

UNIVERSIDADE FEDERAL DO RIO GRANDE
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

GIANE MARIA DOS SANTOS ULLOA

**ANÁLISE DAS TÉCNICAS TMR E DTMR APLICADA A CIRCUITOS
COMBINACIONAIS NANOMÉTRICOS**

Dissertação apresentada como requisito parcial
para a obtenção do grau de Mestre em
Engenharia de Computação.

Prof^a. Dr^a. Cristina Meinhardt
Orientadora

Rio Grande, 03 de setembro de 2018

Ficha catalográfica

U42a Ulloa, Giane Maria dos Santos.
 Análise das técnicas TMR e DTMR aplicada a circuitos
 combinacionais nanométricos / Giane Maria dos Santos Ulloa. –
 2018.
 56f.

 Dissertação (mestrado) – Universidade Federal do Rio Grande –
 FURG, Programa de Pós-Graduação em Engenharia de Computação,
 Rio Grande/RS, 2018.

 Orientadora: Dra. Cristina Meinhardt.

 1. Microeletrônica 2. Projeto Digital 3. Tolerância a Falhas
 4. TMR 5. DTMR I. Meinhardt, Cristina II. Título.

CDU 004



MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO RIO GRANDE
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

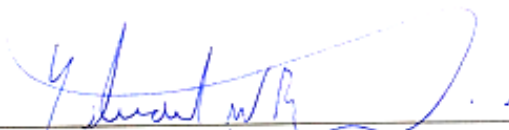
DISSERTAÇÃO DE MESTRADO


**Análise das técnicas TMR e DTMR aplicada a circuitos combinacionais
nanométricos**

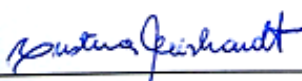
Giane Maria dos Santos Ulloa

Banca examinadora:


Prof. Dr. Ricardo Augusto da Luz Reis


Prof. Dr. Eduardo Wenzel Brião


Prof. Dr. Paulo Francisco Butzen


Prof.ª. Dr.ª. Cristina Meinhardt
Orientadora

AGRADECIMENTOS

Diante de tantos imprevistos nesses últimos dois anos não poderia deixar de agradecer todas as pessoas que contribuíram para essa conquista. Em primeiro lugar eu agradeço a minha família, avó, mãe e irmãos por sempre me apoiarem em momentos de dificuldade. Amo muito vocês e obrigada por todo apoio ao longo deste caminho. Obrigada por entender todos os momentos em que precisei estar ausente.

Agradeço a minha orientadora Cristina Meinhardt por toda orientação, motivação, apoio, amizade, paciência e conselhos. Obrigada por não ter me permitido desistir todos esses anos, desde minhas orientações na iniciação científica.

Também gostaria de agradecer as orientações, momentos de descontração e amizade do professor Paulo Butzen ao longo desses anos no laboratório e também sobre suas valiosas contribuições sobre o meu trabalho na qualificação. Também agradeço as contribuições e incentivos do professor Eduardo Brião a respeito do meu trabalho durante qualificação.

Gostaria de agradecer a todos os colegas do laboratório de Sistemas Digitais e Embarcados com quem tive mais convívio no meu primeiro ano de mestrado e pelas amizades contruídas, momentos de angústia compartilhados e momentos de descontração. Também quero deixar meu agradecimento aos amigos do mestrado com quem criei laços independente do grupo de pesquisa e que vou levar para sempre a amizade de vocês: Raquel, Júlia, Miguel e Bruno.

Meu agradecimento também aos amigos que conheci quando fui para o mercado de trabalho e que tanto me apoiaram nesses últimos meses de preparação para a defesa e também não me deixaram desistir. Muito obrigada a todos vocês e em especial ao Lucas que sempre me incentivou a continuar.

