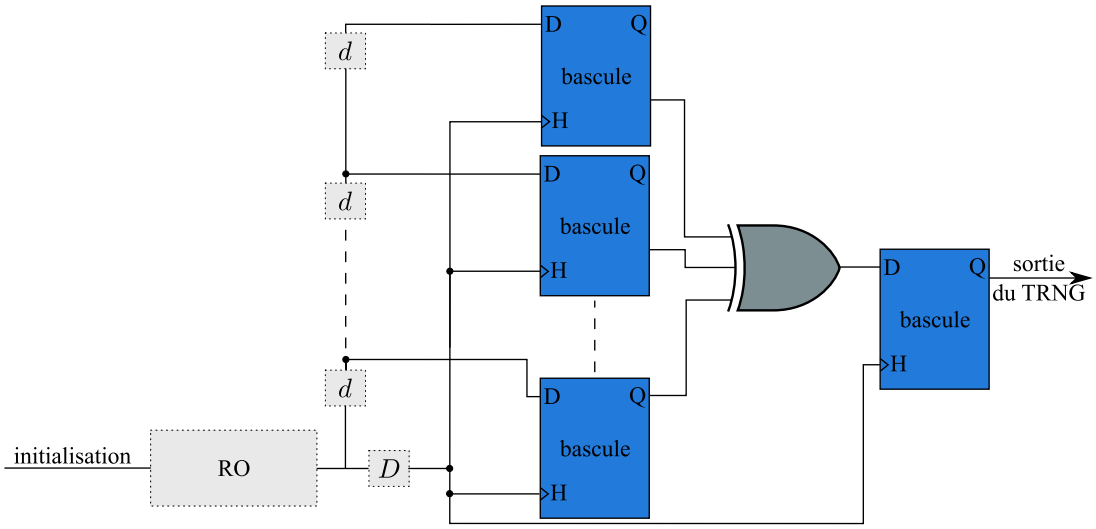


FIGURE 1.21: Structure du TRNG de Danger *et al.*

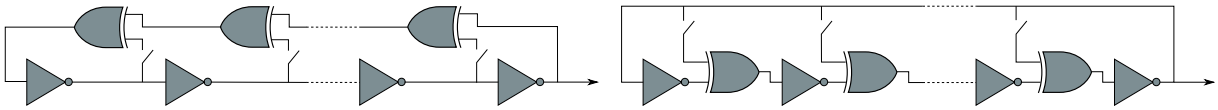
Par la suite, Danger *et al.* proposent une version améliorée de ce TRNG en intégrant un contrôle dynamique des délais d en fonction de tests de biais embarqués [45].

1.2.2.4 TRNG exploitant le chaos

Le chaos représente le comportement imprévisible d'un système censé être déterministe. Ce phénomène se produit sur des systèmes déterministes très sensibles à leurs conditions initiales. Dans ces conditions initiales particulières, le système va avoir un comportement complètement chaotique influencé par le bruit et impossible à prévoir. Puisqu'ils sont imbriqués dans un système

déterministe, ces TRNG peuvent être considérés comme hybrides (HRNG) et sortent légèrement du cadre de notre état de l'art. Nous n'illustrerons donc que brièvement ce principe de TRNG avec un exemple.

TRNG de Golic *et al.* Dans [46] et [47], Golic *et al.* proposent une architecture de TRNG basée sur deux oscillateurs asynchrones mixant le fonctionnement du RO et celui du LFSR : le RO de Fibonacci (FIRO) et le RO de Galois (GARO) représentés sur la figure 1.22. Lorsque les basculements dans un nœud (OU-EXCLUSIF logique) de ces oscillateurs sont proches dans le temps, le bruit va influencer l'état interne du système et ses états suivants. Le TRNG de Golic *et al.* combine la sortie de ces deux oscillateurs à l'aide d'un OU-EXCLUSIF logique avant de l'échantillonner. Dans [48], Dichtl et Golic constatent que le GARO a un comportement beaucoup plus chaotique que le FIRO et propose une amélioration n'utilisant que le GARO.



(a) Oscillateur en anneau de Fibonacci (b) Oscillateur en anneau de Galois

FIGURE 1.22: Oscillateur en anneau de Fibonacci et Galois

1.2.3 Principes de PUF dans les circuits électroniques numériques

Rapidement après la première introduction de PUF optiques par Pappu en 2001 [5], des PUF basées sur les circuits numériques ont émergé. Gassend *et al.*, ont été les premiers à présenter un dispositif de PUF silicium [49]. Depuis, le dispositif de Gassend *et al.* a servi de base pour l'invention de nombreux autres principes de PUF sur silicium. Dans cette partie, nous allons présenter les différents grands principes de PUF numériques dans l'ordre chronologique.

1.2.3.1 RO-PUF

Principe de Gassend *et al.* Cette PUF est basée sur un seul anneau oscillant. Un dispositif à retard configurable est placé dans un circuit auto-oscillant dont la fréquence dépend du retard du circuit. Le signal périodique résultant est synchronisé et ses fronts montants sont comptés par un compteur. Le compteur est activé pendant un nombre prédéfini de cycles d'horloge, après quoi la fréquence de la boucle auto-oscillante peut être lue hors du compteur. Le principe est représenté sur la figure 1.23.

Le dispositif à retard est composé de $n-1$ étages, où n est le nombre de bits du challenge. Chaque étage est composé de deux multiplexeurs et de registres. Ainsi le dispositif est composé de deux chemins comme représentés sur la figure 1.24. À l'entrée du dispositif à retard, un front

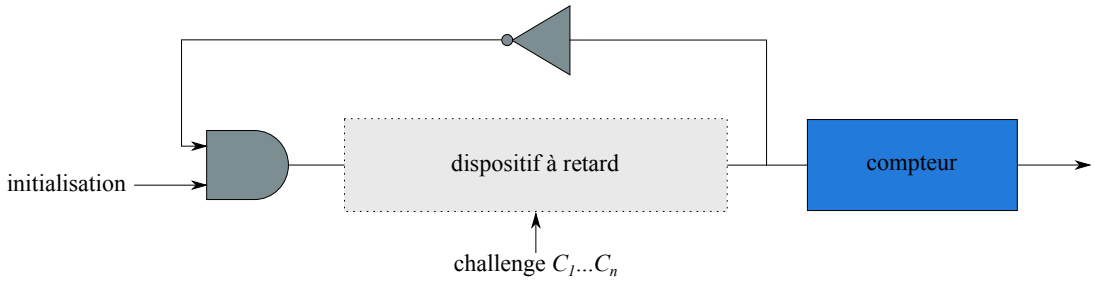


FIGURE 1.23: PUF de Gassend *et al.*

est envoyé sur les deux chemins à la fois. À chaque étage, les fronts peuvent se croiser, c'est-à-dire que le front partant du chemin bas passe au chemin haut et vice-versa. L'un des deux fronts est plus rapide que l'autre et est sélectionné par le multiplexeur de sortie pour être rebouclé à l'entrée du circuit produisant des oscillations. Le nombre de chemins possible est exponentiel en le nombre d'étages du dispositif à retard.

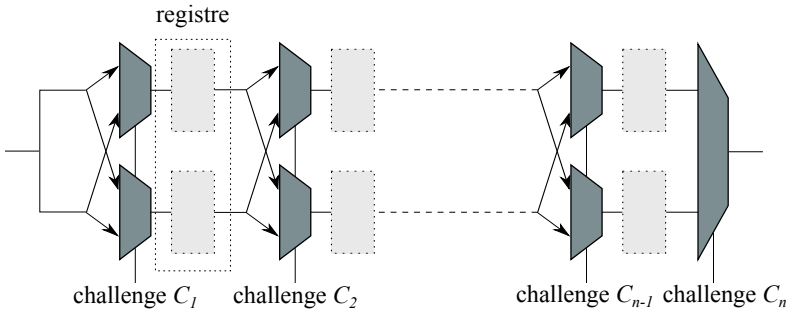


FIGURE 1.24: Dispositif à retard

Gassend *et al.* proposeront une amélioration complexifiant le dispositif à retard à la suite d'une étude montrant la sensibilité de leur principe aux attaques par modélisation [50]. Ils proposeront notamment une ligne à retard à anticipation comme représentée sur la figure 1.25.

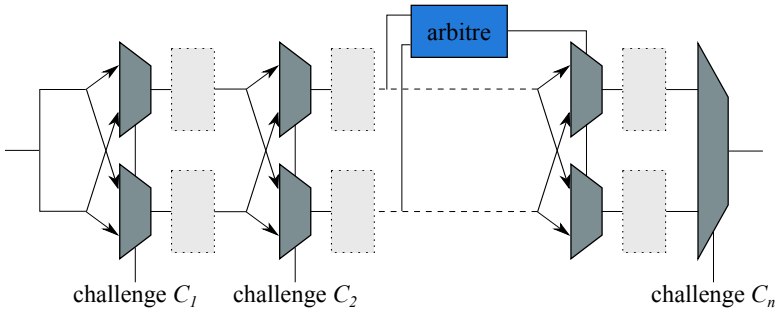


FIGURE 1.25: Dispositif à retard à anticipation

Ce principe sera reprit par Cherif *et al.* en 2012 pour proposer la Loop-PUF [51].

Principe de Suh *et al.* Plus tard, Suh *et al.* proposent un principe de RO-PUF tel qu'il est utilisé aujourd'hui, basé sur plusieurs RO identiques [52]. Chaque RO est un élément de base de la PUF qui oscille à une fréquence particulière en raison des variations de procédés de fabrication. Afin de générer un bit de la réponse de la PUF, les fréquences d'une paire de RO sont comparées. Les résultats de la comparaison de la même paire de RO varient d'une puce à l'autre. La figure 1.26 représente le principe de fonctionnement de cette PUF.

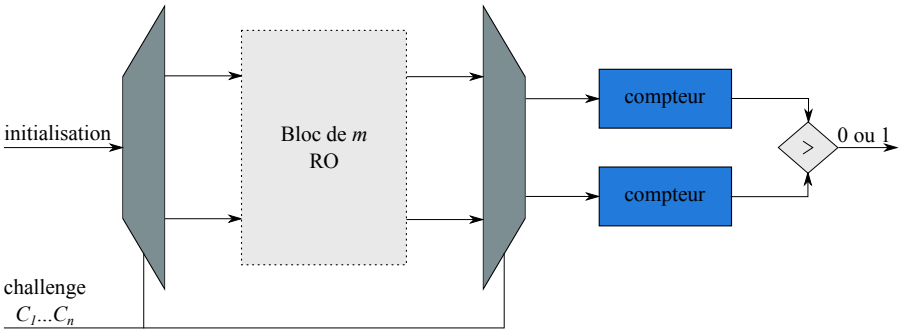


FIGURE 1.26: Structure RO-PUF

1.2.3.2 PUF à base d'arbitre

Basés sur le dispositif à retard de Gassend *et al.*, Lee *et al.* proposent un autre principe de PUF [6] [53] : la PUF à base d'arbitre. Le dispositif à retard est connecté à une bascule D. À l'entrée du dispositif à retard, un front est envoyé sur les deux chemins à la fois. Le front empruntant le chemin le plus court arrivera en premier sur la bascule D qui génèrera un '0' ou '1' logique en fonction du gagnant de "la course". La figure 1.27 montre la structure de cette PUF.

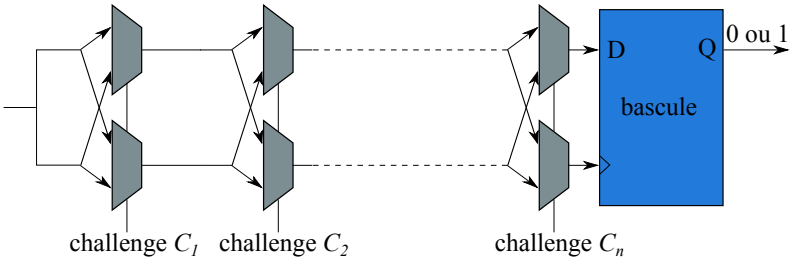


FIGURE 1.27: Structure de PUF à base d'arbitre

Tout comme la PUF de Gassend *et al.*, ce principe sera amélioré avec un dispositif à retard à anticipation. Il sera aussi dérivé à l'aide de registre à trois états [54].

1.2.3.3 PUF à base de mémoire volatile statique ou de bascule

SRAM-PUF Un autre principe de PUF introduit en 2007 par Guarjardo *et al.* repose sur les mémoires volatiles statiques — en anglais static random access memory based PUF (SRAM-PUF) [55]. Une cellule SRAM comme représentée sur la figure 1.28 est formée de deux inverseurs couplés. Le couplage croisé est construit de manière à fournir une boucle à rétroaction positive pour stocker la valeur de bit requise dans la boucle.

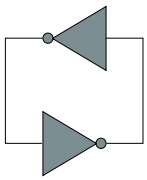


FIGURE 1.28: Schéma d'une cellule SRAM

Dus à leur structure influencée par les variations de procédés de fabrication, les deux inverseurs composant la cellule seront déséquilibrés. Si les cellules SRAM ne sont pas initialisées au démarrage d'un circuit, elles vont se stabiliser dans un état non prévisible. Ainsi, une SRAM-PUF générant n bits sera composée de n cellules SRAM non initialisées qui n'auront qu'à être lues après le démarrage du circuit. La figure 1.29 montre un exemple de génération de réponse d'une SRAM-PUF où les cellules SRAM stabilisées à '1' sont représentées en bleu et les cellules stabilisées à '0' sont en gris.

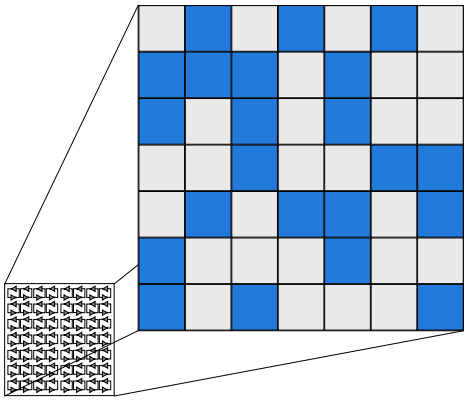


FIGURE 1.29: Exemple d'une réponse de SRAM-PUF

Butterfly-PUF Suivant cette même approche, Kumar *et al.* proposent une variante de la SRAM-PUF [56] qu'ils nomment *Butterfly PUF*. Dans ce principe, les cellules SRAM sont remplacées par deux bascules D en couplage croisé comme représenté sur la figure 1.30. Ce couplage forme une cellule métastable. Un signal d'excitation va forcer au même moment une bascule D à '1' et l'autre à '0' entrainant la métastabilité. Chaque cellule métastable va se stabiliser à '0'

ou '1' en fonction des variations de procédés de fabrication, de la même manière qu'une cellule SRAM.

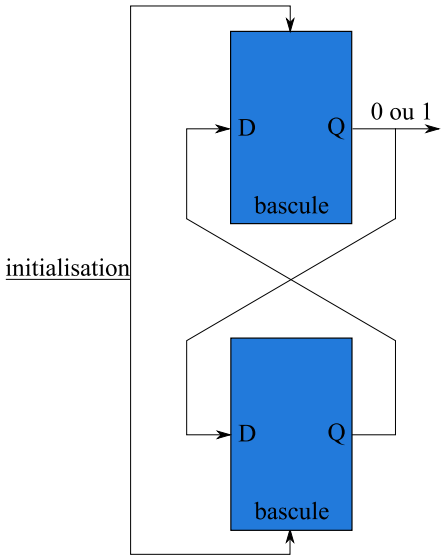


FIGURE 1.30: Schéma d'une cellule *Butterfly*

Flip-flop-PUF Maes *et al.* proposent aussi une version légèrement différente de PUF à base de bascule D — en anglais Flip-flop based PUF (Flip-flop-PUF) — [57].

SR-PUF Également inspirés de la SRAM-PUF, Habib *et al.* conçoivent la PUF à base de bascule RS — en anglais SR lacth based PUF (SR-PUF) — [58].

1.2.3.4 TERO-PUF

Sur la même structure que la RO-PUF de Suh *et al.* (voir figure 1.26), Bossuet *et al.* proposent en 2014 une PUF basée sur les cellules TERO (TERO-PUF) [59]. Contrairement aux oscillateurs en anneaux classiques (RO), l'oscillateur en anneau à effet transitoire (TERO) produit des oscillations temporaires. Afin de générer les bits de la réponse de la PUF, les nombres d'oscillations de paires de TERO sont comparés. Ce principe présente l'avantage par rapport à la RO-PUF de pouvoir extraire plusieurs bits par challenge. Le fonctionnement de la cellule TERO est détaillé dans la section 1.2.1.2.

Dans cette partie, nous venons de voir la multitude de différents principes de TRNG et PUF existants dans les circuits numériques. Auparavant, nous avons détaillé les nombreux efforts faits par la communauté scientifique pour standardiser l'approche d'évaluation et de certification des générateurs d'aléa pour finalement soulever les points d'amélioration possibles. Cela nous a

permis de constater que la qualité d'un générateur physique d'aléa ne dépendait pas seulement du principe, mais aussi de la manière dont il est implanté sur le circuit numérique. Focalisons-nous à présent sur les implantations existantes dans la littérature de TRNG et PUF dans les circuits numériques.

1.3 Implantations sur ASIC et FPGA

1.3.1 Implantations de TRNG

Comme nous l'avons vu précédemment, la plupart des circuits numériques tels que les FPGA ne contiennent pas de blocs analogiques (à l'exception de PLL). Ainsi leurs sources d'aléa sont liées au fonctionnement de portes logiques où le bruit est transformé en variations temporelles (jitter [31]), comportements analogiques de portes logiques entre deux niveaux logiques (métastabilité [41]) ou comportements imprévisibles (chaos [46]). Le fait est que les composants numériques ne sont pas initialement prévus pour ce type d'application. Les fabricants de FPGA ou de circuits intégrés numériques vont avoir pour objectif de réduire au maximum l'impact du bruit sur leurs composants pour éviter tout comportement non prévu, ce qui va à l'encontre des TRNG. C'est pour cette raison que l'implantation de TRNG sur les circuits numériques n'est pas une chose aisée.

Ainsi, certaines implantations de TRNG basées sur des composants numériques nécessitent un développement sur mesure les rendant réalisables seulement sur ASIC et non sur FPGA. C'est le cas du générateur proposé par Intel en 2011 qui exploite la métastabilité des cellules SRAM [60] ou de celui proposé par Bucci *et al.* qui exploite le jitter de deux RO oscillants à des fréquences très proches [61].

Dans notre cas, les TRNG visent à sécuriser les systèmes embarqués au sens large. Pour être largement utilisable, il faut que leurs principes soient suffisamment génériques pour être réalisables dans la plupart des familles de FPGA récentes. Pour ces raisons, les deux principes présentés ci-dessus ne sont pas viables.

De plus, ils sont utilisés pour des applications cryptographiques et doivent donc être conformes au standard AIS31. Parmi les TRNG présentés dans la partie 1.1.1.1, les principes de TRNG basés sur le chaos ne respectent pas cette condition. Bien qu'il semble évident que le fonctionnement des oscillateurs FIRO et GARO présentés par Golic *et al.* dans [46] soit fortement influencé par le bruit, le mécanisme d'extraction d'aléa reste difficilement modélisable et dissociable de la fonction déterministe. Cela les rend non conformes à la norme AIS31.

Les TRNG basés sur la métastabilité analogique quant à eux présentent deux défauts. Premièrement, il est assez difficile de quantifier l'influence du bruit sur la résolution des états métastables. Deuxièmement, il est complexe d'obtenir un état métastable de manière périodique

comme l'explique Fischer [15]. Prenons, par exemple, le principe de TRNG introduit par Epstein *et al.* [41]. Sa structure composée d'un multiplexeur et d'un inverseur logique peut dans certaines situations se mettre à osciller plutôt que de rester dans un état métastable analogique.

Il se trouve que les TRNG réalisables sur une large gamme de circuits numériques incluant les FPGA, et qui sont conformes au standard AIS31 sont tous basés sur des cellules oscillantes. Ce sont les suivants :

— Le STR-TRNG — [38] ;
— Le MURO-TRNG — [35] ;
— Le TERO-TRNG — [40] ;
— Le PLL-TRNG — [32] ;
— Le COSO-TRNG — [34] ;
— Le ERO-TRNG — [37].

En effet, la réalisation d'une cellule oscillante ne nécessite que des cellules standards. Par conséquent, ils sont tous réalisables sur les familles récentes et futures de FPGA. De plus, ils sont tous basés sur une source d'aléa simple et compréhensible. Leur signal de sortie est disponible à l'extérieur du TRNG et un modèle stochastique existe ou est réalisable. Ils sont ainsi tous conformes au standard AIS31.

Cependant, il n'existe pas dans la littérature d'études permettant de comparer ces TRNG une fois implantés sur un circuit numérique en matière de sécurité, d'efficacité et de complexité d'implantation. En analysant l'ensemble des implantations matérielles de ces TRNG, nous remarquons qu'ils sont tous implantés sur des circuits différents, évalués avec des instruments de mesure différents et dans des conditions différentes.

Par exemple, la caractérisation et l'évaluation du STR-TRNG par Cherkaoui *et al.* sont réalisées sur deux familles de FPGA, un FPGA Cyclone III du fabricant Intel et un FPGA Virtex 5 du fabricant Xilinx [38]. La caractérisation du jitter est faite à l'aide de sondes différentielles permettant une très bonne acquisition du signal originel sans (ou presque) ajout de bruit supplémentaire. Wold *et al.* quant à eux, ont réalisé leur implantation sur un FPGA Cyclone II du fabricant Intel et observé les signaux sur l'oscilloscope en les faisant sortir sur des entrées/sorties classiques [36] (par opposition aux entrées/sorties différentielles). Le TERO-TRNG est implanté par Varchola *et al.* sur un FPGA Spartan 3 du fabricant Xilinx [40]. La version la plus récente du PLL-TRNG de Fischer *et al.* est implantée sur plusieurs FPGA provenant tous du fabricant Intel [33]. Le jitter est mesuré à l'intérieur du FPGA et le TRNG est caractérisé pour différentes températures environnantes. Kohlbrenner *et al.* de leur côté ont implanté le COSO-TRNG dans un FPGA Virtex XCV1000 de Xilinx [34]. Enfin, Baudet *et al.* implantèrent le ERO-TRNG sur un FPGA Stratix II de Intel [37].

La technologie (taille des transistors) ainsi que l'architecture sont différentes d'un FPGA

à l'autre. De plus, mesurer une estimation du jitter à l'aide d'instruments différents rend la comparaison totalement impossible. Enfin, la température et les variations de tension d'alimentation vont changer le comportement des bruits à l'intérieur des circuits numériques. Il n'est donc pas rigoureux d'utiliser comme éléments de comparaison les performances proclamées dans ces articles. Ces performances sont propres à la plateforme d'évaluation ainsi qu'aux conditions environnementales (température, tension, *etc.*),

Pour comparer les principes de TRNG et leurs implantations de la manière la plus équitable et la plus rigoureuse possible, il faut une plateforme d'évaluation ayant une structure permettant de tester plusieurs familles de FPGA provenant de plusieurs fabricants dans les mêmes conditions. De plus, cette plateforme doit utiliser le moins possible de composants générant du bruit (*ex.* des alimentations non bruyantes). C'est ce que nous proposons de faire dans le chapitre 2.

1.3.2 Implantations de PUF

1.3.2.1 Variations de procédés de fabrication

Toutes les PUF sont basées sur les disparités entre les éléments les constituant. Ces disparités servent à générer la réponse de la PUF. Dans notre cas, les PUF sont basées sur des circuits électroniques numériques. Par conséquent, la source principale de disparités est le transistor à effet de champ à structure métal-oxyde-semiconducteur — en anglais metal oxide semiconductor field effect transistor (MOFSET). Il existe deux types de variations de procédés de fabrication affectant les transistors : les variations globales et les variations locales.

Variations globales Les variations globales viennent d'imprécision des processus de production d'un circuit intégré, telles qu'un gradient de température sur le wafer lors du recuit, le processus de gravure ou les variations du processus photolithographique. Ces variations entrainent des effets globaux sur les caractéristiques du circuit produit. Par exemple, des gradients de valeurs de tension de seuil peuvent apparaître sur l'ensemble du wafer dans une certaine direction ou les propriétés du dispositif peuvent changer en fonction de leur positionnement / direction. Il s'agit de variations déterministes qui ne sont pas souhaitées pour les PUF.

Variations locales Les variations locales proviennent de différences stochastiques au niveau atomique du transistor. Il n'y a actuellement aucun moyen de les contrôler ou de les réduire. À l'inverse, comme l'ont montré Pelgrom *et al.* dans [62], ces variations sont inversement proportionnelles à la taille des transistors. En effet, plus le transistor est petit, plus l'impact de chaque atome le constituant est important. Ce sont les variations exploitées par une PUF pour générer un identifiant unique.

La figure 1.31 extraite de [22] représente l'influence des variations globales et locales sur un circuit ou un wafer.

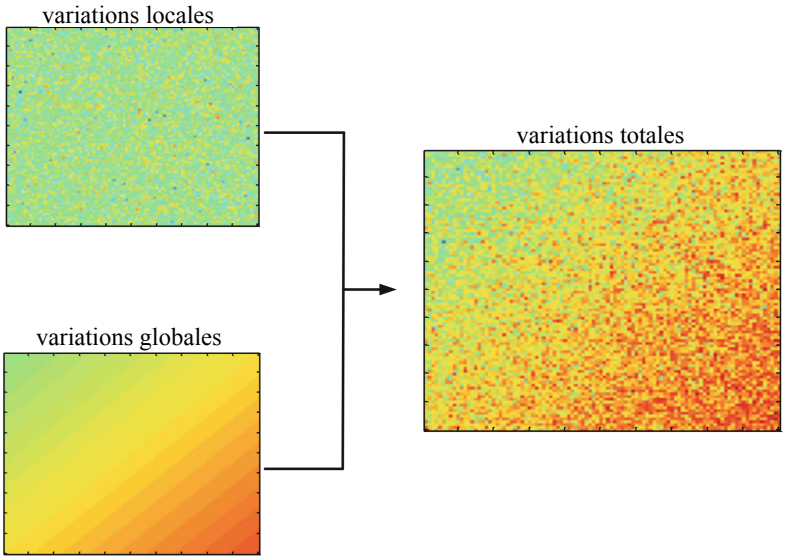


FIGURE 1.31: Influence des variations globales et locales sur un circuit ou un wafer [22]

1.3.2.2 Implantations de PUF sur ASIC

Réduire l'impact des variations globales sur une PUF Pour maximiser l'imprévisibilité des réponses de PUF, les disparités entre les transistors de la PUF ne doivent présenter aucun gradient. Autrement, la réponse de la PUF pourra présenter un biais la rendant prévisible comme expliqué dans la partie 1.1.2. Il existe certaines méthodes pour réduire les variations globales sur ASIC. Une liste de recommandations est donnée dans [63]. En voici quelques exemples :

— Les transistors comparés doivent être de la même taille (longueur et largeur).
— Les transistors doivent être orientés dans la même direction sur le wafer.
— Les transistors comparés doivent être proches les uns des autres.
— Aucune ligne de métal ne doit traverser les transistors

Augmenter les variations locales sur une PUF Comme nous l'avons vu avec le modèle introduit par Pelgrom *et al.*, la seule manière d'augmenter les variations locales est de réduire la taille des transistors [62].

Travaux de Katzenbeisser *et al.* En 2012, Katzenbeisser *et al.* réalisent une analyse à grande échelle sur ASIC de tous les principes de PUF introduit dans la section 1.2.3 (RO-PUF, PUF à base d'arbitre, SRAM-PUF, Butterfly-PUF, Flip-flop-PUF) [64], à l'exception de la TERO-PUF qui sera publiée plus tard. C'est la première comparaison de ce genre. Elle est réalisée sur 96 ASIC

en technologie CMOS 65nm du fabricant TSMC. Chaque ASIC contient de multiples instances de chaque principe de PUF. Les résultats montrent que les qualités de la Butterfly-PUF et la Flip-flop-PUF sont assez largement influencées par les variations de température. À l'inverse, les auteurs de [64] affirment que les SRAM-PUF et les RO-PUF semblent répondre à toutes les propriétés nécessaires d'une PUF. Les résultats de comparaison de la PUF à base d'arbitre dans cette étude ne semblent pas montrer d'influence particulière de la température. Cependant, la PUF à base d'arbitre, en raison de sa structure, présente un gros défaut ; elle est très sensible aux attaques par modélisation. Delvaux montre dans son manuscrit de thèse [65] que cela reste vrai même dans sa version améliorée (avec un dispositif à retard à anticipation présenté dans [50]).

1.3.2.3 Implantations de PUF sur FPGA

D'après l'étude de Katzenbeisser *et al.* de la partie précédente, les deux principes de PUF fournissant les meilleures propriétés sont les PUF basées sur les mémoires SRAM et les PUF basées sur les cellules oscillantes. Intéressons-nous à présent aux performances de ces deux principes de PUF implantées sur FPGA.

SRAM-PUF sur FPGA Pour pouvoir être implantée sur FPGA, la SRAM-PUF nécessite une mémoire SRAM qui ne soit pas automatiquement initialisée au démarrage du circuit. Malheureusement, comme l'explique Maes *et al.* [57], ce n'est pas le cas sur beaucoup de familles de FPGA. De plus, Helfmeier *et al.* [66] et Oren *et al.* [67] ont montré qu'il était relativement simple de cloner la réponse d'une SRAM-PUF. Par conséquent, ce principe n'est pas viable.

PUF à cellules oscillantes sur FPGA Le défaut des SRAM-PUF sur FPGA, ajouté à l'étude de Katzenbeisser *et al.*, nous amène à conclure que le meilleur candidat de PUF pour des cibles FPGA sont les PUF à base de cellules oscillantes. C'est d'ailleurs ce qu'on conclut d'autres études d'implantations sur FPGA [16], [68] [69].

Cependant, la structure de la RO-PUF proposée par [52], telle que présentée dans la section 1.2.3.1, présente certaines failles d'implantation et n'est pas utilisable directement. Un certain nombre de travaux ont contribué à l'amélioration de l'implantation de PUF à base de cellules oscillantes.

Les caractéristiques d'une bonne PUF à base de cellules oscillantes une fois implantées sont les suivantes :

— Les bits des réponses de PUF ne doivent pas être corrélés ;
— Les différences extraites des comparaisons de cellules oscillantes ne doivent dépendre que des variations locales de procédés de fabrication ;
— Les réponses doivent être stables ;

— Le système doit consommer le moins d'énergie possible ;
— Le design devrait être facilement répétable d'un FPGA à l'autre.

Amélioration de Maiti *et al.* Dans [70], Maiti *et al.* montrent que la RO-PUF de Suh *et al.* est influencée par des variations globales entrainant un biais sur les réponses des PUF. Contrairement à un ASIC, il n'existe pas de moyen de réduire ces variations sur FPGA puisque les circuits ont déjà été fabriqués. Cependant, ils proposent de résoudre ce problème en rapprochant physiquement sur le FPGA les cellules RO comparées afin qu'elles soient affectées par les mêmes variations globales.

Amélioration de Bernard *et al.* Bernard *et al.* prouvent plusieurs défauts sur l'implantation de Suh *et al.* [71]. Premièrement, toutes les cellules RO oscillent en permanence, même quand elles ne sont pas comparées pour générer un bit de la réponse de la PUF. Ainsi la consommation de la RO-PUF est très grande. Deuxièmement, ils montrent que si le placement et le routage des cellules sur le FPGA n'est pas imposé, le logiciel de compilation du FPGA va avoir tendance à emprunter les chemins qui l'arrangent. Le résultat est que les cellules comparées ne sont pas implantées de manière identique. Ils montrent aussi la dépendance entre certains challenges de la PUF. Par exemple, si la cellule RO_1 est comparée à la cellule RO_2, ceci donne une information sur le résultat de comparaison de la cellule RO_1 avec la cellule RO_3. Ainsi toutes les cellules ne peuvent pas être comparées avec toutes les cellules pour générer plus de bits de réponse de la PUF, car cela entraîne une corrélation de ceux-ci. Enfin, ils montrent la sensibilité des cellules physiquement trop proches au phénomène de verrouillage. Nous détaillerons ce phénomène dans la partie suivante. Ils introduisent ainsi une nouvelle version de la cellule RO qui n'oscille pas en permanence, mais peut être activée quand nécessaire. C'est la version de la cellule RO qui sera présentée dans la partie 1.2.1.1 et qui sera utilisée dans le reste de ce manuscrit.

Amélioration de Marchand *et al.* Marchand *et al.* reprendront ces améliorations pour les appliquer à la TERO-PUF [72]. Ils proposeront une structure de TERO-PUF divisée en deux blocs de cellules où une cellule du premier bloc sera toujours comparée à une cellule du deuxième bloc pour éviter le phénomène de locking et la corrélation. Ils introduiront aussi une méthode très rigoureuse pour implanter de manière identique les cellules TERO composant la PUF sur les FPGA du fabricant Xilinx (à l'aide de l'outil "HardMacro") et sur les FPGA du fabricant Intel (à l'aide de l'outil "logic lock").

Amélioration de Wild *et al.* Dans [73] Wild *et al.* montrent que des cellules RO oscillant à trop haute fréquence peuvent mettre en défaut le compteur qui les suit. Ils montrent l'impact

de cette défaillance de compteur sur les résultats de PUF et préconisent de ne pas utiliser de cellules oscillantes trop rapides.

Conclusion Tous les travaux d'implantation présentés ci-avant ne concernent que des FPGA à base de mémoire RAM (Xilinx et Intel). Pourtant, le troisième plus grand fabricant de FPGA (Microsemi) utilise des FPGA à base de mémoire Flash. Cette technologie de FPGA présente plusieurs avantages dans un contexte IoT. Premièrement, la taille du circuit est réduite, car les mémoires Flash ne sont pas volatiles. Ainsi, une mémoire non volatile supplémentaire de configuration n'est pas nécessaire. Ensuite, les FPGA à base de mémoire Flash vont consommer moins d'énergie puisqu'ils n'ont pas besoin d'être configurés au démarrage. Enfin, les FPGA à base de mémoire Flash pourront sans problème être utilisés dans un environnement à fort rayonnement électromagnétique (*ex.* applications spatiales, automobile) sans craindre d'être perturbés. Ce n'est pas le cas des mémoires volatiles de type RAM. Nous avons vu à travers l'étude de Marchand *et al.* que les contraintes d'implantation de PUF étaient spécifiques d'un FPGA à l'autre. Ainsi, il semble nécessaire de réaliser une étude visant à implanter de manière rigoureuse et à caractériser les RO-PUF et les TERO-PUF sur FPGA de type Flash. Ce travail sera réalisé dans le chapitre 2.

1.4 Menaces sécuritaires sur les cellules oscillantes

À travers les parties précédentes, nous avons vu que les meilleurs candidats de générateurs d'aléa pour sécuriser le plus grand nombre de circuits numériques sont les générateurs basés sur des cellules oscillantes. Malgré tout, les cellules oscillantes présentent certaines vulnérabilités qui méritent d'être étudiées : leur sensibilité au phénomène de verrouillage et la possibilité d'une analyse du rayonnement électromagnétique qu'elles émettent.

1.4.1 Phénomène de verrouillage

Comme nous l'avons vu précédemment, dans le domaine de la sécurité, les cellules oscillantes servent de source d'aléa pour les TRNG et les PUF puisqu'elles sont largement impactées par le bruit et les variations de procédés de fabrication [74] [75]. Cette propriété, ajoutée au fait que leur conception ne nécessite que des cellules standards, en fait des candidats de choix pour l'extraction d'aléa au sein des circuits numériques.

Cependant, les cellules oscillantes sont aussi sensibles à un autre phénomène assez peu étudié et peu pris en compte par la communauté scientifique et industrielle des générateurs d'aléa lors de leur conception : le phénomène de verrouillage. Le phénomène de verrouillage représente la capacité de deux systèmes oscillants à des fréquences proches de se verrouiller, c'est-à-dire à

se synchroniser et ainsi osciller ensemble à une fréquence proche de leur fréquence d'oscillation naturelle.

Pourtant, l'interaction entre deux systèmes oscillants physiquement proches et fonctionnant à des fréquences proches est un phénomène bien connu, et ce depuis plusieurs siècles. Dans leur article, Mesgarzadeh et Alvandpour, faisant référence à la synchronisation des horloges de Huygens, mentionnent qu'il a été observé pour la première fois au $XVII^{eme}$ siècle [76].

Dans cette partie, nous nous intéressons à l'impact du phénomène de verrouillage sur les générateurs d'aléa à base de cellules oscillantes. Il convient de limiter ce phénomène pour le bon fonctionnement de ces générateurs.

Cependant, pour mieux comprendre ce phénomène nous ne nous focalisons pas seulement sur les articles traitant du phénomène de verrouillage sur les générateurs d'aléa. Nous élargissons notre état de l'art en étudiant des articles analysant le phénomène de verrouillage de manière théorique, ainsi que des articles présentant des applications tirant parti de ce phénomène. En effet, certains systèmes comme les diviseurs de fréquences profitent du phénomène de verrouillage [77].

Cette étude est divisée en deux parties. La première partie dresse un état de l'art des études théoriques du verrouillage sur les systèmes électroniques. La seconde partie traite des études expérimentales exploitant le verrouillage (*ex.* diviseur de fréquence ou attaque sur des générateurs d'aléa) sur des circuits numériques récents tels que les FPGA.

1.4.1.1 Étude théorique du phénomène de verrouillage

Un signal périodique perturbateur injecté à l'intérieur d'un oscillateur électronique, et qui oscille à une fréquence proche de la fréquence d'oscillation naturelle de cet oscillateur, va déroger aux conditions de fonctionnement normales de celui-ci. Il le forcera à osciller à une autre fréquence. C'est ce que s'appliqueront à démontrer, entre 1945 et 1947, Tucker [78], Adler [79] et Huntoon *et al.* [80]. Ils étudieront de manière théorique dans quelles conditions un oscillateur électronique va se verrouiller à un autre oscillateur ou à un signal perturbateur.

Tucker commencera par mettre en évidence le fait que la suppression des oscillations naturelles pour être remplacées par des oscillations forcées dans un oscillateur à triode dépend de deux conditions :

— la puissance (ou tension) du signal perturbateur doit être supérieure à une certaine valeur ;
— la fréquence d'oscillation du signal perturbateur doit être proche de la fréquence d'oscillation naturelle de l'oscillateur.

Adler complétera cette étude en dérivant les conditions de synchronisation d'un oscillateur avec un signal perturbateur. Il établit la formule suivante :

$$\frac{V_{pert}}{V_{osc}} > 2Q \left| \frac{f_{pert} - f_{osc}}{f_{osc}} \right| \tag{1.15}$$

où V_{pert} représente l'amplitude de tension du signal perturbateur, V_{osc} l'amplitude de tension du signal de sortie de l'oscillateur, Q le facteur de qualité de l'oscillateur, f_{osc} la fréquence du signal de sortie de l'oscillateur en fonctionnement normal et f_{pert} la fréquence du signal perturbateur. Ainsi, plus la fréquence du signal perturbateur sera proche de la fréquence du signal de sortie de l'oscillateur en fonctionnement normal, moins la puissance du signal perturbateur aura besoin d'être importante pour verrouiller l'oscillateur.

Enfin, Huntoon *et al.* généraliseront l'équation de Adler pour montrer qu'elle est valable pour d'autres types d'oscillateurs électroniques que seulement l'oscillateur à triode.

À cette période, le transistor n'a pas encore été inventé. Ces études n'ont donc pas montré que ces conditions de verrouillage s'appliquent aux cellules oscillantes CMOS. Ce n'est que plus tard, en 2005, que Mesgarzadeh et Alvandpour reprendront l'équation de Adler pour confirmer qu'elle s'applique également aux cellules oscillantes CMOS [76]. Ils simuleront l'impact du verrouillage sur un RO composé de 5 inverseurs logiques en technologie CMOS 180 nanomètres. Dans cette simulation, le RO oscille à une fréquence de 3,9 GHz et est verrouillé sur des fréquences de 3,8 GHz, 3,7 GHz et 3,6 GHz.

1.4.1.2 Phénomène de verrouillage dans les circuits numériques

Dans la littérature, il n'existe que très peu d'études qui traitent du phénomène de verrouillage dans les circuits numériques. Toutes ces études portent sur un seul type de cellule oscillante, le RO. Ces études sont divisibles en deux catégories.

Diviseurs de fréquences La première représente toutes les utilisations du phénomène de verrouillage pour réaliser des diviseurs de fréquence. Le principe est d'injecter un signal périodique dans un RO à une fréquence proche d'une harmonique de la fréquence du RO pour se verrouiller sur celle-ci. Cette méthode permet de réaliser des diviseurs de fréquence à très haute fréquence et présente l'avantage de ne consommer que très peu d'énergie. Parmi les études les plus populaires dans ce secteur, il y a une étude de Mirzaei *et al.* [81]. Dans ces travaux, les auteurs dérivent la plage de verrouillage en fréquence d'un RO et la démontrent sur deux prototypes de diviseur de fréquence basés sur un RO : un diviseur de fréquence par 2 et un diviseur de fréquence par 6. Ils prouvent aussi que la plage de verrouillage est plus importante en injectant le signal sur plusieurs étages du RO (plusieurs inverseurs logiques). D'autres études similaires dans le domaine sont disponibles ici : [77] [82].

Verrouillage sur les générateurs d'aléa à base de RO La seconde catégorie représente toutes les études sur les générateurs d'aléa pour des applications cryptographiques. Dans cette partie, plusieurs façons de verrouiller un RO sont présentées.

Verrouillage de cellules RO avec un signal injecté à travers l'alimentation La première attaque sur un générateur d'aléa à base de RO est présentée par Marketos et Moore en 2009 [83]. Dans cette étude, ils démontrent qu'un RO peut se verrouiller sur un signal perturbateur injecté à travers l'alimentation du circuit. Ils prouvent l'efficacité du verrouillage sur trois cibles différentes. La première est constituée de deux RO réalisés à l'aide de portes logiques CMOS 74HC04 alimentées en 0-5V. Sans signal injecté, les deux RO oscillent à des fréquences distinctes. Ensuite, le signal perturbateur est envoyé sur la masse des portes logiques. Ce signal a une fréquence de 24 MHz et une amplitude de tension crête-à-crête de 900mV. Avec le signal de perturbation, les deux oscillateurs vont se verrouiller sur celui-ci et osciller à la même fréquence. Ils montrent aussi que les deux oscillateurs vont avoir des phases synchronisées. Avec cette première expérience, ils montrent qu'il est possible de verrouiller des RO à travers l'alimentation du circuit.

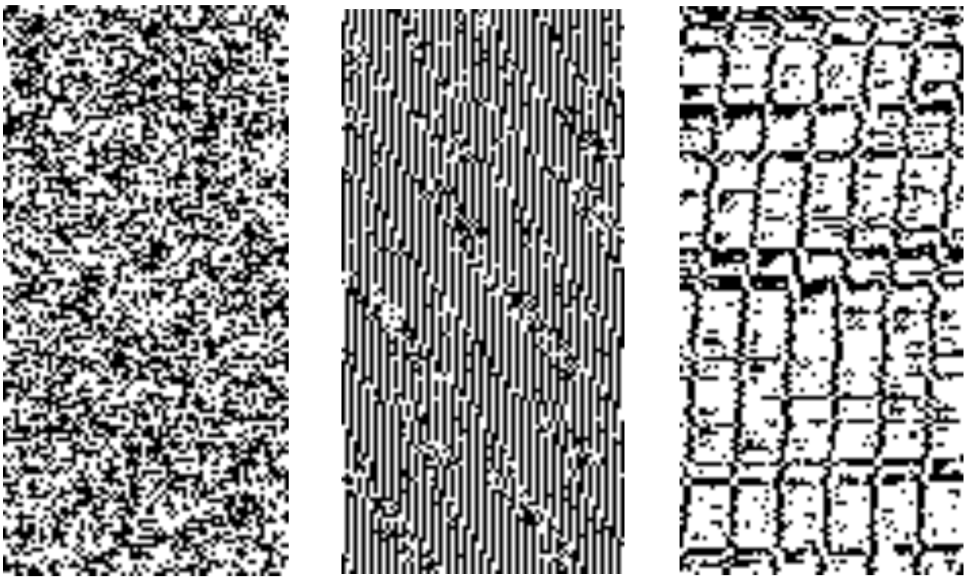
Ils réalisent ensuite la même expérience sur un microcontrôleur 8051 qui embarque un TRNG à base de deux RO (voir sec 1.2.2.1). A cette période, c'est le microcontrôleur communément utilisé pour fabriquer les cartes de paiement. Le flot de bits générés par le TRNG est représenté dans un tableau à deux dimensions où les bits à l'état logique haut ('1') sont représentés en blanc et les bits à l'état logique bas ('0') en noir. Cette représentation est réalisée pour trois cas distincts. Un cas où le TRNG n'est pas perturbé (voir figure 1.32a), un cas où le TRNG est perturbé avec un signal à une fréquence de 1,822 MHz (voir figure 1.32b) et le dernier cas où le TRNG est perturbé avec un signal à une fréquence de 1,929 MHz (voir figure 1.32c). Sur la figure 1.32 extraite de [83], nous pouvons noter la formation de motifs particuliers liés à la fréquence de perturbation. Ces motifs traduisent clairement une réduction de l'aléa sur les bits en sortie du générateur, provoquée par le verrouillage. En effet, s'il y a plus de bits à '1' ou à '0', les bits en sortie du générateur présenteront un biais compromettant l'aléa, comme expliqué dans la partie 1.1.1.1.

Enfin, Marketos et Moore réalisent avec succès cette même attaque sur une carte de paiement émise par la banque British High Street et datant de 2004.

Verrouillage de cellules RO entre elles Pour verrouiller deux RO entre eux, il suffit qu'ils soient physiquement proches et qu'ils oscillent à des fréquences proches. C'est ce que montrent Bochard *et al.* avec le MURO-TRNG (voir sec 1.2.2.1) implanté sur FPGA [84].

Bernard *et al.* montreront la même chose sur la RO-PUF de Suh *et al.* implantée sur FPGA comme nous l'avons vu précédemment (voir sec 1.2.3.1) [71].

Verrouillage de cellules RO avec un signal électromagnétique harmonique injecté Finalement, Bayon *et al.* présentent une nouvelle attaque sur le TRNG de Wold *et al.* (voir sec 1.2.2.1) composé de 50 RO et implanté sur FPGA [85]. Cette expérience montre qu'un signal électro-



(a) Sans perturbation (b) Signal injecté : 1,822 MHz (c) Signal injecté : 1,929 MHz

FIGURE 1.32: Tableaux de 70 x 140 bits sous différentes conditions [83]

magnétique harmonique injecté au-dessus d'un circuit est capable de verrouiller un ou plusieurs RO. Cette méthode présente l'avantage d'être sans contact.

1.4.1.3 Conclusion

Dans cette partie, nous venons de voir que le verrouillage de cellules oscillantes, entre elles ou avec un signal perturbateur, peut totalement compromettre la qualité en matière d'imprévisibilité des nombres fournis par un générateur physique d'aléa. Face à cette menace, plusieurs autres points importants méritent d'être explorés. Premièrement, toutes les études présentées ci-avant ne traitent que des RO. La sensibilité au phénomène de verrouillage de cellules oscillantes spécifiques telles que le TERO ou le STR n'a encore jamais été étudié. Deuxièmement, bien que ces études montrent l'impact du verrouillage sur un générateur physique d'aléa à base de RO, aucune d'elles n'analyse les causes du verrouillage ou ne donne de conseils pour éviter ou, au moins, limiter au maximum le verrouillage. Dans le chapitre 4, nous réaliserons une analyse poussée des causes et de l'impact du verrouillage sur les trois cellules oscillantes utilisées pour la génération d'aléa : les RO, les TERO et les STR. Enfin, nous verrons comment ce phénomène peut être limité.

1.4.2 Analyse électromagnétique

1.4.2.1 Attaques passives

Une attaque passive fait référence à toute exploitation par l'analyse d'informations qui s'échappent du système ciblé. Ce type d'attaque, par opposition aux attaques actives, ne vient aucunement perturber le système durant son fonctionnement. Il s'agit d'extraire des informations issues de canaux "cachés" du circuit en fonctionnement normal. Les canaux "cachés" représentent tous les canaux physiques auxiliaires aux entrées et sorties classiques du système et délivrant de l'information.

Les canaux "cachés" les plus utilisés en cryptographie sont la consommation de puissance du circuit électronique, son rayonnement électromagnétique, son rayonnement photonique, le temps nécessaire pour effectuer un calcul et tout ce qui peut être mesuré depuis l'extérieur du circuit.

Dans le cas des RO, l'analyse du rayonnement électromagnétique se trouve être très efficace. Dans la partie suivante, nous nous intéressons aux différentes études scientifiques de la littérature qui analysent le rayonnement EM des RO pour attaquer un générateur physique d'aléa.

1.4.2.2 Analyse EM des générateurs physiques d'aléa à base de RO

Analyse EM de Merli *et al.* Merli *et al.* ont été les premiers en 2011 à envisager la possibilité d'une analyse par canaux "cachés" des PUF [86]. Cette même année, ils réaliseront la première analyse EM avec succès d'un générateur physique d'aléa à base de RO [87]. Dans cette étude, ils démontrent qu'une fois implantée sur un circuit électronique, une RO-PUF est sensible à l'analyse par canaux "cachés" de type électromagnétique. Ils réalisent une décapsulation d'un FPGA Spartan XC3S200 du fabricant Xilinx. Cette décapsulation permet de réaliser une cartographie fréquentielle du FPGA à l'aide d'une sonde EM de champ proche. La sonde utilisée est une sonde ICR HH 150 du fabricant Langer. La figure 1.33 extraite de [87] montre la configuration pour réaliser une cartographie à l'aide d'analyse EM sur un FPGA décapsulé. Avec cette cartographie, ils arrivent à identifier les RO ainsi que leurs fréquences d'oscillation. De cette manière, ils reconstruisent un modèle complet de la RO-PUF implantée sur le FPGA. .

Ils présenteront ensuite une amélioration de cette analyse EM en la réalisant sur des RO de très petite taille (3 inverseurs logiques) implantés sur un FPGA Spartan 3A du fabricant Xilinx [88]. Dans cette étude, ils montrent comment filtrer le signal EM enregistré pour améliorer la détection d'un RO en particulier. De plus, ils montrent qu'une analyse EM des compteurs placés après les RO permet d'identifier quels RO sont en train d'être comparés.

Analyse EM de Bayon *et al.* Dans la partie précédente, nous avons vu comment un générateur physique d'aléa pouvait être attaqué à l'aide du phénomène de verrouillage par injection EM,

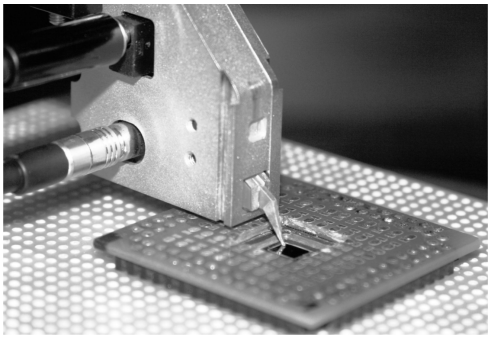


FIGURE 1.33: Analyse EM d'un FPGA décapsulé à l'aide d'une sonde de champ proche [87]

injection d'un signal par l'alimentation ou encore juste par la proximité de deux oscillateurs oscillant à des fréquences proches. Ce type d'attaque pourrait être encore plus efficace et rapide si l'attaquant pouvait connaitre la position et la fréquence de fonctionnement des cellules qui composent le générateur d'aléa. Bayon *et al.* montrent que cela est possible dans une étude d'analyse EM d'un TRNG à base de RO sur FPGA [89]. Ils réalisent une cartographie fréquentielle du circuit grâce aux émissions EM de celui-ci. En chaque point du circuit ils enregistrent le signal EM dans le domaine temporel, effectuent une transformée de Fourier rapide — en anglais fast Fourier transform (FFT) — sur ce signal pour le passer dans le domaine fréquentiel et identifier la fréquence d'oscillation des RO qui constituent le TRNG. La figure 1.34 extraite de [89] montrent les étapes de réalisation de cette cartographie.

