Année 2021

Code de L'UE : HMEE 215

STAGE MASTER 1 ÉLECTRONIQUE, ENERGIE ELECTRIQUE, AUTOMATIQUE

Montpellier le, 06/07/2021

Réalisé par :
Yacine MEHADJI

Domaine :
Microélectronique

Spécialité :
Systèmes Électroniques Intégrés et Embarqués

Maitre de stage :
Arnaud VIRAZEL
Maître de conférences à l'Université de Montpellier

<u>Sujet :</u> Mise en place d'un démonstrateur d'architecture de tolérance aux fautes hybride

L'évolution technologique consiste à réduire la taille des composants électroniques, cependant les systèmes intégrés doivent faire face aux problèmes de fiabilité. Ce rapport de stage présente une technique pour améliorer la fiabilité et garantir un fonctionnement correct d'un système numérique malgré la présence des fautes.

1 Présentation de l'organisme d'accueil - LIRMM

Le laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier- LIRMM est une unité mixte de recherche, dépendant conjointement de L'Université de Montpellier et du Centre National de la Recherche Scientifique - CNRS. Il est situé sur le campus Saint Priest de L'Université de Montpellier.

Ses activités de recherche positionnent pleinement le LIRMM au cœur des sciences et technologies de l'information, de la communication et des systèmes.

Ainsi, de l'information aux systèmes, de la technologie à l'humain et aux usages, les activités de recherche du LIRMM concernent : la conception et la vérification de systèmes intégrés, mobiles, communicant, la modélisation de systèmes complexes à base d'agents, les études en algorithmique, bio-informatique, interaction homme-machine, robotique, etc.

Ce stage est mené dans le département Microélectronique, un département scientifique de recherche, organisé en équipes. Le département Microélectronique du LIRMM est spécialisé dans la recherche de solutions innovantes pour embarquer, dans les systèmes électroniques intégrés, toujours plus d'intelligence et de technologie émergentes afin d'améliorer la qualité, la fiabilité, l'adaptabilité, l'efficacité (notamment énergétique) et la sécurité de ces systèmes. La plupart de ces activités trouvent des applications dans le domaine large des objets communicants pour l'environnement, y compris les environnements difficiles (spatial, radiatif, haute température) et le vivant.

D'autre travaux sont menés dans deux autres départements scientifiques de recherche, eux-mêmes organisés en équipes.

Les thématiques du département Informatique s'étendent des frontières des mathématiques à la recherche appliquée : algorithme des graphes, bio-informatique, cryptographie, réseaux, bases de données et systèmes d'information (intégration de données, fouille de données, maintien de la cohérence), génie logiciel (langage de programmation, objets, composants, modèles), intelligence artificielle (apprentissage, contraintes, représentation des connaissances, systèmes multi-agents), interaction homme-machine (langage naturel, visualisation, web sémantique et e-learning).

Enfin, le département Robotique développe des nouveaux systèmes robotique et des outils fondamentaux associés avec pour objectif de les amener jusqu'à la valorisation et le transfert industriel. Cette politique scientifique s'inscrit également dans une démarche qui vise à répondre, pour un certain nombre de travaux, à des problèmes sociétaux, économiques et environnementaux. Le département mène ainsi des activités de recherche appliquées à l'industrie manufacturière, la santé, l'environnement quotidien.

Les recherches menées au LIRMM trouvent généralement une finalisation dans des domaines applicatifs aussi divers que la biologie, la chimie, les télécommunications, la santé, l'environnement… et dans les domaines propres du laboratoire : l'informatique, l'électronique et l'automatique.

Les missions du LIRMM est donc de produire :

- Des connaissances (en moyen 300 publications d'audience internationale chaque année).
- Des chercheurs directement (docteurs, post doc) ou indirectement (participation LMD).
- Des objets matériels et/ou logiciels prototypes.
- De l'activité économique : partenariat industriels, création d'entreprises innovantes.
- De l'animation scientifique à l'échelle nationale mais aussi internationale.

2 Introduction

Les dispositifs sont devenus de plus en plus sensibles à l'impact des particules de haute énergie. Il y a de fortes probabilités qu'elles puissent causer des perturbations isolées (Single Event Upset – SEU) en frappant la surface du circuit silicium. Cela peut entrainer des erreurs transitoires (soft errors) qui se manifestent comme un bruit dans la logique combinatoire appelées (Single Event Transient – SET) ou des inversions de bits dans la mémoire c'est-à-dire dans la partie séquentielle du circuit qui sont appelées SEU. Elles peuvent être déclenchées par des facteurs environnementaux tels que les décharges statiques ou les fluctuations de températures et de tension d'alimentations. L'évolution de la technologie permet de réaliser des systèmes de plus en plus complexes sur une seule puce, cependant leur sensibilité aux effets de l'environnement s'est accrue, parce que les fréquences d'horloges augmentent alors que la taille des blocs fonctionnels diminue.

Les approches conventionnelles pour la conception d'un dispositif fiable exploitent une architecture hybride tolérante aux fautes avec quatre types de redondances pour améliorer la robustesse des circuits et systèmes numériques.

3 Problématique de la fiabilité dans les circuits intégrés

3.1 Notion de fiabilité

La loi empirique de Gordon Moore « cofondateur d'Intel » prédit que la densité d'intégration des circuits intégrés double tous les 18 mois. Cependant cette intégration croissante qui permet de réaliser des systèmes intégrés sur une seule puce, système SOC (Système on chip) a un impact négatif sur la fiabilité des circuits. En effet, à cause de la miniaturisation croissante des procédés de fabrication « 90nm, 65nm, 45nm, … » il est de plus en plus difficile de réaliser un circuit intégré sans aucun défaut de fabrication.

La fiabilité est un attribut de la sûreté de fonctionnement et correspond à la probabilité qu'un système accomplisse la fonction par laquelle il a été conçu. Comme elle signifie la capacité d'un système à accomplir une tâche dans des conditions défavorables pendant une durée déterminée et dans une plage spécifiée cela concerne donc la continuité du service assuré par ce système spécifié dans son cahier de charges.

L'objectif des fiabilistes est de limiter au minimum le taux de défaillances dans les circuits intégrés, car elles sont liées généralement aux défauts des procédés de fabrication et de conception et aux variabilités de tension et de la température. La deuxième catégorie de défaillance est essentiellement due au vieillissement donc aux conditions d'utilisation et la durée de vie du dispositif. Les circuits intégrés sont devenus aussi plus sensibles aux effets temporaires dus à des influences électromagnétiques au rayonnement de particules alpha ou aux radiations cosmiques, cette catégorie de défaillance est provoquée par les interférences dues à l'environnement.