Resumo

Os avanços no campo da microeletrônica possibilitaram fabricar dispositivos que utilizam tecnologias nanométricas, aumentando as funcionalidades disponíveis em um chip, e, conseqüentemente, o número de transistores compondo um mesmo sistema integrado. No entanto, também surgem uma gama de desafios para os projetistas de circuitos integrados. Dentre os principais desafios estão a variabilidade, o envelhecimento e, principalmente, a sensibilidade a falhas. Para lidar com esse último desafio, aplicam-se diversas técnicas capazes de tolerar ou mascarar falhas. A técnica mais utilizada atualmente é a técnica *Triple Modular Redundancy* ou TMR, que consiste em triplicar um módulo do circuito e fazer com que suas saídas apontem para um circuito votador majoritário. Porém, a desvantagem deste método é que ele aumenta em mais de três vezes a área do circuito, considerando a triplicação dos módulos mais a área do circuito votador majoritário. Uma das alternativas para minimizar este problema é o uso de uma técnica chamada de *Diverse Triple Modular Redundancy* ou DTMR. A diversidade de projeto visa evitar que erros sejam replicados no circuito e, também, pode, de acordo com os circuitos escolhidos, minimizar a área ocupada, ao mesmo tempo que o torna mais robusto com relação a falhas. O objetivo deste trabalho é comparar as técnicas TMR e DTMR aplicadas ao projeto de circuitos em tecnologias nanométricas. Como estudo de caso, as técnicas são aplicadas a circuitos somadores completos de 1 bit e a portas lógicas XOR. Estas funções foram escolhidas baseado na importância para todo e qualquer sistema de computação e também pela possibilidade de serem implementados com diferentes arranjos, capazes de explorar diferentes números de transistores por topologia. Os circuitos utilizados nos experimentos foram implementados utilizando em um primeiro momento a tecnologia CMOS de 32 nm HP. Resultados mostram que, além de adicionar a diversidade de projeto reduzindo as chances de que um mesmo vetor de entrada seja sensível em todos os módulos, a técnica DTMR mostrou apresentar o mesmo grau de robustez que a técnica TMR.

Palavras chave — Microeletrônica, Projeto Digital, Tolerância a falhas, TMR, DTMR.

Abstract

Advances in the field of microelectronics have made it possible to manufacture devices that use nanometric technologies, increasing the functionalities available on a chip, and consequently the number of transistors composing the same integrated system. However, there is also a range of challenges for integrated circuit designers. Among the main challenges are variability, aging and, especially, sensitivity to failure. To deal with this latter challenge, several techniques that tolerate or mask failures are applied. The most commonly used technique today is the Triple Modular Redundancy or TMR technique, which consists of tripling a circuit module and having its outputs point to a majority voting circuit. However, the drawback of this method is that it increases by more than three times the area of the circuit, considering the triplication of the modules plus the area of the majority voting circuit. One of the alternatives to minimize this problem is the use of a technique called Diverse Triple Modular Redundancy or DTMR. The diversity of the project is designed to avoid errors being replicated in the circuit and also, according to the chosen circuits, to minimize the area occupied, at the same time that it makes it more robust with respect to failures. The objective of this work is to compare the TMR and DTMR techniques applied to circuit design in nanometric technologies. As a case study, the techniques are applied to complete 1-bit adder circuits and XOR logic gates. These functions were chosen based on the importance to each and every computer system and also the possibility of being implemented with different arrangements, capable of exploring different numbers of transistors per topology. The circuits used in the experiments were implemented using CMOS technology at 32 nm HP. Results show that, in addition to adding design diversity, reducing the likelihood that the same input vector will be sensitive in all modules, the DTMR technique showed the same degree of robustness as the TMR technique.

Keywords – Microeletronic, Digital design, Fault Tolerance, TMR, DTMR.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS	7
LISTA DE FIGURAS	8
LISTA DE TABELAS	10
1 INTRODUÇÃO	11
2 PRINCIPAIS CONCEITOS SOBRE FALHAS DE HARDWARE	14
2.2 Máscaramento	18
2.4.1 Máscaramento lógico	18
2.4.2 Máscaramento elétrico	19
2.4.3 Máscaramento temporal	20
2.3 Tratamento de falhas em <i>hardware</i>	21
3.1 Circuitos Somadores, XORs e Votadores	26
3.2 Ambiente de simulação	32
3.3 Avaliação das características elétricas das arquiteturas	34
3.4 Modelagem e simulação de falhas transientes	35
4 RESULTADOS	38
4.1 Robustez dos módulos	38
4.2 Robustez das Arquiteturas TMR e DTMR	40
4.3 Tempos de propagação, potência e área	41
CONCLUSÕES	49
5.1 Trabalhos Futuros	50
REFERÊNCIAS	53