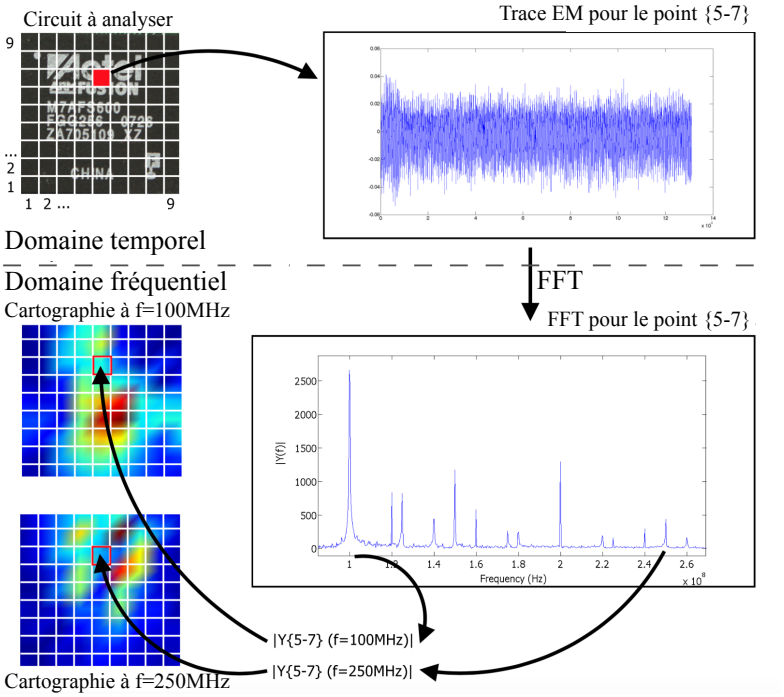


FIGURE 1.34: Cartographie fréquentielle à l'aide de l'analyse EM [89]

Ensuite, en se basant sur le fait que la fréquence d'oscillation d'un RO est influencée par la température et la tension d'alimentation, ils réalisent une analyse différentielle de fréquence sous deux conditions de tension (ou température) différentes. Cela leur permet d'identifier de manière claire les RO sur le circuit et leur fréquence d'oscillation.

Enfin, ils prouvent l'efficacité de cette analyse en identifiant un TRNG à base de RO implanté dans un FPGA à côté d'un algorithme de chiffrement en fonctionnement.

1.4.2.3 Conclusion

Tout comme pour le phénomène de verrouillage, l'analyse EM des générateurs d'aléa à base de cellules oscillantes n'a été réalisée que sur des générateurs à base de RO. Bien qu'il ne semble pas nécessaire de vérifier le potentiel d'émission EM d'un oscillateur à oscillation permanente, le cas particulier du TERO mérite d'être étudié. En effet, le TERO n'oscille que temporairement avant de se stabiliser. Intuitivement, nous pourrions penser que les émissions EM résultantes vont être plus faibles. Dans le chapitre 5, nous réalisons une analyse EM du TERO, ainsi qu'une analyse EM d'une TERO-PUF pour vérifier sa résistance face à une attaque de ce genre.

1.5 Conclusion

Dans ce chapitre, nous avons vu les différents grands principes de TRNG et PUF sur les circuits numériques existant dans la littérature. Pour répondre à la problématique soulevée dans l'introduction — répondre à un besoin pressant de sécuriser les systèmes embarqués du monde de l'IoT —, nous avons étudié, à travers la littérature scientifique existante, les candidats potentiels de TRNG et PUF qui peuvent être implantés de manière sûre au sein d'un grand nombre de circuits numériques incluant les FPGA. Cette analyse nous a permis d'identifier certaines améliorations à apporter pour parachever les travaux scientifiques existants.

Premièrement, nous avons dressé un historique de la façon d'évaluer la qualité statistique et l'imprévisibilité des générateurs physiques d'aléa. Les nombreux efforts de standardisation de l'évaluation des TRNG nous ont permis de voir que la meilleure solution pour évaluer l'entropie d'un générateur physique d'aléa était d'établir un modèle stochastique de celui-ci. Cette approche n'a encore jamais été appliquée aux PUF. Dans le chapitre 3, nous tentons d'amorcer des travaux dans ce sens.

Deuxièmement, nous avons vu que les générateurs physiques d'aléa, aussi bien les TRNG que les PUF, présentant les meilleures propriétés pour des implantations sur une large gamme de circuits numériques étaient tous basés sur des cellules oscillantes. Malgré tout, les études

scientifiques dans le domaine des TRNG ne permettent actuellement pas de comparer de manière rigoureuse les différents principes de TRNG à base de cellules oscillantes.

De plus, nous avons vu qu'un certain nombre de travaux visant à implanter des PUF à base de cellules oscillantes avait été réalisé sur FPGA de type SRAM. Nous avons aussi remarqué à travers les travaux de Marchand *et al.* qu'une implantation de PUF était très dépendante du circuit qui l'accueille [90]. Il se trouve qu'il n'existe aucune implantation rigoureuse de PUF à base de cellules oscillantes sur les FPGA de type Flash qui représente une part importante des circuits numériques.

Ainsi, dans le chapitre 2, nous proposons tout d'abord de réaliser une implantation de tous les TRNG à base de cellules oscillantes dans les mêmes conditions environnementales, en utilisant la même cible et en suivant une approche rigoureuse pour pouvoir les comparer.

Ensuite, nous verrons comment implanter de la meilleure des manières possibles, des PUF à base de cellules oscillantes sur les FPGA de type Flash.

Enfin nous présenterons des travaux d'intégration de TRNG et PUF sur FPGA au sein de trois démonstrateurs complets. Le premier est un système USB portable de stockage de données sécurisées réalisé à l'aide d'un TRNG, d'une PUF et d'un algorithme de chiffrement authentifié. Le deuxième est un système de messagerie sécurisée incluant les mêmes primitives cryptographiques. Le dernier est un module permettant de protéger un noyau IP au moment de la conception et de l'activer à distance plus tard.

Ensuite, nous avons pu prendre conscience de la sensibilité des RO au phénomène de verrouillage et les possibles attaques induites par cette sensibilité sur les générateurs physiques d'aléa à base de RO. Au cours de cette analyse nous avons remarqué que seule la sensibilité des RO au phénomène de verrouillage avait été étudiée. La sensibilité de cellules oscillantes spécifiques telles que le TERO ou le STR n'a encore jamais été étudiée.

De plus, aucune analyse des causes du verrouillage ou aucun conseil pour éviter ou limiter au maximum le verrouillage n'ont été proposés à ce jour.

Dans le chapitre 4, nous réalisons une analyse complète du verrouillage sur les trois cellules oscillantes utilisées pour la génération d'aléa : les RO, les TERO et les STR. Nous proposons ensuite des solutions pour limiter ce phénomène dans les circuits numériques.

Finalement, nous avons vu le potentiel d'une attaque passive par analyse du rayonnement EM sur des générateurs physiques d'aléa à base de RO présentée dans différentes études. Ces études, à l'instar du phénomène de verrouillage, ayant été réalisées seulement sur des RO, nous proposons d'étudier dans le chapitre 5, la possibilité d'une analyse EM de la cellule TERO ainsi que d'une TERO-PUF.

Chapitre 2

Conception et caractérisation de TRNG et de PUF à base de cellules oscillantes

Le chapitre 1 nous a permis de prendre conscience que les meilleurs candidats de générateurs physiques d'aléa pour des implantations dans les circuits électroniques numériques étaient les générateurs basés sur les cellules oscillantes. Nous avons vu que, dans le cas des TRNG, il n'existait aucune comparaison équitable des différents principes dans la littérature. Nous avons aussi vu que, dans le cas des PUF, aucune étude pour implanter de manière efficace des PUF sur les FPGA à base de mémoire Flash n'avait encore été menée. Enfin, nous avons montré que l'efficacité d'un générateur physique d'aléa ne dépendait pas seulement du principe, mais aussi de la manière dont il est implanté sur le circuit numérique et du système qui l'entoure.

L'objectif de ce chapitre est de répondre à ces problématiques. Il est divisé en trois parties. Dans la première, nous réalisons une étude d'implantation détaillée sur FPGA de tous les principes de TRNG à base de cellules oscillantes conformes à la norme AIS31. La deuxième partie présente des travaux d'implantation efficace de PUF à base de cellules oscillantes sur les FPGA de type Flash. Enfin, la troisième partie traite de l'implantation de TRNG et de PUF au sein de trois systèmes complets.

2.1 Étude des principes de TRNG conformes à la norme AIS31 sur FPGA

Avec le chapitre précédent, nous avons pu prendre conscience que les meilleurs candidats de TRNG pour une implantation sur FPGA étaient les TRNG à base de cellules oscillantes (voir sec. 1.3.1). Cependant, il n'existe pas d'études scientifiques dans la littérature qui permettent de comparer de manière rigoureuse les résultats d'implantations des différents principes de TRNG à base de cellules oscillantes.

Dans cette partie, nous réalisons une implantation rigoureuse de tous les principes de TRNG sélectionnés dans le chapitre précédent sur plusieurs familles de FPGA. Nous comparons ces générateurs en utilisant toujours la même plateforme matérielle faible bruit dédiée. Nous proposons aussi deux nouvelles métriques d'évaluation pour une meilleure comparaison des TRNG : l'efficacité énergétique et le produit entropie $\times$ débit binaire de sortie. Les TRNG étudiés sont : le ERO-TRNG, le COSO-TRNG, le MURO-TRNG, le PLL-TRNG, le TERO-TRNG et le STR-TRNG.

Ce travail a été réalisé en étroite collaboration avec Oto Petura et tous les travaux présentés dans cette partie sont le résultat d'une publication commune dans laquelle Oto Petura est, en raison de son implication plus importante, le premier auteur. Aussi, il est possible que ces résultats soient présentés en suivant une approche similaire dans le manuscrit de thèse de Oto Petura.

2.1.1 Stratégie d'implantation et d'évaluation des TRNG étudiés

Afin d'évaluer le plus équitablement possible et sur plusieurs familles de FPGA les TRNG sélectionnés, la plateforme de test est composée de trois parties : un FPGA contenant seulement le TRNG à évaluer, une carte d'acquisition chargée de traiter et d'enregistrer le flux de données produit par le TRNG avant de l'envoyer au PC pour évaluation. Le FPGA contenant le TRNG et la carte d'acquisition communiquent grâce à une liaison série. Le flux de données et un signal de synchronisation des données sont envoyés à la carte d'acquisition à travers deux liaisons de données différentielles basses tensions — en anglais low voltage differential signaling (LVDS). Le flux de bits généré est sauvegardé à l'intérieur d'une mémoire SRAM de 4 Mo de la carte d'acquisition puis envoyé au PC à l'aide d'un bus USB.

Cette méthode d'acquisition présente plusieurs avantages. Premièrement, l'interface de transfert des données sur FPGA contenant le TRNG est simple et son impact sur le fonctionnement du TRNG est réduit à son minimum. Deuxièmement, grâce à la liaison LVDS, le transfert des données est rapide et robuste face au bruit. Enfin, l'utilisation de la mémoire SRAM garantit la continuité du flux de données entre le FPGA contenant le TRNG et le PC. Cette continuité n'aurait pas pu être assurée avec l'utilisation de la communication USB seule.

Pour comparer les TRNG dans un cas réel d'utilisation et donc réduire leur vulnérabilité face aux manipulations externes, aucune horloge externe (*ex.* horloge à quartz) n'est utilisée. Tous les signaux d'horloges ont été générés à l'intérieur du FPGA à l'aide de RO.

Cette évaluation est faite sur plusieurs familles de FPGA : Xilinx Spartan 6, une famille de FPGA à base de mémoire SRAM en technologie 45 nm utilisant des tables de correspondances à 6 entrées — en anglais look-up table (LUT) — ; Intel Cyclone V, une famille de FPGA à base de mémoires SRAM en technologie 28 nm utilisant des LUT à 6 entrées ; et Microsemi SmartFusion 2, une famille de FPGA à base de mémoires Flash en technologie 65 nm utilisant des LUT à 4 entrées.

Pour la plupart des TRNG sélectionnés, le jitter d'horloge des RO est utilisé comme source d'aléa. Ainsi, nous avons d'abord caractérisé le jitter des horloges générées en fonction de leur fréquence d'oscillation. Pour cette étude, seul un RO est implanté sur le FPGA et le jitter en sortie de ce RO est mesuré à l'aide d'un oscilloscope Lecroy WaveRunner 640ZI (4 GHz de bande passante, 40 G échantillons/seconde) et d'une sonde différentielle D420 WaveLink 4 GHz. Les résultats de mesure du jitter sont représentés sur la figure 2.1. Les FPGA Spartan 6 et Cyclone V produisent un jitter compris entre 2 et 4 ps pour une période d'horloge entre 4 et 8 ns. Les FPGA SmartFusion2 produisent un jitter plus élevé, entre 8 et 10 ps pour la même période d'horloge. Nous pouvons aussi remarquer que le jitter augmente avec la période d'horloge. Ceci est probablement dû à un jitter global déterministe (*ex.* jitter venant de l'oscillateur RC embarqué). Pour toutes ces mesures, en dessous d'une période d'horloge de 3 ns le bruit dans les appareils de mesure prend le dessus.

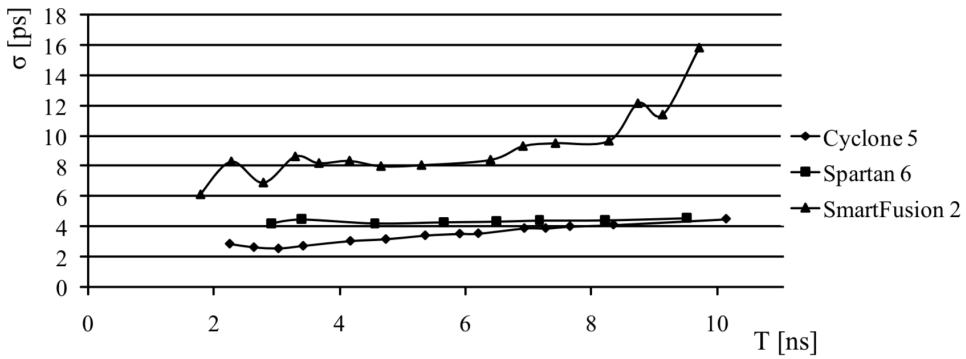


FIGURE 2.1: Résultats du jitter mesuré sur les familles de FPGA sélectionnées

Tous les principes de TRNG testés dans cette étude présentent des degrés de liberté de configuration importants (nombre d'inverseurs par RO, facteurs de divisions, *etc.*). Pour cette comparaison, nous avons choisi les paramètres produisant le meilleur taux d'entropie par bit pour chaque TRNG.

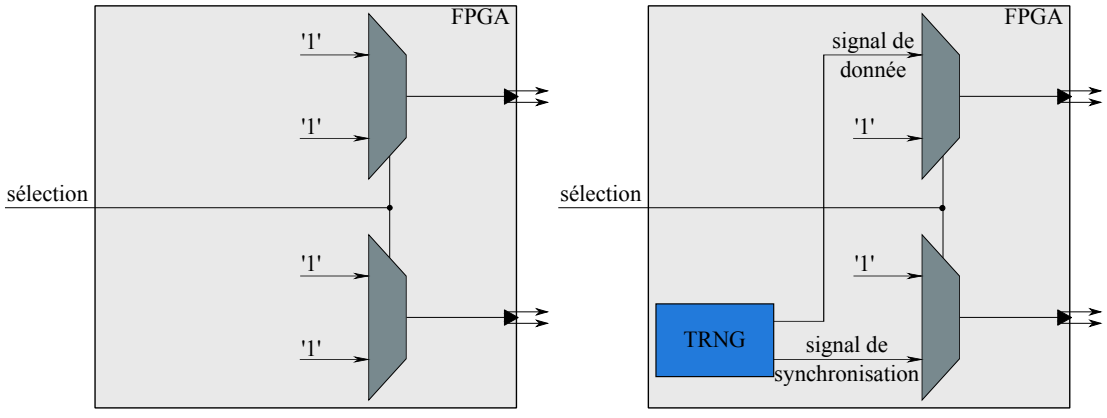
2.1.2 Critères d'évaluation

Afin de réaliser une comparaison des résultats d'implantation des différents principes de TRNG, les critères d'évaluation suivants ont été définis. Ils sont divisés en deux catégories : les critères technologiques et les critères sécuritaires.

2.1.2.1 Critères technologiques

— **Surface d'occupation du FPGA :** La surface totale d'occupation du FPGA est exprimée en nombre de LUT et de registres occupés par l'implantation.

— **Consommation de puissance (mW) :** Ce paramètre donne la consommation de puissance du TRNG (sans l'interface de transfert des données). Pour ce faire, il faut tenir compte de deux choses. Premièrement, un FPGA alimenté, même non configuré, a une consommation statique non nulle. Deuxièmement, sortir en continu les données du TRNG à l'extérieur du FPGA entraîne une surconsommation de celui-ci dû aux basculements permanents des entrées et sorties. Afin de soustraire la consommation statique du FPGA, la consommation d'une implantation de référence a été mesurée. Dans cette implantation, le FPGA est programmé avec simplement une entrée de sélection et deux multiplexeurs comme représentés sur la figure 2.2a. Ensuite, les différents TRNG sont implantés sur les FPGA avec cette implantation de référence. Ainsi l'entrée de sélection permet de désactiver les entrées/sorties du FPGA sans arrêter le générateur lors de la mesure de consommation comme représentée sur la figure 2.2b. Les consommations des FPGA Spartan 6, Cyclone V et SmartFusion2 avec l'implantation de référence sont, respectivement, $3,5\ mW$, $29,7\ mW$, et $12,5\ mW$. Cette consommation est soustraite à la consommation totale mesurée pour toutes les expériences présentées ci-après.



(a) Design de référence (b) Design avec TRNG

FIGURE 2.2: Designs pour la mesure de consommation de puissance

— **Débit binaire ($Mbits/s$) :** Puisque l'interface de transfert des données de la carte d'acquisition est plus rapide que les TRNG testés, elle ne limite pas la vitesse de transfert des données. Le débit binaire de sortie dépend donc seulement du générateur.

— **Efficacité énergétique ($bits/\mu Ws$) :** Ce paramètre exprime la relation entre le débit binaire de sortie du TRNG (en $bits/s$) et la puissance consommée (en μW). Autrement dit, il traduit le nombre de bits obtenus par unité d'énergie (un μWs).

— **Faisabilité et répétabilité :** Ce dernier paramètre est un paramètre arbitraire représentant la difficulté de réalisation d'une implantation et sa répétabilité d'une famille de FPGA à l'autre. Ce critère est classé en 6 niveaux allant de 0 à 5.

  — **Un score de 5**, le plus haut score, définie une implantation qui ne requiert pas d'intervention spécifique manuelle et les résultats obtenus seront toujours satisfaisants, quelle que soit la famille sur laquelle le TRNG est implanté.

  — **Un score de 4** représente une implantation qui nécessite quelques configurations manuelles simples comme le placement des composants sur le FPGA. Les résultats obtenus sont répétables sur tous les circuits FPGA d'une même famille.

  — **Un score de 3** est donné à une implantation qui a besoin d'une configuration manuelle (*ex.* optimisation des paramètres) et une configuration complexe dans certaines familles, mais la conception est réalisable dans toutes les familles de FPGA. Ces optimisations manuelles ne sont pas transférables automatiquement d'une famille de FPGA à une autre, mais restent les mêmes au sein d'une famille de FPGA.

  — **Un score de 2** définit une implantation qui nécessite l'utilisation d'une configuration manuelle complexe en fonction de la famille (optimisation de la topologie et du routage). Les résultats obtenus restent valables pour les circuits FPGA d'une même famille.

  — **Un score de 1** est attribué à une implantation qui nécessite une configuration manuelle pour chaque circuit individuellement, même à l'intérieur de la même famille, mais une solution satisfaisante peut toujours être obtenue.

  — **Un score de 0** est donné à une implantation pour laquelle une configuration satisfaisante ne peut pas être garantie.

2.1.2.2 Critères sécuritaires

— **Taux d'entropie par bit de sortie :** L'entropie est estimée à l'aide du test T8 de l'AIS31 (voir sec. 1.1.1.2). Ce test permet de réaliser une estimation du taux d'entropie. En aucun cas, il ne remplace un modèle stochastique du TRNG. Pour une estimation précise de l'entropie, il convient d'utiliser un modèle stochastique. Ce travail ne fait pas partie de cette étude et ne sera donc pas détaillé outre mesure.

— **Le produit entropie $\times$ débit binaire :** Le taux d'entropie et le débit binaire de sortie

sont fortement liés puisqu'un signal de sortie présentant un faible taux d'entropie couplé à un haut débit binaire de sortie peut subir un post-traitement pour augmenter l'entropie aux dépens du débit binaire. Ainsi, ils doivent être évalués ensemble en utilisant le même paramètre : le produit entropie × débit binaire.

2.1.3 Implantations des TRNG dans les FPGA

2.1.3.1 ERO-TRNG

Conception La conception du ERO-TRNG est celle décrite dans la partie 1.2.2.1. Le compteur synchrone réalisant la division de fréquence est choisi avec une taille de 17 bits. L'utilisation de deux RO identiques réduit l'impact des sources aléatoires globales (*ex.* la tension d'alimentation et la température), qui peuvent être facilement manipulables. Le nombre d'éléments N des RO a été choisi pour obtenir une fréquence de signal de sortie d'environ 300 MHz sur les trois familles de FPGA : $N = 3$ pour les FPGA Spartan 6 et $N = 5$ pour les FPGA Cyclone V et SmartFusion2. La période d'échantillonnage (donc le facteur de division K_d) a été sélectionnée en fonction de la fréquence des RO et de la quantité de jitter mesurée (voir fig. 2.1).

La limite d'entropie inférieure définie dans [37] peut être adaptée au ERO-TRNG comme suit :

$$H_{min} = 1 - \frac{4}{\pi^2 \ln(2)} e^{\frac{-\pi^2 \sigma_{th}^2 K_d T_{osc2}}{T_{osc1}^3}} \tag{2.1}$$

où σ_{th}^2 représente la variance du jitter dûe au bruit thermique, K_d le facteur de division et T_{osc1} et T_{osc2} les périodes des signaux générés par RO_1 et RO_2.

L'équation 2.1 et la mesure du jitter représentée sur la figure 2.1 nous permettent de déduire les valeurs de K_d pour chaque famille de FPGA. Pour une fréquence de RO de 300 MHz, le jitter est respectivement de 4 ps, 3 ps et 8 ps sur les FPGA Spartan 6, Cyclone V et SmartFusion2. Ceci donne des facteurs de division respectifs de 80 000, 135 000 et 20 000. Afin d'assurer la répétabilité de l'implantation, les deux RO ont été placés manuellement.

Résultats d'implantation La surface d'occupation des FPGA par le ERO-TRNG et sa consommation de puissance sont relativement faibles, aux dépens d'un faible débit binaire.

Les deux RO doivent avoir la même structure pour compenser l'impact des sources aléatoires globales sur le TRNG. Cela requiert un placement manuel simple des éléments. Ce générateur présente un potentiel de sécurité très élevé. Pour garantir un taux d'entropie suffisant, pour le facteur de division K_d donné, il suffit que les tests embarqués vérifient que les anneaux oscillent et qu'ils ne sont pas verrouillés. Il convient aussi de noter qu'un modèle stochastique solide existe pour ce TRNG.

Tous les résultats d'implantation du ERO-TRNG sont présentés dans le tableau 2.2.

2.1.3.2 COSO-TRNG

Conception La conception du COSO-TRNG est celle décrite dans la partie 1.2.2.1. Comme pour le ERO-TRNG les deux RO utilisés sont identiques. Pour extraire l'aléa venant du jitter, la différence de période entre les deux RO (Δ_T) doit respecter la condition suivante :

$$\Delta_T < \sqrt[3]{\sigma_T^2 \cdot T_{osc}} = \Delta_{T_{max}} \tag{2.2}$$

où T_{osc} représente la période des RO et σ_T^2 la variance du jitter. Satisfaire cette condition n'est pas aisé. Nous avons mesuré la période T_{osc} des RO ainsi que le jitter pour calculer $\Delta_{T_{max}}$. Ensuite, nous avons testé différents placements pour obtenir un Δ_T plus petit que $\Delta_{T_{max}}$. Dans notre cas, nous avons obtenu les résultats suivants :

— $N = 8$ sur Spartan 6 donnant $T_{osc} = 6.92$ ns, $\sigma_T \approx 4$ ps, $\Delta_{T_{max}} \approx 50$ ps ;
— $N = 6$ sur Cyclone V donnant $T_{osc} = 3.17$ ns, $\sigma_T \approx 2{,}5$ ps, $\Delta_{T_{max}} \approx 30$ ps ;
— $N = 10$ sur SmartFusion 2 donnant $T_{osc} = 5.4$ ns, $\sigma_T \approx 8$ ps, $\Delta_{T_{max}} \approx 70$ ps.

Résultats d'implantation Comme attendu au regard de la figure 1.13, la surface d'occupation du FPGA est très faible. Le réglage de Δ_T devra toujours être un compromis entre le taux d'entropie et le débit binaire. En effet, un Δ_T petit augmentera le taux d'entropie, mais réduira le débit binaire et inversement.

La principale contrainte dans la conception du COSO-TRNG est le besoin de régler Δ_T de manière très précise. La différence de période entre les deux RO doit être suffisamment petite (comparable à la quantité de jitter accumulé). Malheureusement, même les RO qui ont exactement la même structure peuvent générer des périodes T_{osc} trop éloignées. Ceci est dû notamment aux variations de procédés de fabrication.

Pour obtenir deux périodes suffisamment proches, nous avons placé un RO à une position sur le FPGA et le second RO a été déplacé automatiquement, à l'aide d'un script en langage TCL, à différentes positions sur le FPGA, jusqu'à ce que Δ_T soit suffisamment petit. Une fois cette configuration obtenue, le placement des deux RO est sauvegardé pour prévenir de tout changement lors de possibles compilations futures.

Bien que l'automatisation du processus permettant de trouver une solution rend l'implantation plus simple, ce réglage doit être fait pour chaque circuit. Cela remet en question l'utilisation pratique de ce générateur.

D'un point de vue sécuritaire, tout comme pour le ERO-TRNG, le fait que les deux RO aient des structures identiques réduit l'impact des sources aléatoires globales.

Tous les résultats d'implantation du COSO-TRNG sont présentés dans le tableau 2.2.

2.1.3.3 MURO-TRNG

Conception Le MURO-TRNG utilise m RO comme source d'aléa (voir fig. 1.14). En admettant que les RO oscillent de manière indépendante, les phases de leurs signaux de sortie sont distribuées uniformément. Ainsi, le nombre de RO nécessaires doit remplir la condition suivante :

$$m > \frac{T_{osc}}{\sigma_{acc}} \tag{2.3}$$

où T_{osc} représente la période moyenne des RO et σ_{acc} le jitter accumulé pendant une période d'échantillonnage.

Puisque les phases des RO sont distribuées de manière uniforme, la probabilité que la bascule D de sortie échantillonne un front d'horloge parmi les m théoriquement disponibles à la sortie du OU-EXCLUSIF logique est aussi distribuée uniformément.

Nous avons implanté la version améliorée du MURO-TRNG introduite par Wold *et al.* permettant de subvenir au problème de temps de réaction d'une porte OU-EXCLUSIF logique à plusieurs entrées (voir sec.1.2.2.1) [36].

Les m RO sont composés de $N = 5$ inverseurs logiques produisant des fréquences d'oscillations entre 200 et 350 MHz selon la famille de FPGA. Ensuite, le nombre m de RO est dimensionné par rapport aux FPGA Cyclone V qui produisent la plus petite quantité de jitter parmi les trois familles sélectionnées. De cette manière, le TRNG satisfera les conditions de l'équation 2.3 quelle que soit la famille de FPGA. Sur Cyclone V, la quantité de jitter est d'environ 3 ps. Cela donne $m > 1200$. Afin de réduire le nombre de RO, le diviseur de fréquence d'échantillonnage est fixé à $K_d = 100$ permettant d'accumuler la quantité de jitter $\sigma_{acc} \approx 30$ ps et réduisant ainsi m à 120 RO.

Résultats d'implantation Avec 120 RO, la surface d'occupation du FPGA et la consommation sont très grandes.

De plus, le MURO-TRNG présente l'inconvénient que les RO, en raison de leur nombre important et de leur proximité, peuvent se verrouiller entre eux, comme l'ont montré Bochard *et al.* [84]. Le verrouillage d'oscillateurs entre eux réduit fortement l'entropie en sortie du générateur comme nous l'avons vu dans le chapitre précédent (voir sec.1.4.1.2).

Cependant, le MURO-TRNG présente deux avantages : aucun placement manuel n'est requit et le débit binaire de sortie est très important, augmentant ainsi le produit entropie $\times$ débit binaire.

Tous les résultats d'implantation du MURO-TRNG sont présentés dans le tableau 2.2.

2.1.3.4 PLL-TRNG

Conception Nous avons choisi d'implanter le PLL-TRNG composé de deux PLL, car il présente une meilleure sensibilité au jitter (voir fig.1.12). Grâce au principe d'échantillonnage cohérent, les échantillons de l'horloge soumise au jitter obtenus en sortie de la bascule D sont distribués de manière uniforme sur la période T_{jit}. Par conséquent, la distance entre les échantillons est définie par $\Delta = T_{jit}/K_D$ et les K_D échantillons doivent être décimés grâce à la fonction OU-EXCLUSIF logique pour obtenir un bit aléatoire en sortie du générateur (voir sec. 1.2.2.1).

Le débit binaire R et la sensibilité S sont définis comme suit :

$$R = f_{ref}/K_D \tag{2.4}$$

$$S = \Delta^{-1} = K_D/T_{jit} \tag{2.5}$$

Afin d'obtenir un taux d'entropie par bit élevé, la distance Δ entre les échantillons doit respecter les conditions suivantes :

$$\Delta \ll \sigma_r \tag{2.6}$$

où σ_r représente le jitter relatif entre les signaux de sortie des deux PLL.

Au regard des équations (2.4) et (2.5), l'objectif du concepteur est d'avoir un Δ le plus petit possible tout en maintenant le débit binaire R dans une gamme acceptable. Pour cela, il faut régler la fréquence d'entrée f_{in} et les facteurs de multiplication K_M et de division K_D des deux PLL.

Avec une fréquence d'entrée d'environ 200 MHz, générée avec un RO, pour chaque famille de FPGA, les paramètres obtenus sont les suivants :

FPGA	**PLL1**		**PLL2**		**Total**		**Δ**
	K_{M1}	K_{D1}	K_{M2}	K_{D2}	K_M	K_D	[ps]
Spartan 6	37	17	17	7	1 377	259	4,82
Cyclone V	31	29	23	18	667	558	4,25
SmartFusion2	74	162	18	22	729	407	9,10

TABLE 2.1: Paramètres des PLL et distance entre les échantillons (Δ) pour les familles de FPGA sélectionnées

Résultats d'implantation Le PLL-TRNG ne requiert aucun placement manuel et fournit un niveau de sécurité élevé puisque les PLL, étant isolées du reste du FPGA, ne subissent pas les bruits déterministes globaux. Le débit binaire est aussi relativement élevé.

Cependant, le choix des paramètres des PLL n'est pas trivial. Plusieurs contraintes doivent être respectées, dont les contraintes physiques des PLL fournies par les fabricants (*ex.* gamme

de fréquences d'entrées, valeur maximale pour K_M et K_D) et les contraintes de sécurité (équation 2.6).

La surface d'occupation du FPGA, si la taille des PLL, qui n'occupe pas la partie logique du FPGA, est exclue, est faible pour une implantation FPGA. Par contre, il est nécessaire que le FPGA embarque une ou plusieures PLL. Ce n'est pas le cas de toutes les familles de FPGA.

Enfin, la consommation d'un PLL-TRNG est très importante à cause des PLL. Dans notre cas, ceci est particulièrement vrai pour les FPGA Spartan6 et SmartFusion2, car elles sont désactivées par défaut. Pour le cas particulier du Cyclone V, les PLL sont actives en permanence, donc la consommation du TRNG par rapport à l'implantation de référence sera faible.

Tous les résultats d'implantation du PLL-TRNG sont présentés dans le tableau 2.2.

2.1.3.5 TERO-TRNG

Conception Le TERO-TRNG implanté est représenté sur la figure 1.17. Le signal de contrôle chargé de redémarrer de manière périodique la cellule TERO est généré par un RO. Il définit le débit binaire du générateur. Dans notre cas, la cellule TERO est composée de $N = 11$ inverseurs par branche. La fréquence du RO est de 150 MHz et la taille du compteur est de 9 bits pour les trois familles de FPGA. Enfin, la fréquence d'oscillations f_{osc} de la cellule TERO est environ 90 MHz pour Spartan 6 et 150 MHz pour Cyclone V et SmartFusion 2.

D'après le modèle de Bernard *et al.*, pour obtenir un taux suffisant d'entropie par bit, le nombre d'oscillations de la cellule TERO (N_{osc}) doit être compris entre [91] :

$$100 < N_{osc} < \frac{T_{ctrl_{1/2}}}{T_{osc}} \tag{2.7}$$

où $T_{ctrl_{1/2}}$ représente le temps d'activation de la cellule (une demi-période du signal de contrôle) et T_{osc} représente la période du signal de sortie de la TERO.

Résultats d'implantation La surface d'occupation du TERO-TRNG est faible, le débit binaire est élevé, la consommation est faible et l'architecture du générateur est simple et efficace.

Malheureusement, le TERO-TRNG nécessite une configuration manuelle des cellules TERO sur chaque circuit individuel pour obtenir un N_{osc} respectant les conditions de l'équation 2.7. La cellule TERO est très influencée par les variations de procédés de fabrication. Il est donc presque impossible d'obtenir une implantation répétable. Même des circuits FPGA d'une même famille, configurés avec le même bitstream, donnent des N_{osc} différents.

Tous les résultats d'implantation du TERO-TRNG sont présentés dans le tableau 2.2.

2.1.3.6 STR-TRNG

Conception Le schéma du STR-TRNG implanté est représenté sur la figure 1.16. L'horloge d'échantillonnage est générée à l'aide d'un RO. Chaque étage du STR est échantillonné à l'aide d'une bascule D et les sorties des bascules D sont décimées à l'aide d'un OU-EXCLUSIF logique.

Si la cellule STR est en mode régulièrement espacé (voir sec.1.2.1.3), le principe de fonctionnement est le même que celui du MURO-TRNG (échantillonnage d'un front d'horloge parmi les m théoriquement disponibles).

Si le nombre d'événements e et le nombre d'étages N sont premiers entre eux, le STR peut produire autant de phases équidistantes que le nombre d'étages N. Dans ce cas, la différence de phase entre deux étages voisins peut être exprimée :

$$\triangle\varphi = \frac{e}{2 \times N} \tag{2.8}$$

La période d'oscillation T_{osc} d'une cellule STR ne dépend pas seulement du nombre d'étages N, mais aussi du nombre d'évènements e. Il est donc possible d'augmenter N en gardant le même T_{osc} et ainsi, théoriquement, régler précisément la différence de phase.

Pour garantir un taux d'entropie par bit suffisant en sortie du TRNG, il faut :

$$\triangle\varphi < \sigma_{acc} \tag{2.9}$$

où $\triangle\varphi$ représente la différence de phase définie dans l'équation 2.8 et σ_{acc} le jitter accumulé.

Notre implantation donne une fréquence d'oscillation f_{osc} du STR d'environ 300 MHz. La plus faible quantité de jitter (celle du FPGA Cyclone V) à cette fréquence est d'environ 3 ps. Ainsi, d'après les équations (2.8) et (2.9), N devrait être supérieur à 550. Un STR de 550 cellules de Muller occuperait une place bien trop importante sur le FPGA. Nous avons donc choisi un STR de $N = 255$ étages.

Résultats d'implantation Bien que la conception de cellule de Muller dans les FPGA ne soit pas très complexe, le STR-TRNG nécessite un placement manuel des éléments pour garantir que la cellule STR fonctionne en mode régulièrement espacé.

Le point fort du STR-TRNG est que le niveau d'entropie par bit et le débit binaire de ce TRNG sont très élevés.

Cependant, la surface d'occupation et la puissance consommée par ce TRNG sont aussi très élevées.

Aussi, d'après l'équation 2.9, le nombre d'étages N devrait être plus grand, mais nos expérimentations ont montré qu'il était difficile d'obtenir un mode régulièrement espacé avec un STR avec un grand nombre d'étages.

Ainsi, un STR avec $N = 255$ semble, dans notre cas, être un compromis raisonnable pour les trois familles de FPGA.

Tous les résultats d'implantation du STR-TRNG sont présentés dans le tableau 2.2.

2.1.4 Comparaisons

Le tableau 2.2 résume tous les résultats d'implantation des TRNG sélectionnés.

TRNG	**FPGA** famille	**Surface** (LUT/Reg)	**Puissance** $[mW]$	**Débit** $[Mbits/s]$	**Efficacité** $[bits/\mu Ws]$	**Entropie** par bit	**Entropie** × **Débit**	**Faisabilité & Répétabilité**
ERO	Spartan 6	46/19	2,16	0,0042	1,94	0,999	0,004	**5**
	Cyclone V	34/20	3,24	0,0027	0,83	0,990	0,003	
	SmartFusion 2	45/19	4	0,014	3,5	0,980	0,013	
COSO	Spartan 6	**18/3**	**1,22**	0,54	442,6	0,999	0,539	1
	Cyclone V	**13/3**	**0,9**	1,44	**1 600**	0,999	1,438	
	SmartFusion 2	**23/3**	**1,94**	0,328	169	0,999	0,327	
MURO	Spartan 6	521/131	54,72	2,57	46,9	0,999	2,567	4
	Cyclone V	525/130	34,93	2,2	62,9	0,999	2,197	
	SmartFusion 2	545/130	66,41	3,62	54,5	0,999	3,616	
PLL	Spartan 6	34/14	10,6	0,44	41,5	0,981	0,431	3
	Cyclone V	24/14	23	0,6	43,4	0,986	0,592	
	SmartFusion 2	30/15	19,7	0,37	18,7	0,921	0,340	
TERO	Spartan 6	39/12	3,312	0,625	188,7	0,999	0,624	1
	Cyclone V	46/12	9,36	1	106,8	0,987	0,985	
	SmartFusion 2	46/12	1,23	1	813	0,999	0,999	
STR	Spartan 6	346/256	65,9	**154**	**2 343,2**	0,998	**154,121**	2
	Cyclone V	352/256	49,4	**245**	**4 959,1**	0,999	**244,755**	
	SmartFusion 2	350/256	82,52	**188**	**2 286,7**	0,999	**188,522**	

TABLE 2.2: Résumé des résultats d'implantation des TRNG sélectionnés

En observant le tableau 2.2, la première chose que nous pouvons remarquer est que la surface d'occupation, exprimée en nombre de LUT, pour un type de TRNG est très similaire d'une famille de FPGA à l'autre. Ceci est dû au fait que chaque porte de délai est implantée dans une LUT.

Le ERO-TRNG occupe une surface sur le FPGA qui est très faible et consomme très peu de puissance. C'est aussi le plus simple à implanter, car il n'exige presque aucune contrainte de placement. Il obtient d'ailleurs la meilleure note en matière de répétabilité et de faisabilité. Son gros point faible est son débit binaire très faible, qui va lui procurer le moins bon produit Entropie × Débit, quand bien même la sécurité et le taux d'entropie par bit de ce TRNG sont grands.

Le COSO-TRNG occupe lui aussi une surface sur le FPGA très faible. Il obtient même la plus faible surface d'occupation, ainsi que la plus faible consommation de puissance. Son débit binaire est acceptable sans être dans le haut du tableau, lui procurant une efficacité énergétique et un produit Entropie × Débit satisfaisant. Cependant, le COSO-TRNG présente un inconvénient

majeur. L'implantation correcte de ce TRNG requiert un placement minutieux des éléments, et ce, sur chaque circuit. Ce désavantage est éliminatoire pour beaucoup d'applications pratiques. En particulier dans un contexte industriel ou la répétabilité de l'implantation est nécessaire pour une production de masse.

Le MURO-TRNG est un générateur simple à implanter et possédant un débit binaire élevé. Il obtient le deuxième meilleur rapport Entropie × Débit, mais son efficacité énergétique n'est pas remarquable. Ceci est dû au fait qu'il consomme beaucoup de puissance. De plus, il a la surface d'occupation la plus importante. C'est un gros point faible dans un contexte comme le nôtre — l'IoT —, car il est fréquent que les circuits électroniques utilisés soient embarqués dans un environnement restreint.

Le PLL-TRNG obtient le meilleur résultat dans aucun des critères d'évaluation choisis, mais se trouve être un très bon compromis puisqu'il occupe une surface relativement faible, sans prendre en compte la surface occupée par les PLL, il consomme relativement peu d'énergie et il présente un bon débit sans être remarquable. Le gros point fort de ce TRNG est que les PLL sont isolées du reste du circuit, renforçant ainsi sa résistance face aux manipulations et aux bruits globaux. Son principal point faible réside dans sa complexité pour trouver des paramètres satisfaisants. Cependant, une fois les paramètres trouvés, l'implantation est répétable sur les circuits d'une même famille.

Le TERO-TRNG occupe une surface du FPGA relativement faible. Il présente un bon débit binaire et une consommation de puissance assez basse. Cependant, comme pour le COSO-TRNG, il nécessite un placement minutieux des éléments, et ce, sur chaque circuit. Cela représente un gros handicap pour l'utilisation de ce TRNG.

Le STR-TRNG présente des résultats très intéressants. Il a un débit binaire extrêmement élevé et un bon taux d'entropie par bit. De ce fait, le produit Entropie × Débit est aussi extrêmement élevé. Il présente le désavantage d'occuper une grande place sur le FPGA et de consommer beaucoup de puissance. Malgré cela, son efficacité énergétique est la meilleure, grâce à son débit binaire. Un autre point faible de ce générateur est la complexité de son implantation. Il nécessite un placement minutieux des éléments, notamment pour qu'il fonctionne en mode régulièrement espacé.

Si nous regardons à présent les deux nouvelles métriques d'évaluation que nous avons introduites dans cette partie, nous remarquons que l'efficacité énergétique est très utile pour mesurer l'énergie utilisée pour générer un bit aléatoire. L'avantage de cette métrique est très visible pour le cas du STR-TRNG qui consomme énormément d'énergie, mais possède aussi un débit binaire extrêmement grand

À l'inverse, le produit Entropie × Débit ne semble pas apporter un avantage remarquable dans le cas de notre évaluation. Nous pensons que ceci est dû à notre stratégie d'implantation : nous

avons choisi les paramètres produisant le meilleur taux d'entropie par bit pour chaque TRNG. Si le taux d'entropie par bit n'était pas aussi proche de 1, ce qui est le cas bien souvent dans la réalité, le produit Entropie × Débit permettrait certainement de trouver un compromis entre le débit binaire et le taux d'entropie.

À présent, intéressons-nous aux résultats comparatifs des TRNG pour chaque critère d'évaluation. Au regard du tableau 2.2, il est clair qu'aucun générateur n'obtient le meilleur résultat pour tous les critères. Par exemple, le ERO-TRNG possède la meilleure répétabilité et faisabilité, mais le plus faible débit binaire. Le COSO-TRNG est parfait en matière de surface d'occupation du FPGA, mais il se trouve être parmi les plus compliqués à implanter correctement. Le MURO-TRNG est un bon compromis entre le débit binaire et la faisabilité, mais il est très sensible au phénomène de verrouillage. Le PLL-TRNG représente sans doute le meilleur compromis parmi tous les critères d'évaluation, mais n'obtient aucun résultat remarquable. Le TERO-TRNG semble prometteur notamment d'un point de vue de son efficacité énergétique, mais, comme le COSO-TRNG, il est très compliqué à implanter correctement. Le STR-TRNG avec un débit binaire fulgurant est très approprié pour des applications à grande vitesse, mais il consomme beaucoup d'énergie et occupe une surface très importante sur le FPGA.

Ainsi, cette comparaison nous amène à conclure qu'aucun candidat idéal de TRNG n'existe. Un compromis entre les critères d'évaluation présentés ci-devant devra toujours être fait. Le choix du type de TRNG devra prendre en compte l'environnement et les contraintes liées au système qui l'accueil.

2.1.5 Conclusion

Dans cette partie, nous avons étudié les résultats d'implantation des TRNG à base de cellules oscillantes. À travers cette étude, nous avons pu voir que tous les TRNG sélectionnés étaient réalisables sur les principales familles de FPGA existantes.

Cependant, deux des six TRNG étudiés ne sont, en l'état, pas utilisable pour des applications industrielles nécessitant une répétabilité de l'implantation. En effet, le COSO-TRNG et le TERO-TRNG nécessitent un placement manuel minutieux des éléments pour chaque circuit.

Cette analyse nous a aussi permis de comprendre qu'aucun TRNG idéal n'existe et qu'un compromis devra toujours être fait. Le TRNG le plus approprié doit être sélectionné en fonction des contraintes de l'application.

Les deux critères d'évaluation proposés (l'efficacité énergétique et le produit Entropie × Débit) aideront le concepteur à faire ce choix.

La plupart des critères d'évaluation que nous avons utilisés sont dépendants du matériel d'implantation choisi. C'est pourquoi, il était essentiel d'utiliser la même plateforme d'évaluation et

dans les mêmes conditions pour comparer correctement les TRNG.

Intéressons-nous à présent à la conception de PUF à base de cellules oscillantes sur les FPGA de type Flash.

2.2 Conception de PUF à base de cellules oscillantes sur FPGA de type Flash

Dans le chapitre précédent, nous avons pu constater que, tout comme pour les TRNG, les meilleurs candidats de PUF pour une implantation sur FPGA étaient les PUF à base de cellules oscillantes (voir sec. 1.3.2). Les études d'implantation de TERO-PUF de Marchand *et al.* sur FPGA de type SRAM nous ont aussi permis de voir qu'une implantation rigoureuse de PUF était fortement liée au type de FPGA [72], [90].

Dans cette partie, nous allons voir comment réaliser une implantation de RO-PUF et TERO-PUF rigoureuse sur les FPGA de type Flash. Toutes les implantations ont été réalisées sur des FPGA SmartFusion 2 du fabricant Microsemi à l'aide du logiciel d'implantation Microsemi Libero v11.7.

2.2.1 Structure des PUF implantées

Les PUF à base de cellules oscillantes implantées sont composées de $2 \times m$ cellules oscillantes, 2 compteurs et 1 comparateur comme représenté sur la figure 2.3. Les cellules sont divisées en deux blocs, A et B, de m cellules. Afin d'éviter tout risque de corrélation, comme expliqué précédemment (voir sec. 1.3.2), une cellule du bloc A sera toujours comparée à une cellule du bloc B. De plus, une cellule est utilisée pour une seule comparaison. Ainsi, une cellule de chaque bloc est sélectionnée à l'aide de deux démultiplexeurs. Les sorties de ces deux cellules sont envoyées sur les compteurs.

Pour rappel, la RO-PUF extrait les variations de procédés de fabrication d'un circuit électronique numérique en comparant les fréquences d'oscillations de deux cellules RO implantées identiquement. Elle est composée de 128 cellules RO par bloc, elles-mêmes composées de 9 inverseurs. Les cellules comparées sont démarrées en même temps. À partir du moment où un des deux compteurs atteint sa valeur maximale, un arbitre arrête les compteurs. Si le compteur du bloc A atteint sa valeur maximale en premier, l'arbitre génère un '1' logique en sortie de la PUF. Sinon, il génère un '0' logique. Cette comparaison est faite pour toutes les cellules RO des blocs A et B et les bits résultants sont concaténés pour former une réponse de 128 bits.

. Le code VHDL associé à cette section est disponible à l'adresse suivante : https://gitlab.univ-st-etienne.fr/ugo.mureddu/flash_fpga_puf_source_code

Dans le cas de la TERO-PUF, c'est le nombre d'oscillations qui est comparé. De ce fait, l'arbitre à la sortie des compteurs est remplacé par un soustracteur. Cette structure permet d'extraire entre 1 et 3 bits par comparaison. Comme l'expliquent Marchand *et al.* [90], les compteurs et le temps d'activation des cellules TERO doivent être dimensionnés en accord avec le nombre d'oscillations moyen des cellules. En effet, si les compteurs sont surdimensionnés, les bits de poids forts ne passeront jamais à '1' et la comparaison de ceux-ci ne permettra d'extraire aucun bit. De plus, ils occuperont une place supplémentaire sur le FPGA. À l'inverse, s'ils sont sous-dimensionnés, il y aura un risque fort de dépassement des compteurs. Pour cette étude, la TERO-PUF est composée de $m = 64$ cellules TERO par bloc. Les cellules TERO sont composées de 7 inverseurs par branche ($N = 7$), les compteurs sont de 10 bits et le temps d'activation est configuré à 1 μs. Les bits 5 et 9 du soustracteur sont extraits après chaque comparaison pour former une réponse de 128 bits.

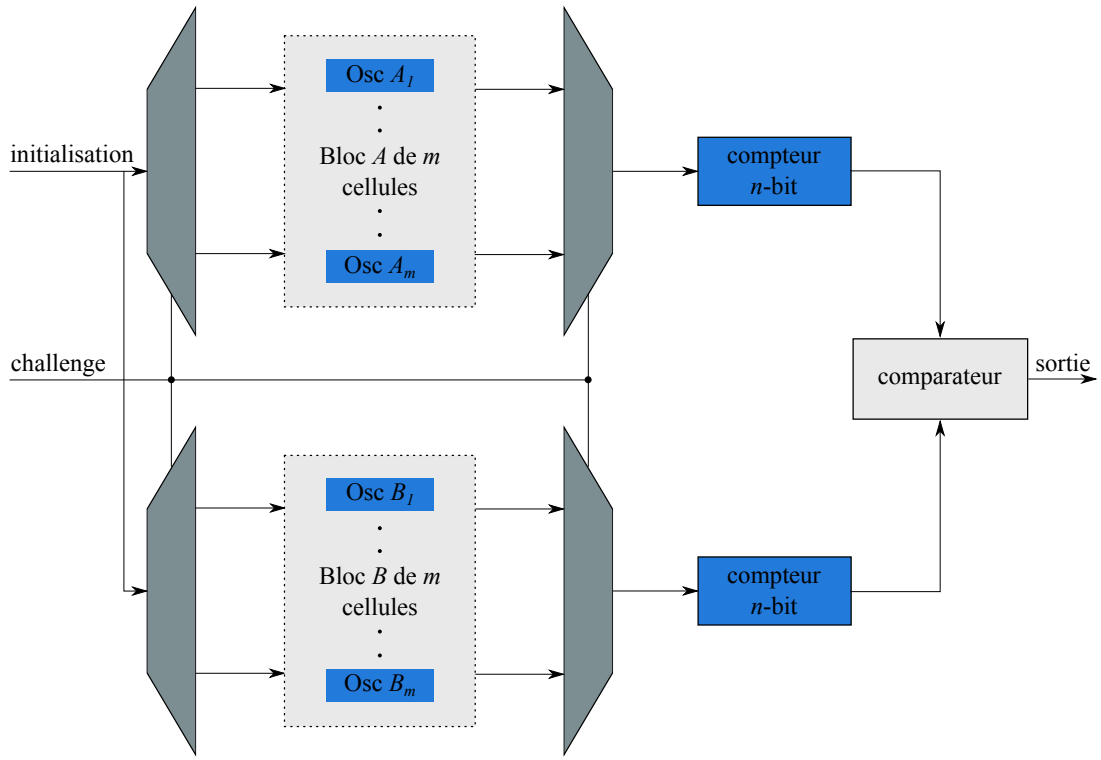


FIGURE 2.3: Architecture de PUF à base de cellules oscillantes

2.2.2 Méthodologie de conception

2.2.2.1 Conception de PUF à base de cellules oscillantes

Si l'implantation n'est pas réalisée avec attention, il peut arriver que le résultat de comparaison de deux cellules soit dépendant de la position de ces cellules sur le FPGA. Lorsque c'est le cas, les

résultats de comparaisons de ces deux cellules implantées sur plusieurs circuits seront biaisés. Par exemple, si la cellule oscillante du bloc A est routée sur le FPGA avec un délai plus important que la cellule oscillante du bloc B avec laquelle elle est comparée, la fréquence d'oscillation de la cellule du bloc A sera plus faible en moyenne et le résultat sera biaisé vers '0' sur tous les circuits. C'est pourquoi, pour obtenir des différences de fréquences d'oscillations dûes aux variations de procédés de fabrication et non à des différences de routage, il est primordial que les cellules soient implantées de manière identique. Les recommandations pour implanter les cellules de la même manière sur les FPGA du fabricant Microsemi sont les suivantes.

Non-modification de la structure de la cellule lors de la compilation La première chose à faire est de s'assurer que la structure de la cellule n'est pas altérée tout au long de la synthèse, du placement et du routage par le logiciel d'implantation Libero. En effet, Libero interprète et optimise constamment le code VHDL du concepteur avant de produire le bitstream final qui sera réellement implanté sur le FPGA. Malheureusement, il n'y a aucun moyen d'empêcher ces optimisations. Malgré tout, il y a certaines assurances que le concepteur doit prendre.

Premièrement, désactiver l'option de synthèse «retiming optimization», car celle-ci va insérer des «casseurs de boucles» — en anglais loop breakers. En effet, les boucles logiques sont fortement déconseillées lors d'utilisation classique de FPGA et le compilateur va avoir tendance à les supprimer. Dans notre cas, elles sont essentielles.