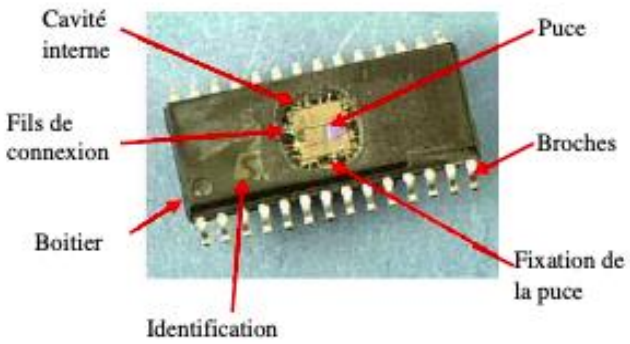

Figure 1 : Identification des défaillances dans un circuit intégré qui sont dues aux défauts de fabrication

Figure 2 : Défauts dus aux phénomènes d'électromigration (augmentation de température)

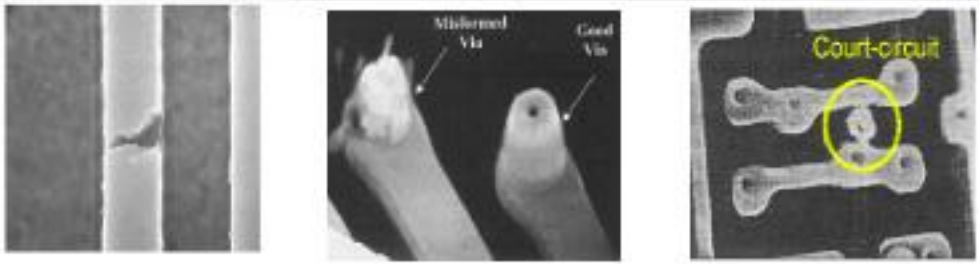

Figure 3 : Défauts de fabrication

3.2 Méthodologies conceptuelles pour améliorer la fiabilité

Le développement d'un circuit fiable passe par l'utilisation d'un ensemble de méthodologies conceptuelles et structurelles qui peuvent être classées ainsi :

- Prévention de fautes : comment empêcher l'occurrence ou l'introduction de fautes lors de la fabrication du circuit. Elle comprend toute technique qui tente de prévenir l'apparition de fautes. Il peut s'agir de vérification du design, de l'inspection de composants, du test ou d'autres méthodes de contrôle de la qualité.
- Tolérance aux fautes : comment assurer par redondance la continuité du service conduisant à la fiabilité du circuit en dépit de fautes qu'il s'agisse de dégradation physique du matériel, de défauts logiciels, d'attaques malveillants, d'erreurs d'interaction homme-machine.
- Élimination de fautes : comment réduire la présence (nombre, sévérité) des fautes, elle peut être effectuée durant les processus de maintenance corrective ou préventive. La maintenance corrective vise à éliminer les fautes qui ont déjà produit une erreur et commence après la détection de l'erreur, tandis que l'entretien préventif est destiné à éliminer les fautes avant qu'elles ne puissent causer des erreurs.
- Prévision de fautes : comment estimer la présence, le taux futur et les possibles conséquences des défauts. Elle est réalisée en effectuant une évaluation du comportement du système à l'égard de la survenance ou de l'activation de fautes. Elle a deux aspects qui sont d'ordre qualitatif et quantitatif. Les principales approches probabilistes de prévision de défaillances visant à dériver des estimations probabilistes sont la modélisation et le test.

4 Entraves de la sûreté de fonctionnement

Les entraves à la sureté de fonctionnement se décomposent en 3 classes : les fautes (fault), les erreurs (errors), et les défaillances (failure). Une faute est une déviation d'au moins un élément du système ou d'une de ses propriétés caractéristiques. Une faute est la cause adjugée ou supposé d'une erreur. Une erreur est la partie de l'état du système qui est susceptible d'entrainer une défaillance. Enfin, une défaillance est une déviation du service rendu par le système par rapport à son service nominal. Un exemple est décrit sur la figure 4 :

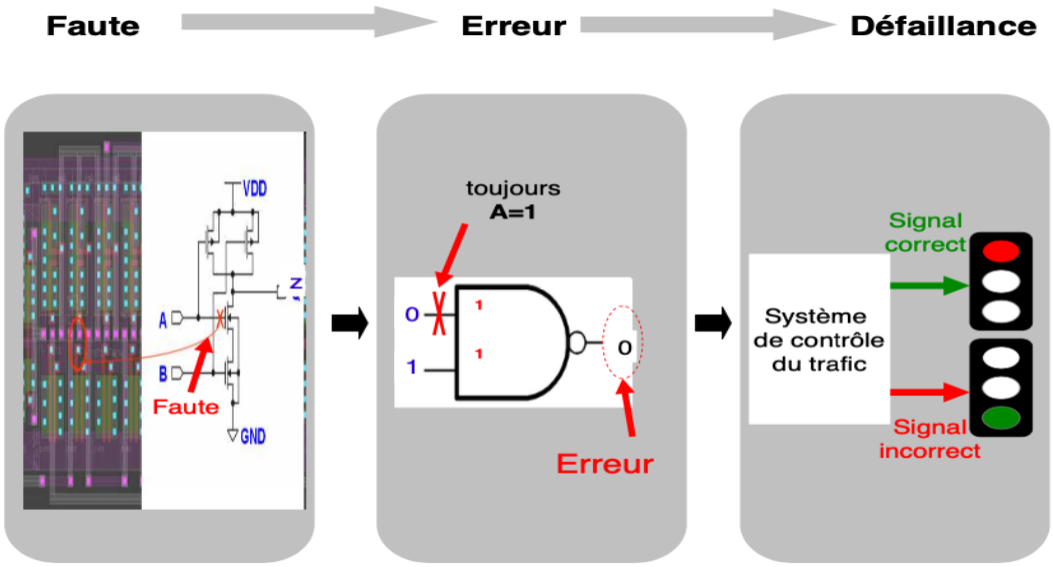

Figure 4 : Une seule faute a causé la défaillance du système de contrôle du trafic