LISTA DE ABREVIATURAS E SIGLAS

ATMR	<i>Aproximate Triple Modular Redundancy</i>
CI	Circuito Integrado
CMOS	<i>Complementary Metal-Oxide-Semiconductor</i>
DDR	<i>Design Diversity Redundancy</i>
FURG	Universidade Federal do Rio Grande
GND	<i>Ground</i>
HP	<i>High Performance</i>
ITRS	<i>International Technology Roadmap for Semiconductors</i>
L	<i>Length</i>
LET	<i>Linear Energy Transfer</i>
LP	<i>Low Power</i>
MOS	<i>Metal Oxide Semiconductor</i>
NAND	<i>Not AND</i>
NMOS	transistor MOS do tipo N
NOR	<i>Not OR</i>
PMOS	transistor MOS do tipo P
PTM	<i>Predictive Technology Model</i>
SEE	<i>Single Event Effects</i>
SET	<i>Single Event Transient</i>
SEU	<i>Single Event Upset</i>
SPICE	<i>Simulation Program with Integrated Circuit Emphasis</i>
TDMR	<i>Triple Diverse Modular Redundancy</i>
TMR	<i>Triple Modular Redundancy</i>
VDD	<i>Supply Voltage</i>
VLSI	<i>Very large-system Integration</i>
W	<i>Width</i>

LISTA DE FIGURAS

FIGURA 2. 1 – MODELO DOS TRÊS UNIVERSOS	14
FIGURA 2. 2 - PROCESSO DE COLETA E DIFUSÃO DE CARGA	16
FIGURA 2. 3 - PROPAGAÇÃO DE UM SET EM CIRCUITO COMBINACIONAL.....	18
FIGURA 2. 4 - MASCARAMENTO LÓGICO	19
FIGURA 2. 5 - MASCARAMENTO ELÉTRICO.....	20
FIGURA 2. 6 - MASCARAMENTO TEMPORAL.....	20
FIGURA 2. 7 - REDUNDÂNCIA MODULAR TRIPLA.....	21
FIGURA 3. 1 – RESUMO DOS EXPERIMENTOS REALIZADOS	26
FIGURA 3. 2 - SOMADOR COMPLETO DE 1 BIT	27
FIGURA 3. 3 - SOMADOR COMPLETO CMOS	28
FIGURA 3. 4 - SOMADOR COMPLETO HÍBRIDO	28
FIGURA 3. 5 - SOMADOR COMPLETO TGA	29
FIGURA 3. 6 - SOMADOR COMPLETO TFA	29
FIGURA 3. 7 - TOPOLOGIA PORTA LÓGICA XOR.....	31
FIGURA 3. 8 - VOTADOR NAND.....	32
FIGURA 3. 9 - AMBIENTE DE SIMULAÇÃO.	33
FIGURA 3. 10 – FLUXO DE DESENVOLVIMENTO DAS ATIVIDADES.	34
FIGURA 3. 11 - FORMAÇÃO DO PULSO DE CORRENTE.....	35
FIGURA 4. 1 - RESULTADOS NORMALIZADOS DE ATRASO MÁXIMO, POTÊNCIA E ÁREA COM BASE NO TMR_CMOS.....	43
FIGURA 4. 2 - RESULTADOS NORMALIZADOS DE ATRASO MÁXIMO, POTÊNCIA E ÁREA COM BASE NO TMR_TFA	44
FIGURA 4. 3 - RESULTADOS NORMALIZADOS DE DTMR POR TMR XOR_V4.....	48
FIGURA 5. 1 - FLUXOGRAMA DA FERRAMENTA DE COMPARAÇÃO TMR E DTMR	52