Deuxièmement, le code VHDL doit être aussi bas niveau que possible pour sûr être que le circuit décrit ne soit pas modifié lors de la synthèse logique. Par exemple, si le compilateur Libero identifie un RO composé de 5 inverseurs, il sera en mesure de comprendre que la même fonction logique peut être réalisée avec seulement un inverseur et supprimera les 4 autres inverseurs. Pour éviter cela sur les FPGA du fabricant Microsemi, il est nécessaire de n'utiliser que des composants de la bibliothèque associée à la famille de FPGA. Dans notre cas, nous utilisons la bibliothèque SmartFusion2. Elle fournit des éléments de base, comme une porte ET-logique à deux entrées ($AND2$) et un inverseur logique (INV), qui ne seront pas optimisés par l'outil.

Pour insérer cette bibliothèque, il faut utiliser les instructions VHDL suivantes :

```
library smartfusion2 ;
use smartfusion2.all ;
```

Contrôle du délai De la même manière, les routes empruntées entre les éléments de base des cellules vont être définies par le compilateur Libero et il n'existe aucun moyen d'imposer le routage au compilateur. S'il est nécessaire de traverser plusieurs registres ou portes pour connecter deux éléments entre eux, le délai sera très variable et son contrôle sera impossible. Bien qu'il ne soit pas

possible d'imposer le routage sur un FPGA, il est possible d'imposer le placement des éléments de base (*AND2* et *INV*). Ainsi, les éléments composants les cellules devront être placés côte à côte pour assurer que le compilateur choisira les routes les plus directes pour les relier. Ce placement des éléments est fait à l'aide de l'outil «Chip Planner Constraint Manager» de Libero ou grâce à un fichier de contraintes dont le format est .pdc. Chaque élément est placé dans une LUT du FPGA, représentant l'élément logique de base d'un FPGA. En effet, la partie logique d'un FPGA est généralement composée de LUT.

Les lignes suivantes donnent un exemple de placement de deux éléments *INV* côte à côte. Ce sont les lignes à ajouter au fichier de contraintes (.pdc).

```
set_location RO_PUF/ring_array_1.1.ring_n/gen_buff.0.buff_i/buff_wrp_i -fixed yes 13 54
set_location RO_PUF/ring_array_1.1.ring_n/gen_buff.1.buff_i/buff_wrp_i -fixed yes 14 54
```

Régions exclusives Afin d'assurer qu'aucun élément ne vient se placer au milieu des cellules et ne les perturbe, des régions exclusives doivent être créées. Ce sont des régions physiques sur le FPGA où seuls les éléments assignés peuvent y être placés. De cette manière, lors des processus de synthèse, de placement et de routage aucun autre élément ne sera ajouté au sein de ces régions. De plus, pour éviter qu'une ligne de routage ne traverse les cellules, le concepteur doit sélectionner l'option «Constrain routing» à la création de la région exclusive. Pour l'implantation des PUF à base de cellules oscillantes, il est nécessaire de créer une région par bloc de cellules.

Voici un exemple de création d'une région exclusive à l'aide du fichier de contrainte, puis de l'assignation d'une cellule RO à l'intérieur de cette région :

```
define_region -name RO_A -type exclusive -route YES -push_place NO 12 54 83 129
assign_region RO_A RO_PUF/ring_array_1.1.ring_n
```

La figure 2.4 montre une impression écran de l'outil «Chip Planner Constraint Manager» de Libero pour un exemple de placement des cellules d'une RO-PUF à l'intérieur de deux régions exclusives A et B.

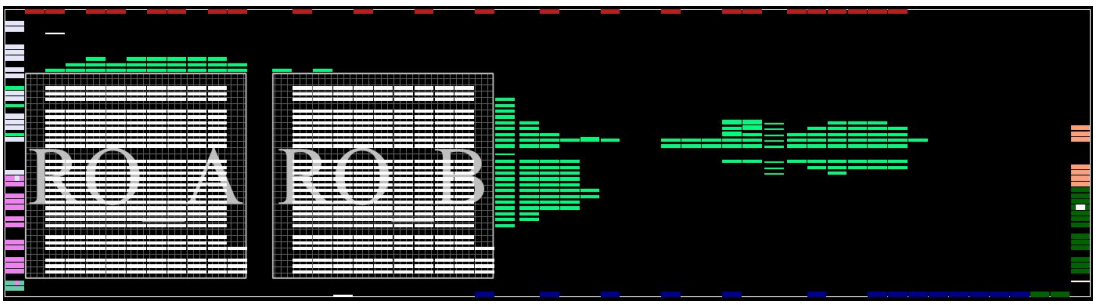


FIGURE 2.4: Exemple de placement des cellules RO à l'intérieur de deux régions exclusives A et B

Maximisation des variations locales En considérant à présent que les cellules sont implantées identiquement, la différence de fréquence ou de nombre d'oscillations entre les cellules n'est due qu'aux variations de procédés de fabrication. Cependant, comme nous l'avons vu dans le chapitre 1, il existe deux types de variations de procédés de fabrication : les variations globales qui sont déterministes et les variations locales qui sont stochastiques. Les variations globales induisent un gradient sur le FPGA qui, s'il impacte les cellules, créera un biais sur les réponses des PUF. Ainsi, pour limiter autant que possible ce gradient, les deux régions de cellules oscillantes doivent être le plus proche possible.

2.2.2.2 Contraintes additionnelles pour la RO-PUF

La ligne de délai à contrôler pour la RO-PUF est beaucoup plus longue que celle de la TERO-PUF. En effet, pour obtenir une comparaison de fréquences équitable des cellules, elles doivent être déclenchées en même temps et les délais entre les sorties des cellules et les entrées des compteurs doivent être les mêmes. Dans le cas de la TERO-PUF, si les cellules sont déclenchées avec un léger décalage, le résultat de l'arbitre ne sera pas influencé puisque ce qui compte c'est le nombre d'oscillations. Pour la RO-PUF, il y a un risque non nul que le décalage de déclenchement ou une différence de délai entre la sortie de la cellule et l'entrée du compteur fasse pencher la balance dans un sens comme dans l'autre.

Par conséquent, les deux multiplexeurs de la RO-PUF sont implantés au plus proche des cellules et de manière symétrique et identique. De plus, les compteurs sont placés côte à côte et les sorties des multiplexeurs sont envoyées aux compteurs à travers un registre d'horloge (*CLKBUF*) qui route le signal sur le réseau global d'horloge. Le réseau global d'horloge est composé de registres globaux qui distribuent les signaux d'horloge sur le FPGA avec une faible déviation. Cela permet l'implantation de fonctions logiques à haute fréquence sur FPGA. Le réseau global d'horloge permet de distribuer les horloges avec un faible décalage, peu importe la position des éléments sur le FPGA. De cette façon, le délai entre les sorties des cellules RO et les entrées des compteurs est mieux contrôlé. Pour se faire, le concepteur doit utiliser un autre élément de base de la bibliothèque SmartFusion2 : *CLKINT*.

Il n'est pas possible de réaliser la même chose avec le signal d'activation des cellules RO, car les ressources globales sont limitées. Cependant, nous venons de voir que l'horloge était distribuée de manière équilibrée sur le FPGA. Ceci signifie que les bascules D du FPGA vont commuter à des instants relativement proches. Ainsi, une bascule D est placée devant chaque cellule RO pour un contrôle plus fin de l'activation de celle-ci.

Ensuite, il convient d'améliorer la qualité du signal de sortie d'un RO en plaçant directement après la sortie de la cellule une bascule T. En effet, bien souvent le signal de sortie d'un RO présente de très faibles caractéristiques (*ex.* fronts de commutation distordus ou rapport cyclique

asymétrique). Placer une bascule T directement à la sortie va permettre d'obtenir un signal symétrique avec des fronts plus propres en sortie de la bascule. De plus, cela réduira la fréquence d'oscillation de la cellule par deux ce qui contribue à limiter les possibles défauts de compteurs identifiés par Wild *et al.* [73] sans affecter l'extraction des variations locales.

Enfin, les oscillations en sortie des cellules doivent être comptées pendant un temps suffisamment long. Autrement, la comparaison sera bruitée. Plus le comptage dure plus la fréquence d'oscillation sera moyennée efficacement. En effet, bien que le jitter soit cumulatif sa valeur moyenne en fonction du temps est nulle alors que les différences induites par les variations locales ne feront qu'augmenter avec le temps de comparaison. Pour cette raison, la taille des compteurs doit être dimensionnée en accord avec le reste du circuit. Cela permettra d'améliorer la stabilité des réponses des PUF.

2.2.3 Protocole expérimental

La caractérisation de la RO-PUF et de la TERO-PUF est réalisée sur 24 FPGA SmartFusion2 de Microsemi. Afin d'effectuer cette caractérisation sur un plus grand nombre d'implantations, les circuits sont verrouillés et placés à différentes positions sur les FPGA. De cette manière, la même structure de PUF utilise d'autres transistors du FPGA et ainsi extrait d'autres variations locales. Chaque PUF est placée à deux positions différentes sur le FPGA. La caractérisation est donc réalisée sur 48 implantations de PUF. Chaque réponse de 128 bits est collectée un millier de fois. Cette caractérisation est faite à l'aide d'une plateforme FPGA développée spécialement pour le projet HECTOR. Le fonctionnement et la structure de la plateforme HECTOR sont décrits dans la section suivante.

2.2.3.1 La plateforme HECTOR

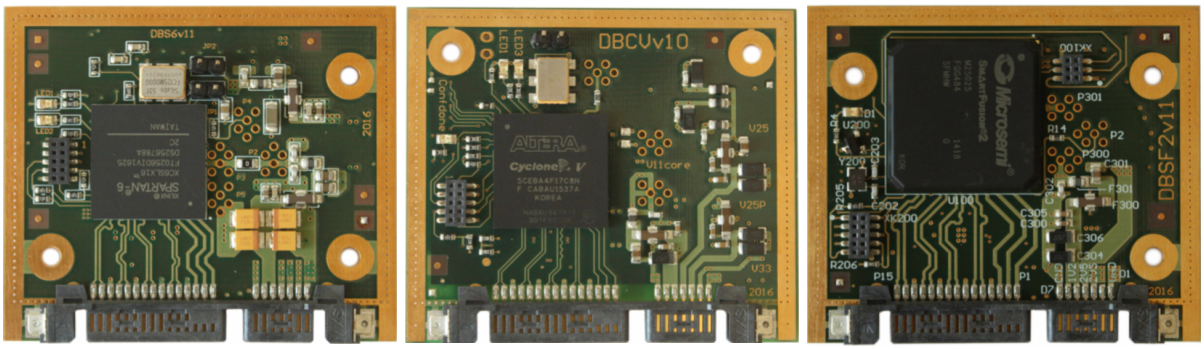
La plateforme HECTOR a été conçue pour être un outil modulaire, optimisé pour l'évaluation de primitives cryptographiques implantées sur les circuits FPGA [92]. Elle est composée d'une carte mère et de multiples cartes filles interchangeables. Les primitives cryptographiques sont implantées sur les cartes filles dont le circuit a été pensé pour n'inclure que l'essentiel et ainsi perturber le moins possible les primitives testées. Les données sont stockées, traitées et transmises à un ordinateur par la carte mère proposant un large choix de périphériques et d'interfaces. La connexion entre la carte mère et une carte fille est effectuée soit à l'aide d'un connecteur SATA soit à l'aide d'un connecteur HDMI permettant de déporter la carte fille pour des tests en environnement dit «hostiles» pour réaliser des attaques.

Cartes filles Les modules cartes filles de la plateforme HECTOR ont été conçus pour permettre l'évaluation de primitives cryptographiques sur différentes familles de FPGA. Cette architecture

présente deux avantages. Le premier est que les cartes filles ne contiennent que les blocs matériels nécessaires et minimisent l'impact de composants bruités sur le fonctionnement des primitives. Le second est que de cette manière les cartes filles ont un coût de fabrication faible permettant de créer un grand nombre de cartes pour caractériser les PUF.

Trois types de cartes filles ont été réalisés : une carte fille avec un FPGA Cyclone V du fabricant Intel, une carte fille avec un FPGA Spartan 6 du fabricant Xilinx et une carte fille avec un FPGA SmartFusion2 du fabricant Microsemi. Ces cartes filles sont visibles sur la figure 2.5. Les connecteurs SATA et HDMI ont été choisis pour leur qualité de transmission de signal, mais n'utilisent pas les protocoles de communications SATA ou HDMI. À la place, ils servent à alimenter les cartes filles avec des alimentations fournies par la carte mère et assurent la communication à l'aide de quatre lignes LVDS et trois lignes de données standards.

Enfin, les cartes filles contiennent des filtres d'alimentation de haute qualité et une horloge à quartz.



(a) Carte fille avec FPGA Xilinx Spartan 6 (b) Carte fille avec FPGA Intel Cyclone V (c) Carte fille avec FPGA Microsemi SmartFusion2

FIGURE 2.5: Cartes filles de la plateforme HECTOR

Carte mère La fonction principale de la carte mère est de contrôler les cartes filles et d'assurer le transfert des données entre les cartes filles et un ordinateur. Pour communiquer avec l'ordinateur, la carte mère utilise une interface USB. Elle est composée d'un système sur une puce — en anglais system on chip (SoC) — Microsemi SmartFusion2 qui intègre un FPGA de type Flash et un processeur Cortex-M3 de ARM. La partie FPGA du SoC traite les parties critiques et la partie processeur gère la communication.

La carte mère contient une mémoire externe DDR SDRAM de 64 Mo nécessaire pour l'acquisition de longs flux continus de données (*ex.* données d'un TRNG). Elle est alimentée par une alimentation externe de 5 V. Pour limiter le coût des cartes filles, elles sont alimentées par des régulateurs de tension linéaires placés sur la carte mère. Une photo de la carte mère est représentée sur la figure 2.6.

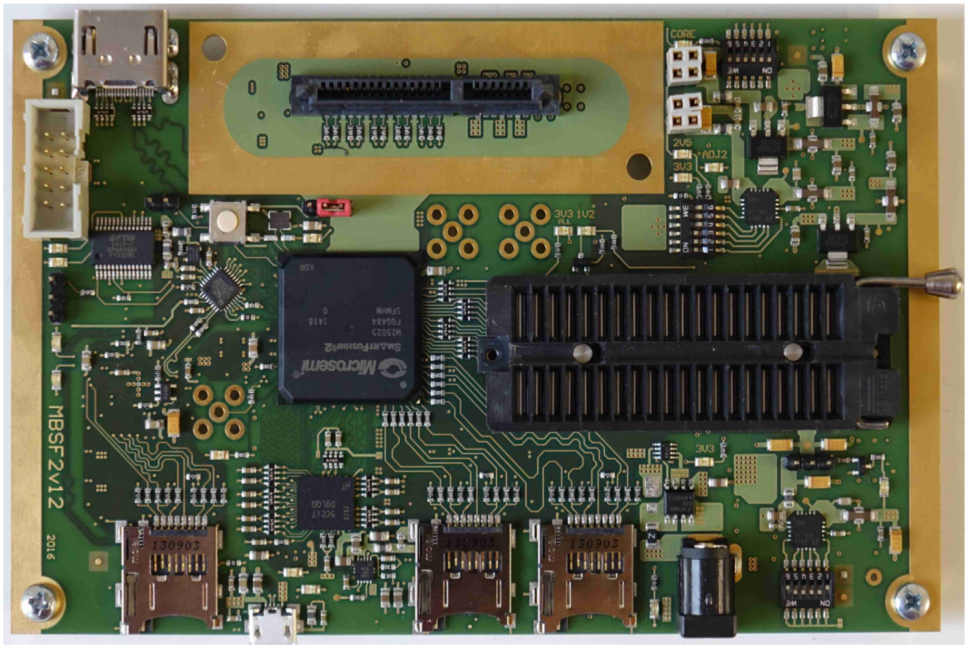


FIGURE 2.6: Carte mère de la plateforme HECTOR

2.2.4 Métriques de caractérisation

Afin d'évaluer la qualité d'implantation des RO-PUF et TERO-PUF, nous utiliserons les métriques de caractérisation introduites dans la partie 1.1.2. Dans cette partie, nous avions vu que la mesure de biais était redondante avec la mesure de l'unicité. De plus, dans notre cas, chaque cellule n'est utilisée que pour une seule comparaison, excluant toute possibilité de corrélation entre les bits d'une réponse. Ainsi, nous utiliserons la mesure de la stabilité (HD_{int}) et la mesure de l'unicité (HD_{ext}). Pour rappel, ces métriques permettent d'identifier les défauts d'une implantation de PUF, mais ne garantissent pas l'imprévisibilité de celle-ci.

2.2.5 Résultats de caractérisation

2.2.5.1 Unicité

La figure 2.7 représente les distributions discrètes de l'unicité d'une RO-PUF pour une implantation non optimisée ainsi que pour une implantation qui suit les méthodologies proposées dans la section 2.2.2. Une approximation de la loi normale correspondante est tracée pour chaque cas. De plus, la loi normale d'une unicité parfaite ($\mathcal{N}(\mu = 50\%, \sigma^2 = 25\%)$) est aussi tracée en pointillés pour comparaison où μ représente la moyenne et σ^2 représente la variance. Pour rappel, ces tests d'unicité sont réalisés sur 48 implantations de chaque principe de PUF. L'amélioration de l'unicité à la suite des optimisations de l'implantation est très nette. En effet, la valeur moyenne (μ) de l'unicité passe de 12,1 % à 42,2 %.

La figure 2.8 représente les mêmes distributions de l'unicité pour une TERO-PUF. Cette fois encore, l'amélioration de l'unicité est très importante avec une valeur moyenne passant de 28,3 % à 48,1 %.

Pour les deux types de PUF, la variance (σ^2) reste dans une gamme de valeurs appropriées puisqu'elle est inférieure à 25 %.

Malgré ces améliorations, les résultats d'unicité de la RO-PUF pour une implantation optimisée reflètent soit une corrélation entre les réponses des différentes implantations soit un biais vers '0' des réponses. Pour le découvrir, il convient dans un premier lieu de vérifier s'il n'y a pas de bits des réponses qui sont toujours à '0'. Dans notre cas, il se trouve qu'il y a effectivement un bit qui est à '0' sur toutes les réponses de toutes les implantations. Ceci explique le décalage vers 0 de l'unicité de la RO-PUF. Si après avoir effectué toutes les améliorations de la section 2.2.2, il existe encore des différences de routage plus importantes que les variations de procédés de fabrication, cela signifie soit que le principe de RO-PUF n'extrait pas suffisamment les variations de procédés de fabrication pour être réalisable sur les FPGA de type Flash, soit que l'implantation a besoin d'améliorations supplémentaires.

Un point d'amélioration envisageable et que nous n'avons pas eu le temps de tester durant ces travaux, est de mieux contrôler la structure du multiplexeur à la sortie cellules RO. Pour cela, il est possible de réaliser des multiplexeurs plus symétriques en utilisant un élément de base de la bibliothèque SmartFusion2, l'élément *MUX4*.

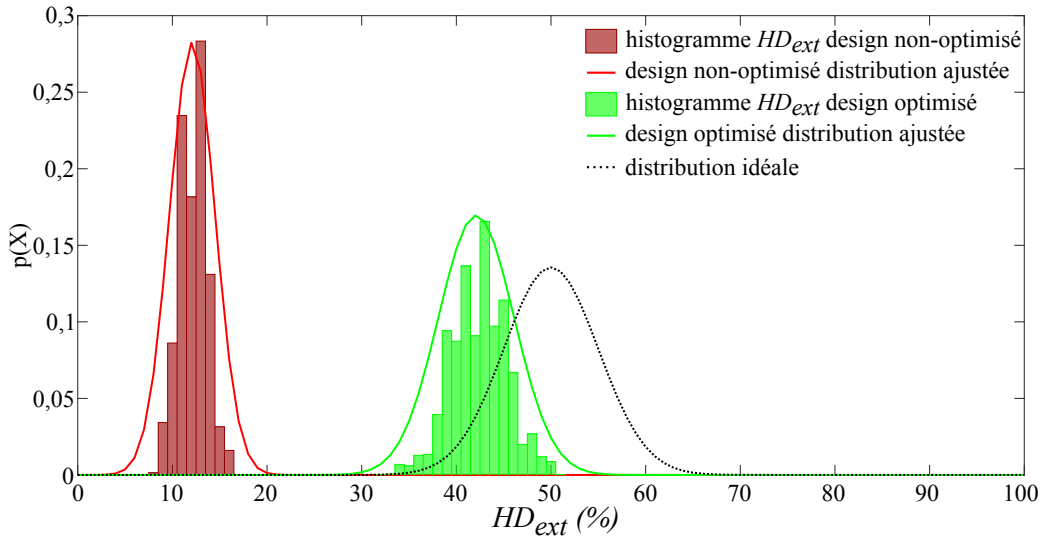


FIGURE 2.7: Unicité de la RO-PUF

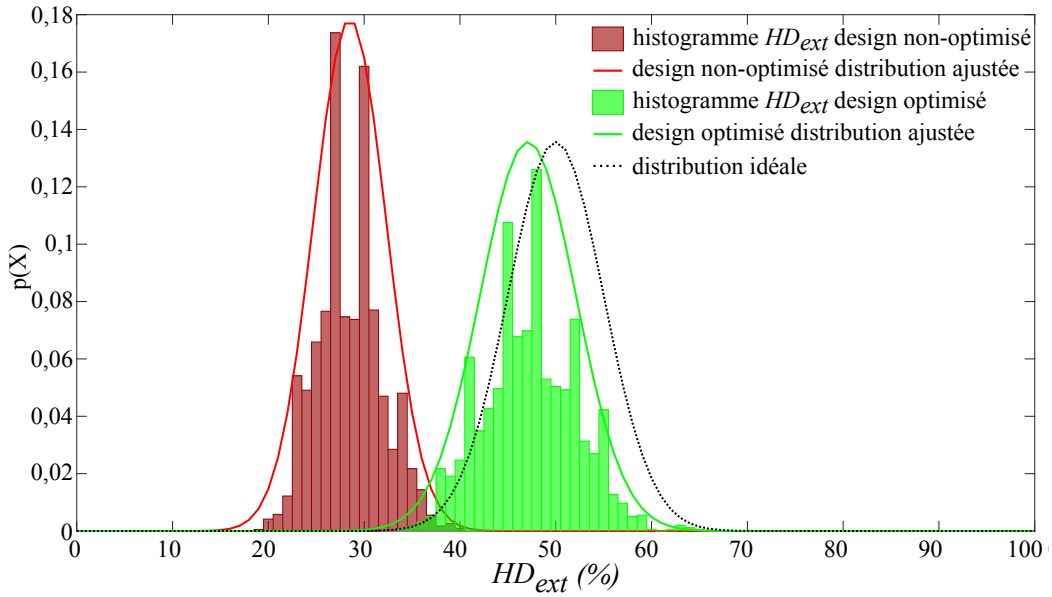


FIGURE 2.8: Unicité de la TERO-PUF

2.2.5.2 Stabilité

Après la prise en compte de toutes les améliorations d'implantation proposées ci-devant, la stabilité moyenne obtenue pour la RO-PUF est de $HD_{int} = 1,68\%$. Celle de la TERO-PUF est de $HD_{int} = 1,52\%$. Pour rappel, pour réaliser ces mesures de stabilité, les réponses de chaque implantation de chaque type de PUF sont récupérées un millier de fois.

Ces mêmes mesures pour des implantations non optimisées donnent une stabilité $HD_{int} = 0,69\%$ pour la RO-PUF et $HD_{int} = 1,3\%$ pour la TERO-PUF. À première vue, les implantations de RO-PUF et TERO-PUF semblent présenter une meilleure stabilité avant optimisation. Cependant, quand l'unicité d'une PUF est trop éloignée de 50 %, les résultats de stabilité même proches de 0 % ne peuvent pas témoigner d'une bonne implantation. En effet, les réponses de plusieurs implantations de PUF peuvent être à la fois très stables et présenter une unicité médiocre, si par exemple elles sont fortement influencées par le routage. Ainsi, observer un seul paramètre statistique à la fois peut être trompeur. Il est important de toujours observer la stabilité à la lumière de l'unicité et inversement.

Ensuite, nous avons étudié l'impact de la taille du compteur sur la stabilité en tension de la RO-PUF. La tension nominale du circuit est de 1,2 V. Les tests de stabilité ont été réalisés pour des compteurs de 7, 8, 9, 10, 11, 12 et 18 bits avec à chaque fois une tension variant de 1,1 à 1,26 V par pas de 0,02 V. Les résultats sont représentés sur la figure 2.9. Ils montrent très clairement que, en particulier pour des tensions d'alimentation extrêmes, la taille du compteur influe sur la stabilité de la RO-PUF. Pour un compteur de 7 bits, HD_{int} peut aller jusqu'à une valeur maximale de 27 %, alors qu'avec un compteur de 11 bits il ne dépasse jamais 5 %.

Cependant, après une certaine taille de compteur, la stabilité ne s'améliore plus. C'est pour cette raison que nous avons choisi pour nos expérimentations des compteurs de 11 bits. Ce dimensionnement des compteurs est spécifique à la technologie utilisée et doit être réalisé pour chaque nouvelle implantation.

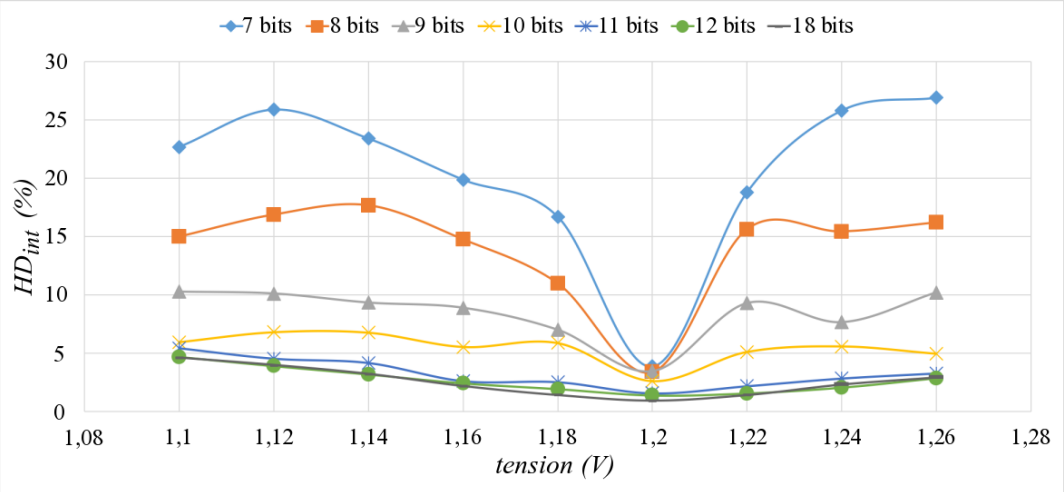


FIGURE 2.9: Stabilité de la RO-PUF en fonction de la tension et de la taille du compteur

2.2.6 Comparaison

Les résultats d'implantation de la RO-PUF et de la TERO-PUF pour des implantations non optimisées et des implantations suivant les recommandations données dans la section 2.2.2 sont résumés dans le tableau 2.3.

Métriques	**RO-PUF**			**TERO-PUF**			
Cible matérielle	Xilinx [16] Spartan-3E	Microsemi SmartFusion 2		Xilinx [72] Spartan-6	Altera [72] Cyclone-V	Microsemi SmartFusion 2	
		non optimisé	optimisé			non optimisé	optimisé
Unicité	47,3 %	12,1 %	42,2 %	48,5 %	47,6 %	28,3 %	48,1 %
Stabilité	0,9 %	non interprétable	1,68 %	2,6 %	1,8 %	non interprétable	1,52 %

TABLE 2.3: Résumé des résultats de caractérisation des RO-PUF et TERO-PUF

À titre de comparaison, les résultats d'implantation de RO-PUF [16] et TERO-PUF [72] sur les FPGA de type SRAM sont aussi représentés dans ce tableau. Ce tableau représente la stabilité et l'unicité moyenne pour chaque implantation. Cette comparaison reflète pleinement l'amélioration de la qualité des PUF sur les FPGA de type Flash en suivant les méthodologies proposées précédemment.

Comme nous l'avons vu auparavant, la stabilité ne peut pas être interprétée si l'unicité est trop éloignée de 50 %. C'est pourquoi dans le tableau, pour une implantation non optimisée, la

stabilité est reportée comme «non interprétable».

Nous pouvons constater que l'implantation de la TERO-PUF sur les FPGA de type Flash présente de meilleurs résultats, notamment en matière d'unicité, que la RO-PUF. Ceci est probablement dû au fait que le délai à contrôler pour comparer deux cellules TERO de manière équitable est plus court que celui permettant de comparer deux cellules RO.

Enfin, la comparaison des résultats de caractérisation de la TERO-PUF optimisée avec les travaux réalisés sur les FPGA de type SRAM montre que sa stabilité ($HD_{int} = 1,52\%$) et son unicité ($HD_{ext} = 48,1\%$) se situent dans des plages de valeurs appropriées.

2.2.7 Conclusion

Dans cette partie, nous venons de voir, pour la première fois, la démarche à suivre pour réaliser une implantation efficace de PUF à base de cellules oscillantes sur les FPGA de type Flash. Ces implantations ont été réalisées au plus bas niveau de conception possible. C'est-à-dire, au niveau le plus proche du matériel. Un certain nombre d'améliorations a été proposé. Premièrement, nous avons vu comment assurer que la structure de la cellule ne soit pas modifiée au cours du processus de synthèse, de placement et de routage par le compilateur Libero. Deuxièmement, nous avons vu comment contrôler le délai des éléments critiques de la PUF. Troisièmement, nous avons vu comment isoler les cellules oscillantes du reste du circuit afin d'éviter toute possibilité de perturbation. Ensuite, nous avons vu comment maximiser les variations locales. Enfin, nous avons vu comment améliorer la qualité du signal en sortie des cellules RO.

Bien que les résultats de caractérisation confirment l'efficacité de ces améliorations, la qualité de la RO-PUF reste faible avec une unicité de 42 %. Pour cette raison, nous avons choisi d'utiliser la TERO-PUF pour l'implantation au sein des démonstrateurs présentés dans la partie suivante.

2.3 Intégration de TRNG et de PUF au sein d'un système complet

2.3.1 Système sécurisé USB portable de stockage de données

L'utilisation de clés USB est très répandue pour le stockage ou le transfert de données. Cet outil peut contenir des informations personnelles sensibles ou des documents professionnels confidentiels. La confidentialité de ces informations peut être compromise dans de nombreux cas de figure. Par exemple, brancher sa clé USB sur un ordinateur inconnu, l'égarer ou encore se la faire dérober.

Pour prévenir de tels risques, un système sécurisé USB portable de stockage de données est nécessaire. Ce type de système est déjà disponible sur le marché, mais il existe un manque de solution fiable conçue et fabriquée en Europe. De plus, afin de prouver l'efficacité des primitives cryptographiques étudiées pendant le projet HECTOR au sein d'un système haut de gamme de sécurité des données, les membres du projet HECTOR ont décidé de réaliser un prototype de ce système.

Ce système inclut les primitives suivantes : un PLL-TRNG conforme à la norme AIS31 avec ses tests embarqués, un système de chiffrement authentifié appelé ASCON et une TERO-PUF. L'intégration de ces primitives au sein d'un système clé en main n'a encore jamais été réalisée. Le système décrit dans la partie suivante est un prototype qui démontre la faisabilité industrielle d'un tel produit. Il a été réalisé par les membres du projet HECTOR dans lequel s'inscrit cette thèse. Ainsi, tous les travaux présentés ci-après sont le fruit d'un travail de collaboration entre les partenaires du projet. La plateforme matérielle de ce démonstrateur est inspirée d'un prototype de l'entreprise Micronic AS, membre du projet, déjà existant au début du projet.

2.3.1.1 Plateforme matérielle

Le circuit de la plateforme est composé d'un SoC FPGA SmartFusion 2 du fabricant Microsemi, d'un clavier tactile capacitif intégré, d'un écran LCD, d'un lecteur de carte SD, de LED et d'un connecteur USB assurant la connexion avec le monde extérieur et l'alimentation comme représenté sur la figure 2.10. L'écran affiche l'état de l'appareil et le résultat des opérations effectuées par l'utilisateur. Le clavier permet de contrôler le périphérique et de saisir une phrase secrète. La carte SD est connectée au FPGA à l'aide des broches d'entrées/sorties. Le FPGA communique avec la carte à un débit allant jusqu'à 25 Mo/s. Le SoC Microsemi SmartFusion2 assure l'interface USB avec l'ordinateur, l'authentification du système, le chiffrement et

. La vidéo du démonstrateur associé à cette section est disponible à l'adresse suivante : https://vimeo.com/289259101

le déchiffrement des données stockées. Le SoC est composé d'un FPGA de type Flash à faible consommation avec un processeur ARM 32 bits.

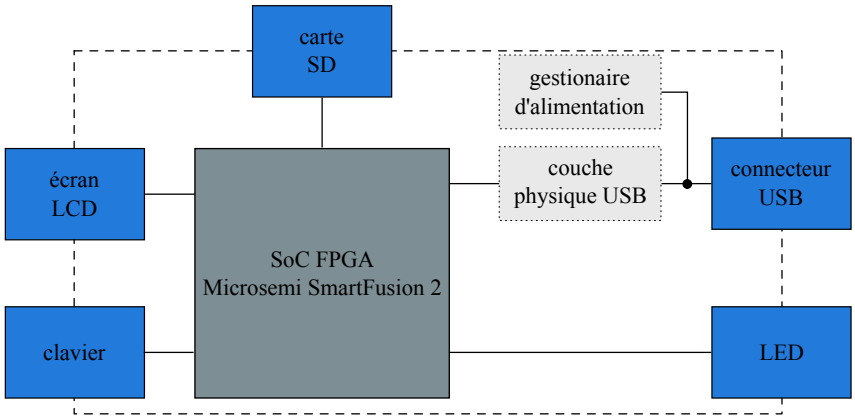


FIGURE 2.10: Schéma bloc du prototype USB portable de stockage de données sécurisées

Toutes les données de la carte SD sont chiffrées, protégées par un algorithme cryptographique puissant, résistant ainsi aux attaques hors ligne. Dans le cadre du scénario des objets perdus et trouvés/volés, la solution doit être sécurisée contre les attaques par force brute et aucune donnée sensible ne doit être stockée sur l'appareil en clair.

2.3.1.2 Blocs élémentaires du démonstrateur

Les blocs élémentaires essentiels au bon fonctionnement du système USB sécurisé sont les suivants.

Phrase secrète La phrase secrète est composée de 22 caractères parmi les suivants : $\alpha = [A - Z,\ a - z,\ 0 - 9,\ @\ \&\ :\ \_\ !\ ?\ |\ .\ ,\ \star\ +\ /]$. Elle permet de générer une clé de chiffrement de 128 bits, notée K_{phs}. Cette clé masquée par la réponse R de la PUF va servir de clé de chiffrement de l'appareil, notée K_{sys}.

TERO-PUF La PUF utilisée est strictement la même que celle présentée dans la partie précédente (voir sec. 2.2 et fig.2.3), une TERO-PUF composée de $m = 64$ cellules TERO par bloc, elles-mêmes composées de $N = 7$ inverseurs par branche. Les compteurs sont de $n = 10$ bits et le temps d'activation est configuré à 1 μs. Les bits 5 et 9 du soustracteur sont extraits après chaque comparaison pour former une réponse de 128 bits.

Afin d'être utilisée comme clé, la réponse R de la PUF doit être parfaitement stable. Nous avons vu dans la partie 2.2 que ce n'était pas le cas. Pour cette raison, deux post-traitements ont été appliqués à celle-ci. Tout d'abord, une étude de la stabilité de chaque bit des réponses en température et en tension a été réalisée. Les 13 bits les plus instables ont été retirés. Ensuite,

un code correcteur d'erreurs, le code de Golay, est appliqué sur les 115 bits restant pour établir une clé de 32 bits.

Chiffrement authentifié La primitive de chiffrement authentifié, ASCON-128a [93], est utilisée à la fois pour la reconstruction de clé et le chiffrement des données. L'instance ASCON est contrôlée par une machine à états pour traiter (chiffrer/déchiffrer) 32 blocs de données de 128 bits (un secteur de 512 octets). Pour des raisons d'optimisation, le processus peut s'étendre sur plusieurs secteurs de 512 octets. Le chiffrement/déchiffrement authentifié nécessite l'utilisation d'un nonce et créer un tag d'authentification des données. Un nonce est un nombre aléatoire à utilisation unique. Il s'agit souvent d'un nombre aléatoire (ou pseudo-aléatoire) utilisé dans un protocole d'authentification pour garantir que les anciennes communications ne peuvent pas être réutilisées dans des attaques par rejeu. Un nonce peut aussi désigner un vecteur d'initialisation. Les données chiffrées, leurs tags et leurs nonces associés seront stockés sur la carte SD.

PLL-TRNG Le PLL-TRNG implanté est celui étudié dans la partie précédente (voir sec. 2.1 et fig.1.12). L'horloge d'entrée est générée par l'oscillateur RC interne au SoC SmartFusion2. Sa fréquence est de 50 MHz. Les facteurs de multiplication et de division de la première PLL sont : $K_{M1} = 31$ et $K_{D1} = 12$. Ceux de la deuxième PLL sont : $K_{M2} = 37$ et $K_{D2} = 7$. Ainsi, les facteurs de multiplication et de division finaux sont : $K_M = 444$ et $K_D = 217$. Cela donne une fréquence de 129,17 MHz pour l'horloge de référence (H_{ref}), 264,28 MHz pour l'horloge soumise au jitter (H_{jit}) et un débit binaire de 600 kbit/s. Étant donné que les coefficients des deux PLL sont définies de telle sorte que le taux d'entropie par bit à la sortie du décimateur soit supérieur à celui requis par la norme AIS31, le post-traitement algorithmique n'est pas nécessaire.

Le PLL-TRNG utilise un test d'échec total et deux tests en ligne. Si un des trois tests déclenche une alarme, un signal d'interruption est généré.

Le PLL-TRNG génère la clé de chiffrement des données K_{don} et les nonces utilisés pour chaque phase de fonctionnement des processus d'authentification et de chiffrement. Les phases de fonctionnement sont décrites ci-après.

2.3.1.3 Fonctionnement du prototype

Ce système est un appareil à usage personnel. Il protège les données stockées lorsqu'il est à l'arrêt. L'utilisateur doit s'authentifier avant de permettre l'accès aux données. Il est alimenté par le port USB (*ex.* connexion à un ordinateur) et ne nécessite pas de source d'alimentation externe.

Le système a deux phases de fonctionnement : une phase d'enregistrement et une phase d'utilisation. Le système doit être enregistré avant la première utilisation. Lors de la phase

d'enregistrement, le système est initialisé et les clés de chiffrement sont générées. Puis une phrase secrète représentant la clé principale est affichée sur l'écran LCD. Si cette phrase secrète ne convient pas, l'utilisateur peut la régénérer. Cette phrase doit être mémorisée par l'utilisateur. Ensuite, l'utilisateur doit entrer cette phrase à l'aide du clavier du système. Une fois le système enregistré, il doit être redémarré pour l'utilisation.

Après le redémarrage, l'utilisateur s'authentifie auprès de l'appareil en entrant une phrase secrète. La phrase secrète ne quitte jamais le périphérique USB, car elle est tapée sur le clavier intégré et affichée sur l'écran intégré. Une fois la clé entrée, le système de stockage va apparaitre dans le système d'exploitation comme une clé USB classique et pourra être utilisé pour sauvegarder des données de manière sécurisée. La confidentialité des données est assurée par la phrase secrète et la PUF. Les données sont stockées et chiffrées sur la carte SD. Si l'utilisateur entre une mauvaise phrase secrète plusieurs fois, les données sont supprimées du système.

Phase d'enregistrement Durant cette phase, les différentes clés permettant le chiffrement et le déchiffrement des données (K_{don}) et l'authentification de l'appareil (R) et de l'utilisateur (K_{phs}) sont générées. R et K_{phs} sont couplées pour générer une clé d'authentification de la paire utilisateur/appareil (K_{sys}). La clé K_{sys} et un nonce produit par le PLL-TRNG ($Nonce_1$) sont utilisés par l'algorithme ASCON pour chiffrer K_{don} ($enc(K_{don})$). $Nonce_1$, $enc(K_{don})$ et un tag associé au chiffrement de K_{don} ($tag(K_{don})$) sont stockés dans la mémoire non volatile. La figure 2.11 représente un schéma bloc de la phase d'enregistrement.

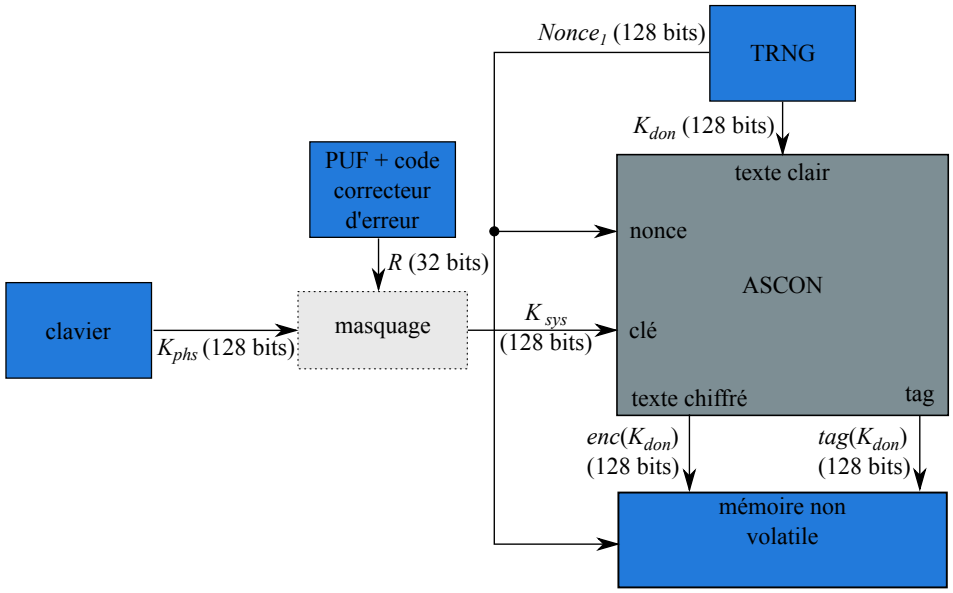


FIGURE 2.11: Phase d'enregistrement

Phase d'activation Au redémarrage du système enregistré, l'utilisateur entre la phrase secrète permettant de régénérer K_{phs}. En parallèle, la PUF régénère la réponse R. Ces deux clés sont couplées pour régénérer K_{sys}. $Nonce_1$, $enc(K_{don})$ et $tag(K_{don})$ sont lues depuis la mémoire non volatile. Le couple K_{sys} et $Nonce_1$ est utilisé par l'algorithme ASCON pour déchiffrer K_{don}. Le tag de déchiffrement ($tag'(K_{don})$) est comparé au tag de chiffrement ($tag(K_{don})$). Si les deux tags sont égaux, le système est activé et K_{don} est stocké dans le registre de clé. Sinon le processus d'activation est relancé. À partir de cet instant, le système peut être utilisé pour stocker ou relire de manière sécurisée les données. La figure 2.12 représente un schéma bloc de la phase d'activation.

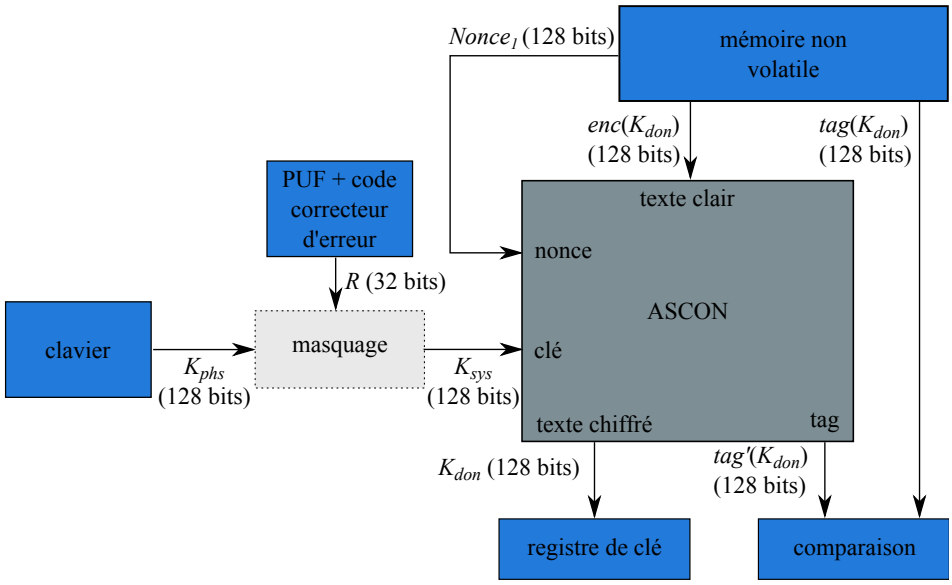


FIGURE 2.12: Phase d'activation

Processus de chiffrement des données Les phases de chiffrement et de déchiffrement des données sur le système USB sécurisé une fois qu'il a été activé sont les suivantes.

Phase de chiffrement Les données envoyées depuis l'ordinateur sont chiffrées et stockées par paquets de 512 octets. Un nonce est généré grâce au PLL-TRNG et à un compteur ($Nonce_x$). L'ajout du compteur dans le processus de création du nonce permet d'ordonner les données chiffrées. K_{don} est lu depuis le registre de clé puis, couplé à $Nonce_x$ pour permettre à l'algorithme ASCON de chiffrer les données par paquet de 512 octets. $Nonce_x$, le paquet de données chiffrées et le tag associé sont stockés ensemble sur la carte SD. La figure 2.13 représente un schéma bloc de la phase de chiffrement des données.

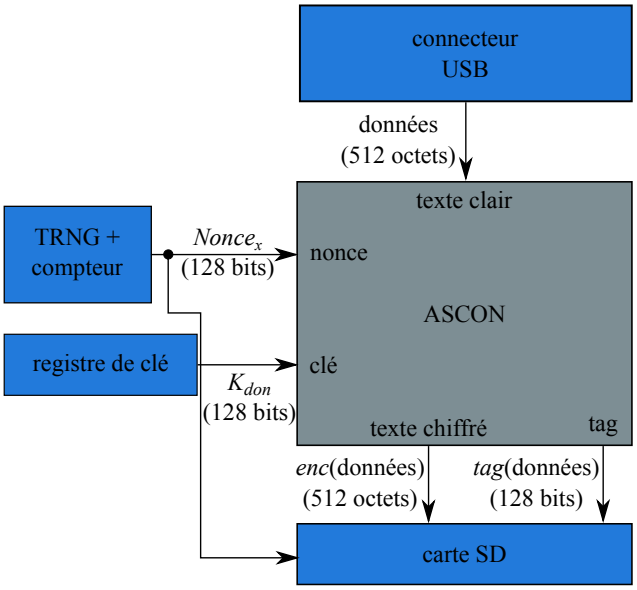


FIGURE 2.13: Chiffrement

Phase de déchiffrement Le principe est similaire pour le déchiffrement. Les données chiffrées, le tag et le nonce associés sont lus depuis la carte SD. Le couple $K_{don}/Nonce_x$ permet à l'algorithme ASCON de déchiffrer les données. Le tag de déchiffrement des données est comparé au tag de chiffrement. S'ils sont égaux, les données déchiffrées sont transférées au PC à l'aide du connecteur USB. Sinon une erreur est générée.

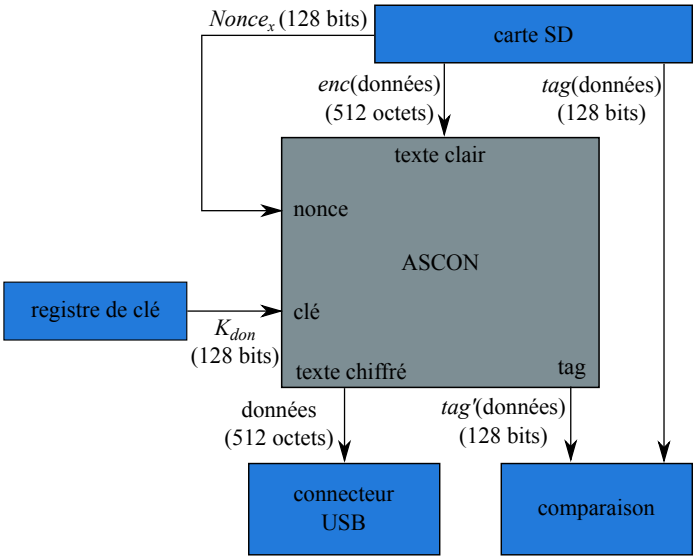


FIGURE 2.14: Déchiffrement

2.3.1.4 Conclusion

À travers cette partie, nous avons vu qu'il était possible d'intégrer au sein d'un système complet et fonctionnel une PUF, un TRNG et un algorithme de chiffrement authentifié. Le système proposé permet le stockage de données de manière sécurisé sur un appareil USB. Ce système fonctionne en plusieurs phases :

— une phase d'enregistrement permettant de générer toutes les clés ;
— une phase d'activation permettant de rendre l'appareil USB utilisable par un ordinateur ;
— des phases de chiffrement pour le stockage et de déchiffrement pour la lecture des données sur l'appareil.

Nous avons pu prendre conscience de l'importance de chaque primitive cryptographique étudiée dans les parties précédentes pour le bon fonctionnement du système. Une photo du prototype final de stockage USB sécurisé est représentée sur la figure 2.15.

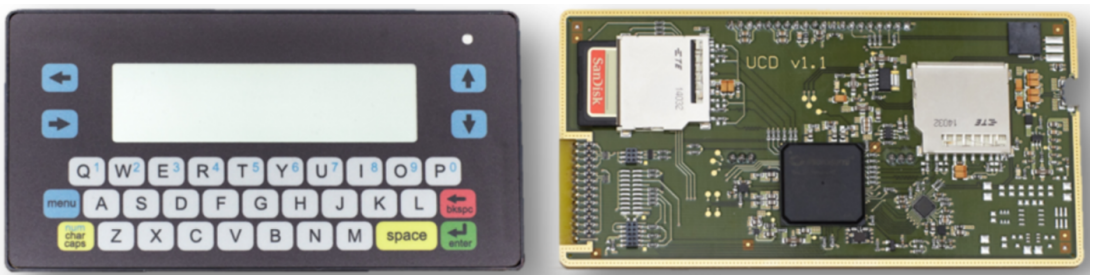


FIGURE 2.15: Photo du prototype USB portable de stockage de données sécurisées

2.3.2 Autres démonstrateurs

2.3.2.1 Système sécurisé de messagerie

Les blocs élémentaires ainsi que la plateforme matérielle utilisés pour ce démonstrateur de système de messagerie sécurisée sont exactement les mêmes que pour le démonstrateur précédent. Deux prototypes tels que celui de la figure 2.15 sont utilisés pour communiquer de manière sécurisé sur le réseau. La connexion des appareils au réseau est effectuée en les connectant via USB sur un ordinateur. Étant donné la grande similarité avec le démonstrateur USB portable de stockage de données sécurisées, ce démonstrateur ne sera pas plus amplement détaillé. Pour plus d'information sur ce démonstrateur, le lecteur peut se référer au chapitre 4 du délivrable D4.2 disponible sur le site officiel du projet HECTOR [1] ainsi qu'à la vidéo de présentation de ce démonstrateur [2].

1. https://hector-project.eu/publications-deliverables/deliverables
2. https://vimeo.com/289260090

2.3.2.2 Infrastructure matérielle/logicielle pour la protection des données de conception

Ce dernier démonstrateur assure la protection des données de conception. Il est le fruit d'une collaboration avec Dr. Brice Colombier, travaillant au laboratoire Hubert Curien de l'Université Jean Monnet, qui était porteur de ce projet. La TERO-PUF décrite dans la section 2.2 a été intégrée au sein de ce démonstrateur qui comprend également un système de verrouillage de la logique, un système de masquage de la logique, une correction d'erreurs utilisant les protocoles de réconciliation de clés et un système de chiffrement léger. Ce système permet de protéger un noyau IP au moment de la conception et de l'activer à distance plus tard. Le fonctionnement détaillé de ce démonstrateur est très bien décrit dans le chapitre 5 du manuscrit de thèse de Brice Colombier [94] et ne sera donc pas redéfinie ici.

2.4 Conclusion

Au cours de ce chapitre, nous avons comparé les résultats d'implantation des TRNG à base de cellules oscillantes conformes au standard AIS31. Cela nous a permis de constater que tous ces TRNG étaient implantables sur les principales familles de FPGA existantes. Malgré tout, le COSO-TRNG et le TERO-TRNG ne sont actuellement pas des implantations viables pour des applications industrielles nécessitant une répétabilité de l'implantation. En effet, ils exigent un placement manuel minutieux des éléments pour chaque circuit.