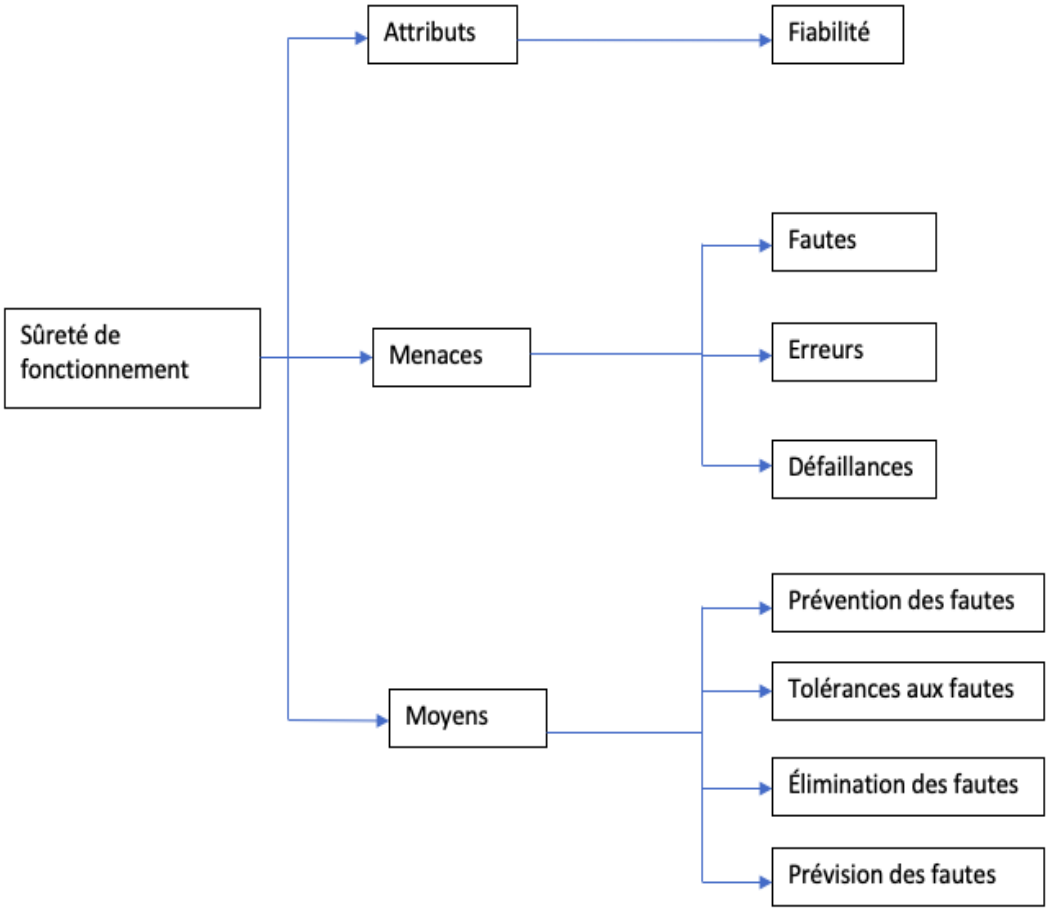

Figure 5 : Arbre de la sûreté de fonctionnement

4.1 Classification temporelle des fautes

La durée d'une faute est une dimension importante qui permet de distinguer classiquement trois types principaux de fautes soudaines : les fautes permanentes qui ont un effet permanent sur le système, les fautes transitoires de durée limitée dans le temps et les fautes intermittentes qui correspond à des fautes transitoires répétitives.

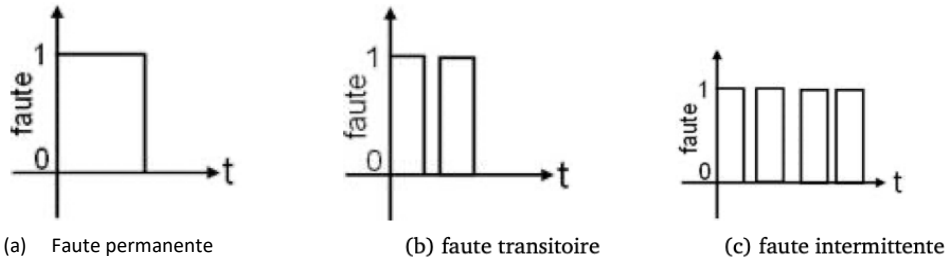

Figure 6 : Caractérisation d'une faute selon sa durée

Une faute dans un circuit intégré peut être classé en fonction de ses caractéristique temporelle, comme elle peut être classée sur la base des techniques pour la tolérer.

Le terme employé pour désigner un effet mesurable résultant du dépôt d'énergie par l'impact d'une unique particule ionisante est SEE (Single Event Effect).

Single Event Upset (SEU) : Le SEU est la plupart du temps une erreur temporaire causée par le signal transitoire induit par l'impact d'une particule isolée de haute énergie. Il se produit lorsqu'un rayonnement provoque une perturbation de charge suffisamment importante pour inverser l'état d'une cellule mémoire, registre, verrou, ou flip-flop. L'erreur est dite « soft » car le circuit n'est pas endommagé de façon permanente par le rayonnement, et lorsqu'une nouvelle donnée est écrite dans la cellule mémoire touchée, le dispositif va le stocker correctement. Le SEU est un problème très grave car il est l'une des principales sources de défaillance dans les systèmes numériques, il constituera probablement une menace sérieuse pour l'avenir du calcul robuste et exige une attention sérieuse, il peut se manifester comme SBU (Single Bit Upset) ou MBU (Multi Bit Upset).

Single Bit Upset (SBU) et Multi Bit Upset (MBU)

Le SBU est un évènement unique dû à un rayonnement qui se traduit par une inversion de bit alors qu'un MBU est un évènement unique dû à un rayonnement qui résulte en ce que plus d'un bit soient inversés. Les SBU et MBU sont donc un sous ensemble de SEU.

Single Event Transient (SET)

Le SET est une impulsion transitoire dans le chemin logique d'un circuit intégré. Semblable à un SEU, il est induit par un dépôt de charge d'une particule ionisante isolée. Un SET peut être propagé le long du chemin logique où il a été créé. Il peut être verrouillé dans un registre, un verrou ou une flip-flop, provoquant le changement de leur état de sortie.

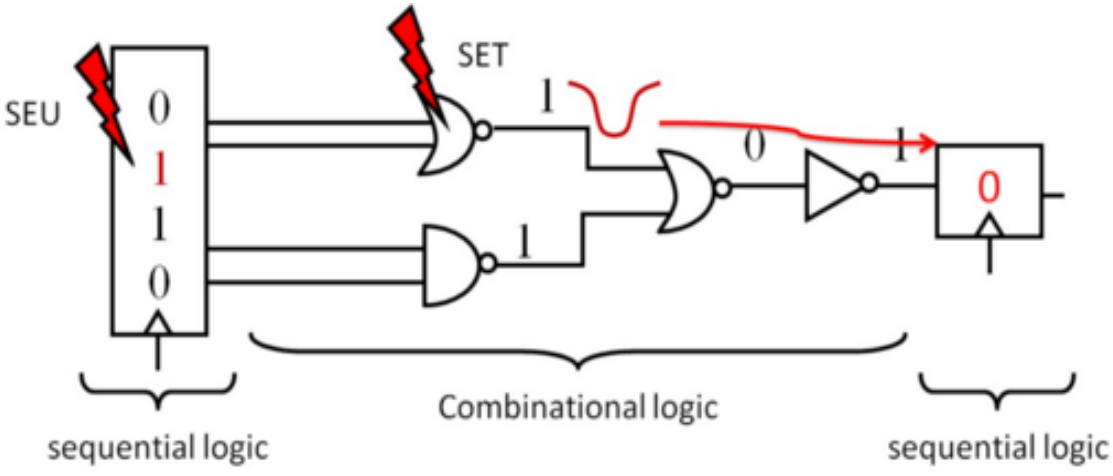

Figure 7 : Les occurrences d'erreurs dans la logique combinatoire et les éléments de stockage

4.2 Classification des erreurs

Nous classons les erreurs sur la base de leur défaut, car cette classification constitue la base des techniques de tolérances.

4.2.1 Les erreurs logicielles

Les erreurs logicielles dites « soft » sont définies comme des évènements dans lesquels les données sont corrompues, mais le dispositif lui-même n'est pas endommagé de façon permanente. Les erreurs logicielles sont un sous ensemble des SEE (Single event effect) qui sont causées par des neutrons à haute énergie provenant des rayons cosmiques ou des particules alpha générées par des impuretés dans les matériaux d'emballage. Afin d'éviter toute ambiguïté, nous utilisons l'interprétation SEU (Single Event Upset) pour designer qu'il y a des erreurs dans les mémoires causées par le transitoire de tension qui se propage à travers le bloc CL depuis le point d'évènement unique jusqu'au l'élément de stockage. Nous utilisons également la terminologie SET (Single Event Transient) pour désigner les impulsions de tensions transitoires induites par un évènement unique générées dans les blocs CLs.

4.2.2 Les erreurs matérielles

Les erreurs matérielles dites « Hard » sont causées par des défauts permanents du silicium, qui sont dues soit aux défauts de fabrications, soit au vieillissement du dispositif.