LISTA DE TABELAS

TABELA 3. 1 - TABELA VERDADE DO SOMADOR COMPLETO DE 1 BIT	26
TABELA 3. 2 - PRINCIPAIS CARACTERÍSTICAS DAS QUATRO ARQUITETURAS DE SOMADOR CMOS	30
TABELA 3. 3 - TABELA VERDADE DA XOR	30
TABELA 3. 4 – RELAÇÃO DE ARQUITETURAS QUE FORAM UTILIZADAS NESTE TRABALHO	33
TABELA 4. 1 - RELAÇÃO DE <i>FAULT MASKING</i> DOS MÓDULOS ISOLADOS DA XOR	39
TABELA 4. 2 - NODOS E ENTRADAS COM MAIOR SENSIBILIDADE A SET POR TOPOLOGIA DE XOR.....	39
TABELA 4. 3 - RELAÇÃO DO <i>FAULT MASKING</i> DOS MÓDULOS ISOLADOS DOS SOMADORES	40
TABELA 4. 4 - RESULTADOS FALHAS DO TIPO SET INSERIDAS A NÍVEL ARQUITETURAL .	41
TABELA 4. 5 - TEMPOS DE PROPAGAÇÃO DOS MÓDULOS INTERNOS DAS ARQUITETURAS	42
TABELA 4. 6 - RELAÇÃO ENTRE OS ATRASOS, POTÊNCIA E NÚMERO DE TRANSISTORES DE SOMADORES.	42
TABELA 4. 7 - TEMPOS DE PROPAGAÇÃO DOS MÓDULOS INTERNOS DA TMR XOR	45
TABELA 4. 8 - TEMPOS DE PROPAGAÇÃO DOS MÓDULOS INTERNOS DE DTMR XOR.....	45
TABELA 4. 9 - RELAÇÃO ENTRE OS ATRASOS, POTÊNCIA E NÚMERO DE TRANSISTORES DAS XORS.....	47

1 INTRODUÇÃO

Os avanços na microeletrônica permitiram aumentar a capacidade de integração dos circuitos e, assim, criar arquiteturas cada vez mais complexas e com dimensões cada vez menores (TAMBARA et al, 2013). Com a redução da escala de fabricação, espera-se que as novas tecnologias apresentem um desempenho superior e um número cada vez maior de funcionalidades, aplicadas a dispositivos comuns no nosso dia a dia como os celulares, automóveis, sistemas médicos e outros. Porém, essa evolução também traz diversos desafios aos projetistas de sistemas digitais e embarcados, tais como variabilidade de processo (SAMAR K. SAHAR, 2014) (BORKAR et al., 2003), envelhecimento (BORKAR, 2005) e aumento da sensibilidade a falhas (DODD, 2010).

Circuitos aritméticos são o foco de diversas pesquisas, pois fazem parte do caminho crítico dos sistemas eletrônicos. Este tipo de circuito tem um papel fundamental no funcionamento de qualquer sistema computacional, dos mais simples controladores aos mais complexos microprocessadores. Existem diferentes tipos de arranjos de transistores que implementam circuitos aritméticos, a maioria explorando circuitos somadores de 1 bit e portas lógicas XOR. Cada uma das abordagens tem vantagens e desvantagens bem exploradas em relação à área, atraso e potência. Entretanto, projetos atuais devem-se considerar também efeitos decorrente da miniaturização dos dispositivos e o aumento da sensibilidade a falhas (BECKETT, 2002), sendo esse último o foco deste trabalho.

Existem diversas técnicas capazes de mascarar falhas em um circuito integrado. Muitas técnicas de tolerância a falhas utilizam algum tipo de redundância. Entretanto, isso implica ter mais recursos, além do mínimo necessário, para executar uma função com o intuito de mascarar falhas e manter um nível de funcionalidade adequado do sistema. Existem três tipos de redundância: redundância de *hardware*, redundância de informação e redundância de tempo (KOREN, 2010).

A redundância de *hardware* pode ser considerada desde uma simples duplicação a outras estruturas complexas com unidades ativas. Todas as formas de redundância de *hardware* têm um custo em área de *chip* e consumo de energia, sendo geralmente justificável sua aplicação apenas em sistemas críticos (KOREN, 2010).

A técnica mais clássica de redundância de *hardware* é chamada de NMR (*N-Modular Redundancy*). Uma estrutura NMR tem N módulos executando a mesma função e a sua saída é escolhida por um votador de maioria. Se $N = 3$, então é chamado de TMR (*Triple Modular Redundancy*) (KOREN, 2010). O *Triple Modular Redundancy* ou TMR é a técnica de redundância mais adotada, devido a sua grande robustez, e consiste em triplicar um mesmo módulo do sistema, fazendo que os três executem a mesma operação e o resultado é processado por um circuito votador (GOMES, 2014).