Cette analyse reflète aussi le fait qu'aucun TRNG idéal n'existe et qu'un compromis devra toujours être fait. Le choix du TRNG sera toujours dépendant des contraintes de l'application. Nous avons introduit deux critères d'évaluation (l'efficacité énergétique et le produit Entropie × Débit) qui aideront le concepteur à faire ce choix.

L'évaluation que nous avons réalisée est dépendante du matériel d'implantation choisi. L'utilisation de la même plateforme d'évaluation pour comparer correctement les TRNG était donc essentiel.

Dans une deuxième partie de ce chapitre, nous avons proposé, pour la première fois, une démarche pour réaliser une implantation efficace de PUF à base de cellules oscillantes sur les FPGA de type Flash. Pour ce faire, un certain nombre de méthodologies d'implantations ont été proposées. Premièrement, s'assurer que la structure de la cellule ne soit pas modifiée au cours du processus de synthèse, de placement et de routage par le compilateur Libero. Deuxièmement, contrôler le délai des éléments critiques de la PUF. Troisièmement, isoler les cellules oscillantes

. Le code associé à cette section est disponible à l'adresse suivante : https://gitlab.univ-st-etienne.fr/b.colombier/demonstrator

du reste du circuit afin d'éviter toute possibilité de perturbation. Enfin, améliorer la qualité du signal en sortie des cellules RO.

Enfin, dans la troisième partie de ce chapitre, nous avons présenté une intégration d'un candidat de TRNG et d'un candidat de PUF au sein de plusieurs démonstrateurs Un système de stockage de données de manière sécurisé sur un appareil USB, un système de messagerie sécurisée et une infrastructure pour la protection des données de conception. Ces intégrations nous ont permis de prouver l'utilité de ces primitives cryptographiques au sein d'un système complet ainsi que de prouver la faisabilité d'une telle intégration.

A présent, dans le chapitre suivant, nous tentons d'amorcer une approche de modélisation des cellules oscillantes exploitées par les PUF : la cellule RO et la cellule TERO.

Chapitre 3

Modélisation des caractéristiques probabilistes des cellules oscillantes exploitées par les PUF

Au cours du chapitre 1, nous avons vu que la meilleure manière de garantir l'imprévisibilité d'un générateur physique d'aléa est d'estimer son entropie en se basant sur un modèle stochastique (voir sec.1.1.2). De nombreux travaux ont été réalisés dans ce sens pour les TRNG. Les normes de certification standardisées des TRNG telles que l'AIS31 imposent aux concepteurs de TRNG de fournir un modèle stochastique pour que celui-ci soit utilisable pour des applications cryptographiques.

À l'heure actuelle, aucun standard semblable n'existe pour les PUF. Nous pensons qu'une démarche scientifique dans ce sens est nécessaire pour une intégration viable des PUF au sein de systèmes industriels. Dans ce chapitre, nous tentons d'initier une telle approche en apportant des éléments de modélisation des caractéristiques probabilistes des cellules oscillantes exploitées par les PUF. Ainsi, les cellules RO et TERO seront étudiées. En complément du chapitre 2, cette étude apportera aux concepteurs des éléments essentiels pour mener à bien la conception de PUF sur les circuits intégrés.

Puisque ce travail se focalise sur des circuits microélectroniques (*ex.* FPGA et ASIC) et que les transistors MOS représentent la principale source de variation au sein des circuits microélectroniques, l'accent est mis sur ce composant.

En suivant l'approche proposée par Haddad *et al.* [95], nous proposons de réaliser un modèle physique (électrique) de la cellule RO et de la cellule TERO pour ensuite dériver un modèle stochastique. Nous commencerons par modéliser un inverseur logique et une ligne de transmission. Ensuite, en partant de l'impact des variations de procédés de fabrication sur les transistors, nous tenterons de déduire l'impact de celles-ci sur une cellule oscillante.

3.1 Modélisation électrique de l'inverseur logique

L'inverseur logique est l'élément de base des cellules oscillantes. Ainsi la modélisation des cellules commence par la modélisation d'un inverseur seul. Un inverseur logique élémentaire est composé de deux transistors MOS, un transistor MOS de type N et un transistor MOS de type P. Pour cela, il est essentiel de comprendre le fonctionnement du transistor MOS.

3.1.1 Transistor MOS

Le transistor à effet de champ à structure métal-oxyde-semiconducteur — en anglais metal oxide semiconductor field effect transistor (MOFSET) — est le composant de base des circuits électroniques numériques. Il est composé de trois électrodes actives : le drain noté D, la source notée S et la grille notée G. Le transistor module le courant qui le traverse à l'aide d'un signal appliqué sur son électrode G. En effet, une différence de potentiel entre G et S supérieure à un certain seuil crée un canal de conduction entre D et S permettant le passage des porteurs de charges (électrons ou trous). Soit V_{gs} la différence de potentiel, ou tension, entre G et S, V_{th} la tension de seuil et I_{ds} le courant traversant le canal. Lorsque $V_{gs} > V_{th}$ le canal devient assez grand pour laisser passer des porteurs de charges et donc créer un courant I_{ds}. Plus V_{gs} est grand, plus le courant I_{ds} pourra être grand. La tension entre D et S, V_{ds} représente la tension de polarisation. Elle impose un champ électrique accélérateur dans le canal qui guide les porteurs de charges de S vers D ou inversement.

L'analogie d'un transistor est souvent faite avec un robinet. Si le robinet est fermé ($V_{gs} < V_{th}$), l'eau ne circule pas à travers la vanne. Plus le robinet est ouvert plus l'eau circule et plus son débit (I_{ds}) est grand, jusqu'à que le robinet soit complètement ouvert et que le débit sature. Ainsi, la tension V_{gs} correspond à l'ouverture du robinet et la tension V_{ds} correspond à la pression de l'eau. La figure 3.1 représente un transistor MOS.

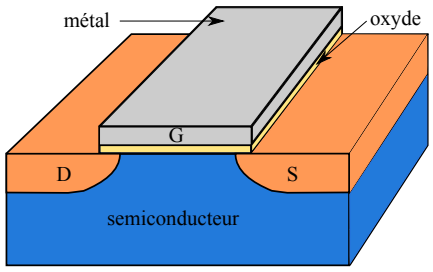


FIGURE 3.1: Transistor MOS

Le transistor MOS a trois régimes de fonctionnement :

— **Bloqué** : lorsque $V_{gs} - V_{th} < 0$. Dans ce cas, le canal de conduction n'a pas été crée et le courant est nulle à l'exception des fuites : $I_{ds} \approx 0$. On peut considérer le transistor éteint.

— **Linéaire** : lorsque $V_{gs} - V_{th} > V_{ds} > 0$. Le transistor est en fonctionnement et un canal de conduction a été créé permettant le passage du courant entre D et S. Dans ce mode le transistor peut être assimilé à une résistance, car la relation liant I_{ds} et V_{ds} est quasi-linéaire. Le courant traversant le canal est défini par l'équation suivante :

$$I_{ds} = I_{lin} = \beta((V_{gs} - V_{th})V_{ds} - \frac{V_{ds}^2}{2}) \tag{3.1}$$

avec :

$$\beta = \mu C_{ox}\frac{W}{L} \tag{3.2}$$

où μ représente la mobilité des porteurs de charges, C_{ox} la capacité de l'oxyde de grille, W la largeur de la grille et L la longueur de la grille.

— **Saturé** : lorsque $V_{ds} \geq V_{gs} - V_{th} > 0$. Le transistor est en fonctionnement et le courant traversant le canal est saturé et ne dépend plus de V_{ds}. Le courant traversant le canal est défini par l'équation suivante :

$$I_{ds} = I_{sat} = \frac{\beta}{2}(V_{gs} - V_{th})^2 \tag{3.3}$$

3.1.2 Inverseur CMOS

Soit un inverseur logique CMOS composé de deux transistors, V_{in} la tension appliquée à son entrée, V_{out} sa tension de sortie, C_L la capacité de la charge de sortie (*ex.* un autre inverseur) et V_{DD} la tension d'alimentation du circuit comme représenté sur la figure 3.2.

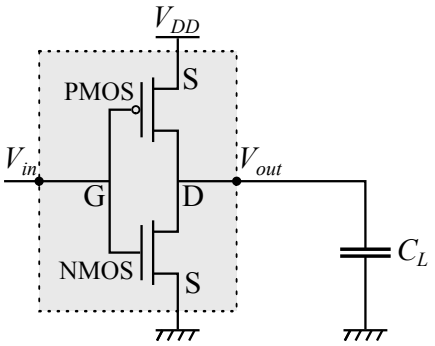


FIGURE 3.2: Schéma électrique d'un inverseur CMOS

Considérons un état stable de l'inverseur où $V_{in} = 0\,\text{V}$ (l'état logique bas) et $V_{out} = V_{DD}$ (l'état logique haut). À ce moment, la tension V_{gs} du transistor NMOS, notée $V_{gs_n}(= V_{in})$, vaut $0\,\text{V}$, la tension V_{gs} du transistor PMOS, notée $V_{sg_p}(= V_{DD} - V_{in})$, vaut V_{DD} et la tension V_{ds} du transistor PMOS, notée $V_{sd_p}(= V_{DD} - V_{out})$, vaut $0\,\text{V}$. Ainsi :

— $V_{gs_n} - V_{th_n} < 0$, le transistor NMOS est **bloqué**.

— $V_{sg_p} - V_{th_p} > V_{sd_p} = 0$, le transistor PMOS est en régime **linéaire**.

À présent, si V_{in} passe de l'état logique bas (0 V) à l'état logique haut (V_{DD}) :

— $V_{ds_n} > V_{gs_n} - V_{th_n} > 0$, le transistor NMOS est **saturé**.

— $V_{sg_p} - V_{th_p} < 0$, le transistor PMOS est **bloqué**.

Le transistor NMOS devient passant et le transistor PMOS bloqué. Ainsi, la capacité C_L est directement reliée à la masse et va se décharger. Au fur et à mesure que V_{out} diminue, V_{ds_n} diminue. Lorsque $V_{gs_n} - V_{th_n} > V_{ds_n}$, le transistor NMOS devient linéaire :

— $V_{gs_n} - V_{th_n} > V_{ds_n} > 0$, le transistor NMOS est en régime **linéaire**.

— $V_{sg_p} - V_{th_p} < 0$, le transistor PMOS est **bloqué**.

Le passage de V_{in} de l'état logique haut (V_{DD}) à l'état logique bas (0 V) s'effectuera de la même manière : le transistor NMOS sera bloqué et le transistor PMOS sera saturé puis linéaire.

Basé sur ce fonctionnement et ainsi que sur les travaux de modélisation de la bascule RS de Reyneri *et al.* [29], l'inverseur logique peut-être décomposé en trois parties :

— **Un comparateur idéal** : si $V_{in} < \frac{V_{DD}}{2}$ alors $V_{out} = V_{DD}$ et 0 V sinon. Le passage entre l'état logique bas et l'état logique haut se fait à $\frac{V_{DD}}{2}$.

— **Un étage de délai** ayant un temps de retard noté T_r et correspondant au délai entre la commutation de V_{in} à l'entrée de la porte et le temps pour que la sortie V_{out} de l'inverseur commence à changer.

— **Un limiteur de pente** qui va produire les temps de montée et de descente de l'inverseur.

La figure 3.3 représente la décomposition de l'inverseur en trois parties.

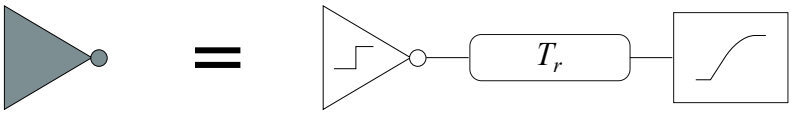


FIGURE 3.3: Décomposition de l'inverseur logique en trois parties

Ceci nous donne un modèle électrique de l'inverseur en fonction du temps lui aussi décomposable en trois parties distinctes comme représentées sur la figure 3.4.

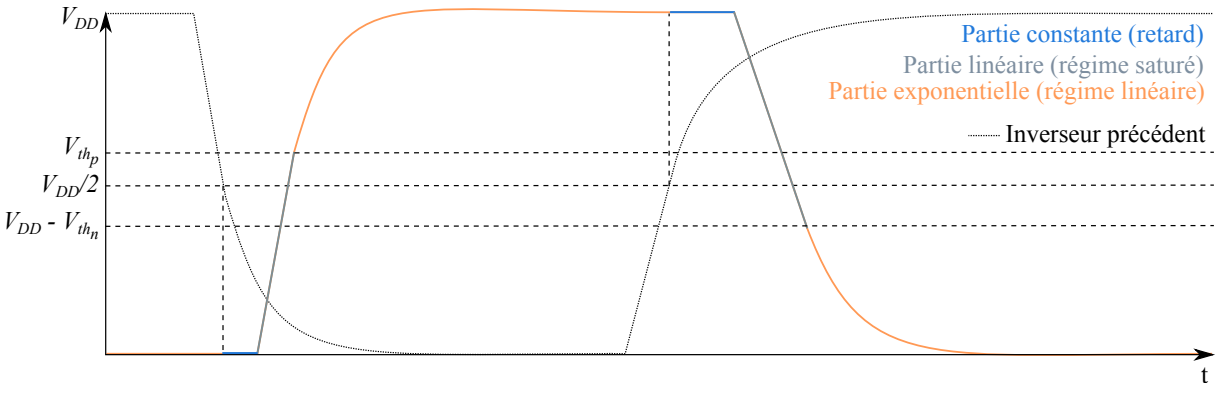


FIGURE 3.4: Modélisation de la sortie d'un inverseur en fonction du temps

— **Une partie constante** qui correspond à T_r.

— **Une partie linéaire** qui correspond au temps durant lequel le transistor (NMOS pour une commutation de V_{in} de '0' vers '1', PMOS pour une commutation de '1' vers '0') est en régime saturé.

— **Une partie exponentielle** qui correspond au temps durant lequel le transistor (NMOS pour une commutation de V_{in} de '0' vers '1', PMOS pour une commutation de '1' vers '0') est en régime linéaire. La transition entre la partie linéaire et la partie exponentielle se fait dans le cas d'un front descendant lorsque :

$$V_{out} < V_{DD} - V_{th_n} \tag{3.4}$$

avec $V_{out} = V_{ds_n}$ et $V_{in} = V_{gs_n} = V_{DD}$.

Dans le cas d'un front montant la transition se fait lorsque :

$$V_{out} > V_{th_p} \tag{3.5}$$

avec $V_{out} = V_{DD} - V_{sd_p}$ et $V_{in} = V_{DD} - V_{sg_p} = 0$.

La tension de sortie de l'inverseur V_{out} est donc définie par 3 fonctions liées aux trois parties définies ci-devant. Prenons l'exemple d'un front descendant comme représenté sur la figure 3.5.

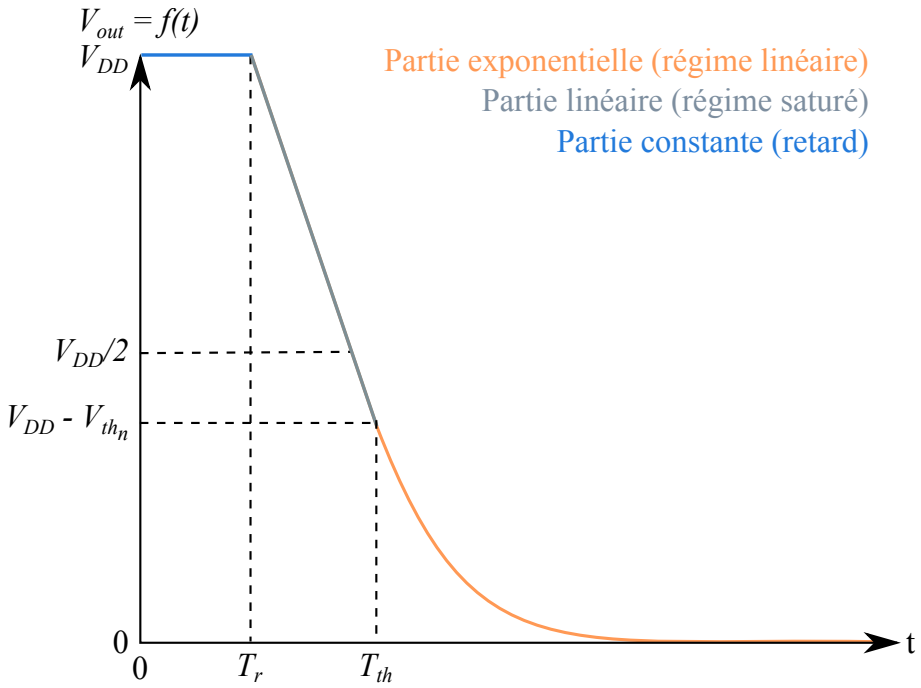


FIGURE 3.5: Front descendant d'un inverseur en fonction du temps ($V_{out} = f(t)$)

— **Partie constante** ($t \leq T_r$) :

$$V_{out} = V_{DD} \tag{3.6}$$

— **Partie linéaire** ($T_r < t \leq T_{th}$) :

$$V_{out} = V_{DD} - \alpha(t - T_r) \tag{3.7}$$

— **Partie exponentielle** ($T_{th} < t$) :

$$V_{out} = (V_{DD} - V_{th_n})e^{-\frac{t-T_{th}}{\tau}} \tag{3.8}$$

où α représente le coefficient de pente de la partie linéaire, T_{th} le temps au bout duquel le transistor passe du régime saturé au régime linéaire et τ la constante de temps de la partie exponentielle.

Le comportement de la porte d'activation ET-logique sera très similaire à celui de l'inverseur élémentaire puisqu'elle sera composée d'une porte NON-ET-logique (deux transistors PMOS en parallèle, deux transistors NMOS en série) suivie d'un inverseur élémentaire comme représenté sur la figure 3.6. Une fois la cellule activée (*ctrl* qui passe à '1', voir sec.1.2.1.1), un des deux transistors PMOS de la porte NON-ET-logique sera toujours bloqué et un des deux transistors NMOS de la porte NON-ET-logique sera toujours passant. Ainsi, la porte ET-logique pourra être assimilée à deux inverseurs élémentaires successifs.

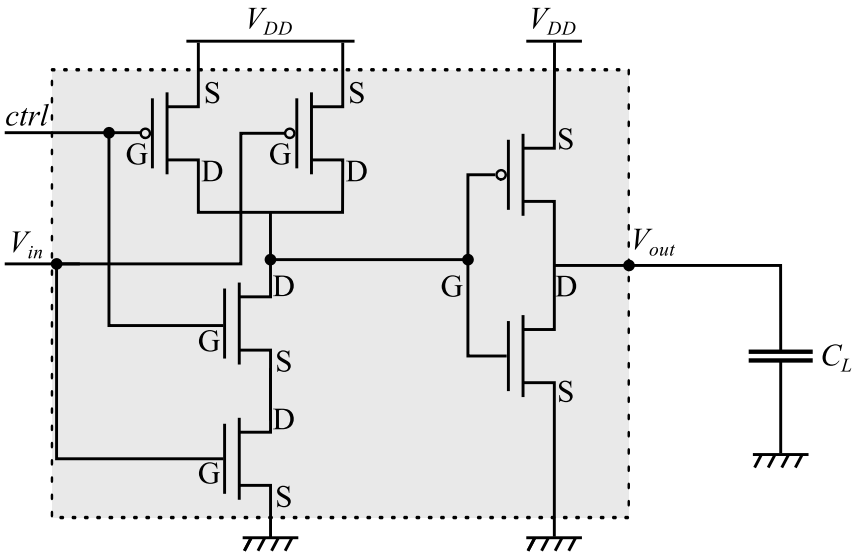


FIGURE 3.6: Schéma électrique d'une porte ET-logique CMOS

À présent, tâchons de modéliser le dernier élément qui constitue une cellule oscillante : les lignes d'interconnexion entre les portes logiques.

3.2 Modélisation électrique d'une ligne de transmission

Si la longueur de la ligne, notée l, est très faible par rapport à la longueur d'onde du signal, notée λ, qui la traverse ($l \ll \lambda$) l'approximation des régimes quasi stationnaires nous permet de considérer comme négligeable le temps de propagation des ondes électromagnétiques devant la période de l'onde. Pour rappel, l'équation de la longueur d'onde est :

$$\lambda = \frac{c}{f} \tag{3.9}$$

où c représente la célérité d'une onde dans le vide ($\approx 300\,000$ km /s) et f la fréquence de l'onde.

En considérant dans notre cas des fréquences d'oscillations de l'ordre de la centaine de MHz, λ sera de l'ordre du mètre. Toutes les lignes considérées pour la modélisation de cellules oscillantes dans les circuits numériques sont des lignes très courtes permettant de connecter les inverseurs logiques entre eux. Ces lignes sont de l'ordre du μm. Ainsi, il parait tout à fait raisonnable d'admettre que la longueur de ces lignes est toujours très faible par rapport à λ. Dans ce cas, l'approximation des régimes quasi stationnaires est respectée et le modèle électrique d'une ligne de longueur l en négligeant les pertes est celui représenté sur la figure 3.7. Les faibles pertes diélectriques à l'intérieur d'un circuit intégré nous permettent de considérer la ligne sans pertes.

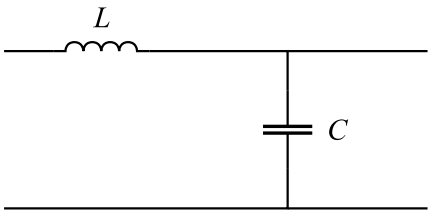


FIGURE 3.7: Représentation schématique d'une ligne de transmission sans pertes

La vitesse de propagation du signal à l'intérieur d'une ligne électrique sans pertes telle que représentée sur la figure 3.7 est définie selon l'équation suivante :

$$v = \frac{1}{\sqrt{LC}} \tag{3.10}$$

où L représente la valeur de l'inductance caractéristique par unité de longueur de la ligne (en $H/\,\mathrm{m}$) et C la valeur de la capacité caractéristique par unité de longueur de la ligne en ($F/\,\mathrm{m}$).

Ainsi, le délai de propagation, noté t_{li}, à travers une ligne de longueur l est :

$$t_{li} = \frac{l}{v} \tag{3.11}$$

3.3 Modélisation électrique de la cellule RO

Dans les deux sections précédentes, nous avons modélisé tous les éléments qui composent une cellule RO. Pour rappel, une RO-PUF compare les fréquences d'oscillations de deux cellules RO pour générer un bit de réponse. Par conséquent, l'objectif est de modéliser la fréquence d'oscillation d'une cellule RO à l'aide des modèles de chaque élément de la cellule. Lorsque le signal d'activation (*ctrl*) va passer de l'état logique bas ('0') à l'état logique haut ('1'), l'évènement électrique e se propageant dans la cellule RO va engendrer les commutations décrites dans la section 3.1. Nous considérons le passage d'un état logique bas à un état logique haut ou inversement à $\frac{V_{DD}}{2}$. Ainsi, le temps de commutation d'un inverseur correspond au temps que va mettre la sortie V_{out} d'un inverseur pour atteindre $\frac{V_{DD}}{2}$ à partir du moment où V_{in} atteint $\frac{V_{DD}}{2}$. Jusqu'à $\frac{V_{DD}}{2}$, les transistors responsables de la commutation sont en régime saturé, donc la sortie de l'inverseur reste dans la partie linéaire (voir fig.3.5). Ainsi, grâce à l'équation 3.7, nous pouvons définir le temps de commutation comme suit.

$$t_{\frac{V_{DD}}{2}} = \frac{V_{dd}}{2 \times \alpha} + T_r \tag{3.12}$$

où α vaut $\frac{I_{sat}}{C_L}$.

Pour un front montant, c'est le transistor PMOS qui sera saturé. Pour un front descendant, ce sera le transistor NMOS. Ceci nous donne le temps de montée, noté t_m, et le temps de descente, noté t_d, suivant.

$$t_m = \frac{V_{dd} \times C_L}{2 \times I_{sat_p}} + T_{r_m} \tag{3.13}$$

avec I_{sat_p} le courant du transistor PMOS en régime saturé et T_{r_m} le temps de retard montant.

$$t_d = \frac{V_{dd} \times C_L}{2 \times I_{sat_n}} + T_{r_d} \tag{3.14}$$

avec I_{sat_n} le courant du transistor NMOS en régime saturé et T_{r_d} le temps de retard descendant.

Ainsi, à chaque porte logique de la cellule RO sera associé un t_m et un t_d et à chaque interconnexion sera associé un t_{li} comme représenté sur la figure 3.8.

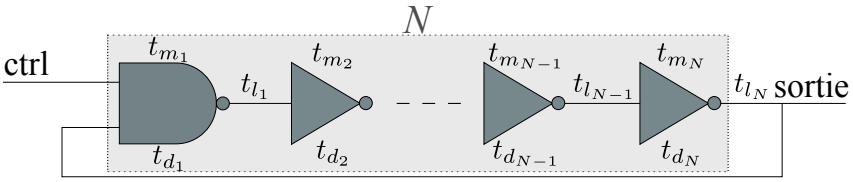


FIGURE 3.8: Modèle d'une cellule RO

Pour effectuer une période complète d'oscillation en sortie de la cellule, l'évènement e passera deux fois dans la boucle en déclenchant ainsi un front montant et un front descendant sur chaque

inverseur. La fréquence d'oscillation sera donc définie comme suit.

$$f_{osc} = \frac{1}{\sum_{i=1}^{N} t_{m_i} + \sum_{i=1}^{N} t_{d_i} + 2\sum_{i=1}^{N} t_{li_i}} \tag{3.15}$$

3.4 Modélisation électrique de la cellule TERO

La modélisation de la ligne de transmission de la section 3.2 reste valable pour la cellule TERO. Cependant, dans le cadre de la cellule TERO, c'est le nombre d'oscillations final qui va être exploité pour les PUF. Comme pour la cellule RO, à chaque porte logique de la cellule TERO sera associé un t_m et un t_d et à chaque interconnexion sera associé un t_{li} comme représenté sur la figure 3.9.

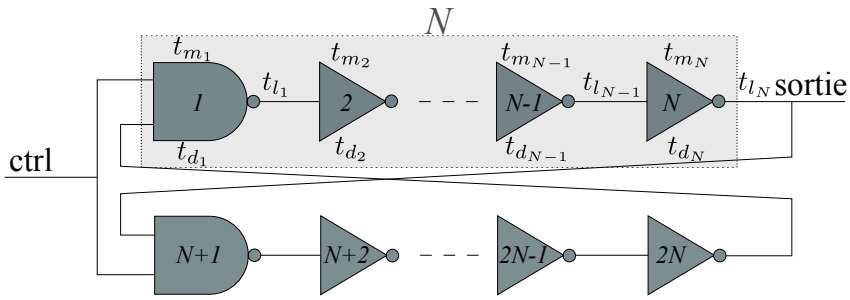


FIGURE 3.9: Modèle d'une cellule TERO

Puisque deux évènements se propagent à travers la cellule TERO (voir sec.1.2.1.2), le comportement de ces inverseurs va être légèrement différent de celui de la cellule RO. Trois effets distincts, très bien présentés par Reyneri *et al.* [29], vont influencer le nombre d'oscillations. Ces effets sont les suivants.

3.4.1 Effet du limiteur de pente

Supposons deux évènements successifs à l'entrée d'un inverseur, comme c'est le cas pour une cellule TERO en fonctionnement. Ces évènements provoqueront un front montant suivi d'un front descendant, ou inversement. Le temps qui sépare l'arrivée de ces deux évènements successifs à l'entrée de l'inverseur représente une impulsion notée T_{in}. L'impulsion produite à la sortie de l'inverseur en réponse à une impulsion T_{in} sera notée T_{out}.

Dans un premier temps, pour simplifier la compréhension de cet effet du limiteur de pente et pour le mettre en valeur, nous considérons le temps de montée (t_m) égale au temps de descente de l'inverseur (t_d) : $t_m = t_d = T_{lim}$, où T_{lim} représente le délai induit par le limiteur de pente introduit dans la section 3.1 (voir fig. 3.3).

Si l'impulsion T_{in} est supérieure ou égale à T_{lim} ($T_{in} \geq T_{lim}$), alors elle traversera l'inverseur sans être perturbée : $T_{out} = T_{in}$. Cependant, si $T_{in} < T_{lim}$, l'effet du limiteur de pente produira une impulsion de sortie réduite de $(T_{lim} - T_{in})$: $T_{out} = T_{in} - (T_{lim} - T_{in})$. La figure 3.10 représente l'effet du limiteur de pente pour le cas où $T_{in} \geq T_{lim}$ (T_{in1}) et pour le cas où $T_{in} < T_{lim}$ (T_{in2}).

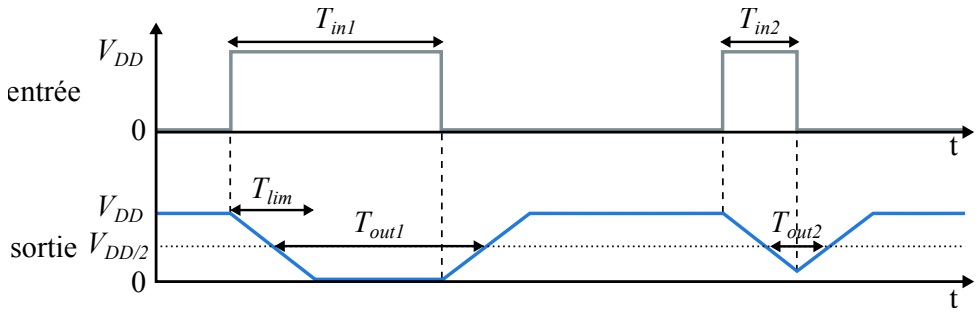


FIGURE 3.10: Effet du limiteur de pente

Comme nous l'avons vu dans la section 1.2.1.2, le rapport cyclique du signal de sortie de la cellule TERO va évoluer en direction de 0 % ou 100 % jusqu'à l'arrêt des oscillations. Soit T_{osc} la période d'oscillation du signal de sortie de la cellule TERO et T_{in} le temps à l'état haut du signal de sortie de la cellule TERO. Si $T_{in} < T_{lim}$ ou $T_{osc} - T_{in} < T_{lim}$ alors l'effet du limiteur de pente va influencer la cellule TERO.

Cet effet va contribuer à l'arrêt des oscillations seulement à partir du moment où les deux évènements seront assez proches.

3.4.2 Effet de l'inhomogénéité des inverseurs

À présent, plaçons-nous dans le cas plus général où les inverseurs ne sont pas homogènes, c'est-à-dire qu'ils comportent des temps de montée et de descente différents d'un inverseur à l'autre. Le rapport cyclique du signal de sortie la cellule TERO va être influencé par cette différence.

Soit e_1 le premier évènement qui va se présenter sur l'inverseur numéro 1 de la cellule TERO représentée sur la figure 3.9 après activation de celle-ci et e_2 le deuxième évènement.

e_1 déclenchera toujours un front descendant sur l'inverseur numéro 1, ainsi que sur tous les inverseurs de la cellule TERO possédant un numéro impair. De la même façon e_2 déclenchera toujours un front montant sur l'inverseur numéro 1, ainsi que sur tous les inverseurs de la cellule TERO possédant un numéro impair. Ce sera l'inverse pour les inverseurs possédant un numéro pair : e_1 déclenche un front montant et e_2 un front descendant.

Par conséquent e_1 parcourt la cellule TERO avec le délai t_{e_1} suivant :

$$t_{e_1} = \sum_{i=1}^{N} t_{m_{2i}} + \sum_{i=0}^{N-1} t_{d_{2i+1}} + 2\sum_{i=1}^{2N} t_{li_i} \tag{3.16}$$

et e_2 parcourt la cellule TERO avec le délai t_{e_2} suivant :

$$t_{e_2} = \sum_{i=1}^{N} t_{d_{2i}} + \sum_{i=0}^{N-1} t_{m_{2i+1}} + 2\sum_{i=1}^{2N} t_{li_i} \tag{3.17}$$

À chaque tour, correspondant à une période du signal de sortie de la cellule TERO, l'impulsion initiale T_{in} sera réduite de $\Delta_{e_1/e_2} = |t_{e_1} - t_{e_2}|$.

Si cet effet était le seul à intervenir sur le rapport cyclique du signal de sortie de la cellule TERO, le nombre d'oscillations, noté N_{osc}, pourrait être défini comme suit :

$$N_{osc} = \frac{T_{in}}{\Delta_{e_1/e_2}} \tag{3.18}$$

Cependant, à partir d'un certain moment, l'effet du limiteur de pente intervient. En considérant l'impact de l'effet du limiteur de pente sur N_{osc} comme une constante représentant la largeur minimale d'impulsion qu'un inverseur peut transmettre, notée T_{min}, N_{osc} peut être approximé comme suit :

$$N_{osc} \approx \frac{T_{in} - T_{min}}{\Delta_{e_1/e_2}} \tag{3.19}$$

De plus, un autre effet va influencer le nombre d'oscillations de la cellule TERO : l'effet mémoire.

3.4.3 Effet Mémoire

Après T_{th} (voir fig.3.5), la sortie de l'inverseur sera en partie exponentielle et elle va tendre lentement vers 0 V ou V_{DD}. Ainsi, un effet mémoire s'effectuera sur l'inverseur. La réponse à un front en entrée sera différente de la réponse à une impulsion en entrée.

En effet, lorsque le deuxième évènement arrivera à l'entrée de l'inverseur, la sortie ne repartira pas exactement de 0 V ou V_{DD}, mais d'une tension proche de 0 V ou V_{DD}. Plus le temps entre deux évènements sera long, plus le signal de sortie aura le temps d'approcher la valeur 0 V ou V_{DD}.

Ainsi, toute asymétrie (*ex.* rapport cyclique différent de 50 %) à l'entrée de l'inverseur produira une asymétrie plus grande en sortie.

Reyneri *et al.* montrent que dans ce cas T_{out} peut être approximé comme suit [29] :

$$T_{out} \approx \frac{T_{osc}}{2} + [T_{in} - \frac{T_{osc}}{2}][1 + H_d] \tag{3.20}$$

où $H_d = 2e^{\frac{K_0 \cdot T_{lim} - \frac{T_{osc}}{2}}{(1-K_0) \cdot T_{lim}}}$ avec K_0 la tension de sortie à l'instant T_{th} (voir fig.3.5). $K_0 = V_{DD} - V_{th_n}$ dans le cas d'un front descendant et $K_0 = V_{th_p}$ dans le cas d'un front montant.

Si cet effet était le seul à intervenir sur le rapport cyclique du signal de sortie de la cellule TERO, la largeur d'impulsion T_{out} après N inverseurs pourrait être définie comme suit :

$$T_{out_N} \approx \frac{T_{osc}}{2} + [T_{in} - \frac{T_{osc}}{2}][1 + H_d]^N \tag{3.21}$$

3.4.4 Conditions initiales influençant la cellule TERO

Nous venons de voir les trois effets responsables de l'arrêt des oscillations d'une cellule TERO. Plusieurs conditions initiales et paramètres de configuration de la cellule TERO vont influencer le nombre d'oscillations et l'état final d'une cellule TERO.

La première condition initiale va être le rapport cyclique de départ : $\frac{T_{in}}{T_{osc}}$. Ce rapport cyclique va être fixé par l'arrivée des évènements sur les portes d'activations et par le nombre d'inverseurs par branche de la cellule TERO. En effet, si les deux évènements n'arrivent pas au même moment sur les portes d'activations, ou si le nombre d'inverseurs n'est pas le même dans chaque branche (par exemple 7 inverseurs sur la branche supérieure de la cellule TERO et 5 inverseurs sur la branche inférieure), alors $T_{in} \neq T_{osc} - T_{in}$. Dans ce cas, le rapport cyclique sera différent de 50 % et l'effet mémoire de l'inverseur influencera le nombre d'oscillations de la cellule TERO. Plus le rapport cyclique sera asymétrique, plus l'influence de l'effet mémoire sera importante.

Le deuxième paramètre est la différence entre les temps de montée et de descente des inverseurs. Cette différence sera d'autant plus grande que les inverseurs qui composent la cellule TERO sont différents. Autrement dit, s'ils ne sont pas homogènes.

Ainsi, le concepteur pourra jouer sur ces paramètres pour influencer le nombre d'oscillations d'une cellule TERO.

3.4.5 Validation du modèle électrique

À partir des trois effets décrits précédemment, Reyneri *et al.* ont établi un modèle analytique des conditions d'oscillation de la cellule TERO ainsi que du temps nécessaire à sa stabilisation [29]. Ce modèle sera réutilisé par Haddad *et al.* pour établir un modèle stochastique de TRNG à base de cellule TERO [95].

Un de nos objectifs est de vérifier la validité de ce modèle. Pour ce faire, nous avons réalisé plusieurs simulations électriques avancées et expérimentations sur différentes structures de cellules TERO.

3.4.5.1 Simulations sur Cadence®

Dans un premier temps, nous avons utilisé le logiciel Spectre® inclut dans la suite Virtuoso® fournit par Cadence®. Ce logiciel permet la simulation de circuits électroniques au niveau du transistor. L'avantage de cet outil est qu'il réalise les simulations en se basant sur le modèle électrique lié à la technologie utilisée. Dans notre cas, nous utilisons le modèle lié à la technologie STMicroelectronics CMOS en 65nm. Plusieurs structures de cellule TERO ont été simulées.

Cellules TERO non homogènes avec branches équilibrées pour différents nombres d'éléments Tout d'abord, nous avons simulé des cellules TERO équilibrées, c'est-à-dire avec le même nombre N d'inverseurs par branche, et avec un élément non homogène, c'est-à-dire en utilisant un inverseur différent des autres, que nous appellerons I. Puisque ce sont des simulations électriques, aucune variation et aucun bruit ne s'applique sur les transistors et le temps de propagation entre les inverseurs n'est pas pris en compte. Ainsi :

— Le temps de montée sera le même pour tous les inverseurs sauf pour l'inverseur I ;
— Le temps de descente sera le même pour tous les inverseurs sauf pour l'inverseur I ;
— Le signal d'activation ($ctrl$) arrivera au même moment sur les deux portes d'activation : au départ $T_{in} = T_{osc} - T_{in}$.

Cette simulation est réalisée pour quatre tailles de cellule TERO : une cellule TERO avec 3 inverseurs par branche, 5 inverseurs par branche, 7 inverseurs par branche et 9 inverseurs par branche. De cette manière, l'impulsion initiale (T_{in} ou $T_{osc} - T_{in}$) augmentera avec le nombre d'inverseurs alors que la différence de délai pour parcourir la cellule TERO (Δ_{e_1/e_2}) entre les deux évènements (e_1 et e_2) restera la même. La figure 3.11 montre les résultats de cette simulation. Ce résultat montre bien l'effet de l'inhomogénéité des inverseurs puisque le nombre d'oscillations, N_{osc}, évolue de manière cohérente avec l'équation 3.19 :

— Pour la cellule TERO 3-3, $N_{osc} = 48$;
— Pour la cellule TERO 5-5, $N_{osc} = 373$;
— Pour la cellule TERO 7-7, $N_{osc} = 754$;
— Pour la cellule TERO 9-9, $N_{osc} = 812$.

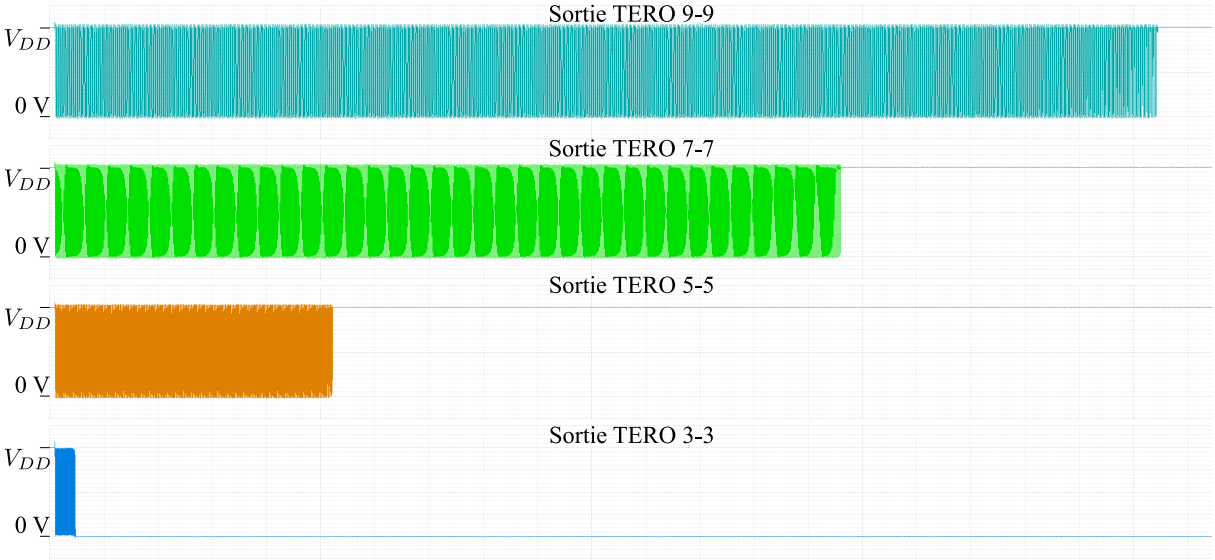


FIGURE 3.11: Simulation de cellules TERO non homogènes avec branches équilibrées pour différents nombres d'éléments

Cellules TERO non homogènes avec branches équilibrées et une charge en sortie Ensuite, nous avons réalisé une seconde expérience sur la cellule TERO non homogènes avec des branches équilibrées composées de sept inverseurs par branche. Une charge de 1 fF est appliquée en sortie de la branche. Cela a pour conséquence de ralentir les temps de commutation de l'inverseur sur lequel la charge est appliquée (voir équation 3.13 et 3.14).

La figure 3.12 montre les résultats de simulation pour le cas où la charge est appliquée sur la branche supérieure de la cellule TERO et pour le cas où la charge est appliquée sur la branche inférieure. La figure 3.13 montre le résultat de comparaison avec la cellule TERO non chargée. Cette simulation soulève deux points importants.

Premièrement, le nombre d'oscillations est très fortement réduit par l'ajout d'une charge. Ceci reste cohérent avec la simulation précédente puisque le fait d'ajouter une charge augmente la différence Δ_{e_1/e_2}. Cependant, le concepteur doit en tenir compte lors de la réalisation d'une PUF à base de TERO puisque les sorties des cellules TERO seront envoyées sur un multiplexeur qui aura pour conséquence de rajouter une charge sur une branche de la cellule TERO.

Deuxièmement, nous pouvons constater que la position de la charge (branche supérieure ou branche inférieure) va conditionner l'état final de la cellule TERO après oscillations. En effet, dans un cas la cellule s'arrête à l'état logique haut (V_{DD}) alors que dans l'autre elle s'arrête à l'état logique bas (0 V).

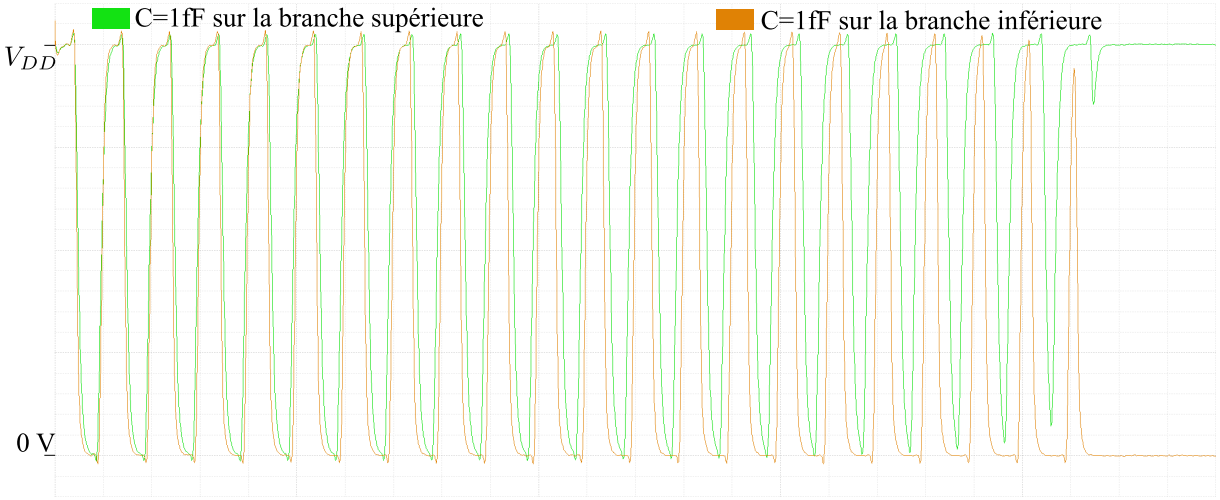


FIGURE 3.12: Simulation de cellules TERO non homogènes avec branches équilibrées et une charge en sortie

Cellules TERO homogènes avec branches déséquilibrées Cette fois-ci, tous les inverseurs qui composent la cellule TERO sont les mêmes. Cependant, le nombre d'inverseurs par branche est différent. La simulation a été réalisée pour une cellule TERO composée de 7 et 9 inverseurs, une cellule TERO composée de 7 et 5 inverseurs et une cellule TERO composée de 7 et 3 inverseurs.

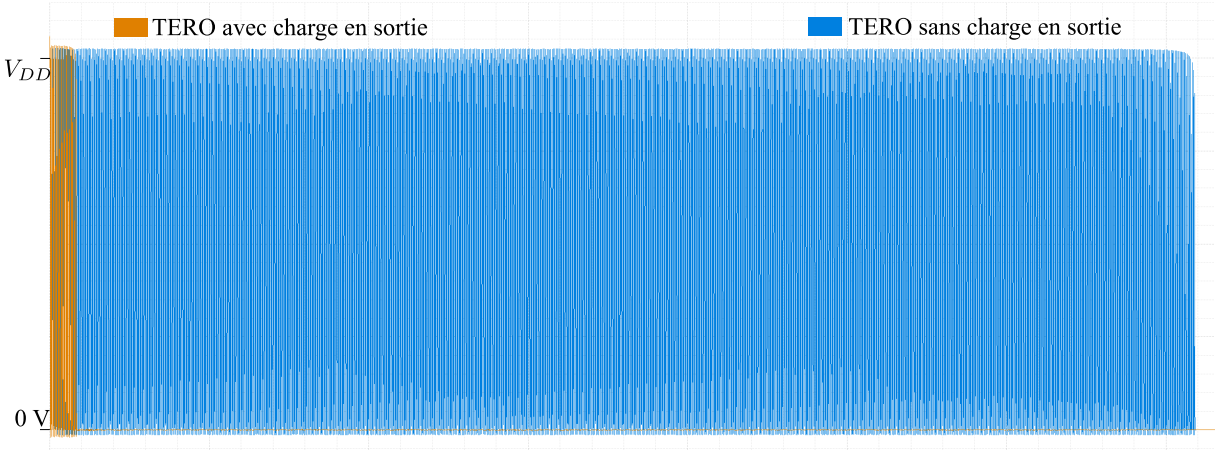


FIGURE 3.13: Simulation de cellules TERO avec ou sans charge en sortie

La simulation a aussi été réalisée pour une cellule TERO composée de 7 inverseurs par branche à titre de comparaison. Ainsi :

— Le temps de montée et de descente sera le même pour tous les inverseurs : $\Delta_{e_1/e_2} = 0$;
— Le signal d'activation (*ctrl*) arrivera au même moment sur les deux portes d'activation, mais le délai de parcourt sera différent d'une branche à l'autre donc : $T_{in} \neq T_{osc} - T_{in}$.

La figure 3.14 montre les résultats de simulation. Ces résultats montrent bien la présence de l'effet mémoire, car plus l'asymétrie entre les branches est importante plus les oscillations vont s'arrêter rapidement.

De plus, pour le cas de la cellule composée de 7 inverseurs par branche, le rapport cyclique n'a toujours pas commencé à changer alors que les autres cellules sont arrêtées.

Enfin, lorsque c'est la branche supérieure qui est la plus grande, la cellule s'arrête à l'état logique haut (V_{DD}). Dans le cas contraire, elle s'arrête à l'état logique bas (0 V).

Évolution du rapport cyclique d'une cellule TERO non homogène avec branches déséquilibrées Si nous regardons à présent l'évolution du rapport cyclique du signal de sortie d'une cellule TERO composée de 7 et 5 inverseurs et un inverseur différent des autres comme représenté sur la figure 3.15, il est intéressant de remarquer trois phases d'évolution du rapport cyclique :

— Il commence par évoluer de manière linéaire, en accord avec l'équation 3.19. Ceci est dû à l'effet de l'inhomogénéité des inverseurs.
— Ensuite, le rapport cyclique évolue de manière exponentielle, quand l'asymétrie devient importante, augmentant ainsi l'impact de l'effet mémoire.
— Enfin, les impulsions s'arrêtent brutalement, lorsque la largeur d'impulsion T_{in} ou $T_{osc}-T_{in}$ devient inférieure à T_{lim}. Ce sont les conséquences de l'effet du limiteur de pente.

L'analyse décomposée de l'évolution du rapport cyclique montre bien l'impact de chaque effet

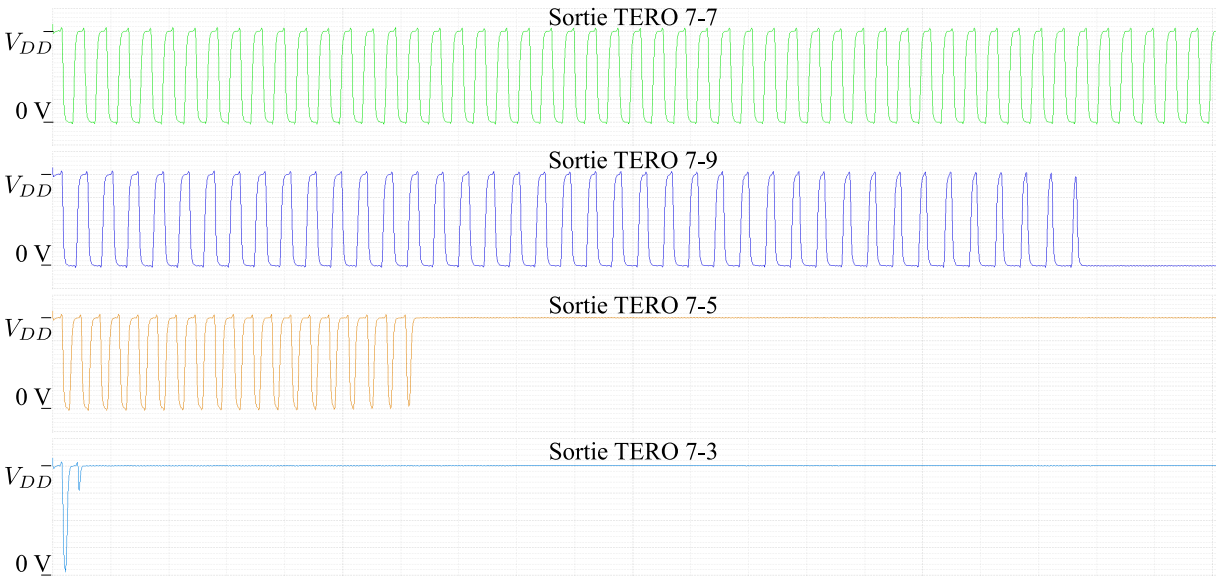


FIGURE 3.14: Simulation de cellules TERO homogènes avec branches déséquilibrées

décrit par Reyneri *et al.* et confirme donc la validité du modèle physique établi au regard du modèle électrique d'un transistor.