4.2.3 Les erreurs de synchronisation

Contrairement aux les erreurs « logicielles » et les erreurs « Matérielles », les composants qui ont une erreur de synchronisation fournissent toujours des sorties logiques correctes. Cependant, ils présentent des délais plus élevés entre les établissements des signaux d'entrée et de sortie. Ces erreurs sont causées par la variabilité de la tension, la température, les défauts de fabrication et le vieillissement du dispositif.

5 Techniques de tolérances aux fautes

Les moyens pour atteindre la sûreté de fonctionnement d'un système sont la prévention des fautes, l'élimination des fautes et la tolérance aux fautes.

La tolérance aux fautes est définie comme la capacité de corriger la fonctionnalité du système en présence de fautes. Idéalement, un système tolérant aux fautes est capable d'exécuter sa tâche correctement indépendamment de la présence de fautes qu'il s'agisse de dégradation physique du matériel, de défauts logiciels, d'attaques malveillants, d'erreurs d'interaction homme-machine. Dans les années 1950, John Von Neumann a lancé l'idée d'utiliser la redondance pour améliorer la fiabilité des systèmes.

Les architectes doivent répondre au problème de la fiabilité en utilisant des architectures tolérantes aux fautes. Ces architectures sont couramment utilisées pour tolérer les fautes en ligne, c'est-à-dire les fautes qui apparaissent pendant le fonctionnement normal du système indépendamment de leur nature transitoire ou permanente. Ils utilisent la redondance pour tolérer les défauts de la logique combinatoire (CL) et les éléments de stockage. Ces techniques sont généralement classées en fonction du type de redondance utilisé.

5.1 Détection d'erreur

Un système ne peut pas tolérer une erreur s'il n'a pas connaissance de son existence. Les mécanismes de détection d'erreur forment la base d'un système résistant aux erreurs car toute faute lors d'une opération doit être détectée avant que le système ne puisse décider d'une action corrective pour la tolérer. Même si un système ne peut pas recouvrir l'erreur détectée, il peut au minimum stopper le processus ou informer l'utilisateur qu'une erreur et que les résultats ne sont plus fiables. Plusieurs techniques basées sur la redondance ont été proposées et utilisées commercialement pour la conception de système de calculs fiables. Elles ont été répertoriées en trois classes : La redondance matérielle, la redondance temporelle, et la redondance d'information.

5.1.1 La redondance matérielle

La redondance matérielle fait référence à l'ajout de ressources matérielles supplémentaires, tel que le doublement du système, en utilisant un comparateur en sortie pour détecter les erreurs. Il est tenu compte ici de la structure du circuit et non pas de la fonctionnalité. Il est tout aussi efficace pour les fautes transitoires que pour les fautes intermittentes ou permanentes Cependant, les exigences en surfaces et en puissances sont très élevées. Elle peut être classée en deux sous types : duplication avec comparaison et duplication avec redondance complémentaire.

La duplication avec comparaison ou double redondance modulaire est une technique de détection d'erreur simple et facile à mettre en œuvre (voir figure 8), elle possède une bonne capacité de détection d'erreur. Le coût matériel du circuit augmente de plus de 100%.

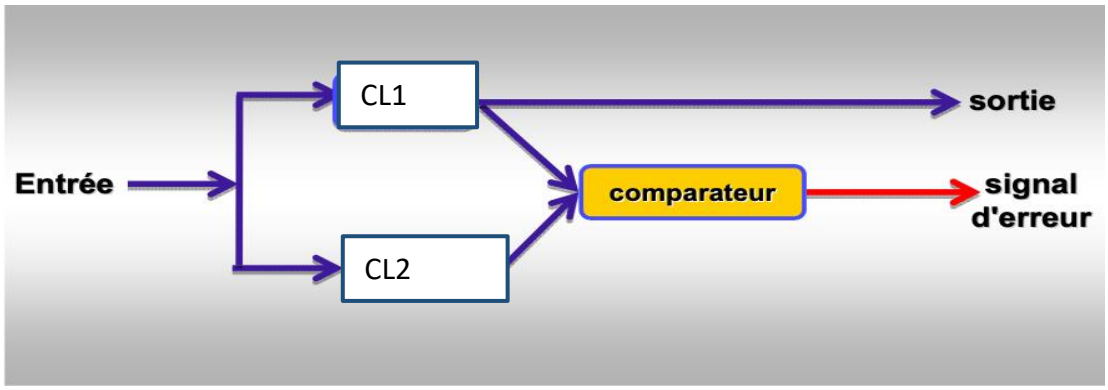

Figure 8 : Duplication avec comparaison

5.1.2 La redondance temporelle

La redondance temporelle est une autre méthode qui a été récemment étudiée. Elle consiste à répéter un calcul ou une transmission de données deux fois ou plus et à comparer les résultats avec les copies précédemment stockées. Ce type de redondance est intéressent lorsqu'il s'agit de distinguer les erreurs transitoires des erreurs permanentes. Si l'erreur est toujours présente après avoir répété le test plusieurs fois, il est probable que l'erreur soit permanente. Dans cette approche, il y a une pénalité en termes de temps supplémentaires, cependant la pénalité matérielle est moindre. Il s'agit d'une technique de réplication temporelle, sans considération de la fonctionnalité du circuit.

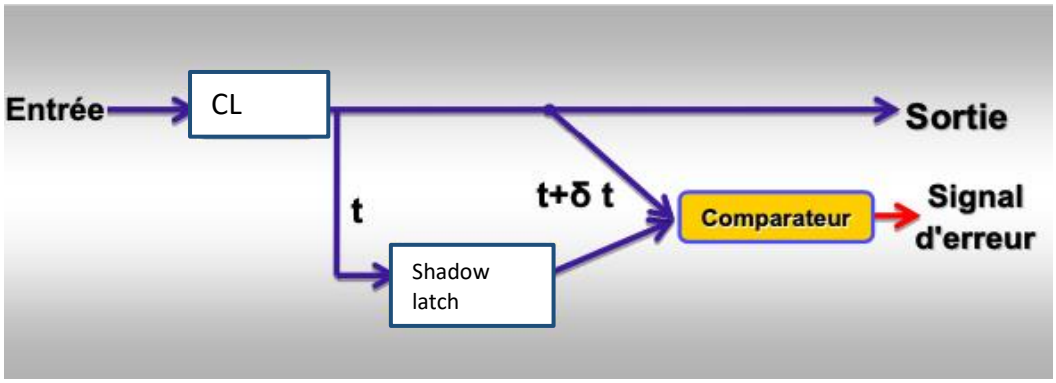

Figure 9 : Redondance temporelle pour la détection des fautes transitoires ou intermittentes

Dans cette technique, les fautes intermittentes et transitoires sont détectées (comme indiquée dans la figure 9) Les résultats du calcul sont stockés dans une shadow latch (mémoire). Les mêmes données sont ensuite utilisées pour répéter le calcul, en utilisant le même bloc fonctionnel à l'instant t+∂t.

Pour la détection des fautes permanentes, le circuit a été modifié comme montré la figure 10. Selon laquelle le comparateur est placé sur le registre de sortie et il compare un signal synchrone avec un signal asynchrone. Tout écart permet de détecter une faute permanente dans le bloc fonctionnel.