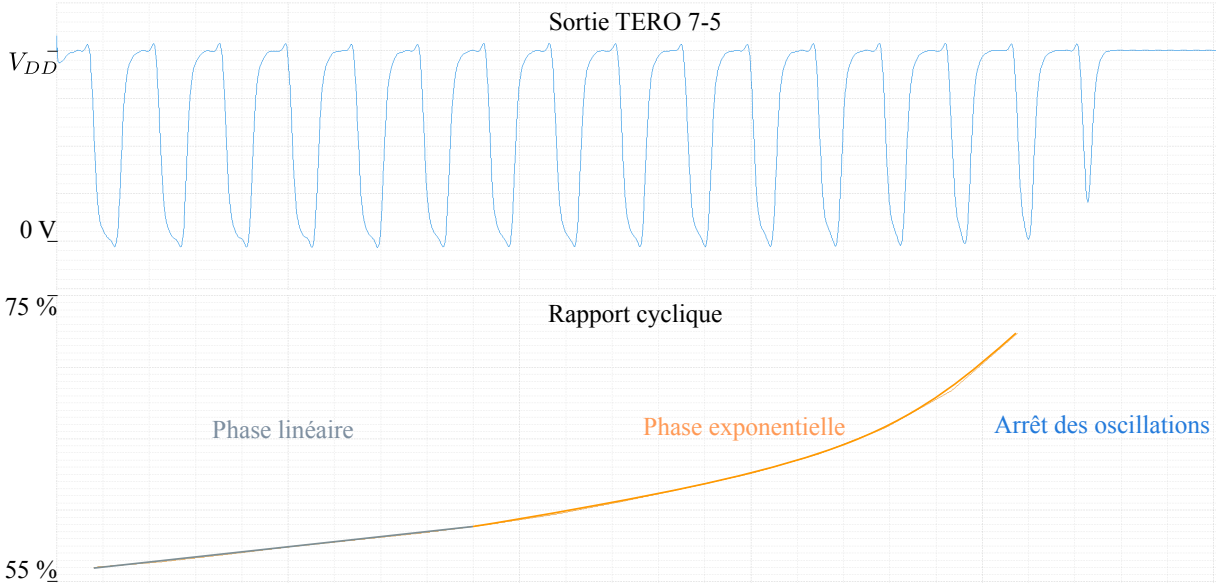


FIGURE 3.15: Évolution du rapport cyclique d'une cellule TERO 7-5

3.4.5.2 Expérimentation sur des circuits intégrés TTL 74HC00/04

Dans un second temps, nous avons réalisé, de la même manière, des expérimentations sur plusieurs structures de cellule TERO cette fois-ci à l'aide de circuits intégrés SN74HC00N et SN74HC04N de Texas Instruments inc.. Le circuit SN74HC00N contient des portes NON-ET-

logique à deux entrées et le circuit SN74HC04N des inverseurs logiques. De cette manière, nous avons réalisé une cellule TERO à l'aide de ces circuits, d'une platine et de fils de connexions. Une photo du montage est représentée sur la figure 3.16. D'après la documentation technique des circuits SN74HC00N et SN74HC04N, les temps de montée, t_m, et de descente, t_d, des portes logiques sont égaux et sont de 6ns. Nous avons donc $\Delta_{e_1/e_2} = 0$. L'expérience a été faite pour une cellule TERO composée de 1 et 1 inverseur, une cellule TERO composée de 1 et 3 inverseurs et une cellule TERO composée de 1 et 5 inverseurs, une cellule TERO composée de 3 et 1 inverseurs et une cellule TERO composée de 5 et 1 inverseurs. Les résultats expérimentaux observés à l'aide d'un oscilloscope sont représentés sur la figure 3.17. Pour chaque structure sont représentés le signal d'activation (*ctrl*) et le signal de sortie (*sortie*). Ces résultats sont en accord avec les simulations précédentes puisqu'ils confirment bien la présence de l'effet mémoire : plus l'asymétrie entre les branches est importante plus les oscillations vont s'arrêter rapidement.

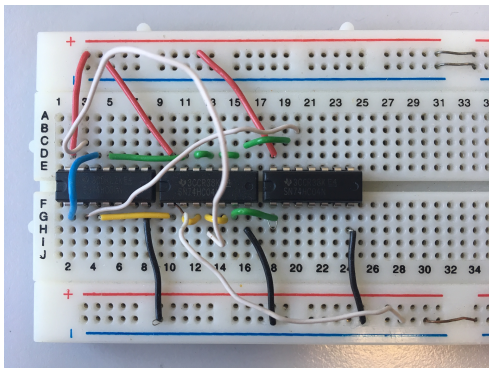


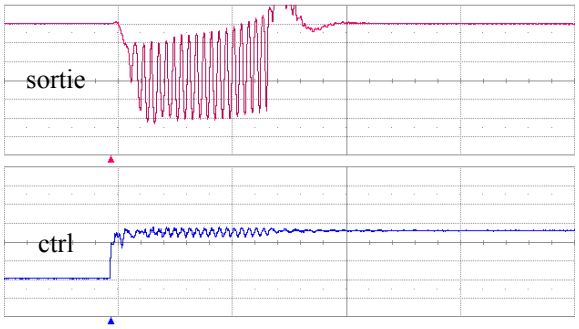
FIGURE 3.16: Cellule TERO réalisée à l'aide de circuits SN74HC00N et SN74HC04N

3.4.6 Conclusion

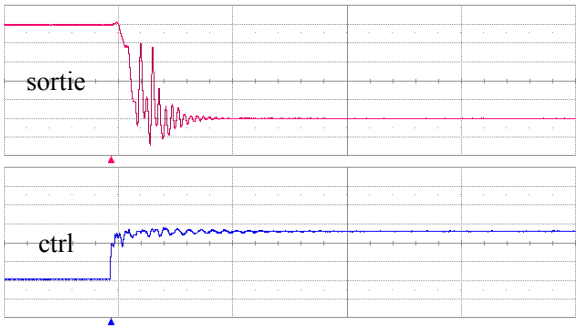
Ces expérimentations et simulations nous ont permis de valider tous les effets décrits par Reyneri *et al.*. Cependant, si l'effet mémoire et l'effet de l'inhomogénéité des inverseurs interviennent tous les deux, ce qui est le cas dans la réalité à cause des variations de procédés de fabrication, nous ne sommes pas en mesure d'établir une relation mathématique liant les paramètres d'entrées (T_{in}, Δ_{e_1/e_2}, etc.) et le nombre d'oscillations, N_{osc}.

En effet, si tous les inverseurs possèdent des temps de montée, t_m, et descente, t_d, différents, chaque inverseur aura un T_{lim} et un K_0 propre, et donc un H_d propre pour la montée et pour la descente. Par conséquent, l'équation 3.21 n'est plus valable et une solution générale dérivant N_{osc} n'existe pas.

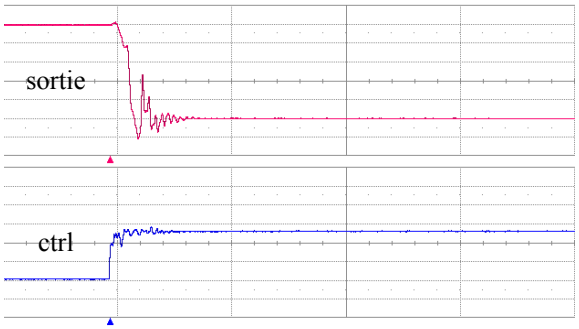
L'absence de modèle physique régissant N_{osc} en fonction des paramètres d'entrées nous empêche d'établir un modèle stochastique de la cellule TERO en suivant cette approche.



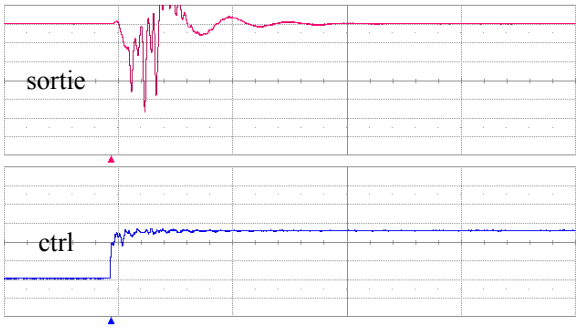
(a) Cellule TERO 1-1



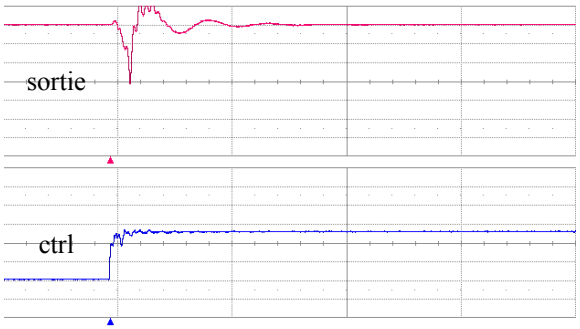
(b) Cellule TERO 1-3



(c) Cellule TERO 1-5



(d) Cellule TERO 3-1



(e) Cellule TERO 5-1

FIGURE 3.17: Signal de sortie de plusieurs structures de cellules TERO à base de circuits SN74HC00/04N

À présent, intéressons-nous aux différentes variations qui impactent les circuits intégrés pour tenter d'établir un modèle stochastique de la cellule RO basé sur son modèle électrique décrit dans la section 3.3 et sur l'impact des variations au niveau du transistor. L'objectif est d'obtenir la distribution de la fréquence d'oscillation des RO en fonction de leurs paramètres d'entrées (nombre d'inverseurs N, temps de montée t_m, temps de descente t_d et délai de propagation t_{l_i}). Dans le mesure où une équation permettant d'obtenir N_{osc} en fonction des paramètres d'entrée existerai, la même démarche aurait été suivi pour la cellule TERO. Malheureusement, aucun modèle physique prenant en compte tous les paramètres d'entrées ayant une influence sur N_{osc} n'existe pour le moment.

3.5 Différentes sources de variations dans les circuits intégrés

Les sources de variations au sein des circuits intégrés peuvent être divisées en quatre familles : les variations de procédés de fabrication, les variations de tension d'alimentation, les variations de température et le vieillissement des circuits.

Comme, nous l'avons vu dans la section 1.3.2, il existe deux catégories de variations de procédés de fabrication : les variations globales et les variations locales.

Pour cette première approche de modélisation des PUF, nous faisons trois hypothèses. Premièrement, la conception de la PUF que nous souhaitons modéliser a été réalisée de manière rigoureuse. Cela signifie que les méthodes pour réduire les variations globales introduites dans la section 1.3.2 ont été appliquées de telle sorte que nous pouvons ignorer les variations globales.

Deuxièmement, en considérant l'alimentation et l'environnement de fonctionnement du circuit de très bonne qualité en matière de stabilité, nous pouvons également ignorer les variations sur ces deux paramètres. De plus, le résultat de la PUF (fréquence d'oscillation de la cellule RO ou nombre d'oscillations de la cellule TERO) est suffisamment moyenné afin de pouvoir éliminer le bruit.

Enfin, en admettant que les composants de la PUF ne soient utilisés que par celle-ci et que le nombre de sollicitations d'une PUF est faible quelque soit son utilisation (authentification, génération de clé, etc.), nous pouvons négliger l'impact du vieillissement du circuit sur une plage d'utilisation humainement raisonnable de celui-ci (quelques dizaines d'années).

Ces hypothèses nous permettent d'affirmer que seules les variations locales affectent le circuit. Les variations locales sont le phénomène stochastique exploité par les PUF et donc le phénomène que nous souhaitons inclure à notre modèle. L'objectif est de partir de l'impact de ces variations locales sur les paramètres physiques du transistor pour ensuite connaitre leurs répercutions sur les paramètres électriques du transistor et déduire leurs influences sur le délai d'un inverseur puis sur la fréquence d'oscillation d'une cellule RO.

Un tel modèle permettrait de décrire mathématiquement le lien entre les variations stochastiques locales au niveau du transistor au moment de sa fabrication et les variations de fréquence d'une cellule oscillante. Ainsi, pour l'exemple de la cellule RO, nous serions en mesure d'estimer les distributions de probabilité de sa fréquence d'oscillation.

Intéressons-nous à présent aux différentes sources de variations locales et leur impact sur le transistor.

3.5.1 Variations locales

Il existe quatre types de variations locales qui affectent un transistor : la fluctuation aléatoire des dopants dans le semiconducteur, la variation de l'épaisseur d'oxyde entre le métal et le

semiconducteur, la granularité du métal (Poly-silicium) et la rugosité des bords de la grille du transistor [96].

La figure 3.18 représente ces variations à l'échelle d'un transistor. À titre de comparaison avec un transistor idéal, se reporter à la figure 3.1.

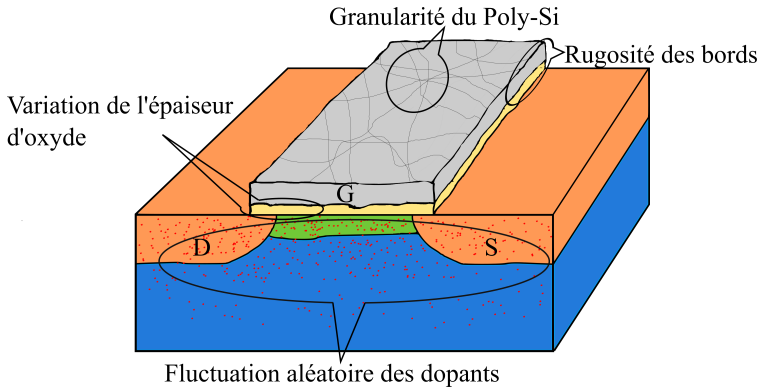


FIGURE 3.18: Impact des variations de procédés de fabrication au niveau du transistor

3.5.1.1 Fluctuations aléatoires des dopants

Le canal de conduction du transistor est dopé avec des atomes impurs qui servent de porteurs de charges. Ces atomes impurs sont placés aléatoirement dans le canal grâce à des techniques telles que l'implantation de dopants qui entrainent des variations statistiques dans le nombre d'impuretés implantées. Ces variations de concentration des porteurs de charges influencent la tension de seuil (V_{th}) et le facteur de courant (β) du transistor. Dans les technologies de transistor plus anciennes (>100 nm), le nombre de dopants était de quelques milliers. Une déviation absolue de quelques dizaines d'atomes était négligeable. Dans les technologies récentes, le nombre de dopants est de l'ordre de la dizaine rendant l'impact des fluctuations aléatoires des dopants très important.

3.5.1.2 Variations de l'épaisseur d'oxyde

L'oxyde de grille peut être développé avec une précision absolue de 1 à 2 couches interatomiques. Dans les technologies précédentes avec une épaisseur d'oxyde de dizaines de couches interatomiques, les variations de la tension de seuil induites par l'épaisseur d'oxyde étaient presque négligeables. Cependant, dans les technologies inférieures à 30 nm avec des épaisseurs d'oxyde comprises entre 1 et 3 nm (environ 5 à 15 espacements interatomiques), les variations de l'épaisseur d'oxyde peuvent contribuer à l'incertitude de la tension de seuil et du facteur de courant autant que les fluctuations aléatoires des dopants.

3.5.1.3 Granularité du métal

La nature polycristalline de la grille métallique du transistor représente aussi une source de variations. Les grains de métal ayant des orientations cristallographiques différentes ont des fonctions de travail différentes, ce qui entraine une variation aléatoire de la tension de seuil dans la région de la grille [97].

3.5.1.4 Rugosité des bords

La rugosité des bords représente la distorsion du bord de la grille, induite par la gravure de grille et le processus de lithographie. Tout comme pour les autres variations, la réduction de la taille de transistor augmente l'impact de la rugosité des bords sur la tension de seuil et le facteur de courant du transistor en raison d'un effet de canal court de plus en plus important.

3.5.1.5 Impact sur le transistor

Les variations locales sont par nature non corrélées. Ainsi chaque transistor sera affecté différemment. Comme nous venons de le voir, ces variations augmentent lorsque la taille des transistors MOS diminue. Par conséquent, la variance de la tension de seuil $\sigma^2_{V_{th}}$ et du facteur de courant σ^2_β augmente. Ceci est dû au fait que, comme l'ont montré Pelgrom *et al.*, la variance de ces deux paramètres est inversement proportionnelle à la taille des transistors [62] :

$$\sigma^2_{V_{th}} = \frac{A^2_{V_{th}}}{WL} \tag{3.22}$$

où $A^2_{V_{th}}$ correspond au paramètre caractérisant la technologie de transistor utilisée et une chaine de fabrication particulière.

$$\sigma^2_\beta/\beta^2 = \frac{A^2_\beta}{WL} \tag{3.23}$$

où A^2_β correspond au paramètre caractérisant la technologie de transistor utilisée et une chaine de fabrication particulière.

Pelgrom *et al.* dérivent ensuite les variations du courant du transistor en fonction des variations de ces deux paramètres comme suit :

$$\sigma^2_{I_{ds}}/I^2_{ds} = \frac{4 \times \sigma^2_{V_{th}}}{(V_{gs} - V_{th})^2} + \sigma^2_\beta/\beta^2 \tag{3.24}$$

Il est important de constater ici que l'impact des variations de procédés de fabrication sur le transistor est propre à la technologie de transistor et à la chaine de fabrication utilisée. Le modèle sera donc dépendant de la technologie.

3.5.1.6 Impact sur le temps de commutation d'une chaine d'inverseurs

En supposant que nous ayons accès aux paramètres $A^2_{V_{th}}$ et A^2_{β} pour une technologie donnée, il est relativement simple, grâce aux travaux de Pelgrom *et al.* de remonter l'impact des variations locales sur le temps de commutation d'un inverseur. Soit T_{porte} une variable aléatoire gaussienne représentant le temps de commutation d'un inverseur. Ainsi, nous avons :

$$T_{porte} \sim \mathcal{N}(\mu_{t_{porte}}, \sigma^2_{t_{porte}}) \tag{3.25}$$

Puisque les variations locales ne sont pas corrélées, le délai de propagation, que nous noterons T_{prop}, à travers N inverseurs chainés ayant chacun un délai T_{porte} est décrit par la loi suivante :

$$T_{prop} \sim \mathcal{N}(N \times \mu_{t_{porte}}, N \times \sigma^2_{t_{porte}}) \tag{3.26}$$

Tachons à présent d'appliquer cela au cas d'une cellule RO.

3.6 Modélisation stochastique de la cellule RO

Soit T_d une variable aléatoire gaussienne représentant le temps de descente (de V_{DD} à $\frac{V_{DD}}{2}$) d'un inverseur en réponse à un front montant sur V_{in}. Soit T_m une variable aléatoire gaussienne représentant le temps de montée (de $0\,V$ à $\frac{V_{DD}}{2}$) d'un inverseur en réponse à un front descendant sur V_{in}. Soit T_{li} une variable aléatoire gaussienne représentant le temps de propagation à travers une ligne de longueur l. Soit μ_{t_d} le temps moyen auquel V_{out} atteint $\frac{V_{DD}}{2}$ après un front montant sur V_{in}, μ_{t_m} le temps moyen auquel V_{out} atteint $\frac{V_{DD}}{2}$ après un front descendant sur V_{in} et $\mu_{t_{li}}$ le temps moyen de propagation à travers une ligne de longueur l. Ainsi, nous avons :

$$T_d \sim \mathcal{N}(\mu_{t_d}, \sigma^2_{t_d}),\ T_m \sim \mathcal{N}(\mu_{t_m}, \sigma^2_{t_m}) \text{ et } T_{li} \sim \mathcal{N}(\mu_{t_{li}}, \sigma^2_{t_{li}}) \tag{3.27}$$

Ce qui nous donne pour T_{osc}, une variable aléatoire représentant la période d'oscillation d'une cellule RO :

$$T_{osc} \sim \mathcal{N}(\mu_{t_{osc}}, \sigma^2_{t_{osc}}) \text{ avec } \begin{cases} \mu_{t_{osc}} = N \times \mu_{t_m} + N \times \mu_{t_d} + 2N \times \mu_{t_{li}} \\ \sigma^2_{t_{osc}} = N \times \sigma^2_{t_d} + N \times \sigma^2_{t_m} + 2N \times \sigma^2_{t_{li}} \end{cases} \tag{3.28}$$

3.6.1 Estimation des paramètres du modèle

Le kit de conception STMicroelectronics CMOS en technologie 65nm dont nous disposons ne nous permet pas d'accéder aux paramètres du modèle de variations locales au niveau du transistor.

Malgré tout, il est possible de réaliser des simulations Monte-Carlo basées sur ce modèle pour estimer les paramètres d'un inverseur. Nous avons donc réalisé une simulation Monte-Carlo de

la réponse d'un inverseur à un stimulus d'entrée. Cela nous permet de mesurer l'écart type et la valeur moyenne du front montant et du front descendant de l'inverseur.

La figure 3.19 représente le résultat de 1000 échantillons successifs pour la simulation Monte-Carlo d'un front descendant de l'inverseur. La figure 3.20 représente les distributions discrètes de T_d et T_m mesurées grâce à la simulation Monte-Carlo.

Les résultats obtenus sont les suivants :

$$T_m \sim \mathcal{N}(\mu_{t_m}, \sigma^2_{t_m}) \text{ avec } \begin{cases} \mu_{t_m} = 60,6762 \text{ ps} \\ \sigma_{t_m} = 1,50865 \text{ ps} \end{cases} \tag{3.29}$$

$$T_d \sim \mathcal{N}(\mu_{t_d}, \sigma^2_{t_d}) \text{ avec } \begin{cases} \mu_{t_d} = 58,3180 \text{ ps} \\ \sigma_{t_d} = 1,02944 \text{ ps} \end{cases} \tag{3.30}$$

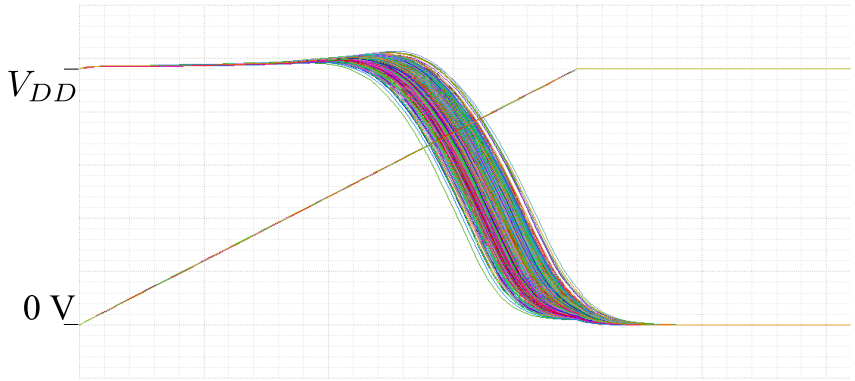
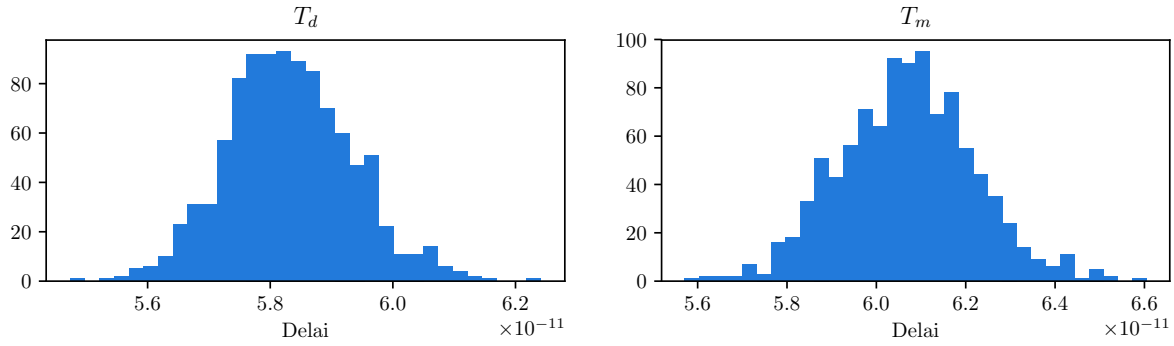


FIGURE 3.19: Simulation Monte-Carlo des variations locales sur un inverseur logique



(a) Distribution discrète de T_d

(b) Distribution discrète de T_m

FIGURE 3.20: Estimation des paramètres T_d et T_m grâce à une simulation Monte-Carlo

En suivant la même démarche pour une ligne de délai, il est ensuite possible de connaître la distribution de T_{osc} pour une cellule RO composée de N inverseurs en technologie STMicroelectronics CMOS 65nm. Dans notre cas, le kit de conception ne nous permet pas de réaliser des

simulations Monte-Carlo sur la ligne de délai. Pour cet exemple nous allons négliger le temps de propagation dans les lignes d'interconnexion. En négligeant les délais t_{li_i}, les résultats des simulations Monte-Carlo de T_d et T_m couplés à l'équation 3.28, nous donne :

$$T_{osc} \sim \mathcal{N}(\mu_{t_{osc}}, \sigma^2_{t_{osc}}) \text{ avec } \begin{cases} \mu_{t_{osc}} = N \times 60,6762 + N \times 58,3180 \text{ ps} \\ \sigma^2_{t_{osc}} = N \times (1,02944)^2 + N \times (1,50865)^2 \text{ ps} \end{cases} \tag{3.31}$$

Si nous prenons l'exemple d'une cellule RO composée de 3 inverseurs par branche, le modèle nous donne les paramètres suivants :

$$T_{osc} \sim \mathcal{N}(\mu_{t_{osc}}, \sigma^2_{t_{osc}}) \text{ avec } \begin{cases} \mu_{t_{osc}} = 356,9826 \text{ ps} \\ \sigma^2_{t_{osc}} = 10,0071 \text{ ps} \end{cases} \tag{3.32}$$

La distribution associée est représentée sur la figure 3.21.

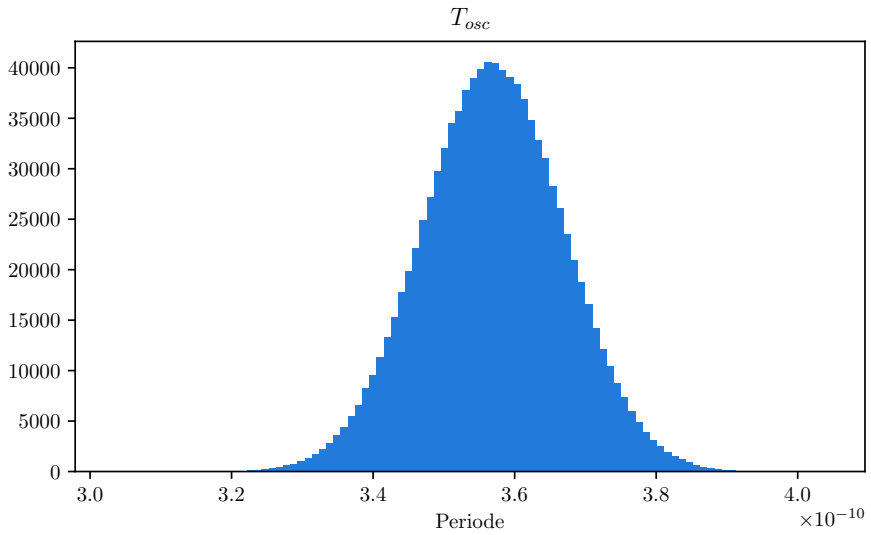


FIGURE 3.21: Distribution de T_{osc}

Ensuite, à l'aide des échantillons de la distribution de T_{osc}, il suffit de tracer la distribution de $\frac{1}{T_{osc}}$. Le résultat est représentée sur la figure 3.22.

Par conséquent, le modèle prédit, pour une cellule RO composée de 3 inverseurs par branche, une fréquence d'oscillation moyenne de 2,8 GHz, une fréquence minimale d'environ 2,6 GHz et une fréquence maximale d'environ 3 GHz.

3.6.2 Validation expérimentale et conception d'un ASIC

Il convient à présent de valider notre modèle avec des résultats expérimentaux. Pour ce faire, il faut tout d'abord obtenir la distribution de T_{osc} par mesure expérimentale pour ensuite la comparer à notre modèle.

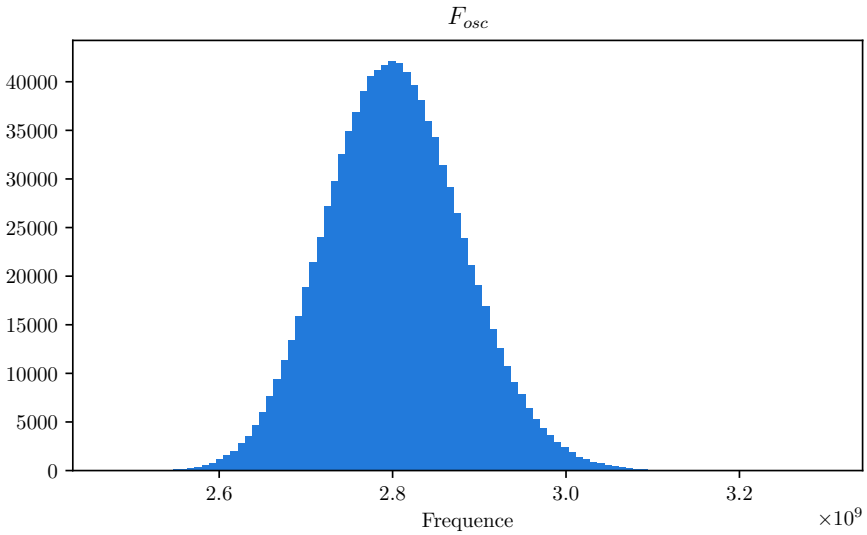


FIGURE 3.22: Distribution de F_{osc}

Nous avons donc réalisé un circuit intégré de test en technologie STMicroelectronics CMOS 65nm avec plusieurs structures de cellules RO implantées dessus. Une image du dessin de l'ASIC conçu est représentée sur la figure 3.23. Cet ASIC contient :

- Une RO-PUF ;
- Une TERO-PUF ;
- Des cellules TERO de différentes tailles, homogènes, non homogènes, avec branches équilibrées ou non, comme présenté dans la partie 3.4.5 ;
- Des cellules RO de plusieurs tailles et en utilisant plusieurs inverseurs logiques disponibles dans la bibliothèque du kit de conception ;
- Un ERO-TRNG ;
- Plusieurs STR-TRNG.

À ce jour, l'ASIC n'est pas encore exploitable. La prochaine étape de la démarche suivie dans ce chapitre est de valider le modèle stochastique de la cellules RO à l'aide des résultats expérimentaux de l'ASIC.

3.7 Conclusion

Au cours de ce chapitre, nous avons amorcé une approche de modélisation stochastique des cellules oscillantes qui composent les PUF en nous basant sur un modèle électrique de celles-ci.

Pour cela, nous avons commencé par modéliser le comportement électrique d'un inverseur logique en se basant sur les transistors qui le composent. Cette modélisation électrique nous a permis d'établir des équations du temps de montée et du temps de descente de l'inverseur. Nous

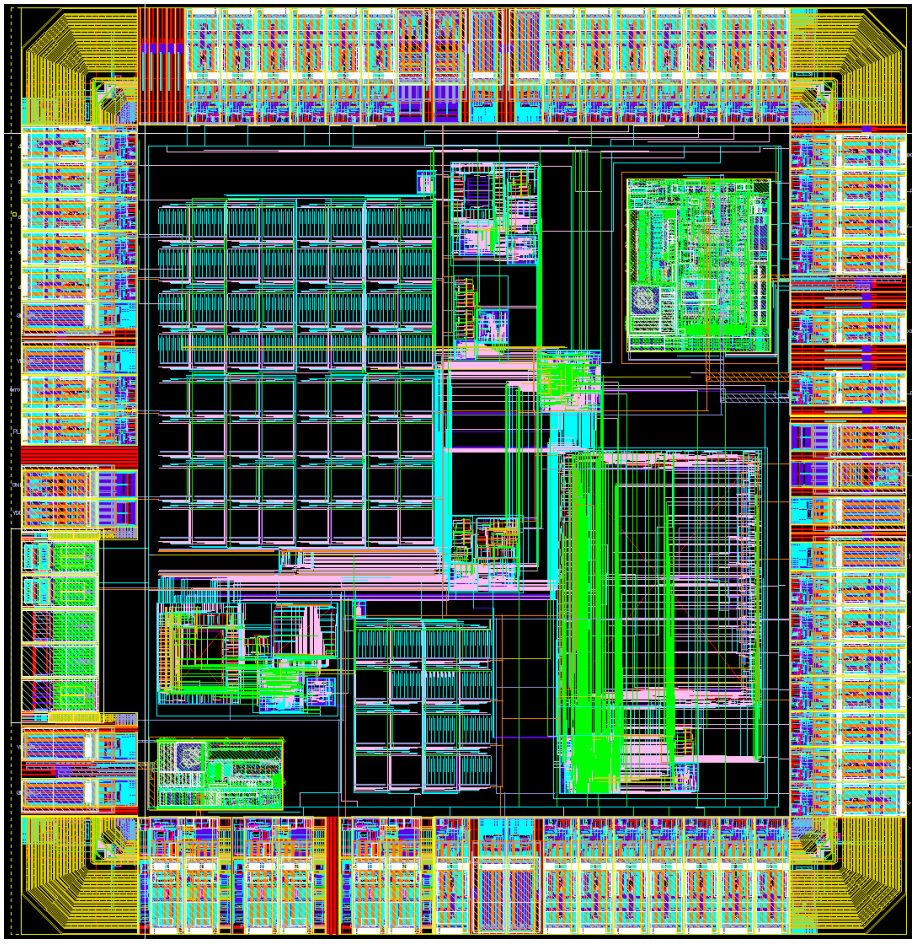


FIGURE 3.23: Impression-écran du dessin de l'ASIC

avons aussi montré que ce modèle était valable pour une porte NON-ET-logique.

Ensuite, en nous basant sur le modèle électrique d'une ligne de transmission sans pertes, nous avons pu établir une équation qui définit le temps de propagation à travers une ligne de longueur l.

La modélisation électrique de ces deux éléments (la porte logique et la ligne de transmission), nous a permis de dériver une équation de la fréquence d'oscillation, f_{osc}, d'une cellule oscillante.

Par la suite, nous avons validé, à l'aide de simulations et d'expérimentations, le modèle analytique de la cellule TERO établi par Reyneri *et al.*. Malheureusement, nous n'avons pas été en mesure d'établir une équation qui définit le nombre d'oscillations, N_{osc}, d'une cellule TERO en fonction des paramètres d'entrées responsables de l'arrêt des oscillations.

En nous plaçant dans un cas de figure rendant l'hypothèse que seules les variations locales affectent un circuit intégré raisonnable, nous avons ensuite établi un modèle stochastique du temps de montée et de descente d'un inverseur logique. Cela nous a permis d'établir un modèle stochastique de la période d'oscillation T_{osc} d'une cellule RO.

À l'aide du kit de conception STMicroelectronics CMOS en technologie 65nm et de la suite Virtuoso® fournit par Cadence®, nous avons estimé les paramètres du modèle du temps de montée et du temps de descente de l'inverseur. Cela nous à permis de déterminer des paramètres pour T_{osc} et ainsi de tracer la distribution de F_{osc}.

La prochaine étape consiste à vérifier notre modèle à l'aide d'expérimentations sur un ASIC en technologie STMicroelectronics CMOS 65nm. Malheureusement, le circuit fabriqué n'est pas encore exploitable à ce jour.

Malgré tout, ces travaux constituent une première initiative de modélisation des cellules oscillantes qui constituent les PUF et nous espérons fortement qu'ils seront réutilisés dans le futur pour finir la démarche initiée à travers ce chapitre.

Dans le chapitre suivant, nous nous intéressons au phénomène de verrouillage sur les trois cellules oscillantes utilisées pour la génération d'aléa : les RO, les TERO et les STR. Nous proposons ensuite des solutions pour limiter ce phénomène dans les circuits numériques.

Chapitre 4

Étude de l'impact du verrouillage sur les cellules oscillantes

Dans le chapitre 1, nous avons pu prendre conscience, à travers les travaux de Marketos et Moore [83], Bochard *et al.* [84], Bernard *et al.* [71] ou encore Bayon *et al.* [85], de l'impact du phénomène de verrouillage sur les cellules RO (voir sec. 1.4.1.2). En effet, le verrouillage peut totalement compromettre la qualité, en matière d'imprévisibilité, des nombres fournis par un générateur physique d'aléa. Nous avons aussi vu qu'un certain nombre de points importants n'avaient pas été explorés. Effectivement, il n'existe dans la littérature que très peu d'études qui traitent du phénomène de verrouillage dans les circuits numériques et toutes portent sur un seul type de cellule oscillante, le RO.

Dans ce chapitre, nous réalisons une étude complète de la sensibilité au phénomène de verrouillage des cellules RO, TERO et STR. Cette analyse est réalisée sur FPGA pour deux raisons :

— Plusieurs architectures et topologies peuvent être étudiées en reconfigurant simplement le FPGA ;

— Les fréquences d'oscillation des cellules, à taille égale, sont plus faibles sur FPGA que sur ASIC ce qui permet d'observer le signal en dehors du circuit à l'aide d'un oscilloscope et d'une interface LVDS.

Le phénomène de verrouillage est étudié sur trois familles de FPGA provenant des trois principaux fabricants : Xilinx Spartan 6 et Intel Cyclone V qui représentent les FPGA à base mémoire SRAM et Microsemi SmartFusion 2 qui représente les FPGA à base mémoire Flash.

Toutes les sources VHDL sont disponibles sur Git pour assurer la répétabilité de nos expériences.

. Le code associé à ce chapitre est disponible à l'adresse suivante :
https://gitlab.univ-st-etienne.fr/ugo.mureddu/locking-osicllating-cells.git

4.1 Banc d'expérimentation

L'impact du phénomène de verrouillage sur les cellules oscillantes est évalué avec le banc d'expérimentation représenté sur la figure 4.1. Il est composé de :

— **Une plateforme matérielle à base de FPGA** – un ensemble de cartes d'évaluation (Evariste) consistant en une carte mère assurant la communication avec le PC hôte et plusieurs cartes filles présentant différents types de FPGA. Comme pour la plateforme HECTOR présentée dans le chapitre 2, la modularité de la plateforme Evariste permet d'étendre notre étude sur plusieurs familles de FPGA. Pour cette étude, la plateforme Evariste, par rapport à la plateforme HECTOR, présente l'avantage de fournir plus d'entrées et de sorties. Plus de détails sur la plateforme matérielle Evariste sont disponibles dans [98].

— **Un générateur de fonctions**, modèle 81160A de la marque Agilent Technologies permettant de générer des signaux carrés à une fréquence allant jusqu'à 330 MHz et des signaux sinusoïdaux à une fréquence allant jusqu'à 500 MHz. Le générateur de fonctions sert à produire le signal de perturbation et à déclencher les oscillations des cellules étudiées.

— **Un oscilloscope**, modèle WaveRunner 640Zi de la marque LeCroy possédant une plage de fréquence allant jusqu'à 4 GHz et une capacité d'enregistrement de 40 giga échantillons par seconde. L'oscilloscope enregistre les signaux de sorties des cellules étudiées ainsi que celui du signal perturbateur. Il sert à déclencher l'enregistrement des signaux provenant à la fois des cellules oscillantes et du signal perturbateur sur un front du signal d'activation.

— **Des sondes différentielles**, modèle WL-PBus de la marque LeCroy pour transférer les signaux à haute fréquence depuis le FPGA vers l'oscilloscope.

— **Un PC** en charge de l'exécution des scripts qui contrôle le FPGA, l'oscilloscope et le générateur de fonctions.

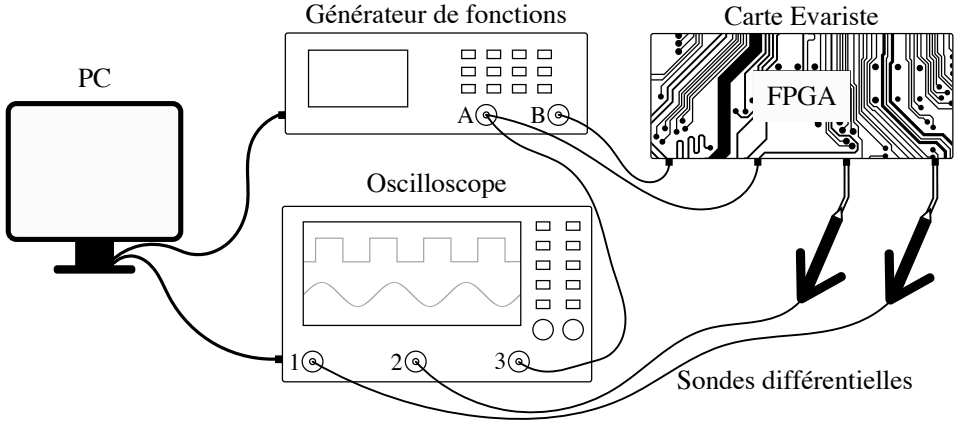


FIGURE 4.1: Banc d'expérimentation

loa 1: Script Python de contrôle du banc d'expérimentation

Programmation du FPGA
for $f_{pert} = f_{init}$ **à** f_{final} **do**
  if front montant de *ctrl* **then**
    Démarrer l'acquisition des données
  end if
end for

La sortie A du générateur de fonctions génère un signal carré basse fréquence de contrôle, utilisé pour activer à la fois les cellules oscillantes et l'enregistrement des données de l'oscilloscope. La sortie B génère un signal sinusoïdal de perturbation à une fréquence proche de la fréquence d'oscillation de la cellule étudiée.

Afin d'imiter les interactions entre la cellule oscillante et la logique environnante et ainsi se placer dans un cas d'application réaliste, le signal de perturbation n'est pas injecté directement dans la cellule oscillante, mais passe à travers une ligne à retard placée près de la cellule. Pour rendre les interactions entre la cellule étudiée et la ligne de délai les plus importantes possible, les éléments qui composent la cellule et la ligne de délai sont entrelacés comme représenté sur la figure 4.2.

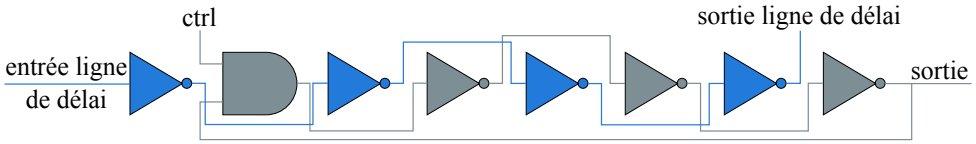


FIGURE 4.2: Placement de la cellule oscillante et de la ligne de délai dans le FPGA

Les sorties des cellules étudiées ainsi que la sortie du signal de perturbation sont observées à l'aide des sondes différentielles. L'expérience est répétée pour plusieurs fréquences du signal de perturbation. L'algorithme 1 décrit le pseudo-code du script Python permettant de contrôler le banc d'expérimentation.

Ici, nous venons de voir le banc d'expérimentation permettant d'évaluer le phénomène de verrouillage sur une cellule oscillante. Intéressons-nous à présent aux moyens qui permettent de détecter la présence de verrouillage.

4.2 Preuves de verrouillage

Soit f_{osc} la fréquence d'oscillation du signal de sortie d'une cellule oscillante et φ_{osc} la phase du signal de sortie d'une cellule oscillante. Lorsque le signal de sortie de la cellule est verrouillé avec le signal de perturbation, la cellule va dévier de ses conditions normales de fonctionnement.

Ainsi, les paramètres électriques suivants pourront être mesurés pour détecter le phénomène de verrouillage :

4.2.1 Différence de fréquence

En présence d'un signal de perturbation, le signal de sortie d'une cellule oscillante peut se verrouiller sur la fréquence de ce signal perturbateur. Une fois verrouillée, la différence de fréquence entre les deux signaux, notée $\overline{\Delta f}$, sera, en moyenne, nulle. Dans le reste du manuscrit, $\overline{\Delta f}$ sera égal à $\left|\overline{f_{pert} - f_{osc}}\right|$, où f_{pert} représente la fréquence d'oscillation du signal de perturbation (en Hz).

4.2.2 Déviation standard de la période

Comme nous avons pu le voir dans le chapitre 1, les cellules oscillantes présentent une grande sensibilité au bruit. Par conséquent, la période (ou fréquence) de leur signal de sortie n'est pas totalement stable. En revanche, une fois verrouillée, la période du signal de sortie de la cellule devient plus stable qu'en fonctionnement normal. Ainsi, la déviation standard de la période, notée $\sigma_{T_{osc}}$ (en seconde), devient plus faible.

4.2.3 Déphasage

Lorsque le verrouillage affecte une cellule oscillante, le signal de sortie de la cellule et le signal de perturbation n'ont pas forcément la même phase. Cependant, le déphasage moyen entre les deux signaux, noté $\overline{\Delta_\varphi}$, reste constant pendant toute la plage de verrouillage. Dans le reste du manuscrit, $\overline{\Delta_\varphi}$ est égal à $|\overline{\varphi_{pert} - \varphi_{osc}}|$, où φ_{pert} représente la phase du signal de perturbation (en °).

4.2.4 Déviation standard du déphasage

De la même manière, deux signaux indépendants auront par définition une fonction cumulative de distribution du déphasage uniforme. Cela correspond à une déviation standard du déphasage de 28,86 % ou 104 °. Cette déviation standard, notée σ_{Δ_φ}, tend vers 0 en présence de verrouillage.

4.2.5 Paramètre additionnel pour la cellule TERO – le nombre d'oscillations

Dans le cas particulier de la cellule TERO, qui n'oscille que temporairement en fonctionnement normal, un autre paramètre permet de détecter le phénomène de verrouillage. En effet, les oscillations ne s'arrêtent pas en présence de verrouillage.

Tout système sensible au phénomène de verrouillage a 2 plages distinctes : la plage de verrouillage et la plage d'accrochage. La plage de verrouillage est définie comme la plage de fréquence sur laquelle le système verrouillé suit les changements du signal perturbateur. En d'autres termes, la fréquence du système verrouillé, f_{osc}, suit f_{pert}. La plage d'accrochage représente la plage de fréquence pendant laquelle un système non verrouillé va venir se verrouiller au signal perturbateur. En théorie, la plage d'accrochage est toujours inférieure à la plage de verrouillage.

Pour toutes nos expériences, nous n'avons pas été en mesure de distinguer une différence entre la plage de verrouillage et la plage d'accrochage. Pour cette raison, dans le reste du chapitre, seule la plage de verrouillage sera considérée.

4.3 Verrouillage des cellules RO

4.3.1 Verrouillage d'une cellule RO à l'aide d'un signal perturbateur

À l'aide du banc expérimental décrit dans la section 4.1, le comportement d'une cellule RO composée de 3 inverseurs ($N = 3$) et implantée sur un FPGA Xilinx Spartan 6 est étudié.

Sans aucune perturbation, la fréquence d'oscillation naturelle de la cellule RO est de 296 MHz.

La figure 4.3 montre le résultat des 4 preuves de verrouillage introduites ci-devant lorsqu'un signal perturbateur est injecté dans la ligne de délai entrelacée avec la cellule RO.

4.3.1.1 Différence de fréquence, $\overline{\Delta f}$

La partie supérieure de la figure 4.3 représente l'évolution de la fréquence d'oscillation du signal de sortie de la cellule RO (f_{osc}) en fonction de la fréquence d'oscillation du signal perturbateur (f_{pert}). f_{pert} varie de 285 MHz à 301 MHz par pas de 0,025 MHz. Lorsque f_{pert} est égal à 285 MHz, la cellule RO commence à être perturbée et dévie légèrement de sa fréquence d'oscillation naturelle. À partir de 288 MHz, la cellule RO se verrouille sur le signal perturbateur et le reste jusqu'à 296,5 MHz. Pendant toute la plage de verrouillage, puisque $f_{osc} = f_{pert}$, $\overline{\Delta f}$ est nulle. Afin de bien mettre en valeur le moment où la cellule RO est perturbée, la fréquence d'oscillation naturelle est aussi représentée sur la figure.

4.3.1.2 Déviation standard de la période, $\sigma_{T_{osc}}$

La deuxième partie de la figure 4.3 montre la déviation standard de la période du signal de sortie de la cellule RO ($\sigma_{T_{osc}}$) ainsi que la déviation standard de la période du signal perturbateur ($\sigma_{T_{pert}}$) en fonction de f_{pert}. $\sigma_{T_{osc}}$ comme $\sigma_{T_{pert}}$ ont été mesurées à l'aide de l'oscilloscope. Si la cellule RO n'est pas perturbée, $\sigma_{T_{osc}}$ moyen est autour de 20 ps. Puis, quand la cellule RO

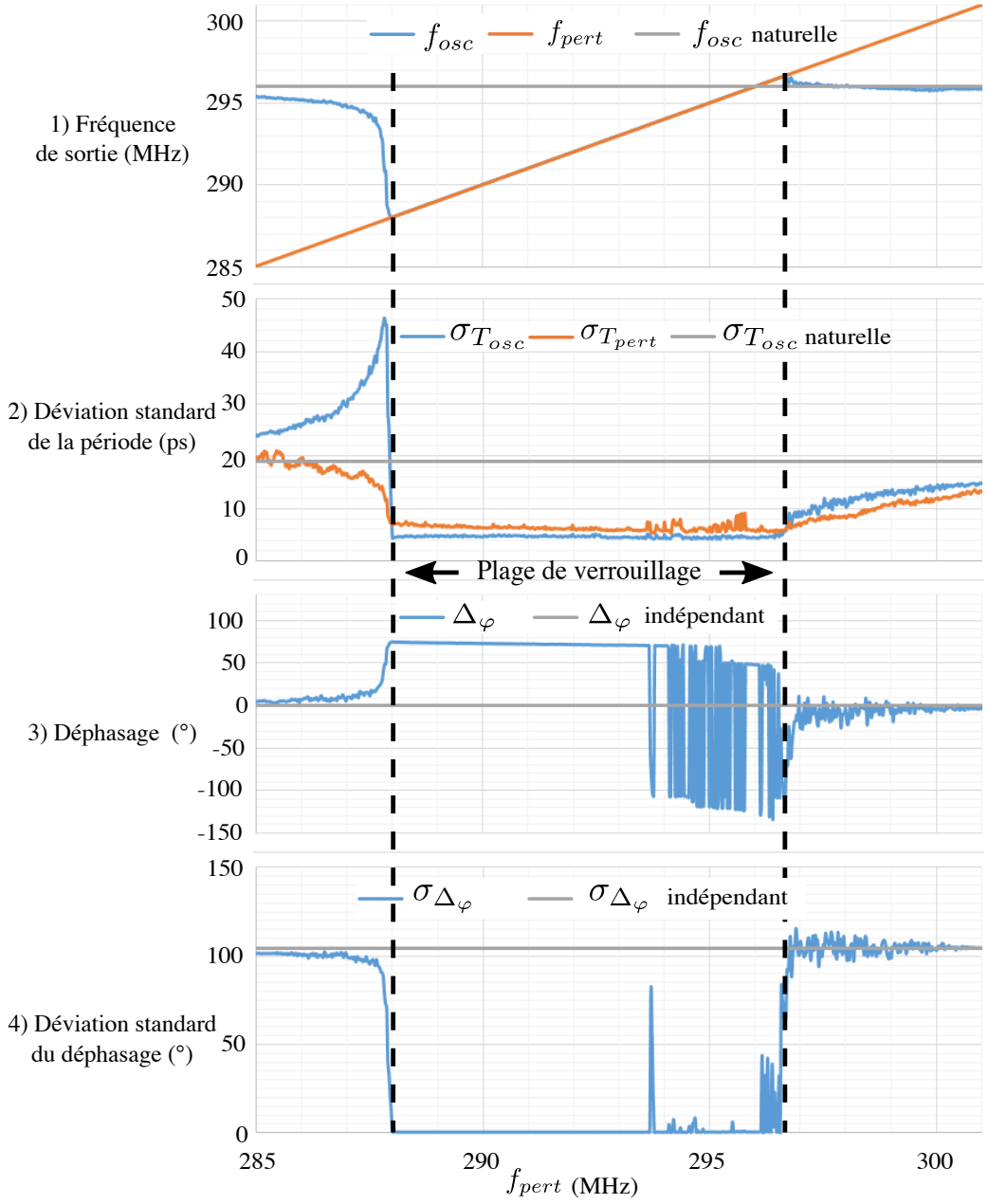


FIGURE 4.3: Preuves de verrouillage dans le cas d'une cellule RO avec $N = 3$ inverseurs et implantée sur Xilinx Spartan 6

commence à être perturbée, $\sigma_{T_{osc}}$ augmente pour atteindre sa valeur maximale à environ 50 ps et ensuite redescendre à environ 5 ps quand la cellule RO est verrouillée avec le signal perturbateur. Il est intéressant de constater que $\sigma_{T_{pert}}$ est aussi plus faible lorsque la cellule RO est verrouillée. Les valeurs de $\sigma_{T_{osc}}$ et $\sigma_{T_{pert}}$ restent basses de 288 MHz à 296,5 MHz, confirmant ainsi la plage de verrouillage observée avec la première méthode. À titre de comparaison, la valeur de $\sigma_{T_{osc}}$ pour cette même cellule RO non perturbée est aussi représentée sur la figure.