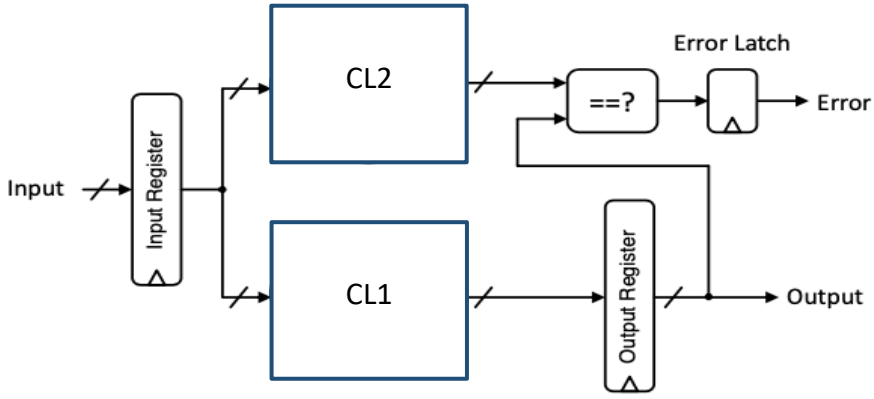

Figure 10 : la comparaison entre un signal synchrone et asynchrone pour détecter les fautes permanentes

La redondance temporelle affecte directement les performances du système, même si le coût matériel est généralement inférieur à celui de la redondance matérielle. Ainsi les systèmes basés sur la redondance temporelle sont comparativement plus lents. Afin de surmonter cet inconvénient de nombreux systèmes utilisent le pipelining pour masquer le problème de latence. Les conséquences énergétiques de la redondance temporelle ne sont pas du tout traitées, à l'exception du fait qu'elle consomme deux fois plus d'énergie qu'une unité non redondante.

5.1.3 La redondance d'information

Le type de redondance le plus répandu dans le contexte des mémoires est la redondance d'information. Elle suppose l'ajout d'informations supplémentaires aux données transmises, stockées ou traitées ce qui permet de vérifier la validité de l'information. Habituellement, ces informations supplémentaires sont des codes qui sont calculés sur la base des données elles-mêmes. Ces codes appelés codes de détection d'erreurs. C'est une façon de protéger les données grâce à un codage mathématique qui peut être réutilisé ensuite pour décoder les données originales. Comme montré dans la figure 11.

Les circuits de codage et de décodage ajoutent des délais supplémentaires ce qui les rends plus lents que la duplication avec comparaison, mais le surcout matériel est beaucoup plus faible. Lors du codage, l'information stockée ou la fonctionnalité du circuit sont considérées, mais la

structure du circuit n'est pas prise en compte. Typiquement, la redondance d'information est utilisée pour protéger des éléments de stockage (comme la mémoire et les registres).

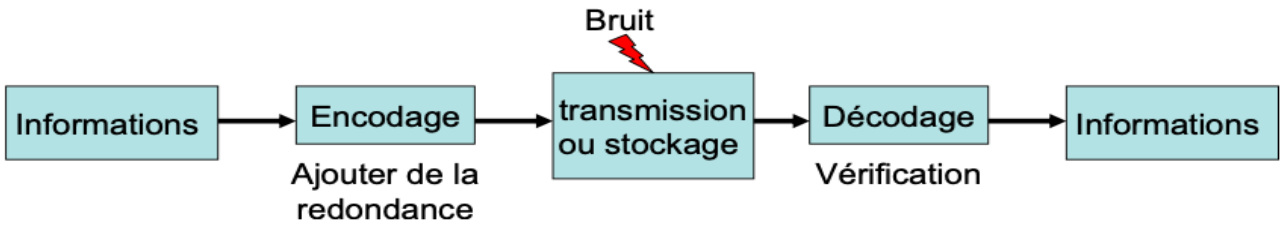

Figure 11 : Principe de la redondance d'information

5.1.4 La redondance logicielle

Les techniques de tolérance aux fautes logicielles sont également basées sur la redondance, qui est appliquée aux procédures, aux processus, aux données ou à l'ensemble du code d'exécution. Le type le plus courant de redondance logicielle dans les systèmes embarqués est la multiplication des données. Une façon simple de procéder consiste à stocker une copie de mot dans une zone de mémoire différente comme la shadow latch montré dans la figure 9, cela permet de détecter par comparaison des résultats égaux ou inégaux. Le principal inconvénient de la redondance logicielle est la consommation de mémoire car la multiplication des données, du code ou des processus nécessite un espace mémoire supplémentaire qui généralement limité dans les systèmes embarqués.

5.2 Correction d'erreurs

Détecter une erreur est une action suffisante pour assurer la sécurité, mais nous souhaitons également que le système puisse recouvrer les états défectueux. Le recouvrement cache les effets de l'erreur à l'utilisateur. Après le recouvrement, le système peut reprendre son fonctionnement et idéalement continuer à fonctionner normalement. Le recouvrement d'erreurs est une caractéristique importante pour un système basé sur les deux attributs de sûreté de fonctionnement et de disponibilité, car les deux mesures nécessitent que le système puisse se remettre de ses erreurs, sans intervention de l'utilisateur.

Les techniques de correction sont aussi classées en trois sous-classes : la redondance matérielle, temporelle ou d'information.

5.2.1 La redondance matérielle

L'ajout d'un troisième bloc supplémentaire et le remplacement du comparateur par un voteur produit une architecture TMR (Triple Modular Redundacy). Comme indiqué dans la figure 12. La TMR, en plus de leur détection, peut également corriger les erreurs. Dans cette technique, tous les composants fonctionnent simultanément et leurs sorties sont envoyés vers un voteur. La sortie du voteur sera correcte si au moins deux composants sont non défectueux. Les techniques de redondance statique sont caractérisées pour être simples, mais elles ont un fort surcoût en surface et en consommation.

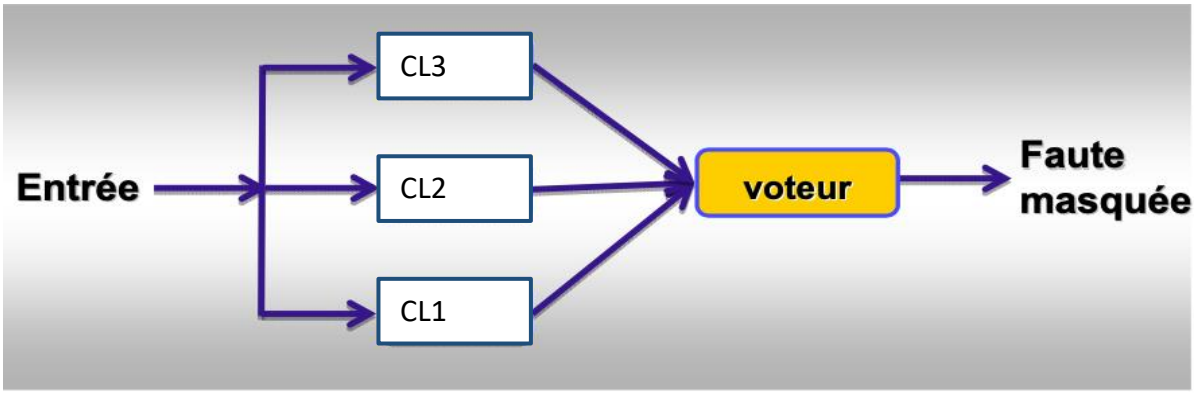

Figure 12 : Redondance modulaire triple

La technique TMR a longtemps constitué la solution de tolérance aux fautes principale dans les avions et les navettes spatiales, où non seulement les processeurs, mais les systèmes entiers sont répliqués pour garantir la robustesse.