4.3.1.3 Déphasage, $\overline{\Delta_\varphi}$

La troisième partie de la figure 4.3 représente la valeur moyenne de la différence de phase ($\overline{\Delta_\varphi}$) entre la phase du signal de sortie de la cellule RO (φ_{osc}) et la phase du signal perturbateur (φ_{pert}) mesurée à l'oscilloscope. Comme décrit dans la section 4.2.3, lorsque la cellule oscillante n'est pas verrouillée avec le signal perturbateur, $\overline{\Delta_\varphi}$ est distribuée de manière uniforme entre -180 ° et 180 °. Par conséquent, $\overline{\Delta_\varphi}$ est centrée autour de 0 °. À l'inverse, lorsque la cellule oscillante est verrouillée avec le signal perturbateur, $\overline{\Delta_\varphi}$ reste constante et différente de 0. L'évolution de la valeur de $\overline{\Delta_\varphi}$ en fonction de f_{pert} confirme, encore une fois, la plage de verrouillage observée avec les deux méthodes précédentes puisque $\overline{\Delta_\varphi}$ est constante et égale à environ 70 ° entre 288 MHz et 293,5 MHz. Cependant, entre 293,5 MHz et 296,5 MHz, bien que $\overline{\Delta_\varphi}$ ait une valeur constante à chaque mesure, quelques inversions de phase apparaissent : $\overline{\Delta_\varphi}$ passe de 70 ° à -110 °. Ceci s'explique par le fait que à chaque augmentation de f_{pert} par pas de 0,025 MHz, le générateur de fonction coupe temporairement le signal perturbateur pour le configurer à sa prochaine valeur et relâche ainsi f_{osc} pour une période très courte. La resynchronisation des deux signaux ne se fait pas forcément sur le même front. Malgré tout, $\overline{\Delta_\varphi}$ reste constante et différente de 0 confirmant donc la plage de verrouillage. La ligne horizontale représente la valeur de $\overline{\Delta_\varphi}$ pour deux signaux indépendants.

4.3.1.4 Déviation standard du déphasage, σ_{Δ_φ}

Enfin, la dernière partie de la figure 4.3 montre la déviation standard du déphasage (σ_{Δ_φ}) en fonction de f_{pert}. À l'exception de quelques pics dus à l'inversion de phase décrite ci-devant, la plage de verrouillage est aussi confirmée avec cette méthode puisque σ_{Δ_φ} est égale à 0 ° entre 288 MHz et 296,5 MHz, et 104 ° sinon. De la même manière que précédemment, la ligne horizontale représente la valeur de σ_{Δ_φ} pour deux signaux indépendants (104 °). Les 4 preuves de verrouillage nous permettent d'affirmer que, pour la cellule RO avec $N = 3$ inverseurs implantée sur un FPGA Xilinx Spartan 6, la plage de verrouillage est de 8,5 MHz.

Puisque toutes les méthodes utilisées ici donnent la même plage de verrouillage, dans le reste du chapitre, seule $\overline{\Delta f}$ sera représentée. Cependant, toutes ces preuves de verrouillage sont utiles et peuvent être utilisées par le concepteur en fonction des moyens de mesure dont il dispose ou de ce qu'il souhaite mesurer. De plus, même si elles ne sont plus représentées sur les figures dans le reste du manuscrit, les 3 autres preuves de verrouillage ont été mesurées pour toutes nos expériences, nous permettant ainsi de valider la plage de verrouillage obtenue avec la mesure de $\overline{\Delta f}$.

4.3.2 Verrouillage des cellules RO en fonction de N et de la famille de FPGA utilisée

Dans cette partie, l'étude de la section précédente est étendue sur des cellules RO avec un nombre d'inverseurs N différents et sur deux autres familles de FPGA : les FPGA Cyclone V de Intel et les FPGA SmartFusion 2 de Microsemi. Puisque la technologie et la structure interne de ces deux FPGA diffèrent de celle des FPGA Spartan 6 de Xilinx, le même nombre d'inverseurs N donnera une fréquence d'oscillation du signal de sortie de la cellule (f_{osc}) différente d'une famille à l'autre. Nous avons sélectionné N de manière à obtenir des f_{osc} comparables pour chaque famille. Le tableau 4.1 donne les résultats de cette étude. Il représente N, la fréquence f_{osc} naturelle de la cellule RO et la plage de verrouillage obtenue pour toutes les cellules étudiées sur chaque famille de FPGA. Ces résultats montrent clairement que plus N est petit (ou plus f_{osc} est grand), plus la plage de verrouillage sera importante.

Famille de FPGA	***N***	**f_{osc} naturelle** (MHz)	**Plage de verrouillage** (MHz)
Xilinx Spartan 6	3	296	8,5
	7	144	2,1
	11	90	1
	15	67	0,6
	23	44	0,3
	31	33	0,2
Intel Cyclone V	7	409	5,8
	11	222	1
	15	144	0,45
	19	115	0,35
	21	106	0,3
	31	71	0,2
Microsemi SmartFusion 2	5	404	4,4
	7	266	3,3
	11	178	2,4
	15	123	1,6
	23	84	1,2
	31	63	0,9

TABLE 4.1: Étude du verrouillage sur les cellules RO en fonction de N et de la famille de FPGA

Dans le but d'évaluer l'impact de f_{osc} sur la plage de verrouillage, une cellule RO avec $N = 5$ inverseurs a été implantée sur un FPGA Spartan 6 de Xilinx avec 5 placements et routages différents afin de faire varier f_{osc}. Le placement et routage donnant f_{osc} la plus élevée est réalisé avec tous les inverseurs dans le même bloc logique configurable — en anglais configurable logic block (CLB). Le placement et routage donnant f_{osc} la plus faible est réalisé avec un inverseur par CLB. La figure 4.4 représente la plage de verrouillage de chacune de ces cellules RO en fonction de f_{osc}. Au vu du tableau 4.1 et de la figure 4.4, il est clair que f_{osc} est le principal paramètre influençant la plage de verrouillage. Puisque cette étude est réalisée sur des circuits numériques,

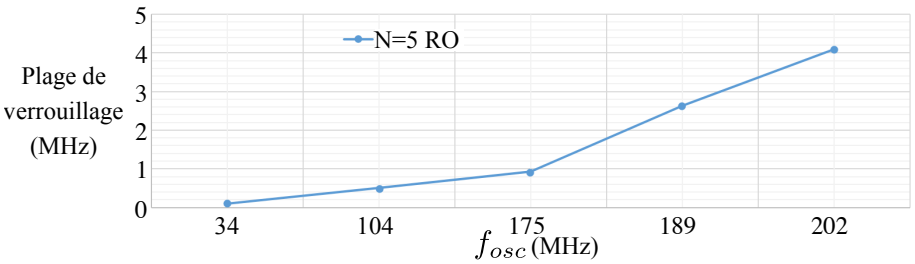


FIGURE 4.4: Plage de verrouillage en fonction de f_{osc} sur un FPGA Spartan 6 de Xilinx

la tension du signal de sortie de la cellule RO et la tension du signal perturbateur sont égales ($V_{pert} = V_{osc}$). Par conséquent, l'équation de Adler (voir sec. 1.4.1.1) établissant les conditions pour que le verrouillage intervienne peut être réarrangée comme suit :

$$\frac{f_{osc}}{2Q} > |f_{pert} - f_{osc}|\,, \tag{4.1}$$

confirmant ainsi les résultats du tableau 4.1 et de la figure 4.4 : f_{osc} est le paramètre principal influençant la plage de verrouillage.

4.3.3 Verrouillage d'une cellule RO sur une harmonique de la fréquence du signal perturbateur

Dans cette partie, la même cellule RO que dans la section 4.3.1 est verrouillée cette fois-ci sur une harmonique de la fréquence d'oscillation du signal perturbateur. Pour rappel, sa fréquence d'oscillation naturelle est de 296 MHz. f_{pert} varie de 145 MHz à 150 MHz par pas de 0,025 MHz de telle sorte que la deuxième harmonique du signal perturbateur est proche de f_{osc}. Les résultats de cette expérience sont représentés sur la figure 4.5. $f_{pert}\times 2$ représente la deuxième harmonique du signal perturbateur. La plage de verrouillage obtenue est de 2 MHz (de 146,5 MHz à 148,5 MHz).

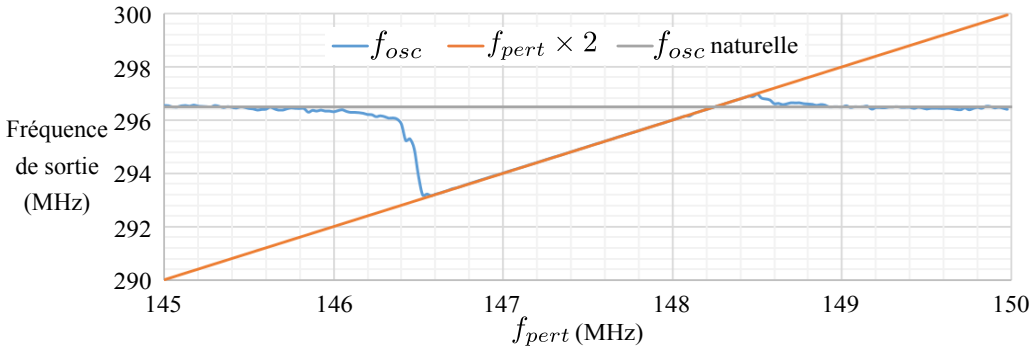


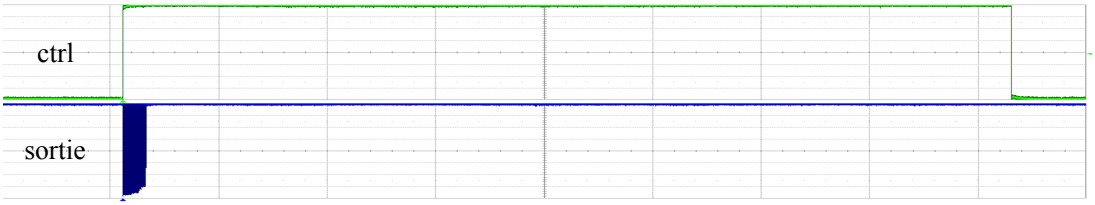
FIGURE 4.5: Évolution de f_{osc} pour une cellule RO avec $N = 3$ inverseurs en fonction de f_{pert} sur un FPGA Spartan 6 de Xilinx

4.4 Verrouillage des cellules TERO

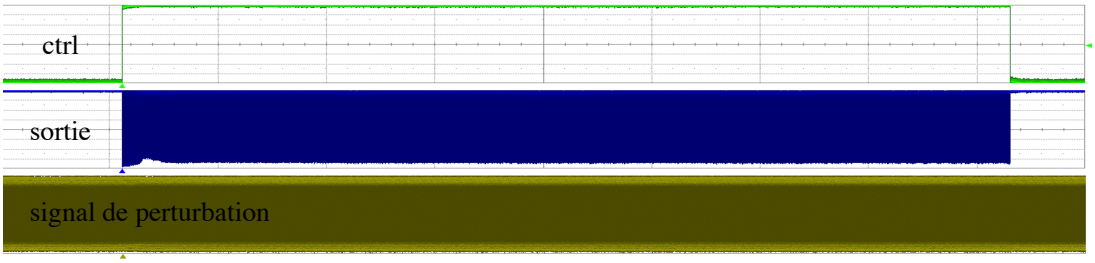
4.4.1 Verrouillage d'une cellule TERO à l'aide d'un signal perturbateur

À présent, la même expérience que précédemment est réalisée sur une cellule TERO composée de $N = 7$ inverseurs par branche et implantée sur un FPGA Spartan 6 de Xilinx.

Sa fréquence d'oscillation naturelle est de 174,5 MHz avec un nombre d'oscillations moyen (N_{osc}) après activation de 228. La figure 4.6a représente le signal de sortie de cette cellule TERO (*sortie*) ainsi que son signal d'activation (*ctrl*) lorsqu'aucun signal perturbateur n'est injecté dans la ligne de délai. La figure 4.6b montre la sortie de la même cellule lorsqu'elle est perturbée. Ces figures montrent clairement la différence de comportement de la cellule TERO avec ou sans perturbations. En effet, sans perturbation la cellule TERO s'arrête d'osciller comme attendu, alors qu'une fois perturbée, elle tend à osciller de manière permanente comme le montre la figure 4.6b.



(a) Signal de sortie d'une cellule TERO en fonctionnement normal



(b) Signal de sortie d'une cellule TERO en présence de perturbations

FIGURE 4.6: Signal de sortie d'une cellule TERO avec et sans perturbation

La figure 4.7 montre l'évolution du rapport cyclique du signal de sortie de la cellule TERO pour 100 activations successives sans perturbation. Comme nous l'avons vu dans la section 1.2.1.2, en fonctionnement normal, le rapport cyclique du signal de sortie évoluera de 50 % en direction de 0 % ou 100 % jusqu'à l'arrêt des oscillations. Ces deux cas de figure (en direction de 0 % ou en direction de 100 %) peuvent être exprimés par le paramètre D en % :

$$D = 50 - |50 - t_h| = 50 - |50 - t_l| \,, \tag{4.2}$$

où t_h et t_l représentent respectivement le temps à l'état logique haut ('1') et le temps à l'état logique bas ('0') du signal de sortie de la TERO.

La figure 4.8 montre l'évolution du paramètre D pour la même TERO perturbée. Le nombre d'oscillations maximales visibles sur la figure ($N_{osc} \sim 450$) est déterminé par la fenêtre d'acquisition de l'oscilloscope, mais il est clair que D reste toujours proche de 50 % et donc que les oscillations continuent au-delà.

Il est aussi important de remarquer que même quand D commence à varier, il peut arriver que la cellule TERO se verrouille à nouveau, ramenant D vers sa valeur initiale.

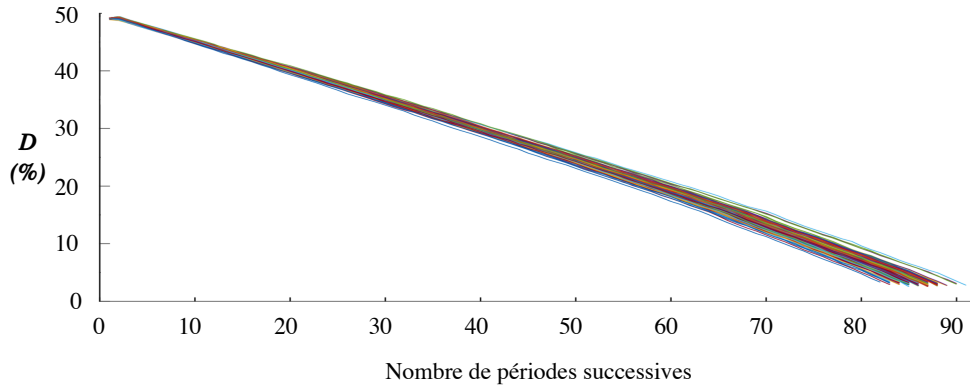


FIGURE 4.7: Évolution du rapport cyclique exprimé à l'aide du paramètre D pour une cellule TERO en fonctionnement normal

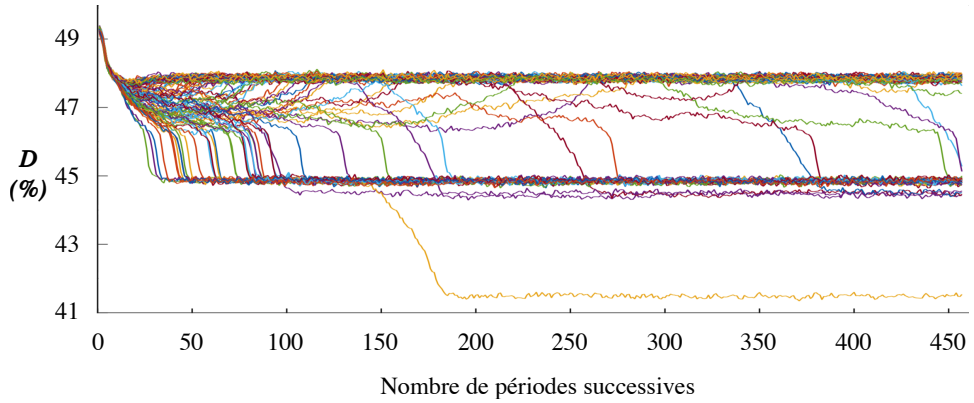


FIGURE 4.8: Évolution du rapport cyclique exprimé à l'aide du paramètre D pour une cellule TERO perturbée

En plus d'observer le nombre d'oscillations (N_{osc}) pour identifier le verrouillage de la cellule TERO, les 4 preuves de verrouillage introduites dans la section 4.2 ont aussi été utilisées confirmant ainsi que les cellules TERO sont, tout comme les cellules RO, sensibles au phénomène de verrouillage. La plage de verrouillage mesurée pour cette cellule TERO avec $N = 7$ inverseurs est de 2,8 MHz.

4.4.2 Auto verrouillage des cellules TERO

Comme nous l'avons vu dans le chapitre 1 (voir sec. 1.2.1.2), deux événements électriques se propagent au sein de la cellule TERO lorsqu'elle oscille. À cause des variations de procédés de fabrication, un événement sera plus rapide que l'autre et les oscillations dureront jusqu'à que les deux événements entre en collision.

Cependant, dans certaines situations (par exemple lorsque les deux branches sont trop proches l'une de l'autre ou même entrelacées), la cellule TERO peut s'auto verrouiller et donc stopper l'évolution du rapport cyclique.

La figure 4.9 montre un exemple de cellule TERO sujette à l'auto verrouillage. Cette cellule TERO implantée sur un FPGA Spartan 6 de Xilinx est de composée $N = 3$ inverseurs par branche. Comme nous pouvons le voir sur la figure, à certains moments, D arrête de varier et reste constant autour de 25 %. Lorsque la cellule ne s'auto verrouille pas, le nombre d'oscillations moyen est d'environ 600 ($N_{osc} \sim 600$). À l'inverse, lorsque la cellule TERO s'auto verrouille, N_{osc} peut devenir extrêmement important.

De plus, il peut arriver que la cellule TERO reste verrouillée pendant un certain temps puis soudainement se déverrouille arrêtant ainsi les oscillations (voir Fig. 4.9 autour de N_{osc} égal 4000 ou N_{osc} égal 11000). Avec le peu d'informations dont nous disposons sur la technologie, la compréhension de ce phénomène est difficile.

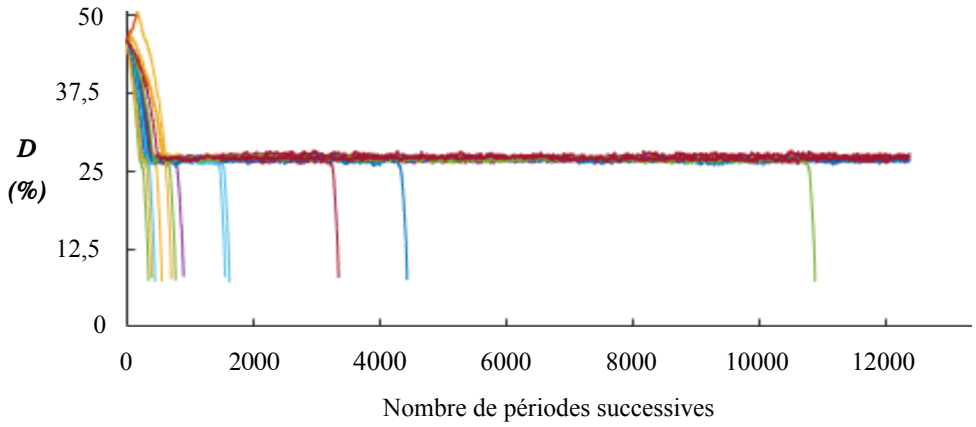


FIGURE 4.9: Évolution du rapport cyclique exprimé à l'aide du paramètre D pour une cellule TERO auto verrouillée

4.4.3 Verrouillage des cellules TERO en fonction de N et de la famille de FPGA utilisée

De la même manière que pour les cellules RO, l'impact du verrouillage en fonction de N est étudié sur trois familles de FPGA différentes. Les résultats affichés dans le tableau 4.2 montrent,

à l'image des résultats de la section 4.3.2 pour les RO, que f_{osc}, qui est déterminée par N, a un impact majeur sur le phénomène de verrouillage.

Famille de FPGA	N	f_{osc} **naturelle** (MHz)	**Plage de verrouillage** (MHz)
Xilinx Spartan 6	3	317	6,4
	15	69	0,5
	31	33	0,1
Intel Cyclone V	3	466	8
	5	258	3,4
	7	188	1,2
Microsemi SmartFusion 2	5	293	3,2
	15	117	1,5
	31	65	0,4

TABLE 4.2: Étude du verrouillage sur les cellules TERO en fonction de N et de la famille de FPGA

Cependant, ce qui est spécifique aux cellules TERO est que le nombre d'oscillations (N_{osc}) a aussi une influence sur la plage de verrouillage. La figure 4.10 montre la plage de verrouillage obtenue pour trois cellules TERO, toutes trois composées de $N = 3$ inverseurs par branche et implantées sur un FPGA Spartan 6 de Xilinx. Les trois cellules TERO ont une fréquence d'oscillations naturelle autour de 350MHz , mais des N_{osc} différents. La figure montre que plus N_{osc} est grand, plus la plage de verrouillage est grande.

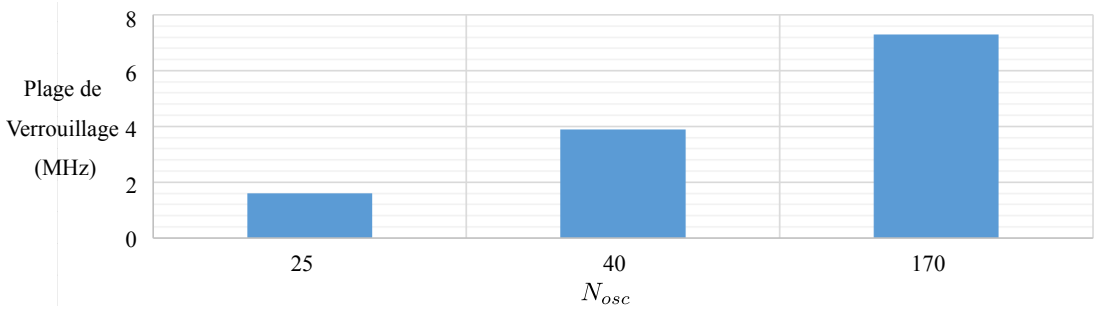


FIGURE 4.10: Plage de verrouillage pour différent N_{osc} pour des cellules TERO composées de $N = 3$ inverseurs sur un FPGA Spartan 6 de Xilinx

4.4.4 Verrouillage d'une cellule TERO sur une harmonique de la fréquence du signal perturbateur

Contrairement à ce que nous avons fait dans la section 4.3.3 pour la cellule RO, où un signal perturbateur à une fréquence environ deux fois inférieure à la fréquence naturelle de la cellule RO était injecté dans la ligne de délai, nous utilisons un signal perturbateur à une fréquence environ deux fois supérieure. Par conséquent, cette fois-ci, la cellule TERO se verrouille au niveau du deuxième harmonique du signal. L'objectif ici est de montrer qu'il est aussi possible de verrouiller

la cellule TERO avec un signal perturbateur ayant une fréquence plus importante. Cette étude complète l'expérience réalisée dans la section 4.3.3. De la même manière que pour la cellule RO, il est aussi possible de verrouiller une cellule TERO avec un signal perturbateur à une fréquence plus faible. La TERO étudié est composée de $N = 5$ inverseurs par branche et implantée sur un FPGA Spartan 6 de Xilinx. Sa fréquence d'oscillation naturelle est de 196 MHz avec un nombre d'oscillations moyen (N_{osc}) de 85. f_{pert} varie de 384 MHz à 398 MHz par pas de 0,025 MHz. Les résultats de cette expérience sont représentés sur la figure 4.11. $f_{pert}/2$ représente la moitié de la fréquence d'oscillation du signal perturbateur. La plage de verrouillage obtenue est de 4 MHz.

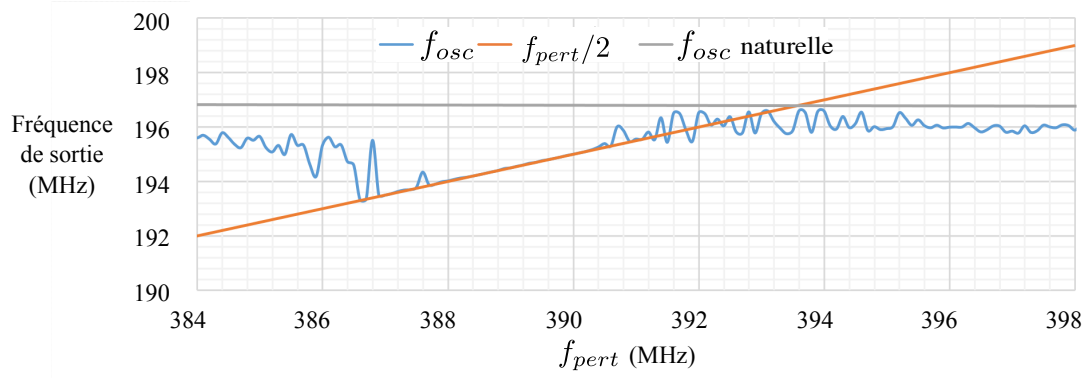


FIGURE 4.11: Évolution de f_{osc} pour une cellule TERO avec $N = 5$ inverseurs en fonction de f_{pert} sur un FPGA Spartan 6 de Xilinx

4.5 Verrouillage des cellules STR

4.5.1 Verrouillage d'une cellule STR à l'aide d'un signal perturbateur

Comme nous l'avons vu dans la section 1.2.1.3, une cellule STR peut opérer selon deux modes de fonctionnement : le mode rafale et le mode régulièrement espacé. En mode régulièrement espacé, la fréquence moyenne de la cellule (f_{osc}) est stable. En mode rafale, plusieurs périodes courtes successives sont suivies par un intervalle sans événements. Pour plus d'information concernant les modes de fonctionnement d'une cellule STR, le lecteur peut se référer à [99]. Pour nos expériences, les cellules STR sont toutes chargées avec $N/2$ événements, garantissant ainsi que les cellules fonctionnent en mode régulièrement espacé.

Ici, le comportement d'une cellule STR, composée de 8 portes de Muller ($N = 8$) et implantée sur un FPGA Xilinx Spartan 6, est étudié. Sans perturbation, la fréquence d'oscillation naturelle de la cellule STR est de 336 MHz. Ensuite, un signal perturbateur à une fréquence f_{pert} variant entre 323 MHz et 344 MHz par pas de 0,025 MHz est injecté dans la ligne de délai. Comme nous pouvons le voir sur la figure 4.12, la cellule STR, à l'instar des autres types de cellules oscillantes, subit le phénomène de verrouillage. La plage de verrouillage obtenue est de 12,5 MHz.

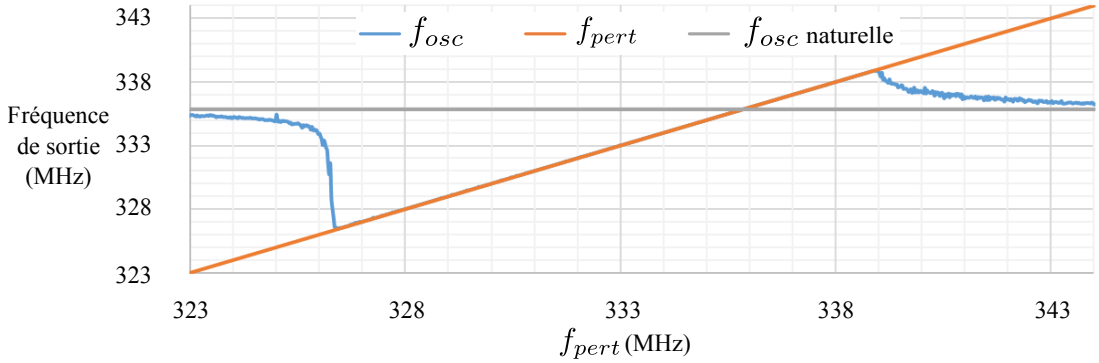


FIGURE 4.12: Évolution de f_{osc} pour une cellule STR avec $N = 8$ éléments en fonction de f_{pert} sur un FPGA Spartan 6 de Xilinx

4.5.2 Verrouillage des cellules STR en fonction de N et de la famille de FPGA utilisée

Comme pour les cellules RO et les cellules TERO, l'impact du phénomène de verrouillage sur des cellules STR en fonction de N et de la famille de FPGA est étudié. Les résultats sont visibles dans le tableau 4.3. Une fois de plus, f_{osc} est le paramètre majeur influençant la plage de verrouillage obtenue.

Famille de FPGA	***N***	**f_{osc} naturelle** (MHz)	**Plage de verrouillage** (MHz)
Xilinx Spartan 6	8	334	12,5
	16	334	12,4
	32	299	10,5
	64	263	4,8
Intel Cyclone V	8	420	2,3
	16	396	2,1
	32	383	0,7
	64	368	0,5
Microsemi Smartfusion 2	8	442	1,4
	16	392	1,4
	32	369	1
	64	255	1,1

TABLE 4.3: Étude du verrouillage sur les cellules STR en fonction de N et de la famille de FPGA

4.5.3 Verrouillage d'une cellule STR sur une harmonique de la fréquence du signal perturbateur

Dans cette dernière expérience, une cellule STR composée de $N = 8$ portes de Muller et implantée sur un FPGA Spartan 6 de Xilinx est verrouillée sur la deuxième harmonique de la fréquence du signal perturbateur.

La fréquence d'oscillation naturelle de la cellule STR est de 334 MHz. f_{pert} varie de 164 MHz à 169 MHz par pas de 0,025 MHz. La figure 4.13 montre l'évolution de f_{osc} en fonction de

f_{pert}. $f_{pert} \times 2$ représente la seconde harmonique du signal perturbateur. La plage de verrouillage obtenue est de 2,1 MHz.

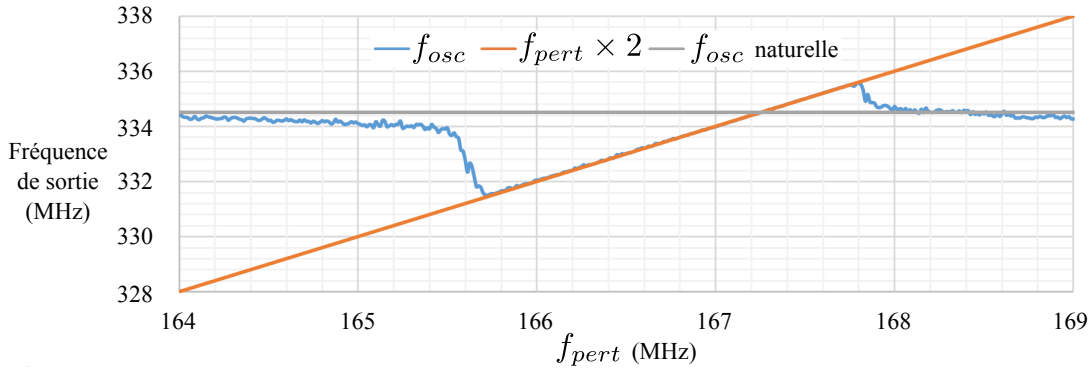


FIGURE 4.13: Évolution de f_{osc} pour une cellule STR avec $N = 8$ éléments en fonction de f_{pert} sur un FPGA Spartan 6 de Xilinx

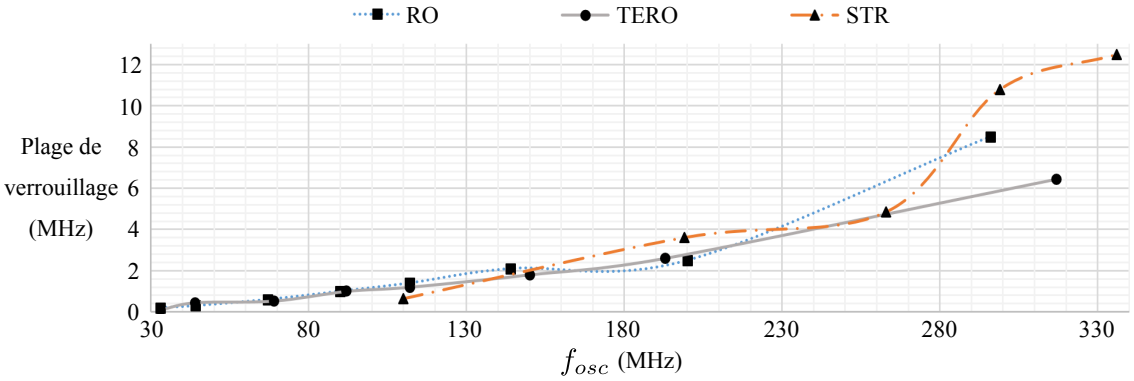
4.6 Résumé et comparaison des résultats

Dans les sections précédentes, nous avons évalué la plage de verrouillage sur des cellules RO, TERO et STR. Cette section résume tous ces résultats et dresse une comparaison.

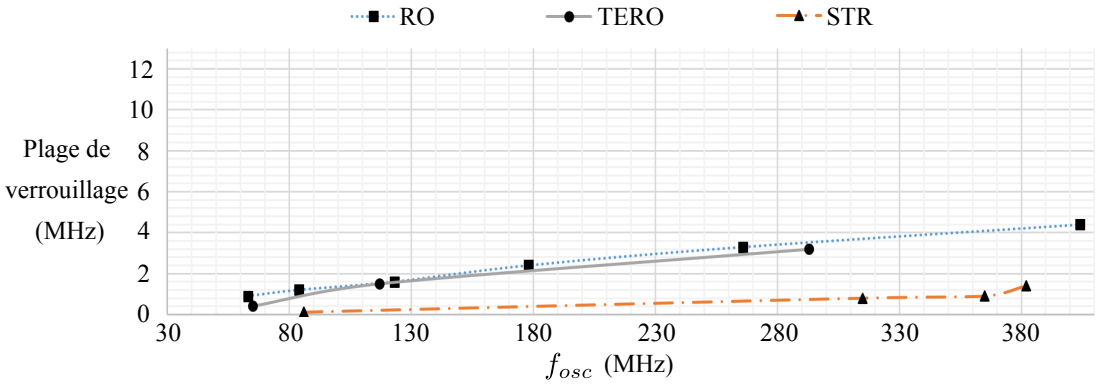
4.6.1 Comparaison des cellules étudiées

Sur la figure 4.14 sont représentées les évolutions des plages de verrouillage des cellules RO, TERO et STR en fonction de f_{osc} sur les FPGA Spartan 6 de Xilinx et SmartFusion 2 de Microsemi. Les résultats suivent la même tendance pour tous les types de cellules. Au vu de la similarité de ces résultats, nous pouvons conclure qu'indépendamment du type de cellule oscillante, toutes les cellules étudiées sont sensibles au phénomène de verrouillage. De plus, plus les cellules oscillent rapidement plus elles sont sensibles au verrouillage. Il est aussi important de noter que les cellules peuvent aussi se verrouiller sur les harmoniques du signal perturbateur.

Nous pouvons aussi constater que les STR tendent à être les plus affectés par le phénomène de verrouillage sur les FPGA Spartan 6 de Xilinx (voir fig. 4.14a) alors que c'est l'inverse sur les FPGA SmartFusion2 de Microsemi (voir fig. 4.14b). Cela démontre que le type de FPGA et sa technologie ont une influence sur l'impact du verrouillage, rendant ainsi difficile la comparaison des cellules sur FPGA.



(a) Évolution des plages de verrouillage des cellules RO, TERO et STR en fonction de f_{osc} sur FPGA Spartan 6 de Xilinx



(b) Évolution des plages de verrouillage des cellules RO, TERO et STR en fonction de f_{osc} sur FPGA SmartFusion2 de Microsemi

FIGURE 4.14: Comparaison des plages de verrouillage des cellules RO, TERO et STR implantées sur les FPGA Spartan 6 et SmartFusion2

4.6.2 Comparaison du phénomène de verrouillage dans les différentes familles de FPGA

En suivant la même approche, la figure 4.15 montre les résultats de verrouillage des cellules RO sur les trois familles de FPGA utilisées. La première conclusion que nous pouvons tirer est que, quel que soit le circuit utilisé, il est possible de verrouiller une cellule RO. Deuxièmement, les résultats montrent que les FPGA Spartan 6 de Xilinx semblent être les plus sensibles au phénomène de verrouillage. Il est probable que cette sensibilité vienne des ressources de routage disponibles et de la manière dont les cellules oscillantes sont routées. Cependant, comme sur FPGA il n'est pas possible de contrôler précisément le routage, nous ne pouvons pas garantir que reproduire cette expérience sur une autre zone du FPGA donne les mêmes résultats. Bien qu'il ne soit pas possible de contrôler le routage sur FPGA, il est possible de l'influencer en plaçant les éléments différemment. Jusqu'à présent, toutes nos expériences étaient réalisées avec

la ligne de délai entrelacée avec la cellule étudiée (voir fig. 4.2). Dans la section suivante, nous analysons l'importance du placement et du routage sur le phénomène de verrouillage.

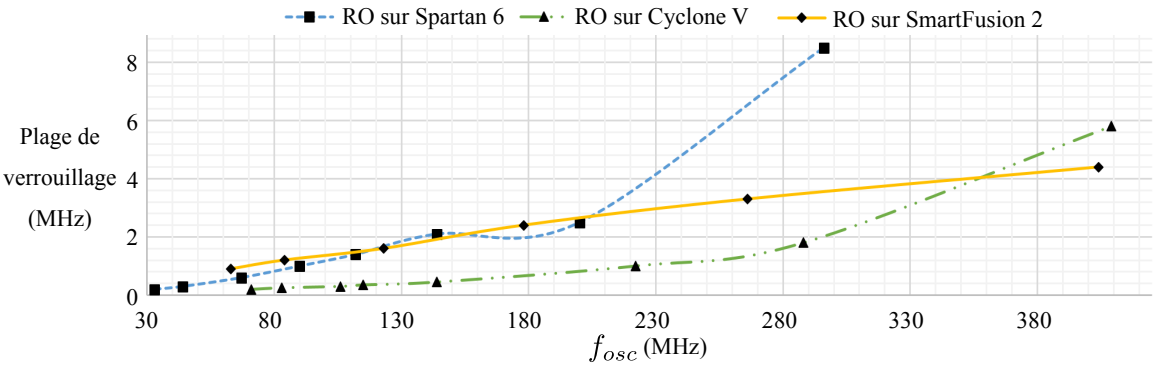


FIGURE 4.15: Évolution des plages de verrouillage sur des cellules RO en fonction de f_{osc} sur des FPGA Spartan 6, Cyclone V et SmartFusion2

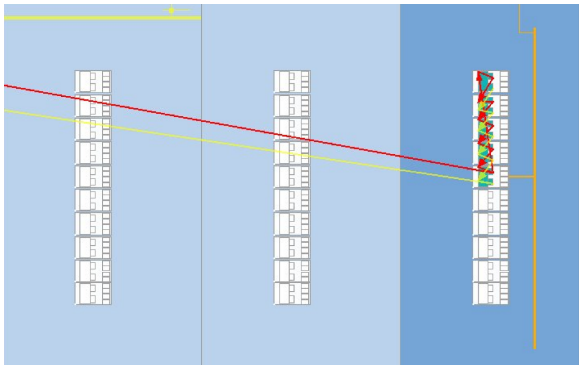
4.7 L'importance du placement et du routage

L'importance du placement et du routage (P/R) est évaluée sur une cellule RO avec $N = 7$ inverseurs implantée sur un FPGA Cyclone V de Intel. Le placement de la cellule RO est le même pour toutes les expériences suivantes. Seul le placement de la ligne de délai est modifié. Au total, 5 placements différents sont testés. La fréquence d'oscillation naturelle de la cellule RO est de 410 MHz.

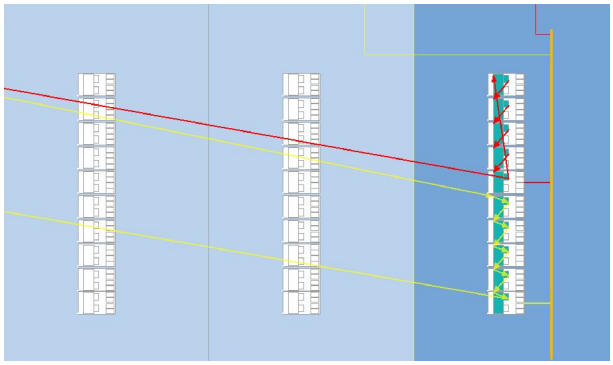
Une impression-écran de l'outil ChipPlanner de Intel pour les 5 placements et routages est représentée sur la figure 4.16. Les éléments de la cellule RO sont représentés en rouge alors que ceux de la ligne de délai sont en jaune.

Pour le premier placement et routage (P/R1 sur la figure 4.16a), les éléments de la cellule RO et ceux de la ligne de délai sont placés, comme pour toutes les expériences présentées ci-devant (voir Fig. 4.2), entrelacés dans le même module logique — en anglais adaptive logic module (ALM). Dans le deuxième placement et routage (P/R2 sur la figure 4.16b), les éléments de la cellule RO et ceux de la ligne de délai ne sont pas dans le même ALM, mais restent toutefois dans le même bloc logique - en anglais logic array block (LAB). Pour la troisième configuration (P/R3 sur la figure 4.16c), la ligne de délai n'est composée que d'un seul inverseur placé dans le même LAB que la cellule RO. Dans le quatrième placement et routage (P/R4 sur la figure 4.16d), la ligne de délai est placée dans un LAB voisin à celui où est placée la cellule étudiée. Enfin, pour le cinquième placement et routage (P/R5 sur la figure 4.16e), la ligne de délai est placée dans l'angle opposé du FPGA.

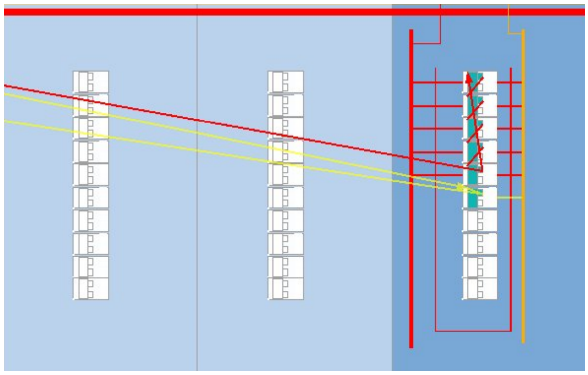
La figure 4.17 montre la plage de verrouillage de la cellule RO pour les 5 différents placements de la ligne de délai sur Cyclone V. Les résultats montrent clairement que, plus la ligne de délai



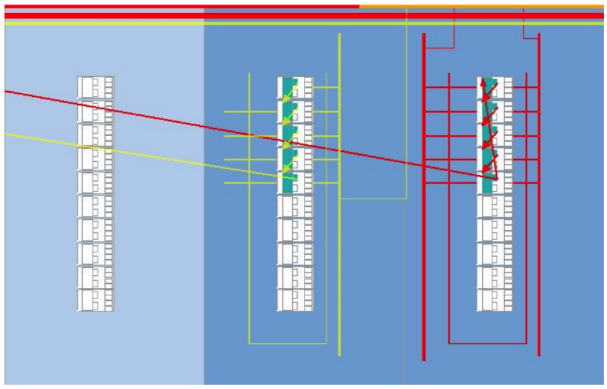
(a) P/R1 : Même ALM



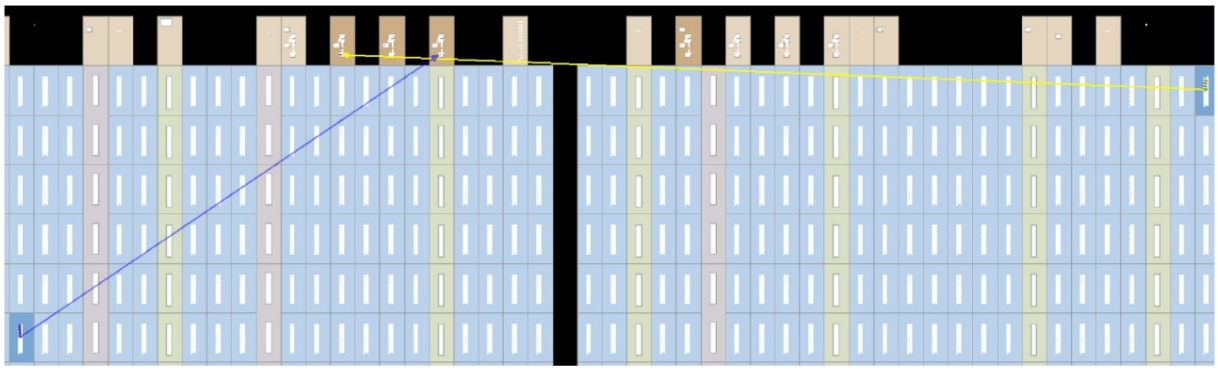
(b) P/R2 : Même LAB



(c) P/R3 : 1 inverseur



(d) P/R4 : LAB voisins



(e) P/R5 : Angles opposés du FPGA

FIGURE 4.16: Différents placements et routages d'une cellule RO avec $N = 7$ inverseurs et d'une ligne de délai sur un FPGA Cyclone V de Intel

sera proche de la cellule, plus la plage de verrouillage sera grande.

En effet, pour le premier placement et routage (P/R1), les éléments sont placés dans le même ALM et la plage de verrouillage obtenue est de 4,4 MHz. Dans les placements et routages 2 et 3 (P/R2 et P/R3), les éléments sont dans le même LAB, mais pas dans le même ALM et la plage de verrouillage obtenue est plus faible de 3 MHz. Elle devient encore plus faible pour le

placement et routage 4 (P/R4), quand les éléments sont dans des LAB voisins. Enfin, dans le dernier cas (P/R5), lorsque les éléments sont éloignés sur le FPGA, le verrouillage n'intervient plus.

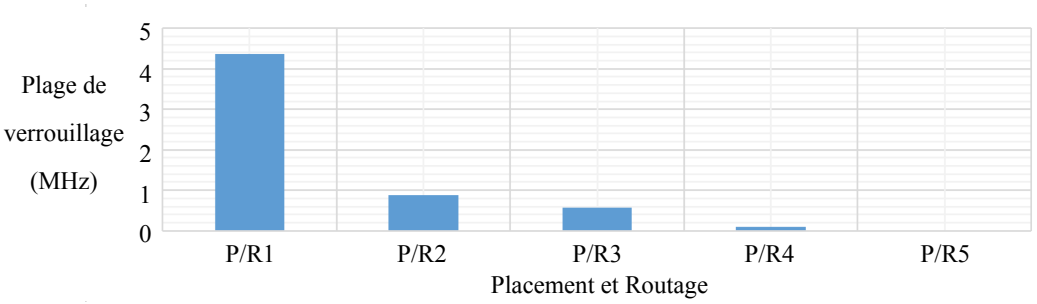


FIGURE 4.17: Plage de verrouillage d'une cellule RO composée de $N = 7$ inverseurs sur un FPGA Cyclone V de Intel en fonction du placement et routage de la ligne de délai

Ces résultats montrent une corrélation claire entre la proximité des éléments et la susceptibilité à se verrouiller des cellules oscillantes. Lorsqu'elles sont proches l'une de l'autre, les cellules utilisent plus de routes en commun et donc les interférences électromagnétiques sont plus importantes. Jusqu'à présent, seule la sensibilité d'une cellule oscillante seule au phénomène de verrouillage a été étudiée. Dans la section suivante, nous réalisons des expériences sur des systèmes complets à base de cellules oscillantes afin de montrer les conséquences du verrouillage sur un système.

4.8 Expérimentation sur différents cas d'applications à base de cellules oscillantes

Dans cette partie, deux cas d'applications à base de cellules oscillantes sont étudiés :

— Un TERO-TRNG fonctionnant dans un environnement qui inclut un signal perturbateur ;

— Un système utilisant deux cellules RO qui oscillent à des fréquences proches dans un environnement sujet aux variations de tensions.

4.8.1 Étude de l'impact du verrouillage sur un TERO-TRNG

Le TERO-TRNG étudié est celui présenté dans le chapitre 1 (voir sec. 1.2.2.2). L'aléa extrait de ce TRNG vient du nombre d'oscillations d'une cellule TERO. Dans cette partie nous allons voir les conséquences sur la sortie du TERO-TRNG lorsque la cellule TERO qui le compose est verrouillée avec un signal perturbateur.

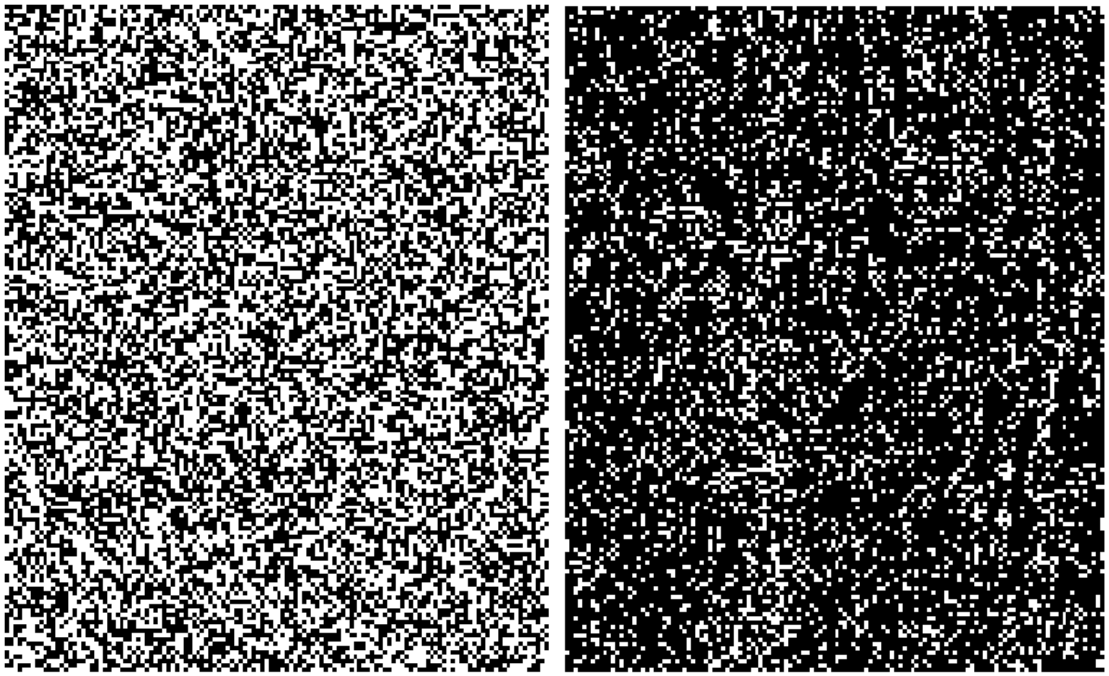
Comme pour les expériences précédentes, les éléments de la ligne de délai dans laquelle le signal perturbateur est injecté, sont entrelacés avec les inverseurs de la cellule TERO. Deux flux d'échantillons de 1 million de bits sont captés du TRNG :

— Un lorsque le TERO-TRNG est fonctionnement normal ;

— Un lorsque la cellule TERO du TRNG est verrouillée avec le signal perturbateur.

La suite de bits générée pour chacun des cas est évaluée à l'aide de tests statistiques simples et rapides de la suite standard FIPS 140-1 [8]. Alors qu'en fonctionnement normal la suite de bits passe les tests avec succès, lorsque la cellule TERO est verrouillée avec le signal perturbateur, la suite de bits passe les tests avec échec. Pour rappel la suite de tests FIPS 140-1 n'est plus appropriée pour évaluer les TRNG, car elle n'est pas assez complète. Cela signifie que si les données récupérées lorsque le TRNG est perturbé ne passent pas avec succès ces tests, elles ne passeront jamais avec succès des tests plus poussés comme ceux de la norme NIST 800-22 ou AIS31.

L'impact du verrouillage sur le TERO-TRNG est visible sur la figure 4.18a et 4.18b. Ces figures représentent les bits de sortie du TRNG par paquet de 128-bit en fonction du temps. Les pixels noirs représentent un zéro logique ('0') et les pixels blancs représentent un un logique ('1'). Sans perturbation (fig. 4.18a), le biais obtenu est de 0,502, reflétant un nombre de '1' logique et de '0' logique équivalents. La sortie du générateur ne semble pas biaisée. À l'inverse, quand la cellule TERO est verrouillée avec le signal perturbateur et donc oscille de manière indéfinie, le biais obtenu est de 0,413, reflétant nombre de '0' supérieur au nombre de '1'. Un tel biais sur la sortie du générateur réduit de manière considérable le taux d'entropie en sortie du TRNG.



(a) Sans perturbations (b) Avec perturbations

FIGURE 4.18: Paquets de 128 bits de sortie du TERO-TRNG en fonction du temps avec et sans perturbations

4.8.2 Effet du verrouillage mutuel entre deux signaux d'horloge générés à l'aide de cellules RO similaires

Pour ce second cas d'application, au lieu de perturber la cellule RO avec un signal perturbateur, deux cellules RO similaires sont implantées sur un FPGA Cyclone V de Intel. Ces cellules RO peuvent être utilisées, par exemple, comme sources de bruit pour le MURO-TRNG (voir sec. 1.2.2.1) ou une RO-PUF (voir sec. 1.2.3.1).

De par leurs structures similaires, la fréquence d'oscillation de ces deux cellules RO est proche.