5.2.2 La redondance temporelle

Pour la correction d'erreur à l'aide de la redondance temporelle, un calcul est répété sur le même matériel à trois intervalles de temps différents puis un vote intervient sur les résultats. Elle exige donc trois fois plus de cycle d'horloge pour exécuter la même tâche. Elle peut uniquement corriger les erreurs dues aux fautes transitoires à condition que la durée de la faute soit inférieure au temps de calcul. Ayant besoin de temps supplémentaires pour répéter les calculs, elle ne peut être employée que dans les systèmes avec peu ou pas de contraintes temporelles. Cependant, elle offre des surcoûts plus faibles par rapport à la TMR.

5.2.3 La redondance d'information

Les codes correcteurs d'erreurs peuvent fournir des solutions moins coûteuse que les autres techniques de redondance connues comme la TMR. Ils sont couramment utilisés pour protéger les mémoires (voir figure 13). Le surcoût d'un code repose sur : des bits supplémentaires nécessaires pour protéger les informations, du matériel supplémentaire et de la latence pour le codage et le décodage. Cependant, la latence du codage et du décodage peut être réduite si ceux-ci sont exécutés en parallèles.

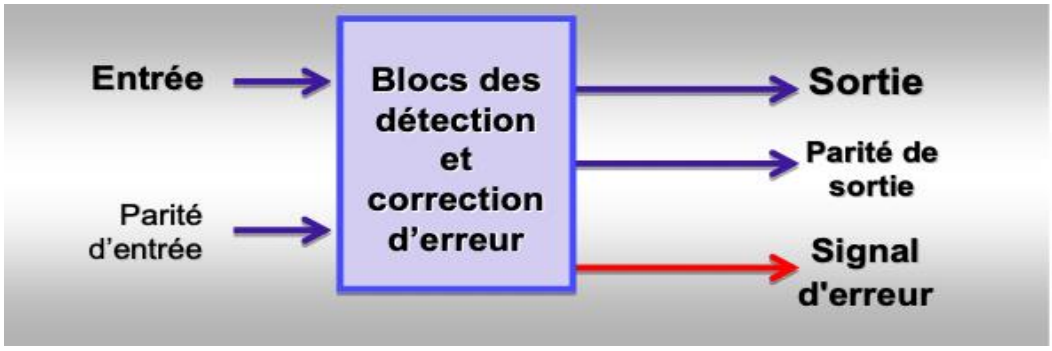

Figure 13 : Bloc mémoire à détection et correction d'erreurs

6 Architecture Hybride Tolérante aux fautes

Les techniques de tolérance aux fautes au niveau du circuit sont généralement basées sur une ou plusieurs redondances. Ces techniques de tolérances aux fautes au niveau du circuit sont généralement classées comme des techniques basées sur le matériel car elles reposent principalement sur la réplication du matériel et sur des modules matériels supplémentaires pour

protéger le système contre les fautes. De plus, les techniques de tolérances aux fautes basées sur le logiciel sont également mises en œuvre au niveau du système en raison de leur non-intrusion, de leur grande flexibilité, de leur faible temps de développement et de leur coût. Cependant, les techniques de tolérances aux fautes basées sur le logiciel ne peuvent pas assurer une protection complète du système en raison de leur incapacité à traiter toutes les erreurs de flux de contrôle possibles.

6.1 Définition de l'architecture hybride

L'architecture étudiée (figure 16) s'appelle architecture hybride parce qu'elle combine 3 types de redondances : la redondance logicielle, la redondance temporelle et la redondance matérielle.

L'architecture Hybride Tolérante aux fautes en anglais HYFT (Hybrid Fault Tolerance) utilise pour la **détection des erreurs** :

- **La redondance matérielle** : elle est sous forme de duplication avec comparaison : le doublement du bloc CL avec le comparateur en sortie. Elle est efficace pour les fautes transitoires.
- **La redondance temporelle** : elle est sous forme de recalcul qui est connu par le signal rollback, elle consiste à répéter le calcul deux fois ou plus et comparer les résultats avec les copies précédemment stockées. Elle est intéressante de distinguer les fautes transitoires des fautes permanentes, si l'erreur est toujours présente après avoir répété le test plusieurs fois, il est probable que la faute est permanente.
- **La redondance logicielle** : elle est sous forme de zone de mémoire qui s'appelle la shadow latch. Elle est incorporée dans le registre d'entré pour prendre en charge la reconfiguration, elle consiste à stocker une copie de mot et la comparer avec le résultat suivant.

Et pour la **correction des erreurs**, l'architecture hybride utilise :

- **La redondance matérielle** : l'ajout d'un troisième bloc CL, et la machine à état fini FSM donne l'ordre à une paire de blocs en fonction de sa commande (voir figure 14 et 15). Les sorties sont envoyées vers le comparateur. La sortie du comparateur sera correcte si les deux blocs sont non défectueux.
- **La redondance temporelle** : le calcul est répété sur les trois paires des blocs CL, le signal rollback intervient sur chaque paire des blocs, puis le comparateur donne le résultat de comparaison. Elle peut uniquement corriger les erreurs dues aux fautes transitoires à condition que la durée de la faute soit inférieure au temps de calcul.

L'architecture utilise la réplication des blocs CL. Un ensemble de multiplexeurs et de démultiplexeurs est utilisé pour sélectionner deux copies primaires de CL et pour mettre la troisième copie de CL en mode veille pendant le fonctionnement normal. L'architecture HyFT est pilotée par un module de logique de commande (FSM) qui génère les signaux de commande nécessaire.

L'architecture HyFT utilise le comparateur pseudo-dynamique pour la détection des erreurs afin d'obtenir une meilleure capacité de détection des glitchs de réduire la consommation d'énergie. Elle utilise un mécanisme de détection d'erreur simultané. Un comparateur pseudo-dynamique compare les sorties de deux copies actives de CL. On peut voir sur l'architecture que le comparateur est placé sur le registre de sortie de manière à pouvoir comparer la sortie du registre de sortie Sout qui est un signal synchrone, avec la sortie de la copie active secondaire Aout, qui est un signal asynchrone.

Le schéma de récupération des erreurs utilise des reconfigurations de la granularité au niveau de l'étage et des retours en arrière profonds a un seul cycle. Les shadow latchs incorporées dans le registre pipeline conservent un cycle d'horloge avant l'état des FF du registre pipeline. La

comparaison a lieu après chaque cycle d'horloge. Ainsi, la détection d'une erreur peut déclencher un cycle de reconfiguration et de retour en arrière, confinant l'erreur et l'empêchant d'affecter le calcul dans les cycles suivants.

La comparaison n'a lieu que pendant de brefs intervalles de temps appelés fenêtre de comparaison. Le moment de la fenêtre de comparaison est défini par la phase haute d'un signal d'horloge retardé DC, qui est généré à partir de CLK à l'aide d'un élément de retard. Ces brèves comparaisons, offrant une réduction de puissance par rapport à un comparateur statique.

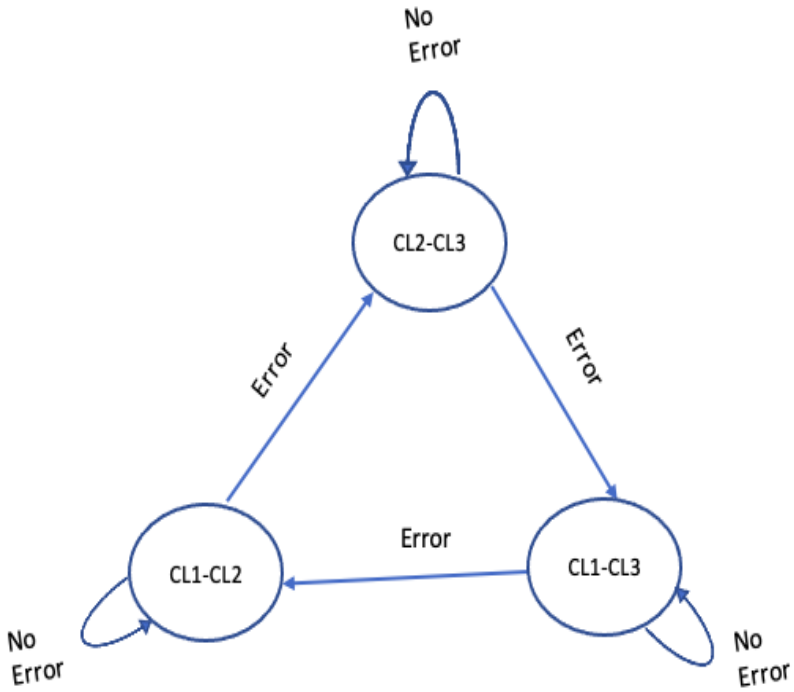

Figure 14 : Graphe d'état de la machine à état fin FSM.

State	Next_state		Output				
	Error	No Error	M1	M2	D1	D2	D3
A (CL1-CL2)	B	A	0	1	1	1	0
B (CL2-CL3)	C	B	1	0	0	1	1
C (CL1-CL3)	A	C	0	0	1	0	1

Figure 15 : Table d'état de la FSM

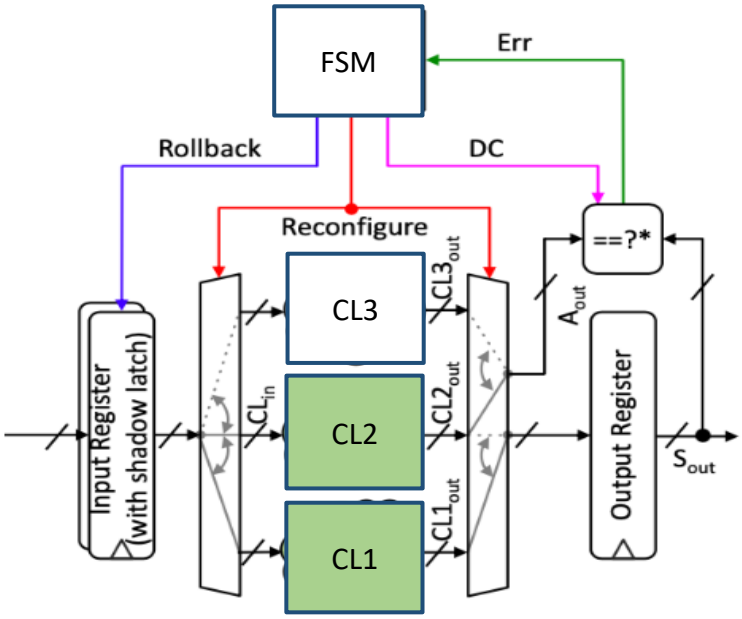

Figure 16 : Architecture Hybride Tolérante aux fautes

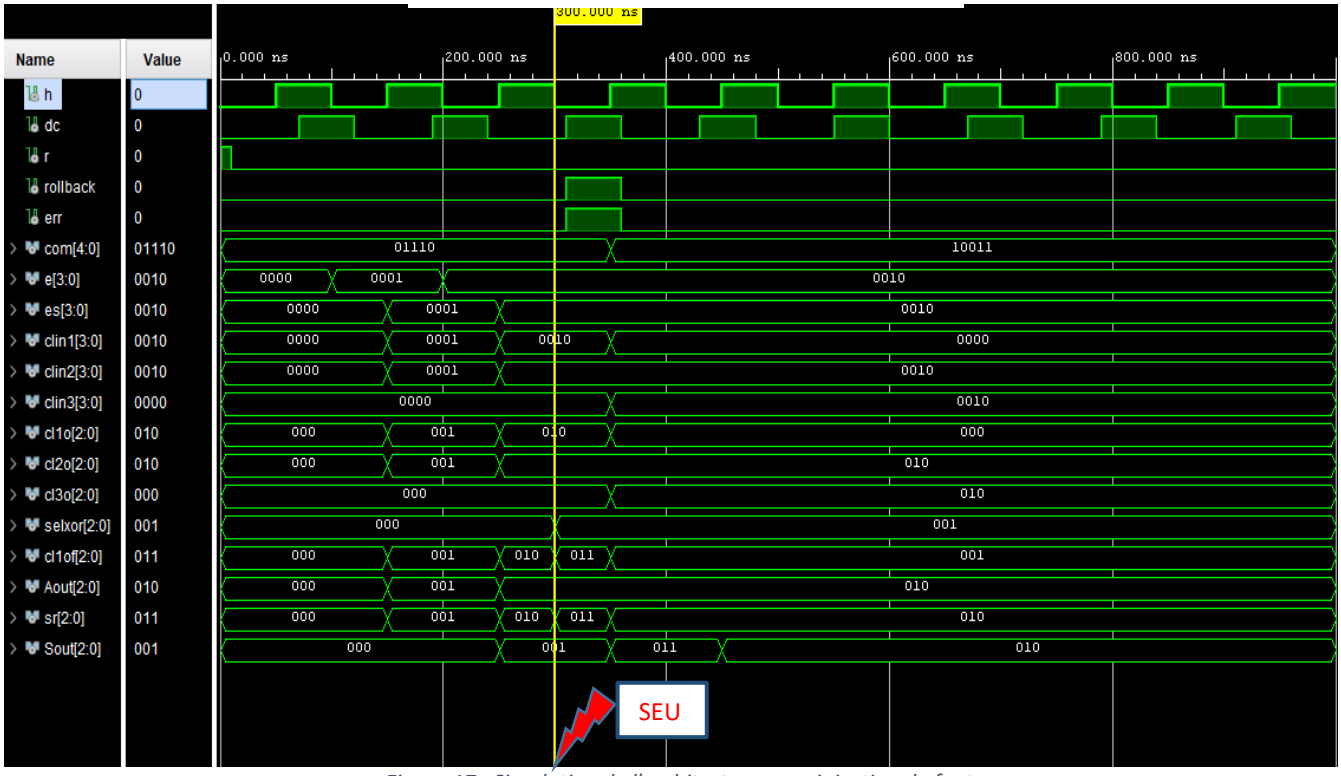

Figure 17 : Simulation de l'architecture avec injection de faute

6.2 Injection de fautes

L'injection de fautes est une activation volontaire de fautes dans le but d'observer le comportement nominal du circuit (sans injection de fautes), avec son comportement en présence de fautes injectées pendant l'exécution d'une application.