En condition normale de fonctionnement, les deux cellules RO oscillent librement et aucun verrouillage ne les affecte. Cependant, les cellules oscillantes sont sensibles aux variations de tension. Lorsque la tension d'alimentation du circuit va augmenter, la fréquence d'oscillation (f_{osc}) de la cellule augmentera aussi. Cette augmentation de fréquence due à l'augmentation de la tension d'alimentation sera propre à chaque cellule. Pour cette expérience, nous avons fait varier la tension d'alimentation du FPGA de 960 mV à 1260 mV.

La figure 4.19 représente l'évolution de fréquence des cellules RO (f_{osc_1} et f_{osc_2}) en fonction de la tension d'alimentation. Les deux cellules RO se verrouillent l'une à l'autre à partir d'une tension d'alimentation de 1100 mV et le restent jusqu'à 1140 mV. La déviation standard de la phase de la première cellule RO ($\sigma_{\Delta_{\varphi osc1}}$) est aussi représentée sur la figure et montre clairement la plage durant laquelle les cellules sont verrouillées.

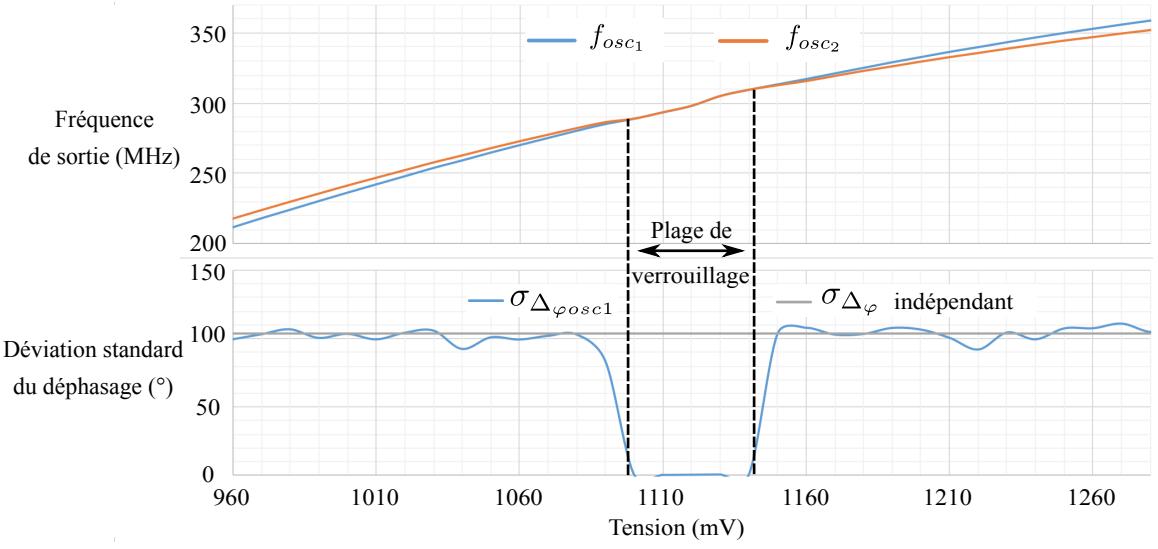


FIGURE 4.19: Évolution de f_{osc_1}, f_{osc_2} et $\sigma_{\Delta_{\varphi osc1}}$ de deux cellules RO en fonction de la tension d'alimentation sur un FPGA Cyclone V de Intel

Cette expérience reflète le danger du verrouillage dans les TRNG et les PUF. En effet, sans signal perturbateur extérieur, les cellules oscillantes peuvent se verrouiller entre elles. De plus, il n'est pas suffisant de vérifier l'absence de verrouillage dans les conditions nominales de fonc-

tionnement. Il faut le vérifier sur toute la plage d'utilisation du système en température et en tension.

4.9 Conclusion

Dans ce chapitre, nous avons réalisé une analyse complète du phénomène de verrouillage sur les cellules oscillantes. Cela nous a permis de prendre conscience que, contrairement à ce que nous pourrions penser en regardant la littérature existante, les cellules TERO et les cellules STR sont tout aussi sensibles au phénomène de verrouillage que les cellules RO.

En montrant la présence de verrouillage sur trois familles de FPGA venant de trois fabricants différents, nous avons montré que le verrouillage intervient indépendamment du circuit utilisé.

De plus, les résultats obtenus confirment les travaux théoriques menés par Tucker [78], Adler [79] et Huntoon *et al.* [80] (voir sec. 1.4.1.1). En effet, comme l'indique l'équation de Adler et puisque notre étude est réalisée sur des circuits numériques ($V_{pert} = V_{osc}$) : plus la fréquence d'oscillation (f_{osc}) sera importante, plus la plage de verrouillage obtenue sera grande. Ce comportement a été vérifié pour toutes nos expériences.

Il est aussi important de souligner que la proximité des cellules oscillantes et leur routage influent de manière significative sur la plage de verrouillage.

Malheureusement, il n'est pas possible de contrôler finement le routage sur des circuits tels que les FPGA. Cela nous empêche donc de quantifier précisément l'impact du routage sur la plage de verrouillage. C'est pourquoi nous pensons qu'une étude de l'impact du routage sur le verrouillage, réalisée sur un circuit intégré dédié où le routage peut être contrôlé, serait un très bon complément à l'étude que nous venons de mener.

Enfin, nous avons montré les conséquences désastreuses du verrouillage sur des systèmes à base de cellules oscillantes utilisés dans le cadre de générateur d'aléa.

Les résultats présentés dans ce chapitre, et les sources VHDL disponibles sur Git, aideront les concepteurs à mieux anticiper le phénomène de verrouillage dans leurs travaux futurs. Nous pensons que la plupart de nos conclusions sont applicables pour des implantations de cellules oscillantes sur un ASIC.

Maintenant que la sensibilité au verouillage de l'ensemble des cellules oscillantes dans les circuits intégrés numériques a été démontrée dans ce chapitre, un dernier point mérite d'être étudié : la sensibilité des cellules TERO à l'analyse du rayonnement EM. C'est ce que nous allons aborder dans le dernier chapitre.

Chapitre 5

Analyse électromagnétique des cellules TERO

Dans la partie 1.4.2 du chapitre 1, nous avons vu les différents travaux existants dans la littérature concernant l'analyse du rayonnement EM des TRNG et des PUF. L'étude de l'état de l'art nous a appris que l'analyse du rayonnement EM permet de retrouver la fréquence d'oscillation d'une cellule RO et ainsi, dans le cadre de la RO-PUF de la cloner [86] et, dans la cadre d'un TRNG à base de RO de trouver sa position sur le circuit intégré pour mieux le perturber ensuite [89]. Nous constatons que tous ces travaux portent uniquement sur l'analyse du rayonnement EM des cellules RO.

Lors de la conception des cellules TERO et de la PUF liée, nous avons pensé que l'utilisation du seul nombre d'oscillations des cellules TERO permettait de rendre la TERO-PUF robuste contre l'analyse spectrale du rayonnement EM telle qu'elle est réalisée sur la RO-PUF. Effectivement, l'information concernant la fréquence d'oscillation des cellules TERO (lors du régime transitoire oscillant) n'est pas utile pour la réponse de la TERO-PUF. Pour autant, cela ne constitue pas une protection suffisante si l'analyse spectrale ne porte pas uniquement sur les valeurs de fréquence mais aussi sur l'amplitude des raies spectrales et leur dynamique temporelle.

C'est ce que nous nous proposons d'étudier dans ce dernier chapitre. Nous débuterons par une étude théorique du rayonnement EM des cellules TERO basée sur un modèle spectral. Ensuite, nous réaliserons, pas-à-pas, les analyses EM nous permettant d'obtenir des informations à partir d'une cellule TERO pour enfin arriver à une TERO-PUF complète.

Toutes les sources VHDL sont disponibles sur Git pour assurer la répétabilité de nos expériences.

. Le code associé à ce chapitre est disponible à l'adresse suivante : https://gitlab.univ-st-etienne.fr/ugo.mureddu/em-analysis-of-transient-effect-ring-oscillator-based-puf.git

5.1 Modèle spectral du signal de sortie d'une cellule TERO

Dans cette première partie, un modèle spectral du signal de sortie d'une cellule TERO est établi. Cela permet d'évaluer de manière théorique le potentiel d'émission électromagnétique du signal de sortie des cellules TERO en fonction du nombre d'oscillations (N_{osc}).

En se basant sur les travaux de modélisation des cellules TERO réalisés dans le chapitre 3, la sortie d'une cellule TERO peut être modélisée à l'aide d'une modulation de largeur d'impulsion — en anglais pulse width modulation (PWM). La modulation de largeur d'impulsion est une technique de modulation qui contrôle la largeur d'une impulsion basée sur un signal d'information.

Dans le cas de la cellule TERO, le rapport cyclique du signal de sortie augmente ou diminue de manière quasi exponentielle (voir fig. 3.15). Par conséquent, le signal de modulation utilisé pour ce modèle ($mod(t)$) est un signal exponentiel défini par l'expression suivante :

$$mod(t) = 1 - e^{-t/\tau} \tag{5.1}$$

où τ représente la constante de temps.

Le signal PWM ($PWM(t)$) est le résultat de la comparaison d'un signal triangulaire périodique ($tri(t)$) et du signal de modulation ($mod(t)$). La figure 5.1 montre le signal $PWM(t)$ généré avec un signal de modulation ($mod(t)$) exponentiel. Ce comportement est modélisé avec Matlab. Le code Matlab est fourni sur Git avec les sources VHDL. Plus τ sera petit, plus le nombre d'oscillations (N_{osc}) sera petit. Ainsi, des cellules TERO produisant différents nombres d'oscillations (N_{osc}) pourront être simulées facilement en faisant varier le paramètre τ.

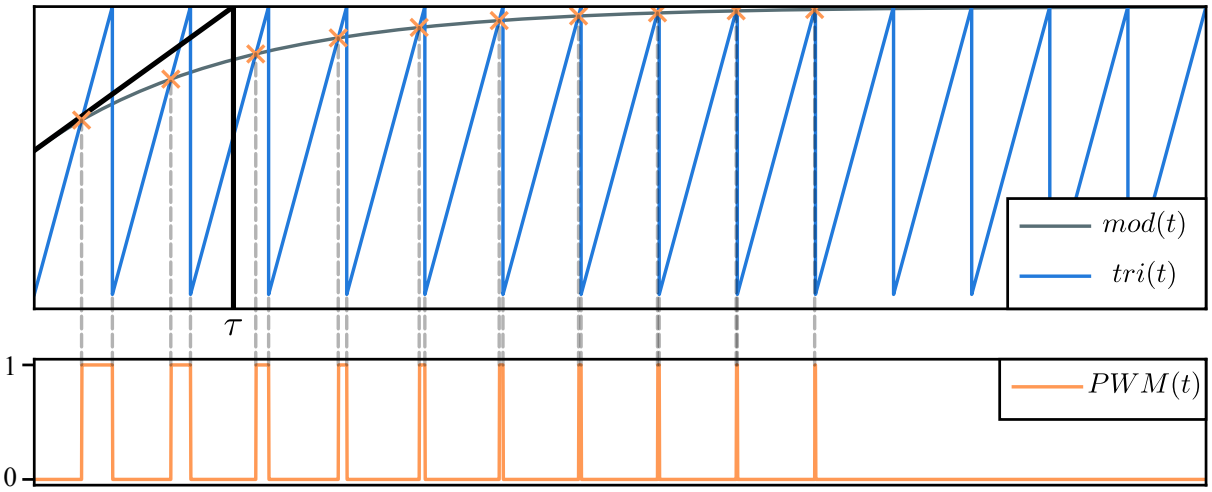


FIGURE 5.1: Modélisation du signal de sortie de la cellule TERO à l'aide d'un signal PWM

La FFT du signal de sortie de la cellule TERO est ensuite calculée pour différentes valeurs de τ (donc de N_{osc}). Les figures 5.2 et 5.3 représentent la FFT du signal de sortie d'une cellule TERO pour $\tau = 0{,}1$ et $\tau = 2$. La FFT du signal de sortie de la cellule TERO est notée $|FFTout(f)|$. Le signal d'activation de la cellule TERO ($ctrl$) est modélisé par un signal carré avec une période de 4 s. Le signal de sortie de la cellule TERO ($sortie$) est modélisé avec $PWM(t)$ et $|FFTout(f)|$ est obtenu en calculant la FFT de $PWM(t)$. La fréquence du signal triangulaire ($tri(t)$) est de 100 Hz donc la fréquence d'oscillation de la cellule TERO (f_{osc}) est de 100 Hz.

Avec $\tau = 0{,}1$ (figure 5.2), respectivement $\tau = 2$ (figure 5.3), $N_{osc} = 9$, respectivement $N_{osc} = 183$. Ces simulations montrent l'impact du nombre d'oscillations (N_{osc}) sur l'amplitude spectral de $|FFTout(f)|$. En effet, plus le nombre d'oscillations (N_{osc}) est grand, plus l'amplitude de $|FFTout(f)|$ sera grande à la fréquence d'oscillation de la cellule TERO (f_{osc}).

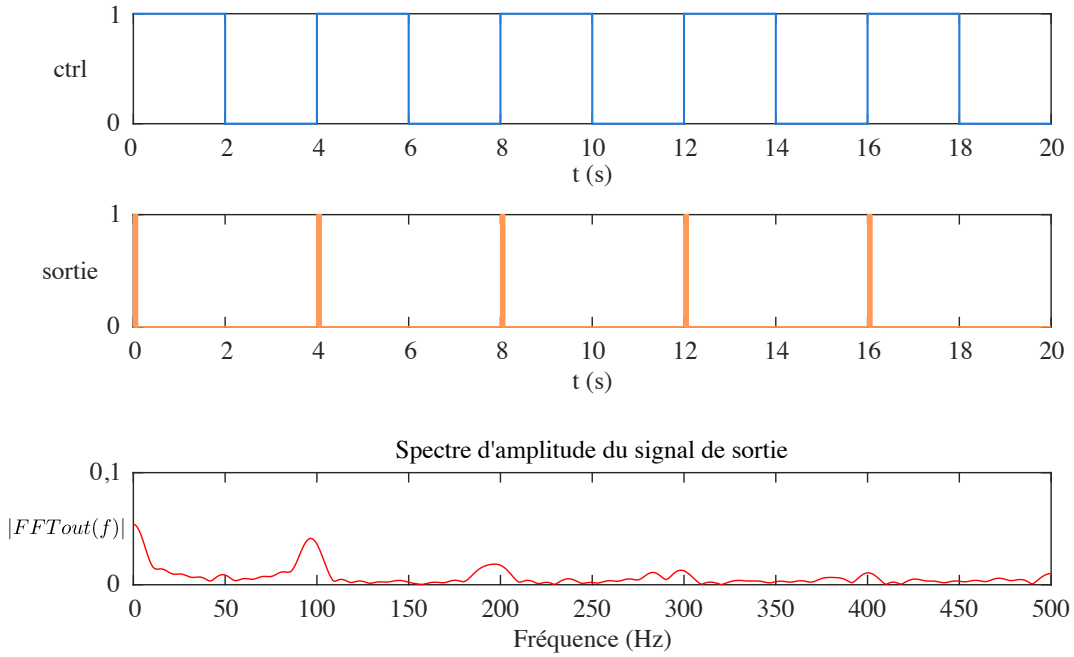


FIGURE 5.2: $|FFTout(f)|$ pour $\tau = 0,1$

En complément, la figure 5.4 montre l'évolution du nombre d'oscillations (N_{osc}) et de l'amplitude de la FFT du signal de sortie de la cellule TERO ($|FFTout(f)|$) à la fréquence d'oscillation (f_{osc}) en fonction de τ. Cette simulation est réalisée pour des valeurs de τ variant de 0,1 à 2,2 par pas de 0,1. Les résultats montrent que le nombre d'oscillations (N_{osc}) croît de manière linéaire avec τ. Pour $|FFTout(f)|$ à f_{osc} les résultats sont légèrement différents. Au début, $|FFTout(f)|$ commence à croitre de manière linéaire avec τ mais ensuite, à partir de $\tau = 1{,}2$, commence à saturer.

Cette simulation nous permet d'anticiper le comportement du rayonnement EM des cellules TERO. Ainsi, nous pouvons nous attendre aux résultats suivants :

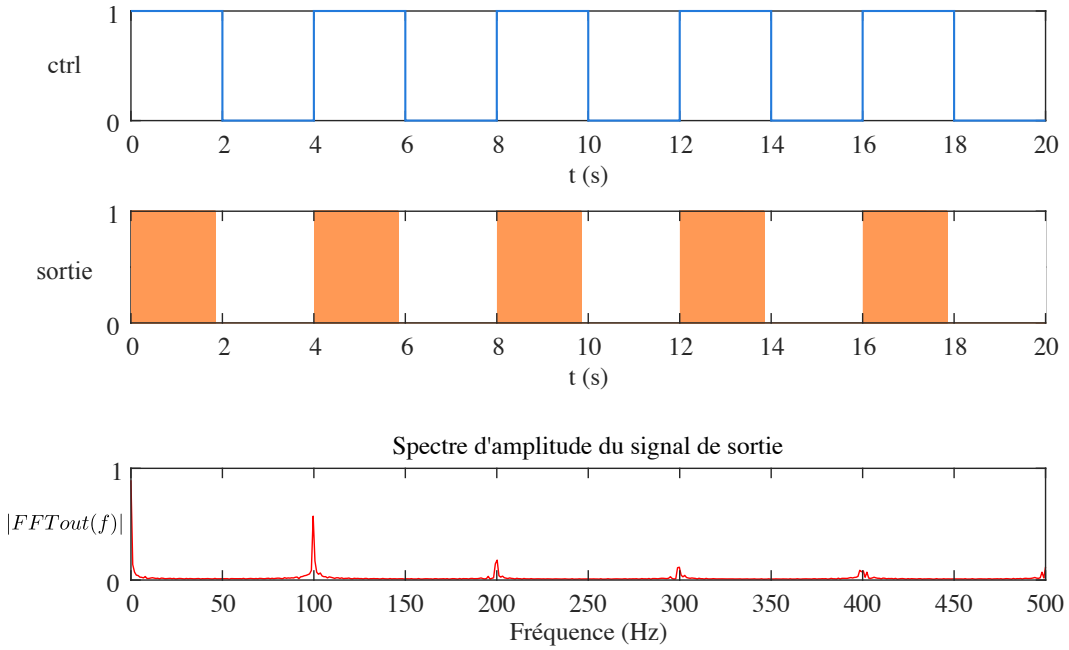


FIGURE 5.3: $|FFTout(f)|$ pour $\tau = 2$

Si le nombre d'oscillation (N_{osc}) est très faible, alors le rayonnement EM de la cellule TERO doit être très faible et il doit probablement être difficile à mesurer en fonction des capacités du banc d'analyse. Ensuite, nous pouvons que l'amplitude du rayonnement EM semble augmenter avec N_{osc} jusqu'à arriver à saturation. Dans ce cas, le rayonnement EM doit certainement être plus aisé à mesurer. Dans la partie suivante, nous tâcherons de vérifier ces résultats théoriques.

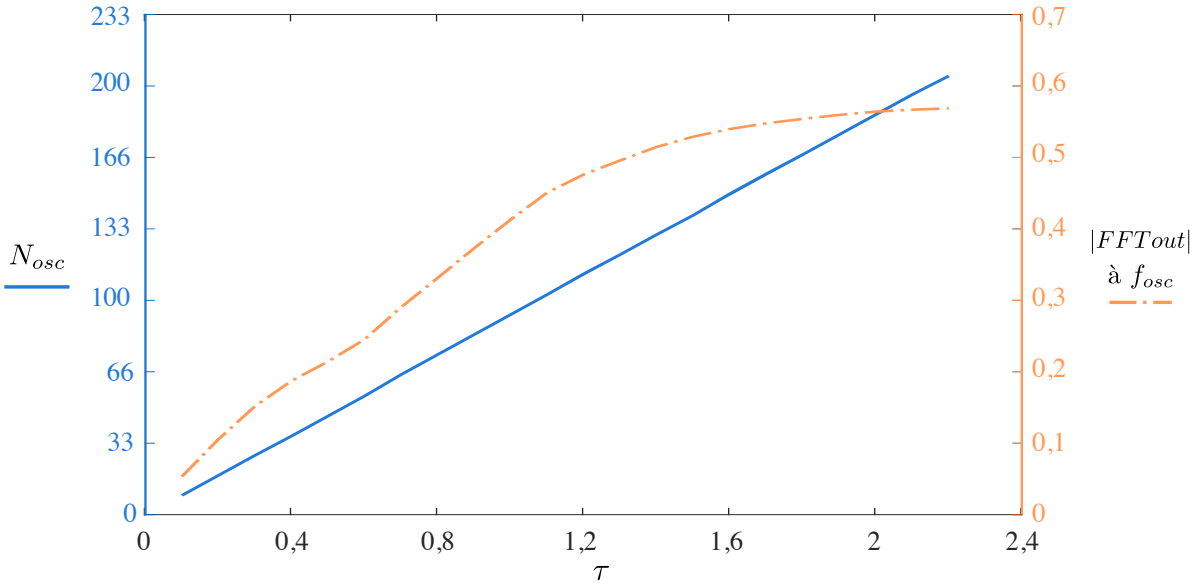


FIGURE 5.4: $|FFTout(f)|$ et N_{osc} en fonction de τ

5.2 Banc d'expérimentation

Le rayonnement EM des cellules TERO est évalué avec le banc d'expérimentation présenté sur la figure 5.5.

Il est composé de :

— **Une plateforme matérielle à base de FPGA**, la plateforme HECTOR décrite et utilisée dans le chapitre 2 (voir sec. 2.2.3.1). Pour rappel, elle est composée d'une carte mère et de plusieurs cartes filles [92].
— **Une sonde EM**, modèle RS H 2,5-2 de la marque Rohde & Schwartz pour capter de manière sélective le spectre en courant émit par les pistes conductrices et les composants conducteurs, tels que des condensateurs ou des circuits intégrés.
— **Un analyseur de spectre temps réel**, modèle RSA607a de la marque Tektronix pour traiter en temps réel les émissions EM captées par la sonde EM.
— **Un amplificateur faible bruit**, modèle HZ-16 de la marque Rohde & Schwartz permettant d'amplifier le signal capté par la sonde EM avant de l'envoyer sur l'analyseur de spectre. Cela permet de faciliter les mesures de signaux à faible émission et à haute fréquence, jusqu'à 3 GHz.
— **Une table XYZ** avec une précision de déplacement de 1 μm sur laquelle la plateforme FPGA est fixée pour se déplacer sous la sonde EM.
— **Un PC** en charge de l'exécution des scripts qui contrôle le FPGA, la table XYZ et l'analyseur de spectre.

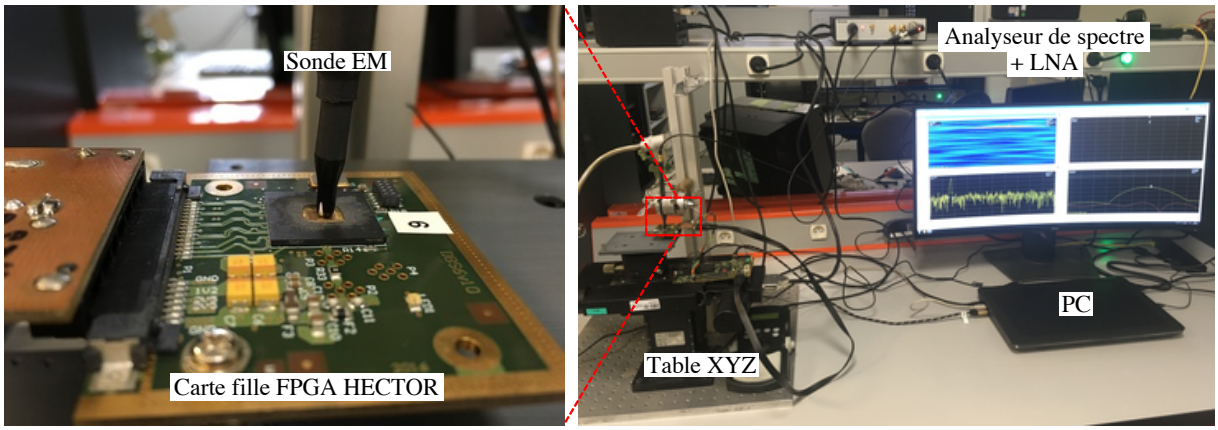


FIGURE 5.5: Banc d'expérimentation

5.3 Analyse électromagnétique des cellules TERO

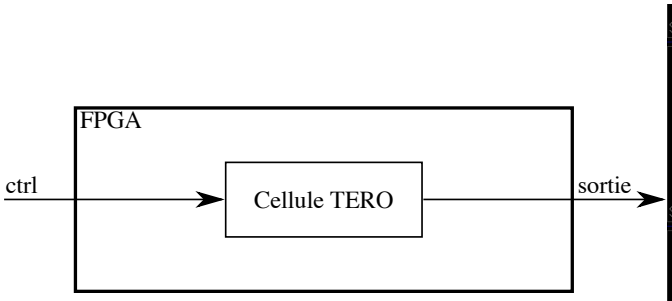
L'objectif des expériences présentées ici est de retrouver le nombre d'oscillations (N_{osc}) des cellules TERO en fonctionnement à l'aide de l'analyse électromagnétique et de montrer à quel point cela peut compromettre la sécurité des générateurs d'aléa à base de cellules TERO. Cette étude est réalisée sur deux familles de FPGA provenant de deux fabricants différents : un FPGA Spartan 6 de Xilinx et un FPGA Cyclone V de Intel. Avant de discuter des résultats d'analyse électromagnétique, nous détaillons la structure de chacune des implantations évaluées.

5.3.1 Structure des implantations évaluées

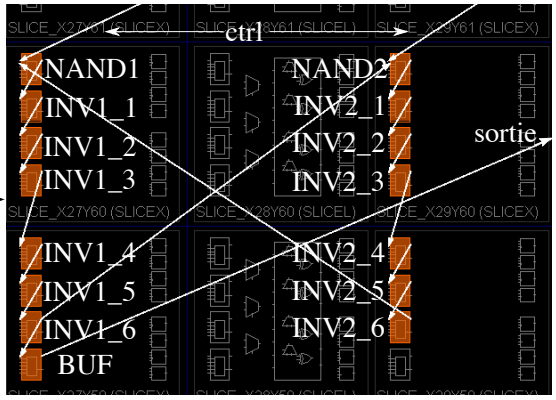
5.3.1.1 Circuit n°1 — Une cellule TERO lorsque son signal de sortie sort du FPGA

Pour ce premier circuit, une cellule TERO seule composée de $N = 7$ inverseurs par branche est implantée sur un FPGA Spartan 6 de Xilinx. La cellule TERO est activée de manière périodique par un signal de contrôle (*ctrl*) à basse fréquence (50 kHz).

Pour cette première expérience, la sortie de la cellule est envoyée à l'extérieur du FPGA et elle est observée sur un oscilloscope. La figure 5.6a représente l'architecture du circuit implanté. La figure 5.6b montre une impression-écran de l'outil ISE floor plan de Xilinx après placement et routage sur laquelle chaque élément qui compose la cellule TERO est visible. Avant d'être envoyée à l'extérieur du FPGA, la sortie de la cellule TERO passe par un buffer.



(a) Architecture



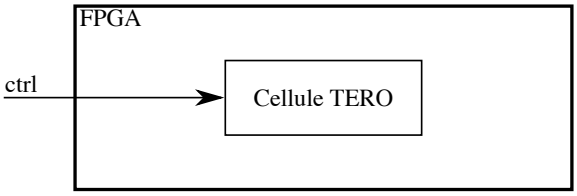
(b) Xilinx ISE floor plan

FIGURE 5.6: Implantation du circuit n°1 - Une cellule TERO

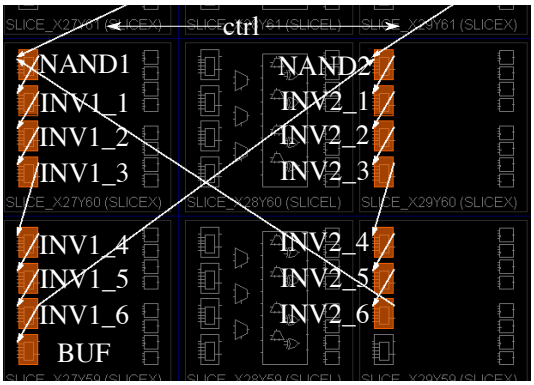
5.3.1.2 Circuit n°2 — Une cellule TERO lorsque son signal de sortie ne sort pas du FPGA

Comme pour le circuit n°1, une seule cellule TERO est implantée sur le FPGA mais, cette fois-ci, le signal de sortie de la cellule TERO ne sort pas du FPGA. L'objectif est d'observer s'il y a une différence d'émission EM lorsque le signal sort ou non sur les entrées/sorties du FPGA.

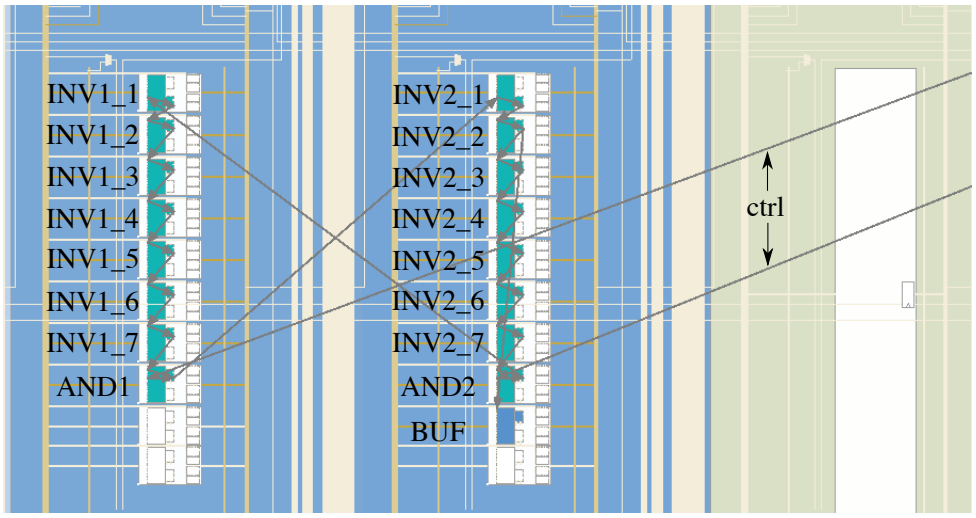
Ce circuit est implanté sur un FPGA Spartan 6 de Xilinx et un FPGA Cyclone V de Intel. La figure 5.7a représente l'architecture du circuit implanté. Les figures 5.7b et 5.7c montrent des impressions-écran des outils ISE floor plan de Xilinx et Quartus floor plan de Intel.



(a) Architecture



(b) Xilinx ISE floor plan



(c) Intel Quartus floor plan

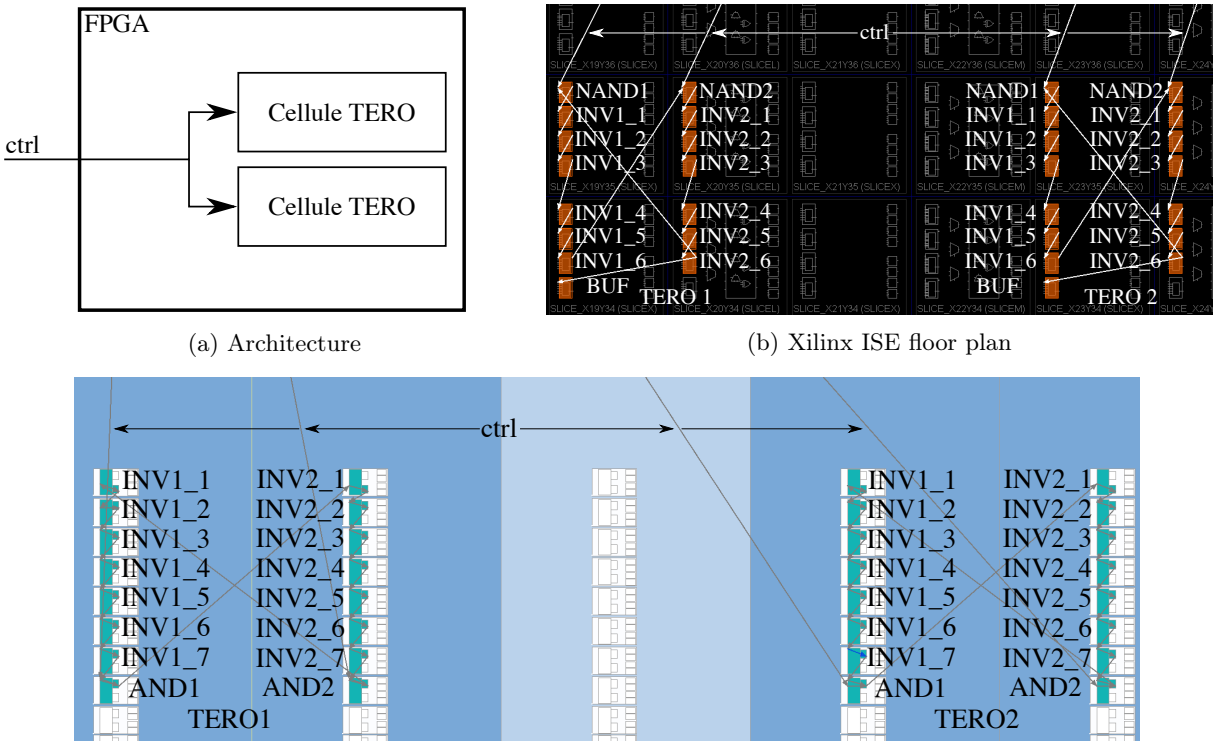
FIGURE 5.7: Implantation du circuit n°2 - Une cellule TERO

5.3.1.3 Circuit n°3 — Deux cellules TERO

Dans ce troisième circuit, deux cellules TERO identiques sont implantées. Les deux cellules sont activées en même temps. L'objectif est de vérifier s'il est possible de dissocier les deux

émissions EM et de retrouver le nombre d'oscillations de chacune des cellules.

La figure 5.8a représente l'architecture du circuit implanté. Les figures 5.8b et 5.8c montrent des impressions-écran des outils ISE floor plan de Xilinx et Quartus floor plan de Intel. Sur les deux impressions-écran, nous pouvons voir que les deux cellules implantées sont identiques. En effet, le placement et le routage sont exactement les mêmes. Ceci est nécessaire pour la conception de PUF afin d'assurer que seule la différence due aux variations de procédés de fabrication est comparée (voir chapitre 2).



(a) Architecture

(b) Xilinx ISE floor plan

(c) Intel Quartus floor plan

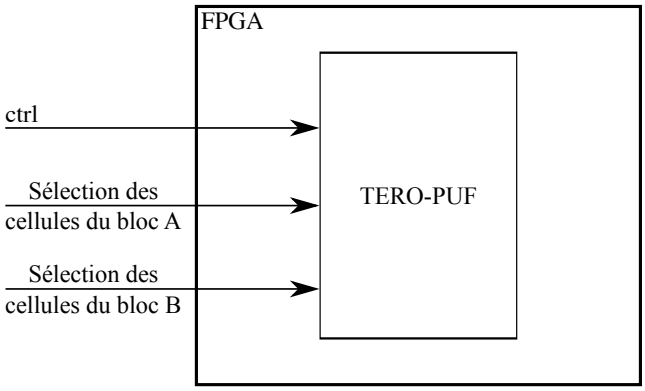
FIGURE 5.8: Implantation du circuit n°3 - Deux cellules TERO

5.3.1.4 Circuit n°4 — TERO-PUF

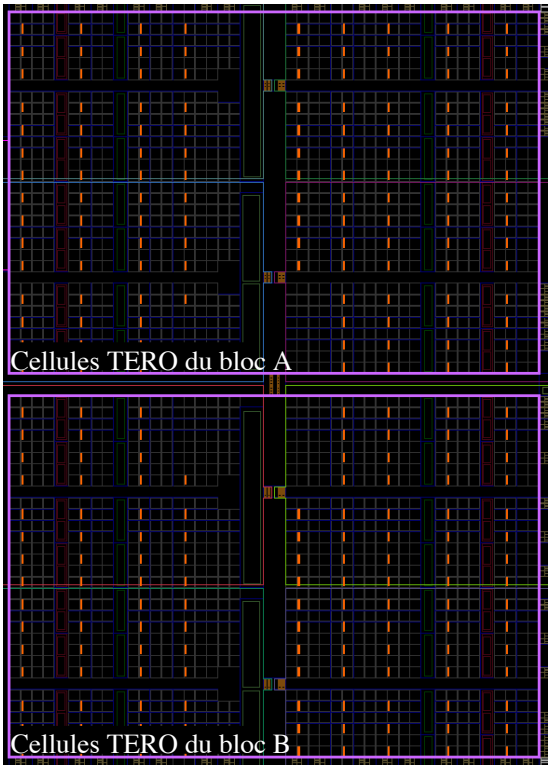
Le dernier circuit analysé est composé d'une TERO-PUF complète implantée sur un FPGA Spartan 6 de Xilinx (voir sec 1.2.3.4). La figure 5.9a représente l'architecture du circuit implanté. Pour rappel, la TERO-PUF est composée de deux blocs (A et B) de 128 cellules TERO chacun. Une cellule du bloc A sera toujours comparée à une cellule du bloc B. Si la cellule TERO du bloc A à un nombre d'oscillations supérieur à la cellule TERO du bloc B, le comparateur de la PUF génère un '1' logique en sortie. Sinon, elle génère un '0' logique. Le signal de contrôle ainsi que des mots de sélection (challenge) pour chaque bloc sont envoyés sur le FPGA depuis l'extérieur.

L'utilisateur a donc accès aux challenges. Cependant, la réponse de la PUF ne quitte jamais le FPGA.

La figure 5.9b montre une impression-écran de l'outil ISE floor plan de Xilinx après placement et routage de la PUF sur laquelle nous pouvons voir clairement les deux blocs A et B. Le nombre d'inverseurs par branche est de 7 ($N = 7$) pour chaque cellule TERO. Comme pour le circuit n°3, les cellules sont toutes implantées de manière identique pour extraire les variations de procédés de fabrication.



(a) Architecture



(b) Xilinx ISE floor plan

FIGURE 5.9: Implantation du circuit n°4 - TERO-PUF

5.3.2 Analyses électromagnétiques

5.3.2.1 Analyse EM du circuit n°1

Comme nous venons de le voir, le premier circuit est composé d'une seule cellule TERO avec 7 inverseurs par branche. Avec l'oscilloscope, nous pouvons observer une fréquence d'oscillation moyenne (f_{osc}) de 174,4 MHz et un nombre d'oscillations moyen (N_{osc}) de 228. La figure 5.10 montre le signal de sortie de la cellule TERO observé à l'aide de l'oscilloscope. Sortir le signal et l'observer avec un oscilloscope nous permet de vérifier l'exactitude des résultats obtenus avec l'analyse EM.

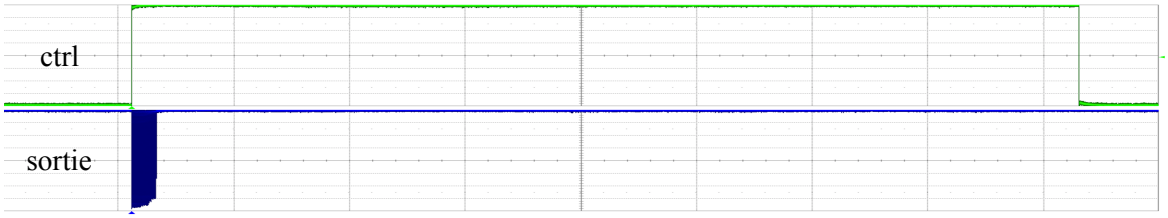


FIGURE 5.10: Observation à l'oscilloscope du signal de sortie d'une cellule TERO

Puisque la fréquence d'oscillation moyenne de la cellule TERO est connue, la fenêtre d'enregistrement de l'analyseur de spectre est centrée à cette fréquence avec une largeur de 7 MHz. En sondant le FPGA avec le banc d'expérimentation représenté sur la figure 5.5, il est possible de capturer les émissions EM de la cellule TERO en fonctionnement. La figure 5.11 montre les résultats obtenus avec l'analyseur de spectre. Le spectrogramme coloré représente l'amplitude d'émission (rouge pour les amplitudes élevées et bleu pour les amplitudes faibles) en fonction du temps (ordonnée) sur une fenêtre de fréquence donnée (abscisse).

Sur la figure 5.11, le signal capté (en rouge) est centré sur une fréquence moyenne de 174,4 MHz ce qui nous permet clairement de reconnaitre la cellule TERO. De plus, il est possible d'identifier combien de temps la cellule TERO oscille. Dans ce cas, la cellule oscille pendant 1,28 μs. En connaissant f_{osc} et le temps d'oscillation, le nombre d'oscillations (N_{osc}) peut très facilement être déduit : $N_{osc} = 1,28 \times 10^{-6} \times 174,4 \times 10^{6} = 223$.

Il est intéressant de constater une faible différence entre le nombre d'oscillations déduit de l'émission EM (223) et le nombre d'oscillations observé à l'oscilloscope (228). Ceci est dû au fait que l'oscilloscope enregistre le nombre d'oscillations moyen de la cellule TERO pour plusieurs activations successives alors que l'analyseur de spectre temps réel enregistre une seule réalisation de la cellule TERO. Une autre représentation proposée par l'analyseur de spectre est l'amplitude d'émission en fonction du temps. Une fois la fréquence d'oscillation de la cellule TERO trouvée, cette représentation permet de mesurer facilement le temps d'oscillation. La figure 5.12 montre l'amplitude d'émission en fonction du temps pour la même réalisation de la cellule TERO que celle de la figure 5.11. Nous pouvons voir sur la figure 5.12 que l'amplitude d'émission augmente avec le temps, donc avec le nombre d'oscillations, ce qui confirme la modélisation de la section 5.1.

5.3.2.2 Analyse EM du circuit n°2

Pour cette expérience, la même analyse, avec l'analyseur de spectre centré sur une fréquence de 174,4 MHz, est réalisée. Cependant, cette fois-ci, le signal de sortie de la cellule TERO n'est pas envoyé à l'extérieur du FPGA. En réalisant une cartographie, en suivant l'approche décrite par Bayon *et al.* [89], il n'est pas possible d'identifier les émissions EM de la cellule TERO. En

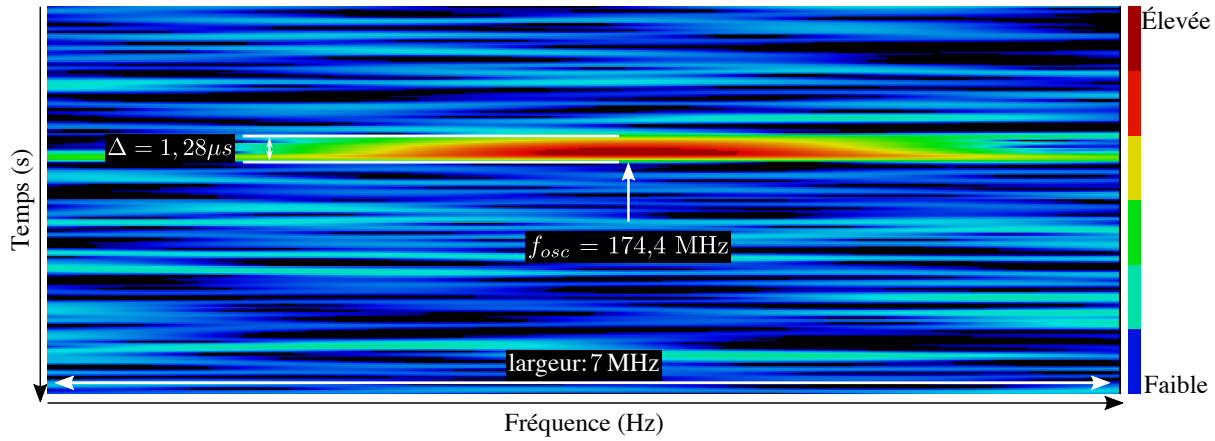


FIGURE 5.11: Spectrogramme d'une cellule TERO en fonctionnement

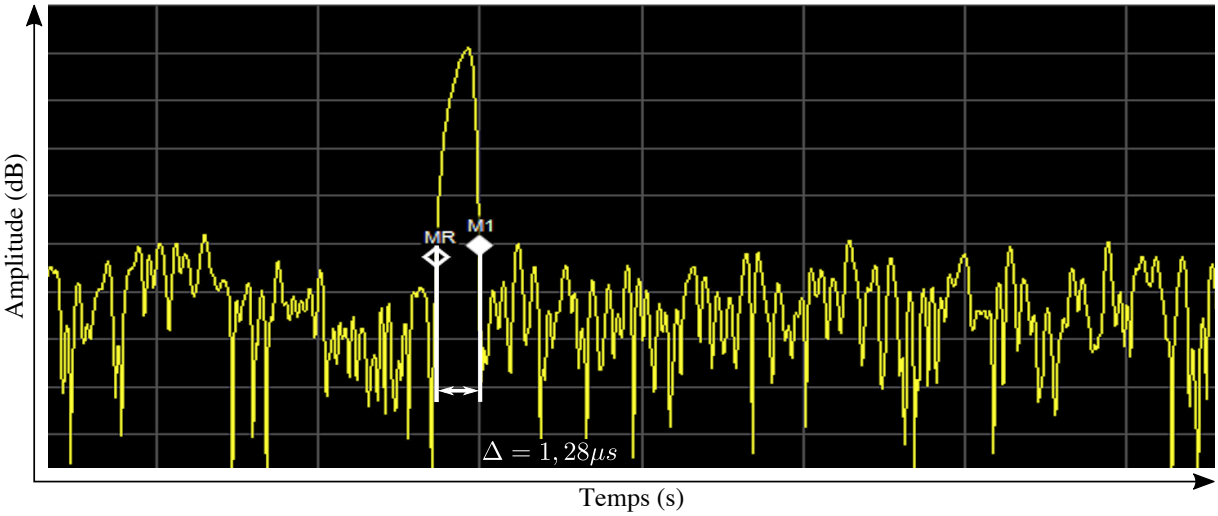


FIGURE 5.12: Amplitude d'émission en fonction du temps de la cellule TERO

effet, il semblerait que l'amplitude d'émission de la cellule TERO ne soit pas assez grande pour être différenciée du bruit ambiant. Il était possible de capter les oscillations de la cellule TERO dans la section 5.3.2.1, car le fait de sortir le signal sur une broche de sortie du FPGA augmente l'amplitude d'émission.

Ensuite, nous avons réalisé la même analyse sur un FPGA Cyclone V de Intel. Cette analyse donne des résultats similaires à ceux obtenus avec un FPGA Spartan 6 de Xilinx : il n'est pas possible d'extraire les émissions EM de la cellule TERO en fonctionnement.

Ainsi, contrairement à une cellule RO, puisqu'elle n'oscille que temporairement, la cellule TERO n'émet pas assez pour être détectée par l'analyseur de spectre sans sortir son signal du FPGA. Pour pouvoir être détectée, il faudrait que la cellule TERO oscille plus longtemps, comme nous l'avons vu dans la partie 5.1.

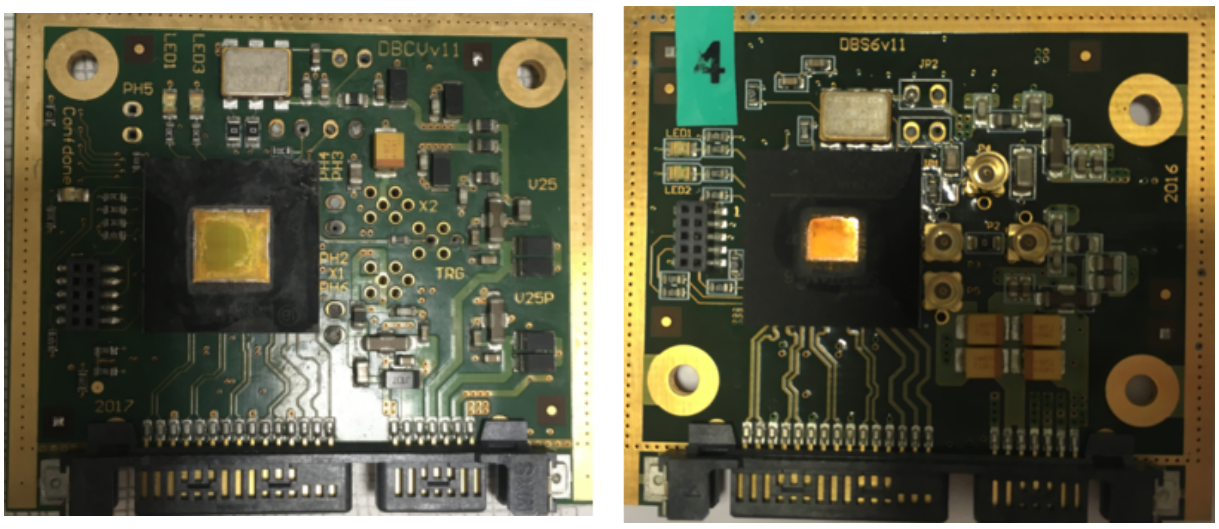
Une autre solution pour pouvoir observer les émissions des cellules TERO avec l'analyseur de

spectre est de décapsuler le FPGA. Pour cette raison, dans la suite de ce chapitre, tous les tests sont réalisés sur des FPGA décapsulés. Le protocole de décapsulation que nous avons suivi est celui décrit par Wittke *et al.* [100]. Pour réaliser la décapsulation, nous avons utilisé le système de décapsulation de circuit intégré JetEth II de Nisene. La recette utilisée pour décapsuler le FPGA Spartan 6 de Xilinx est la suivante :

— Mélange d'acide : **Nitrique (HNO_3) / Sulfurique (H_2SO_4) : 5/1**
— Température de gravure : **44 ° C**
— Temps de gravure : **240s**
— Nettoyage : **Acétone**

Pour le FPGA Cyclone V de Intel, la recette est légèrement différente, car les fils de connexion sont composés de cuivre contrairement à ceux des FPGA Spartan 6 qui sont en or. Or, l'acide nitrique (HNO_3) détruit le cuivre. Nous avons donc utilisé seulement de l'acide sulfurique (H_2SO_4).

La figure 5.13 montre les deux FPGA décapsulés :



(a) Intel Cyclone V

(b) Xilinx Spartan 6

FIGURE 5.13: FPGA décapsulés

Après décapsulation, la même analyse que décrite précédemment nous permet d'observer les émissions EM de la cellule TERO sur les deux FPGA. Pour la cellule TERO implanté sur le FPGA Spartan 6 de Xilinx, le résultat est similaire à celui de la partie 5.3.2.1 : N_{osc} est autour de 225. Pour le FPGA Cyclone V de Intel, une cellule TERO composée de 7 inverseurs par branche est aussi implantée. La fréquence d'oscillation observée (f_{osc}) est de 198 MHz et son nombre d'oscillations (N_{osc}) de 462.

5.3.2.3 Analyse EM du circuit n°3

Dans cette troisième analyse, deux cellules TERO composées de 7 inverseurs par branche sont implantées sur un FPGA Spartan 6 décapsulé. Le signal de sortie des cellules TERO ne sort pas du FPGA et les deux cellules TERO sont activées en même temps par le même signal d'activation.

La figure 5.14a montre le spectrogramme coloré issu de cette analyse. Les émissions EM captées et visibles sur la figure peuvent être divisées en deux parties :

— Une première partie lorsque les deux cellules TERO oscillent ;

— Une seconde partie lorsqu'une seule cellule TERO continue à osciller.

Dans la première partie, qui dure 1,28 μs, nous pouvons retrouver la cellule TERO identifiée dans la section 5.3.2.1. La perte d'amplitude après 1,28 μs rend possible l'identification de la fin des oscillations d'une des deux cellules TERO.