Les techniques d'injection de fautes sont devenues populaires pour évaluer et améliorer la fiabilité des systèmes embarqués à base de processeurs. Elles peuvent être réalisés au niveau physique ou simulées.

6.2.1 Injection de fautes physique

Les fautes sont directement injectées dans le matériel et perturbent l'environnement comme les interférences électromagnétiques, le laser…

6.2.2 Injection de fautes simulées

Une approche de simulation haut niveau, elle a été largement utilisée pour sa simplicité, sa polyvalence et sa contrôlabilité. La simulation permet une analyse plus complète et fournir des résultats plus précis et revenir moins cher que l'injection de fautes physiques. Elle offre une meilleure contrôlabilité et observabilité. Les fautes sont produites en modifiant les valeurs logiques lors de la simulation. Un exemple pour introduire des fautes aux blocs CL de l'architecture étudiée figure (16) est l'utilisation des portes XOR qui permettent d'inverser les bits des mots en sortie du bloc CL(résultat de l'addition).

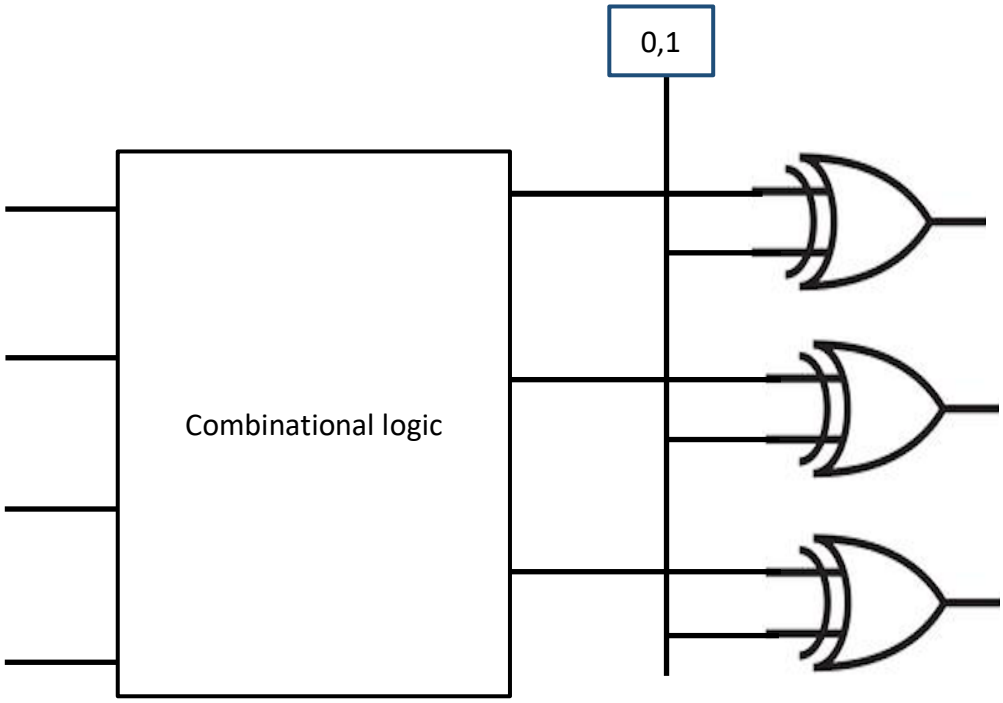

Figure 18 : Modèle d'injection de fautes

7 Conclusion

Ce rapport de stage présente une étude explicite sur le fonctionnement d'un dispositif microélectronique en présence des fautes. Nous avons commencé d'abord par les notions fondamentales de la fiabilité avec les méthodologies conceptuelles pour l'améliorer. Notre étude est essentiellement fondée sur la tolérance aux fautes dans le contexte de la détection des fautes dans les circuits intégrés. Pour cela nous avons étudié les entraves de la sûreté de fonctionnement, nous avons présenté les différents types de fautes ainsi que leurs caractéristiques. Ensuite, les différentes techniques de tolérances aux fautes basées sur les quatre types de redondances, d'information, matérielle, temporelle et logicielle, en expliquant comment détecter une erreur et après la corriger, nous avons les comparer du point de vue coût, surface, et consommation temporelle. De plus nous avons abordé l'étude d'une architecture hybride tolérante aux fautes qui combine trois types de redondances matérielle, logicielle et temporelle. Enfin, nous avons étudié les techniques d'injection de fautes avec une simulation dans le but d'observer le comportement nominal du circuit c'est-à-dire sans injection de fautes avec son comportement en présence de fautes pendant l'exécution.

8 Résumé en français

Compte tenu des récents progrès technologiques, les circuits et les systèmes intégrés sont devenus de plus en plus complexes et plus performants grâce à la miniaturisation de la taille des composants électroniques, tout en réduisant leur consommation énergétique ainsi que leurs coûts de fabrication. Cependant chaque circuit doit faire face aux entraves de la sûreté de fonctionnement grâce aux densités de fautes et d'erreurs croissantes qui sont susceptibles d'entrainer une défaillance. Afin de dépasser cette problématique de fiabilité, en 1950 John Von Neumann a introduit les techniques de tolérance aux fautes qui permettent d'assurer par redondance le fonctionnement correct d'un système. Ces techniques sont utilisées dans les processeurs, les avions et les navettes spatiales afin de garantir la robustesse des systèmes numériques. Ce rapport de stage présente une étude sur une architecture hybride tolérante aux fautes. Pour détecter une erreur logicielle, l'architecture combine 3 types de redondances, la redondance matérielle, temporelle et logicielle et pour la corriger elle combine deux types de redondances, la redondance matérielle et la redondance temporelle.

9 Résumé en anglais

Due to recent technological advances, integrated circuits and systems have become more complex and more efficient cause to the miniaturization of the size of electronic components, while reducing their energy consumption and manufacturing costs. However, each circuit must face the constraints of reliability due to the increasing density of faults and errors that are likely to lead to a failure. In order to overcome this problem of reliability, in 1950 John von Neumann introduced fault tolerance techniques which ensure the correct operation of a system by redundancy. These techniques are used in processors, airplanes and space shuttles to ensure the robustness of digital systems. This internship report presents a study on a fault-tolerant hybrid architecture. To detect a soft error, the architecture combines 3 types of redundancies, hardware, timing and software redundancy and to correct it, it combines two types of redundancies, hardware and timing redundancy.

Bibliographie

Amer Kajmakovic, K. D. (2020). *Challenges in Mitigating Soft Errors in Safety-critical Systems with COTS Microprocesseurs.* Graz: Institute of Technical Informatics, TU-Graz, AT.

Amin, M. (2011). *Conception d'une architecture journalisée tolérante aux fautes pour un processeur à pile de données .* Metz : Univeristé Paul Verlaine .

Durand, B. (2011). *Proposition d'une architecture de contrôle adaptative pour la tolérance aux fautes .* Montpellier : Université Montpellier 2 .

Ghania, A. A. (s.d.). *Etude et modélisation de défauts des circuits fortement submicroniques sécurisés en vue du test.* TIZI-OUZOU: Université MOULOUD MAMMERI DE TIZI-OUZOU.

Wali, I. (2016). *Circuit and System Fault Tolerance Techniques .* Montpellier : Université de Montpellier .