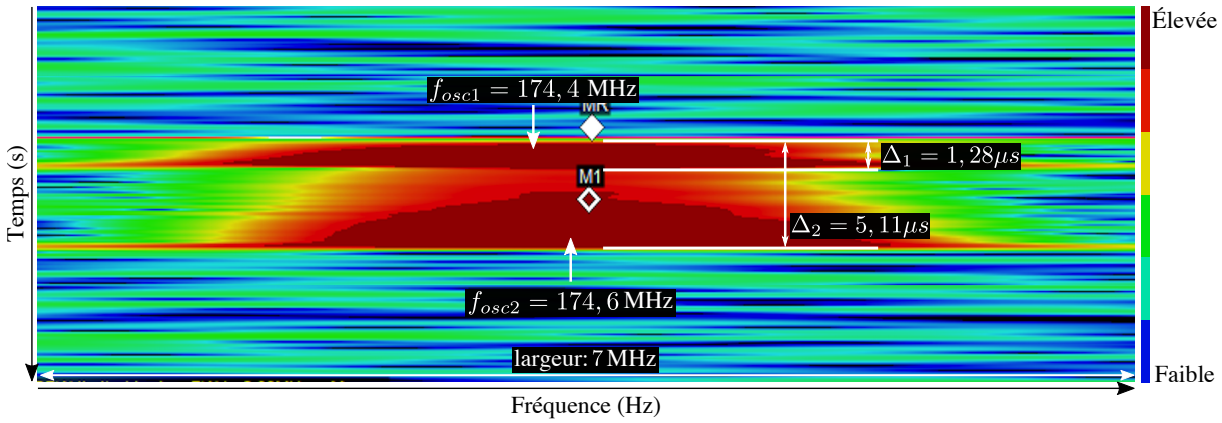
Il est aussi important de remarquer que, pendant la première partie, la largeur des émissions EM est plus grande. En effet, les deux cellules TERO n'oscillent pas exactement à la même fréquence (f_{osc}) à cause des variations de procédé de fabrication. Ceci représente un deuxième indice permettant d'identifier la fin des oscillations d'une des deux cellules TERO. Ainsi, nous sommes en mesure d'identifier clairement le nombre d'oscillations de chacune des deux cellules TERO.

La première cellule TERO a une fréquence d'oscillation (f_{osc1}) de 174,4 MHz et un nombre d'oscillations (N_{osc1}) de 223. La deuxième cellule TERO à une fréquence d'oscillation (f_{osc2}) de 174,6 MHz et oscille pendant 5,11 μs donnant un nombre d'oscillations (N_{osc2}) de 892.

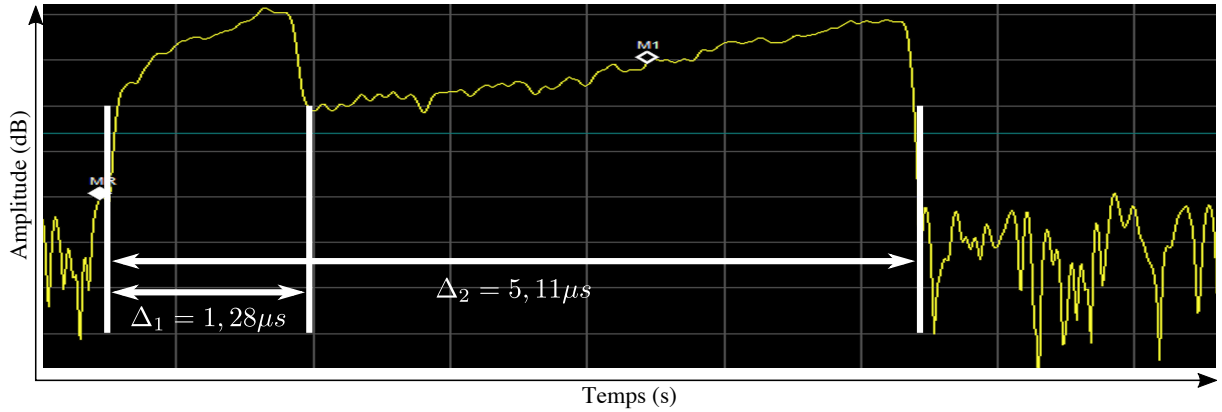
La figure 5.14b montre l'amplitude d'émission en fonction du temps pour cette même acquisition. Il est possible d'identifier de manière très claire l'arrêt d'une des deux cellules TERO grâce à la perte d'amplitude. Ces résultats confirment ceux obtenus avec le spectrogramme de la figure 5.14a.

5.3.2.4 Analyse EM du circuit n°4

Puisque nous avons démontré qu'il était possible d'identifier clairement et de retrouver le nombre d'oscillations (N_{osc}) de deux cellules TERO qui oscillent en même temps, nous avons réalisé une analyse EM de la TERO-PUF complète décrite dans la partie 5.3.1.4. Pour rappel, les cellules TERO qui composent la TERO-PUF sont toutes composées de 7 inverseurs par branche et sont implantées de manière identique sur un FPGA Spartan 6 de Xilinx à l'aide de HardMacro. Pour plus de détail sur l'implantation d'une TERO-PUF sur un FPGA Spartan 6 de Xilinx, le lecteur peut se référer aux travaux de Marchand *et al.* [72]. Puisque toutes les cellules sont, aux variations de procédé de fabrication près, identiques, leur fréquence d'oscillation (f_{osc}) est proche



(a) Spectrogramme pour une réalisation de deux cellules TERO



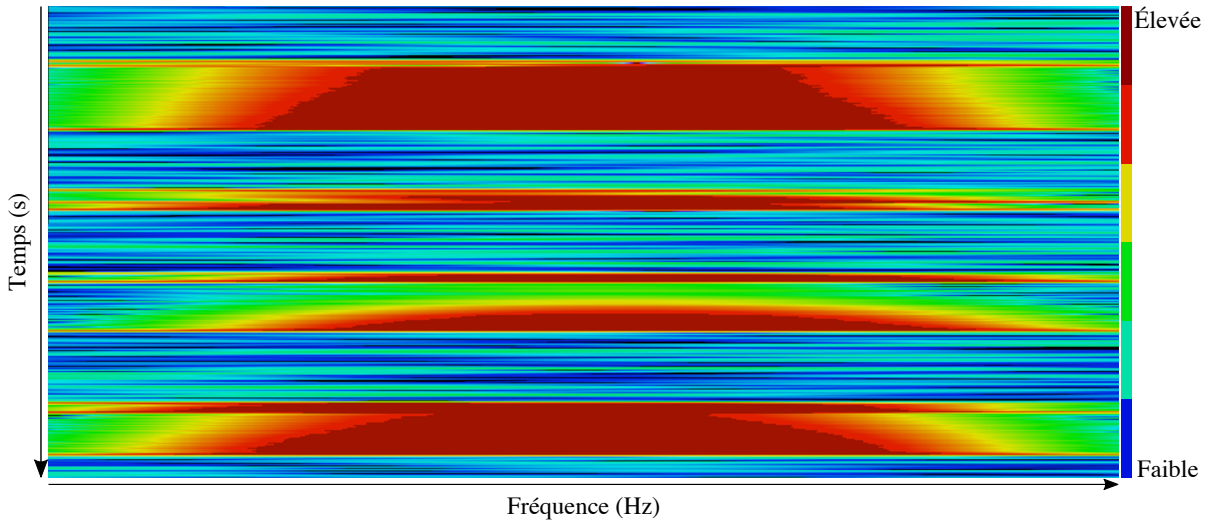
(b) Amplitude d'émission en fonction du temps de deux cellules TERO

FIGURE 5.14: Analyse de deux cellules TERO à l'aide de l'analyseur de spectre

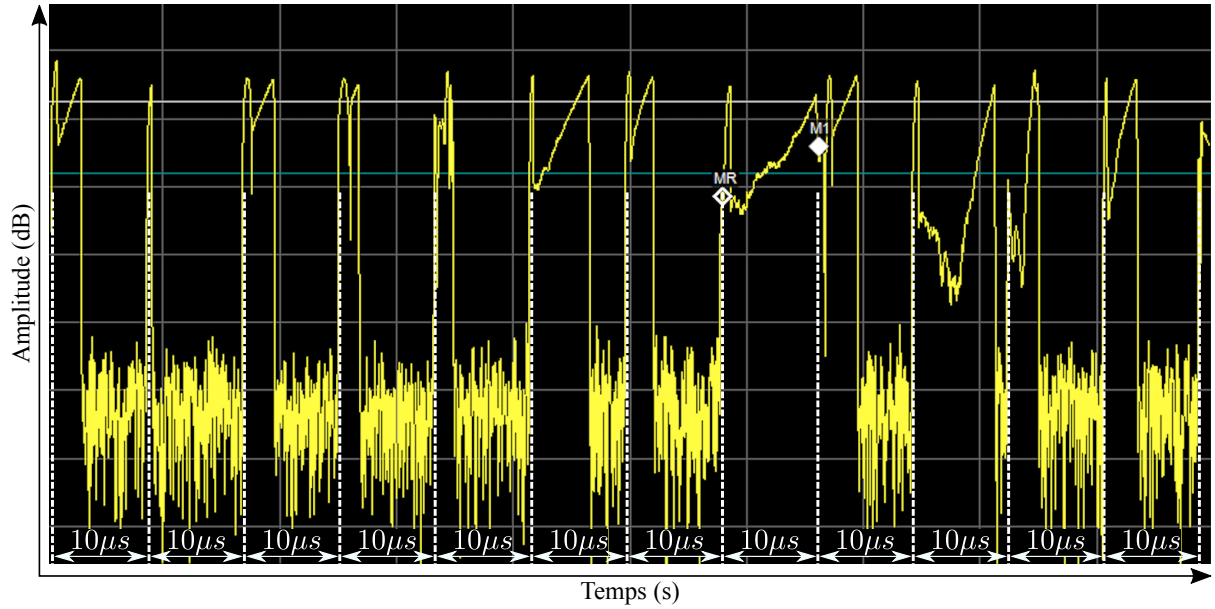
de 174 MHz. Ainsi, l'analyseur de spectre est centré sur 174 MHz avec une largeur de 7 MHz afin d'être certain de capturer toutes les cellules TERO lorsqu'elles oscillent. Le temps d'activation des cellules TERO pour chaque comparaison est de 10 μs.

La figure 5.15 montre le résultat de plusieurs comparaisons successives de cellules TERO deux à deux (avec toujours une cellule TERO du bloc A comparée à une cellule TERO du bloc B). La figure 5.15a représente le spectrogramme coloré pour quatre comparaisons successives. La figure 5.15b représente l'amplitude d'émission en fonction du temps pour 12 comparaisons successives. En divisant la figure 5.15b horizontalement par paquets de 10 μs, nous pouvons retrouver chaque comparaison. Pour cette expérience, les comparaisons successives effectuées sont les suivantes : la cellule TERO A_i est comparée à la cellule TERO B_i, la cellule TERO A_{i+1} est comparée à la cellule TERO B_{i+1}, etc. Ce résultat prouve que l'analyseur de spectre est capable de capter les comparaisons successives.

À présent, plaçons-nous dans le cas de figure d'un attaquant. Pour des raisons évidentes de sécurité, les résultats de comparaison des cellules TERO, autrement dit la réponse de la PUF, ne



(a) Spectrogramme



(b) Amplitude d'émission en fonction du temps

FIGURE 5.15: Analyse d'une TERO-PUF à l'aide de l'analyseur de spectre

sortent pas du FPGA. Cependant, pour ce circuit, et ceci est le cas bien souvent dans la réalité, l'utilisateur (ou l'attaquant) a accès aux challenges de la PUF. Dans cette configuration, seul un nombre limité de comparaisons couplées à l'analyse EM permet de cloner complètement la PUF.

La figure 5.16 montre un exemple de comparaisons successives à effectuer pour identifier le nombre d'oscillations (N_{osc}) de chacune des cellules TERO qui composent la PUF. Deux comparaisons d'une cellule TERO avec deux autres cellules TERO permettent d'identifier cette cellule et son nombre d'oscillations.

Par exemple, la cellule TERO A_1 comparée à la cellule TERO B_1 permet d'identifier deux nombres d'oscillations distincts comme nous l'avons vu dans la partie 5.3.2.3. Ensuite, il suffit de comparer la cellule TERO A_1 à la cellule TERO B_2 pour retrouver le nombre d'oscillations de la cellule A_1. Connaitre N_{osc} de A_1 permet ensuite de déduire N_{osc} de B_1 et B_2. Comparer la cellule TERO B_1 avec la cellule TERO A_2 en connaissant le nombre d'oscillations de B_1 permet de déduire le nombre d'oscillations de la cellule TERO A_2 et ainsi de suite. Par conséquent, $2 \times m - 1$ (avec m le nombre de cellules TERO dans un bloc) comparaisons sont suffisantes pour retrouver le nombre d'oscillations de toutes les cellules TERO qui forment la TERO-PUF.

$$
\begin{aligned}
&A_1 \overset{versus}{\longleftrightarrow} B_1 && \\
& && \Longrightarrow \quad \text{Identification de deux } N_{osc} \\
&A_1 \overset{versus}{\longleftrightarrow} B_2 && \\
& && \Longrightarrow \quad N_{osc} \text{ de } A_1,\ B_1 \text{ et } B_2 \\
&A_2 \overset{versus}{\longleftrightarrow} B_1 && \\
& && \Longrightarrow \quad N_{osc} \text{ de } A_2 \\
&\vdots && \\
&A_m \overset{versus}{\longleftrightarrow} B_m && \\
& && \Longrightarrow \quad N_{osc} \text{ de } A_m \text{ et } B_m
\end{aligned}
$$

FIGURE 5.16: Schéma de comparaisons successives pour cloner une TERO-PUF complète par analyse EM

5.4 Mesures de prévention des fuites EM

Dans les parties précédentes, la susceptibilité des cellules TERO à l'analyse EM a été démontrée. Cette susceptibilité révèle une faiblesse sur la TERO-PUF. Ici, nous dressons une liste de recommandations pour réduire autant que possible les scénarios d'attaque de la TERO-PUF par l'analyse EM.

La première solution, certainement la plus efficace, est de rendre le circuit intégré physiquement inaccessible. Par exemple, en l'enfermant dans un boitier en aluminium pour bloquer les émissions EM. Cependant, en fonction des contraintes du circuit, cette solution n'est pas toujours réalisable. Notamment dans un contexte IoT où les ressources et l'espace sont limités.

Une autre mesure de prévention est de bloquer l'accès aux challenges au moment de la conception. En effet, si les challenges sont générés à l'intérieur du circuit et ne peuvent pas être sélectionnés par l'utilisateur, l'attaquant ne pourra pas retrouver le nombre d'oscillations par l'analyse EM des cellules TERO, et ce quel que soit le degré d'accessibilité physique au circuit.

Nous pourrions penser qu'avoir des cellules avec un nombre d'oscillations proche serait une autre solution puisqu'il serait compliqué, voir impossible, de distinguer les cellules les unes des autres. Cependant, cette approche est à proscrire pour la conception de PUF, car un nombre d'oscillations trop proche diminuerait la stabilité de la PUF.

Enfin, nous avons vu que ne pas sortir le signal de sortie des cellules TERO du FPGA diminuait grandement l'amplitude des émissions EM. Dans le cas où la fonction implantée serait distribuée sur plusieurs composants d'un même circuit, par exemple un FPGA et un processeur, le concepteur pourrait envisager de sortir le signal des cellules TERO du FPGA. L'analyse EM est une raison de ne pas le faire. Bien sûr, nous avons vu dans la partie 5.3.2 qu'en décapsulant le FPGA, une analyse EM était toujours possible. Cependant, cette mesure de prévention rend l'analyse plus compliquée et diminue le nombre d'attaquants potentiels.

5.5 Conclusion

Dans cette étude, nous avons présenté et discuté de l'analyse EM des cellules TERO. Nous avons montré que les cellules TERO émettent suffisamment pour être détectées avec un analyseur de spectre. Ainsi, le nombre d'oscillations (N_{osc}) peut être retrouvé sans accès au signal de sortie de la cellule. Ensuite, cela rend possible de cloner une TERO-PUF complète avec un nombre de comparaisons limité.

En réalisant cette étude sur deux FPGA provenant de deux fabricants, nous avons aussi montré que cette analyse EM fonctionne, quel que soit le circuit.

Il est important de retenir que sortir le signal des cellules TERO sur les broches de sortie du FPGA augmente de manière significative l'amplitude d'émission EM. De plus, laisser un accès libre aux challenges de la PUF la rend clonable.

Cependant, nous devons noter une meilleure résistance des cellules TERO vis-à-vis de l'analyse du rayonnement EM par rapport à celle des cellules RO qui oscillent en permanence et qui produisent donc un rayonnement EM d'amplitude plus importante.

Très récemment (travaux présentés lors de la Conférence COSADE en mai 2019) une équipe de l'Université Technique de Munich et de l'institut Fraunhofer est arrivée aux mêmes conclusions que nous [ref]. Ils sont partis de notre code pour la TERO-PUF [72] et ils ont mise en œuvre une analyse spectrale du rayonnement EM. Ne disposant pas d'un analyseur de spectre temps réel, ils ont mesuré le nombre d'oscillations en calculant le rapport signal à bruit qui augmente lorsque les cellules TERO oscillent. Ces travaux sont tout à fait complémentaires aux nôtres et démontrent la faiblesse des cellules TERO vis-à-vis de l'analyse EM.

Les résultats présentés dans ce chapitre, et les sources VHDL disponibles sur Git aideront les concepteurs à mieux anticiper les fuites EM des cellules TERO lors de leurs futures conceptions.

Conclusion et perspectives

Dans un monde où les objets connectés sont omniprésents, nos lieux de vie et nos environnements de travail ont totalement changé. Cette connexion permanente des appareils d'usage quotidien améliore notre confort, mais peut aussi mener à d'inimaginables problèmes de sécurité. Comme nous l'avons vu dans l'introduction, les problèmes de sécurités liés au vaste déploiement de l'internet des objets touchent à la fois la sécurité des données et la sécurité des systèmes.

Dans ce contexte, les générateurs physiques d'aléa représentent un rempart aux problèmes de sécurités puisqu'ils assurent le bon fonctionnement de certaines fonctions cryptographiques lorsque celles-ci sont réalisées dans les circuits intégrés. En effet, ils exploitent des sources de bruit analogique présentes dans les circuits électroniques pour générer : des clés secrètes permettant de chiffrer les données, des masques aléatoires ou vecteurs d'initialisation pour des protocoles cryptographiques, ou encore, des identifiants uniques permettant l'authentification des circuits.

L'objectif de cette thèse était de proposer des solutions pour améliorer à la fois la conception, la caractérisation, la modélisation et la robustesse contre les menaces sécuritaires des générateurs physiques d'aléa qui exploitent des cellules oscillantes au sein des circuits électroniques numériques.

Principales contributions

Le chapitre 1, nous a permis d'analyser les différents travaux existants dans la littérature scientifique qui traitent des générateurs physiques d'aléa. Dans un premier temps, nous avons vu les différentes méthodes d'évaluations de la qualité statistique et l'imprévisibilité des TRNG et des PUF. Deuxièmement, nous avons étudié les implantations des différents grands principes de TRNG et PUF sur les circuits numériques. Par la suite, la sensibilité des RO au phénomène de verrouillage et les possibles attaques induites par cette sensibilité sur les générateurs physiques d'aléa à base de RO ont été étudiées. Finalement, nous avons vu le potentiel d'une attaque passive par analyse du rayonnement EM sur des générateurs physiques d'aléa à base de RO.

L'analyse réalisée dans le chapitre 1 nous a permis d'identifier plusieurs points d'améliorations pour compléter la littérature scientifique.

Premièrement, nous avons vu que les principes de TRNG et les principes de PUF possédant les meilleurs résultats d'implantations sur les circuits numériques étaient tous basés sur des cellules oscillantes. Pourtant, aucune étude scientifique dans le domaine des TRNG ne permettait de comparer de manière équitable et efficace les différents principes de TRNG à base de cellules oscillantes. De plus, nous avons vu que la plupart des travaux d'implantation des PUF à base de cellules oscillantes avaient été réalisés sur des FPGA de type SRAM.

Aussi, les travaux de Marchand *et al.*, ont monté qu'une implantation de PUF était très dépendante du circuit qui l'accueille [90]. Il se trouve qu'il n'existait pas d'implantation rigoureuse de PUF à base de cellules oscillantes sur les FPGA de type Flash.

Pour ces raisons, dans le chapitre 2, nous avons réalisé une implantation de tous les TRNG à base de cellules oscillantes. Toutes ces implantations ont été effectuées dans les mêmes conditions et en suivant une approche rigoureuse pour pouvoir les comparer de la meilleure des manières possibles. Cette analyse nous a permis de comprendre qu'il n'existait pas de TRNG idéal. Un compromis entre le débit, la surface d'occupation du circuit, la puissance consommée et l'entropie devra toujours être fait. Le TRNG le plus approprié doit être sélectionné en fonction des contraintes de l'application.

Ensuite, nous avons réalisé une implantation rigoureuse de la RO-PUF et de la TERO-PUF sur les FPGA de type Flash. Ces implantations ont été réalisées au niveau le plus proche du matériel. Nous avons vu comment assurer que la structure de la cellule ne soit pas modifiée au cours du processus de synthèse, de placement et de routage, comment contrôler le délai des éléments critiques de la PUF, comment isoler les cellules oscillantes du reste du circuit afin d'éviter toute possibilité de perturbation et comment maximiser les variations locales.

Ces deux études nous ont permis de sélectionner un TRNG et une PUF pour les intégrer au sein de trois démonstrateurs complets. Ces intégrations nous ont permis de prouver l'utilité de ces primitives cryptographiques au sein d'un système complet ainsi que de prouver la faisabilité d'une telle intégration.

Deuxièmement, nous avons vu, à travers les nombreux efforts de standardisation de l'évaluation des TRNG existants dans la littérature scientifique, que la meilleure solution pour évaluer l'entropie d'un générateur physique d'aléa était d'établir un modèle stochastique de celui-ci. Cette approche n'avait, jusqu'à présent, jamais été réalisée pour les PUF.

Ainsi, dans le chapitre 3, inspiré par les travaux de Haddad *et al.* [95], nous avons amorcé cette approche en apportant des éléments de modélisation des caractéristiques probabilistes des cellules oscillantes exploitées par les PUF. Pour ce faire, nous avons réalisé une modélisation électrique d'un inverseur à base de transistors MOS et d'une ligne de transmission.

Cela nous a permis d'établir un modèle électrique de la cellule RO. Ensuite, nous avons fait le lien entre l'impact des variations de procédés de fabrication dans les circuits intégrés et le modèle électrique de la cellule RO pour obtenir un modèle stochastique de celle-ci.

Ensuite, nous avons pu voir les conséquences désastreuses du phénomène de verrouillage sur l'imprévisibilité des nombres produits par un générateur physique d'aléa. Nous avons vu que toutes les études existantes dans la littérature ne traitaient que du phénomène de verrouillage sur les générateurs physiques d'aléa à base de cellules RO. De plus,nous avons pu remarquer qu'aucune analyse des causes du phénomène de verrouillage n'avait été proposée ni aucun conseil pour éviter ce verrouillage n'avait été fourni.

C'est pourquoi, dans le chapitre 4, nous avons effectué une analyse complète du verrouillage sur les trois cellules oscillantes utilisées pour la génération d'aléa : les RO, les TERO et les STR. Nous avons ensuite montré que la proximité des cellules oscillantes et leur routage influent grandement sur la plage de verrouillage. Enfin, nous avons montré l'impact du phénomène de verrouillage sur des systèmes basés sur les cellules oscillantes avant de donner des recommandations pour limiter autant que possible ce phénomène sur les cellules oscillantes.

Enfin, nous avons vu qu'un certain nombre de travaux traitait du rayonnement EM des cellules RO. Tout comme pour le phénomène de verrouillage, l'analyse EM des générateurs d'aléa à base de cellules oscillantes n'a été réalisée que sur des générateurs à base de RO. Intuitivement, nous aurions pu penser que la cellule TERO qui n'oscille que temporairement avant de se stabiliser serait moins sensible à ce genre d'attaque passive. Dans le chapitre 5, nous avons réalisé une analyse EM du TERO, ainsi qu'une analyse EM d'une TERO-PUF pour vérifier sa résistance face à une attaque de ce genre. Nous avons montré que les cellules TERO émettent suffisamment pour être détectées avec un analyseur de spectre.

Conclusion générale du manuscrit

À travers les différents travaux présentés tout au long de ce manuscrit, nous avons pu prendre conscience que réaliser un bon générateur physique d'aléa nécessitait de répondre à plusieurs critères. Répondre à ces critères nécessite d'avoir des connaissances dans des domaines variés comme les mathématiques, la physique et l'électronique. Pour qu'un générateur d'aléa soit efficace en pratique il faut :

— Que le principe sur lequel repose le générateur soit suffisamment efficace pour extraire l'aléa des circuits électroniques. Pour cela, il est très utile de pouvoir s'appuyer sur un modèle stochastique du générateur d'aléa et évaluer sa qualité en amont de la conception ;

— Que l'implantation soit réalisée de manière rigoureuse comme nous l'avons particulièrement vu au cours du chapitre 2 pour le cas des PUF sur FPGA de type Flash. Dans la majorité des cas, l'implantation sera dépendante du circuit utilisé et devra être adaptée. Toutefois, les concepts généraux présentés dans le chapitre 2 restent valables. Par exemple, contrôler le délai lors de la conception d'une PUF, ou encore, maximiser les variations locales. Seule la manière d'y arriver change ;

— Que l'implantation soit résistante aux variations ambiantes (température, tension, etc.) et que son comportement ne soit pas perturbé par son inclusion au sein d'un système plus complet ;

— Que l'implantation soit résistante, autant que possible, aux attaques passives et actives. Dans le cas des cellules oscillantes, nous avons vu que les principales attaques possibles étaient l'analyse EM et le verrouillage, intentionnel ou non.

Au cours de ce manuscrit, nous avons tenté d'aborder tous ces points pour le cas des générateurs physiques d'aléa à base de cellules oscillantes. Les expériences effectuées, les recommandations ainsi que les codes VHDL des implantations aideront les futurs concepteurs à mieux appréhender toutes ces contraintes.

Perspectives

Plusieurs perspectives sont envisageables pour continuer les travaux effectués dans cette thèse.

Tout d'abord, nous pensons nécessaire de continuer les travaux commencés dans le chapitre 3. Pour cela, il faudrait dans un premier temps valider le modèle stochastique de la cellule RO avec des résultats expérimentaux sur un ASIC. La démarche à suivre est de mesurer la fréquence d'oscillation moyenne de plusieurs cellules RO identiques et de comparer l'histogramme des résultats à la distribution du modèle stochastique établi. Si les résultats sont similaires, cela signifie que le modèle est valide pour cette structure de RO. Ensuite, il convient d'appliquer la même démarche sur plusieurs autres structures de RO (en faisant varier le nombre d'inverseurs qui composent la cellule). Une fois le modèle de la cellule RO validé, il pourrait être scientifiquement très bénéfique d'établir un modèle stochastique de la RO-PUF complète et de l'utiliser pour estimer l'unicité et l'imprévisibilité de celle-ci en amont de sa conception. De plus, cette approche pourrait aussi être suivie pour d'autre type de PUF.

Ensuite, nous pensons que, pour mieux comprendre l'impact du verrouillage sur les cellules oscillantes étudié dans le chapitre 4 et pour aider le concepteur à anticiper la sensibilité de la cellule au phénomène de verrouillage, il serait intéressant de pouvoir prédire la fréquence d'oscillation naturelle de la cellule. Cependant, sur FPGA, nous ne possédons pas d'information sur la technologie (modèle des transistors qui composent le FPGA, taille, disposition, etc.). Ainsi, il

n'est pas possible d'estimer la fréquence d'oscillation d'une cellule oscillante. De plus, sur FPGA, nous avons un contrôle limité du placement et du routage des éléments (seules des contraintes de placement sont possibles). Ainsi, deux cellules oscillantes avec le même nombre d'éléments N peuvent avoir des fréquences d'oscillation naturelles très éloignées. Pour toutes ces raisons, nous pensons qu'une étude poussée du verrouillage sur les cellules oscillantes réalisée sur un ASIC s'impose comme la prochaine étape pour compléter les travaux effectués dans le chapitre 4. Cela permettrait de maitriser et pouvoir estimer en amont la fréquence d'oscillation des cellules étudiées. De plus, maitriser le routage permettrait d'étudier plus finement l'importance du routage sur le phénomène de verrouillage.

Enfin, pour faire le lien entre le chapitre 4 et le chapitre 5, nous pourrions envisager de capturer le rayonnement EM de la TERO-PUF pour identifier la position des cellules sur le circuit et ensuite procéder à une injection EM pour les verrouiller, à l'image des travaux de Bayon *et al.* sur le MURO-TRNG [85]. De plus, il serait utile de vérifier le potentiel d'émission EM des cellules TERO sur une cible différente comme par exemple un ASIC ou encore lorsque la TERO-PUF est intégrée au sein d'un système complet comportant d'autres horloges et oscillateurs.

Publications et communications

Journaux internationaux avec comité de lecture

1 Ugo Mureddu, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**Experimental Study of Locking Phenomena on Oscillating Rings Implemented in Logic Devices**". *IEEE Transactions on Circuits and Systems I : Regular Papers (TCAS-I)*, Jun 2019.

2 Cédric Marchand, Lilian Bossuet, Ugo Mureddu, Nathalie Bochard, Abdelkarim Cherkaoui, Viktor Fischer. "**Implementation and characterization of a physical unclonable function for IoT : a case study with the TERO-PUF**". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, Vol.37, Jan 2018.

Conférences internationales avec comité de lecture et actes

3 Ugo Mureddu, Brice Colombier, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**Transient Effect Ring Oscillators Leak Too**". *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Miami, Florida, U.S.A, Jul 2019.

4 Brice Colombier, Ugo Mureddu, Lilian Bossuet, David Hely. "**A comprehensive hardware/software infrastructure for IP cores design protection**". *IEEE International Conference on Field-Programmable Technology (ICFPT)*, Melbourne, Australia, Dec 2017.

5 Ugo Mureddu, Oto Petura, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**Efficient design of Oscillator based Physical Unclonable Functions on Flash FPGAs**". *IEEE 2nd International Verification and Security Workshop (IVSW)*, Thessaloniki, Greece, Jul 2017.

6 Oto Petura, Ugo Mureddu, Nathalie Bochard, Viktor Fischer. "**Optimization of the PLL based TRNG design using the genetic algorithm**". *IEEE International Symposium*

on Circuits and Systems (ISCAS), Baltimore, MD, USA, May 2017.

7 Oto Petura, Ugo Mureddu, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices**". *26th International Conference on Field Programmable Logic and Applications (FPL)*, Lausanne, Switzerland, Aug 2016.

Présentations à des congrès internationaux sans actes

8 Ugo Mureddu, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**Impact of Frequency Locking on Ring Oscillating Cells in FPGA**". *Workshop on Practical Hardware Innovation in Security and Characterisation (PHSIC 2018)*, Gardanne, France, May 2018.

9 Oto Petura, Ugo Mureddu, Nathalie Bochard, Lilian Bossuet, Viktor Fischer. "**Evaluation of AIS-20/31 compliant TRNG cores implemented on FPGAs**". *Workshop on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE)*, Barcelone, Spain, Nov 2016.

Démonstrations

10 Brice Colombier, Ugo Mureddu, Marek Laban, Oto Petura, Lilian Bossuet, Viktor Fischer. "**Hardware Demo : Complete Activation Scheme for IP Design Protection**". *International Symposium on Hardware Oriented Security and Trust (HOST)*, McLean, VA, USA, May 2017.

11 Brice Colombier, Ugo Mureddu, Marek Laban, Oto Petura, Lilian Bossuet, Viktor Fischer. "**Hardware Demo : Complete Activation Scheme for IP Design Protection**". *27th International Conference on Field Programmable Logic and Applications (FPL)*, Ghent, Belgium, Sep 2017.

Séminaire

12 Ugo Mureddu, Lilian Bossuet, Viktor Fischer. "**Étude des cellules oscillantes pour la génération d'aléa dans les circuits électroniques numériques**". *Séminaire sécurité des systèmes électroniques embarqués*, Rennes, France, Dec 2018.

Posters

13 Ugo Mureddu, Lilian Bossuet, Viktor Fischer. "**Sécurité des Fonctions Physiques non-clonables**". *Journée de la recherche de l'école doctorale EDSIS*, Saint-Étienne, France, Jun 2017.

14 Ugo Mureddu, Lilian Bossuet, Viktor Fischer. "**A comparison of PUF cores suitable for FPGA devices**". *Workshop on Trustworthy Manufacturing and Utilization of Secure Devices (TRUDEVICE)*, Barcelone, Spain, Nov 2016.

15 Ugo Mureddu, Lilian Bossuet, Viktor Fischer. "**A comparison of PUF cores suitable for FPGA devices**". *Cryptarchi Workshop*, La Grande Motte, France, Jun 2016.

16 Ugo Mureddu, Lilian Bossuet, Viktor Fischer. "**A comparison of PUF cores suitable for FPGA devices**". *Colloque national du GDR SoC/SiP*, Nantes, France, Jun 2016.

Bibliographie

[1] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom, "Spectre attacks : Exploiting speculative execution," in *40th IEEE Symposium on Security and Privacy (S&P'19)*, 2019.

[2] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg, "Meltdown : Reading kernel memory from user space," in *27th USENIX Security Symposium (USENIX Security 18)*, 2018.

[3] R. Tsang, "Cyberthreats, vulnerabilities and attacks on SCADA networks," *University of California, Berkeley, Working Paper, http ://gspp. berkeley. edu/iths/Tsang_ SCADA% 20Attacks. pdf (as of Dec. 28, 2011)*, 2010.

[4] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society*, pp. 4490–4494, Nov 2011.

[5] R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld, "Physical one-way functions," *Science*, vol. 297, no. 5589, pp. 2026–2030, 2002.

[6] J. Lee, D. Lim, B. Gassend, G. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *VLSI Circuits.*, pp. 176–179, June 2004.

[7] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Transactions on Computer*, vol. 58, no. 9, pp. 1198–1210, 2009.

[8] NIST, "FIPS 140-1 : Security requirements for cryptographic modules," 1994.

[9] NIST, "Special publication 800-22 : A statistical test suite for random and pseudorandom number generators for cryptographic applications," 2010.

[10] G. Marsaglia, "DIEHARD : Battery of tests of randomness.," 1996.

[11] R. Brown, "DIEHARDER : A random number test suite," 2015.

[12] A. Rényi, "On measures of entropy and information," in *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability, pages 547-561*, 1960.

[13] W. Killmann and W. Schindler, "A proposal for : Functionality classes for random number generators, version 2.0," BSI, Germany, 2011.

[14] M. S. Turan, E. Barker, J. Kelsey, K. A. McKay, M. L. Baish, and M. Boyle, "Special publication 800-90 B : Recommendation for the entropy sources used for random bit generation," NIST, Jan. 2018.

[15] V. Fischer, *A Closer Look at Security in Random Number Generators Design*, pp. 167–182. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012.

[16] A. Maiti, J. Casarona, L. McHale, and P. Schaumont, "A large scale characterization of RO-PUF," in *Proceedings of the 2010 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, (Anaheim Convention Center, California, USA), pp. 94–99, 13-14 June 2010.

[17] Y. Su, J. Holleman, and B. Otis, "A digital 1.6 pJ/bit chip identification circuit using process variations," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 69–77, Jan 2008.

[18] Y. Hori, A. Satoh, T. Katashita, and T. Yoshida, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs," in *International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, vol. 00, pp. 298–303, 12 2010.

[19] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Testing techniques for hardware security," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 2, pp. 5 :1–5 :33, Mar. 2009.

[20] A. Maiti, V. Gunreddy, and P. Schaumont, *A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions*, pp. 245–267. New York, NY : Springer New York, 2013.

[21] B. B. Linus Feiten, Matthias Sauer, "On metrics to quantify the inter-device uniqueness of PUFs." Cryptology ePrint Archive, Report 2016/320, 2016.

[22] C. Böhm and M. Hofer, *Physical unclonable functions in theory and practice*. Springer Science & Business Media, 2012.

[23] F. Wilde, B. M. Gammel, and M. Pehl, "Spatial correlation analysis on physical unclonable functions," *IEEE Transactions on Information Forensics and Security*, vol. PP, no. 99, pp. 1–1, 2018.

[24] M. Pehl, M. Hiller, and H. Graeb, "Efficient evaluation of physical unclonable functions using entropy measures," *Journal of Circuits, Systems and Computers*, vol. 25, no. 01, p. 1640001, 2016.

[25] V. van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls, "Hardware intrinsic security from D flip-flops," in *Proceedings of the Fifth ACM Workshop on Scalable Trusted Computing*, STC '10, (New York, NY, USA), pp. 53–62, ACM, 2010.

[26] A. Cherkaoui, L. Bossuet, and C. Marchand, "Design, evaluation, and optimization of physical unclonable functions based on transient effect ring oscillators," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 1291–1305, June 2016.

[27] P. L'Ecuyer, "History of uniform random number generation," in *Proceedings of the 2017 Winter Simulation Conference* (G. Z. N. M. G. W. W. K. V. Chan, A. D'Ambrogio and e. E. Page, eds.), 2017.

[28] F. Galton, "Dice for statistical experiments," in *Nature*, 1890.

[29] L. M. Reyneri, D. D. Corso, and B. Sacco, "Oscillatory metastability in homogeneous and inhomogeneous flip-flops," *IEEE Journal of Solid-State Circuits*, vol. 25, pp. 254–264, Feb 1990.

[30] D. Muller and W. Bartky, "A theory of asynchronous circuits," *Proc. Int'l Symp. Theory of Switching, Part 1, Harvard Univ. Press*, pp. 204–243, 1959.

[31] R. Fairfield, R. Mortenson, and K. Coulthart, "An LSI random number generator (RNG)," in *Advances in Cryptology* (G. R. Blakley and D. Chaum, eds.), vol. 196, (Berlin, Heidelberg), pp. 203–230, Springer Berlin Heidelberg, 1985.

[32] V. Fischer and M. Drutarovský, "True random number generator embedded in reconfigurable hardware," in *Cryptographic Hardware and Embedded Systems - CHES* (B. S. Kaliski, ç. K. Koç, and C. Paar, eds.), (Berlin, Heidelberg), pp. 415–430, Springer Berlin Heidelberg, 2002.

[33] M. Simka, M. Drutarovský, and V. Fischer, "Testing of PLL-based True Random Number Generator in ChangingWorking Conditions," *RADIOENGINEERING,*, vol. 20, pp. 94–101, Apr. 2011.

[34] P. Kohlbrenner and K. Gaj, "An embedded true random number generator for FPGAs," in *Proceedings of the 2004 ACM/SIGDA 12th International Symposium on Field Programmable Gate Arrays*, FPGA '04, (New York, NY, USA), pp. 71–78, ACM, 2004.

[35] B. Sunar, W. J. Martin, and D. R. Stinson, "A provably secure true random number generator with built-in tolerance to active attacks," *IEEE Transactions on Computer*, vol. 56, pp. 109–119, Jan. 2007.

[36] K. Wold and C. H. Tan, "Analysis and enhancement of random number generator in FPGA based on oscillator rings," in *Proceedings of the 2008 International Conference on Recon-*

figurable Computing and FPGAs, RECONFIG '08, (Washington, DC, USA), pp. 385–390, IEEE Computer Society, 2008.

[37] M. Baudet, D. Lubicz, J. Micolod, and A. Tassiaux, "On the security of oscillator-based random number generators," *Journal of Cryptology*, vol. 24, no. 2, pp. 398–425, 2011.

[38] A. Cherkaoui, V. Fischer, A. Aubert, and L. Fesquet, "A self-timed ring based true random number generator," in *19th IEEE International Symposium on Asynchronous Circuits and Systems, ASYNC*, (Santa Monica, CA, USA), pp. 99–106, May 19-22 2013.

[39] E. Böhl, M. Lewis, and S. Galkin, "A true random number generator with on-line testability," in *19th IEEE European Test Symposium (ETS)*, pp. 1–6, May 2014.

[40] M. Varchola and M. Drutarovsky, "New high entropy element for FPGA based true random number generators," *Workshop on Cryptographic Hardware and Embedded Systems CHES*, pp. 351–365, 2010.

[41] M. Epstein, L. Hars, R. Krasinski, M. Rosner, and H. Zheng, "Design and implementation of a true random number generator based on digital circuit artifacts," in *Cryptographic Hardware and Embedded Systems - CHES 2003* (C. D. Walter, Ç. K. Koç, and C. Paar, eds.), (Berlin, Heidelberg), pp. 152–165, Springer Berlin Heidelberg, 2003.

[42] I. Vasyltsov, E. Hambardzumyan, Y.-S. Kim, and B. Karpinskyy, "Fast digital trng based on metastable ring oscillator," in *Cryptographic Hardware and Embedded Systems – CHES* (E. Oswald and P. Rohatgi, eds.), (Berlin, Heidelberg), pp. 164–180, Springer Berlin Heidelberg, 2008.

[43] M. Majzoobi, F. Koushanfar, and S. Devadas, "Fpga-based true random number generation using circuit metastability with adaptive feedback control," in *Proceedings of the 13th International Conference on Cryptographic Hardware and Embedded Systems*, CHES'11, (Berlin, Heidelberg), pp. 17–32, Springer-Verlag, 2011.

[44] J. Danger, S. Guilley, and P. Hoogvorst, "Fast true random generator in fpgas," in *2007 IEEE Northeast Workshop on Circuits and Systems*, pp. 506–509, Aug 2007.

[45] J.-L. Danger, S. Guilley, and P. Hoogvorst, "High speed true random number generator based on open loop structures in fpgas," *Microelectronics Journal*, vol. 40, no. 11, pp. 1650 – 1656, 2009. International Conference on Microelectronics Digital and Mixed-Signal Circuits and Systems.

[46] J. D. Golic, "New paradigms for digital generation and post-processing of random data.," *IACR Cryptology ePrint Archive*, vol. 2004, p. 254, 01 2004.

[47] J. D. Golic, "New methods for digital generation and postprocessing of random data," *IEEE Transactions on Computer*, vol. 55, pp. 1217–1229, Oct. 2006.

[48] M. Dichtl and J. D. Golić, "High-speed true random number generation with logic gates only," in *Cryptographic Hardware and Embedded Systems - CHES 2007* (P. Paillier and I. Verbauwhede, eds.), (Berlin, Heidelberg), pp. 45–62, Springer Berlin Heidelberg, 2007.

[49] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, (New York, NY, USA), pp. 148–160, ACM, 2002.

[50] B. Gassend, D. Lim, D. E. Clarke, M. van Dijk, and S. Devadas, "Identification and authentication of integrated circuits," *Concurrency - Practice and Experience*, vol. 16, no. 11, pp. 1077–1098, 2004.

[51] Z. Cherif, J. Danger, S. Guilley, and L. Bossuet, "An easy-to-design PUF based on a single oscillator : the loop PUF.," in *15th Euromicro Conference on Digital System Design(DSD)*, (Cesme, Izmir, Turkey), Sep. 2012.

[52] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proceedings of the 44th Design Automation Conference, DAC*, (San Diego, CA, USA), pp. 9–14, June 4-8 2007.

[53] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, pp. 1200–1205, Oct 2005.

[54] E. Ozturk, G. Hammouri, and B. Sunar, "Physical unclonable function with tristate buffers," in *2008 IEEE International Symposium on Circuits and Systems*, pp. 3194–3197, May 2008.

[55] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, "Fpga intrinsic pufs and their use for ip protection," in *Cryptographic Hardware and Embedded Systems - CHES 2007* (P. Paillier and I. Verbauwhede, eds.), (Berlin, Heidelberg), pp. 63–80, Springer Berlin Heidelberg, 2007.

[56] S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "Extended abstract : The butterfly PUF protecting IP on every FPGA," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 67–70, June 2008.

[57] R. Maes, P. Tuyls, and I. Verbauwhede, "Intrinsic PUFs from flip-flops on reconfigurable devices," in *3rd Benelux workshop on information and system security*, vol. 17, 2008.

[58] B. Habib, J.-P. Kaps, and K. Gaj, *Efficient SR-Latch PUF*, pp. 205–216. Springer International Publishing, 2015.

[59] L. Bossuet, X. T. Ngo, Z. Cherif, and V. Fischer, "A PUF based on a transient effect ring oscillator and insensitive to locking phenomenon," *IEEE Transactions on Emerging Topics in Computing*, vol. 2, no. 1, pp. 30–36, 2014.

[60] G. Taylor and G. Cox, "Behind Intel's new random-number generator," *IEEE Spectrum*, vol. 24, 2011.

[61] M. Bucci, L. Giancane, R. Luzzi, M. Varanonuovo, and A. Trifiletti, "A novel concept for stateless random bit generators in cryptographic applications," in *Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on*, pp. 4–pp, IEEE, 2006.

[62] M. Pelgrom, A. C. Duinmaijer, and A. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1433–1439, Oct 1989.

[63] A. Hastings, "The art of analog layout," *Upper Saddle River, NJ*, 2001.

[64] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "PUFs : Myth, fact or busted ? a security evaluation of physically unclonable functions (pufs) cast in silicon," in *Cryptographic Hardware and Embedded Systems – CHES* (E. Prouff and P. Schaumont, eds.), vol. 7428 of *Lecture Notes in Computer Science*, pp. 283–301, Springer Berlin Heidelberg, 2012.

[65] J. Delvaux, *Security Analysis of PUF-Based Key Generation and Entity Authentication*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 2017.

[66] C. Helfmeier, C. Boit, D. Nedospasov, and J.-P. Seifert, "Cloning physically unclonable functions," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 1–6, June 2013.

[67] Y. Oren, A.-R. Sadeghi, and C. Wachsmann, "On the effectiveness of the remanence decay side-channel to clone memory-based PUFs," in *Cryptographic Hardware and Embedded Systems - CHES* (G. Bertoni and J.-S. Coron, eds.), (Berlin, Heidelberg), pp. 107–125, Springer Berlin Heidelberg, 2013.

[68] S. Morozov, A. Maiti, and P. Schaumont, "An analysis of delay based PUF implementations on FPGA," in *Reconfigurable Computing : Architectures, Tools and Applications : 6th International Symposium, ARC* (P. Sirisuk, F. Morgan, T. El-Ghazawi, and H. Amano, eds.), (Bangkok, Thailand), pp. 382–387, Springer Berlin Heidelberg, Mar. 2010.

[69] A. Wild, G. T. Becker, and T. Güneysu, "A fair and comprehensive large-scale analysis of oscillation-based PUFs for FPGAs," in *Field Programmable Logic and Applications (FPL), 2017 27th International Conference on*, pp. 1–7, IEEE, 2017.

[70] A. Maiti and P. Schaumont, "Improved ring oscillator PUF : an FPGA-friendly secure primitive," *Journal of Cryptology*, vol. 24, no. 2, pp. 375–397, 2011.

[71] F. Bernard, V. Fischer, C. Costea, and R. Fouquet, "Implementation of ring-oscillators-based physical unclonable functions with independent bits in the response," *International Journal of Reconfigurable Computing*, vol. 2012, pp. 13 :13–13 :13, Jan. 2012.

[72] C. Marchand, L. Bossuet, and A. Cherkaoui, "Design and characterization of the TERO-PUF on SRAM FPGAs," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 134–139, July 2016.

[73] A. Wild, G. T. Becker, and T. Güneysu, "On the problems of realizing reliable and efficient ring oscillator PUFs on FPGAs," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 103–108, May 2016.

[74] A. A. Abidi, "Phase noise and jitter in CMOS ring oscillators," *IEEE Journal of Solid-State Circuits*, vol. 41, pp. 1803–1816, Aug 2006.

[75] M. Mandal and B. C. Sarkar, "Ring oscillators : Characteristics and applications," *Indian Journal of Pure and Applied Physics*, vol. 48, pp. 136–145, 02 2010.

[76] B. Mesgarzadeh and A. Alvandpour, "A study of injection locking in ring oscillators," in *2005 IEEE International Symposium on Circuits and Systems*, pp. 5465–5468 Vol. 6, May 2005.

[77] R. J. Betancourt-Zamora, S. Verma, and T. H. Lee, "1-GHz and 2.8-GHz CMOS injection-locked ring oscillator prescalers," in *2001 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No.01CH37185)*, pp. 47–50, June 2001.

[78] D. G. Tucker, "Forced oscillations in oscillator circuits, and the synchronization of oscillators," *Electrical Engineers - Part III : Radio and Communication Engineering, Journal of the Institution of*, vol. 92, pp. 226–234, September 1945.

[79] R. Adler, "A study of locking phenomena in oscillators," *Proceedings of the IRE*, vol. 34, pp. 351–357, June 1946.

[80] R. D. Huntoon and A. Weiss, "Synchronization of oscillators," *Journal of Research of the National Bureau of Standards*, vol. Volume 38, pp. 397–410, April 1947.

[81] A. Mirzaei, M. E. Heidari, R. Bagheri, and A. A. Abidi, "Multi-phase injection widens lock range of ring-oscillator-based frequency dividers," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 656–671, March 2008.

[82] J. C. Chien and L. H. Lu, "Analysis and design of wideband injection-locked ring oscillators with multiple-input injection," *IEEE Journal of Solid-State Circuits*, vol. 42, pp. 1906–1915, Sept 2007.

[83] A. T. Markettos and S. W. Moore, "The frequency injection attack on ring-oscillator-based true random number generators," *Workshop on Cryptographic Hardware and Embedded Systems CHES*, pp. 317–331, 2009.

[84] N. Bochard, F. Bernard, V. Fischer, and B. Valtchanov, "True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators," *International Journal of Reconfigurable Computing*, vol. 2010, p. ID 879281, Dec. 2010. 12 pages.

[85] P. Bayon, L. Bossuet, A. Aubert, V. Fischer, F. Poucheret, B. Robisson, and P. Maurine, *Contactless Electromagnetic Active Attack on Ring Oscillator Based True Random Number Generator*, pp. 151–166. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012.

[86] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Side-channel analysis of PUFs and fuzzy extractors," in *Trust and Trustworthy Computing* (J. McCune, B. Balacheff, A. Perrig, A.-R. Sadeghi, A. Sasse, and Y. Beres, eds.), vol. 6740 of *Lecture Notes in Computer Science*, pp. 33–47, Springer Berlin Heidelberg, 2011.

[87] D. Merli, D. Schuster, F. Stumpf, and G. Sigl, "Semi-invasive EM attack on FPGA RO PUFs and countermeasures," in *Proceedings of the Workshop on Embedded Systems Security*, WESS '11, (New York, NY, USA), pp. 2 :1–2 :9, ACM, 2011.

[88] D. Merli, J. Heyszl, B. Heinz, D. Schuster, F. Stumpf, and G. Sigl, "Localized electromagnetic analysis of RO PUFs," in *2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 19–24, June 2013.

[89] P. Bayon, L. Bossuet, A. Aubert, and V. Fischer, "Electromagnetic analysis on ring oscillator-based true random number generators," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pp. 1954–1957, May 2013.

[90] C. Marchand, L. Bossuet, U. Mureddu, N. Bochard, A. Cherkaoui, and V. Fischer, "Implementation and characterization of a physical unclonable function for IoT : A case study with the TERO-PUF," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, pp. 97–109, Jan 2018.

[91] F. Bernard, P. Haddad, V. Fischer, and J. Nicolai, "From physical to stochastic modeling of a TERO-based TRNG," *Journal of Cryptology*, Mar 2018.

[92] M. Laban, M. Drutarovsky, V. Fischer, and M. Varchola, "Modular evaluation platform for evaluation and testing of physically unclonable functions," in *28th International Conference Radioelektronika*, pp. 1–6, April 2018.

[93] C. Dobraunig, M. Eichlseder, F. Mendel, and M. Schläffer, "Ascon v1.2, submission to the caesar competition.," tech. rep., Institute for Applied Information Processing and Communications Graz University of Technology, Sept. 2016.

[94] B. Colombier, *Methods for protecting intellectual property of IP cores designers.* Theses, Université Jean Monnet - Saint-Etienne, Oct. 2017.

[95] P. Haddad, V. Fischer, F. Bernard, and J. Nicolai, "A physical approach for stochastic modeling of TERO-based TRNG," in *Cryptographic Hardware and Embedded Systems - CHES* (T. Güneysu and H. Handschuh, eds.), vol. 9293 of *Lecture Notes in Computer Science*, pp. 357–372, Springer Berlin Heidelberg, 2015.

[96] T. Chawla, *Etude de l'impact des variations du procédé de fabrication sur les circuits numériques.* Phd thesis, Télécom ParisTech, 30 Septembre 2010.

[97] A. R. Brown, N. M. Idris, J. R. Watling, and A. Asenov, "Impact of metal gate granularity on threshold voltage variability : A full-scale three-dimensional statistical simulation study," *IEEE Electron Device Letters*, vol. 31, pp. 1199–1201, Nov 2010.

[98] N. Bochard, C. Marchand, O. Petura, L. Bossuet, and V. Fischer, "Evariste III : A new multi-FPGA system for fair benchmarking of hardware dependent cryptographic primitives," in *Workshop on Cryptographic Hardware and Embedded Systems CHES*, Sep 2015. Poster.

[99] J. Hamon and L. Fesquet, "Robust and programmable Self-Timed Ring oscillators," in *IEEE 9th International New Circuits and Systems Conference (NEWCAS)*, pp. 249–252, 2011.

[100] C. Wittke, Z. Dyka, O. Skibitzki, and P. Langendoerfer, "Preparation of SCA attacks : Successfully decapsulating BGA packages," in *Innovative Security Solutions for Information Technology and Communications* (I. Bica and R. Reyhanitabar, eds.), pp. 240–247, Springer International Publishing, 2016.

Table des figures

Liste des tableaux