Outras técnicas como o *Design Diversity Redundancy* ou DDR (TAMBARA et al, 2013) e o *Aproximate Triple Modular Redundancy* ou ATMR (GOMES, KASTENSMIDT, 2013) exploram a diversidade de projeto, se destacando a técnica *Diverse Triple Modular Redundancy* ou DTMR. O TMR diversificado (DTMR), foi inspirado em uma técnica de redundância de software chamada de *N-Version Programming* (ELMENDORF, 1972). Mais tarde a técnica foi estendida para ser utilizada em *hardware* por (AVIZIENIS, 1985). A diferença para o TMR é que o DTMR utiliza três módulos com topologias\ diferentes, ou seja, cada módulo redundante é implementado de uma forma diferente. Alguns trabalhos mostram que o DTMR melhora o mascaramento de falhas comparado ao TMR (TAMBARA et al, 2013) (HIARI; SADEH; RAWASHDEH, 2012). Entretanto, estes trabalhos exploram a DTMR aplicada a FPGA.

Neste contexto, este trabalho tem como objetivo avaliar diferentes arranjos de circuitos utilizando a técnica *TMR* e a técnica *DTMR*. Como estudo de caso, são adotadas diferentes implementações de somadores completos de 1 bit e portas lógicas XOR. O diferencial desenvolvido neste trabalho é o estudo e comparação destas técnicas sob a influência de falhas do tipo *Single Event Transient* a nível elétrico, observando as características elétricas, em tecnologias bulk CMOS nanométricas. Dessa forma, será possível comparar as duas técnicas quanto a sua robustez no mascaramento de falhas, desempenho, consumo de energia e área ocupada pelos circuitos.

Este trabalho está organizado da seguinte forma: o Capítulo 1 apresenta uma introdução ao tema geral do trabalho. No Capítulo 2 é feita uma revisão sobre os principais conceitos que envolvem falhas, mascaramento e técnicas de tolerância a falhas. No Capítulo 3 é descrita a metodologia utilizada em todas as etapas do trabalho. Já no Capítulo 4 são mostrados os resultados deste estudo e, finalmente, no Capítulo 5

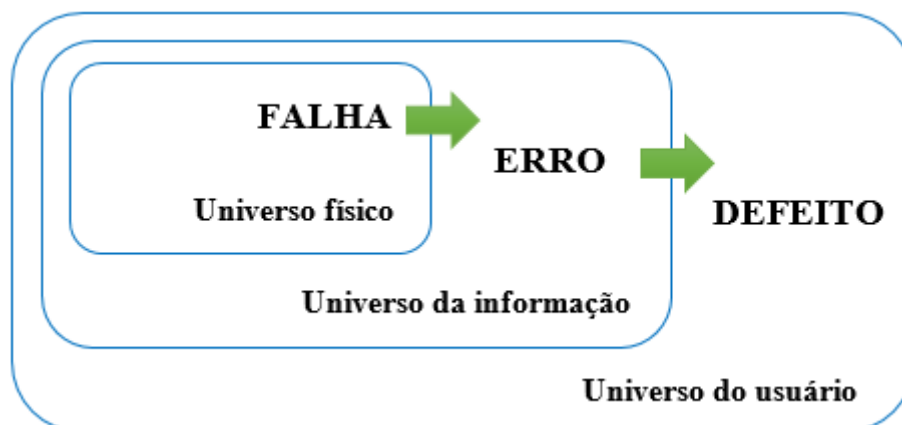
encerramos com as conclusões obtidas através dos dados dos experimentos e apresentamos algumas possibilidades de trabalhos futuros.

2 PRINCIPAIS CONCEITOS SOBRE FALHAS DE HARDWARE

Falhas, erros e defeitos podem assumir diferentes significados de acordo com o autor ou com o contexto. Dessa forma, os conceitos sobre falha, erro e defeito presentes nesse trabalho serão a seguir definidos, de acordo com (FRANCO, 2008).

Existem muitos fatores que podem ocasionar uma falha, como por exemplo erros de projeto ou influências externas. Nesse caso, falhas são consideradas como uma ação inesperada que pode influenciar no comportamento do circuito ou simplesmente não ser notada. De acordo com o ponto onde a falha ocorre, ela pode ou não gerar um erro. Assim, define-se que um sistema se encontra no estado errôneo se o processamento posterior, a partir do momento que ocorreu a falha, conduzir a um defeito. Um defeito, por sua vez, é definido como um desvio da especificação do circuito, podendo ser apenas evitado, e jamais tolerado. A Figura 2.1 ilustra o conceito sobre o modelo dos três universos: o universo físico onde ocorrem as falhas, o universo da informação onde acontecem os erros e o terceiro, o universo do usuário onde aparecem os defeitos (JOHNSON, 1996).

Figura 2. 1 – Modelo dos três universos



Fonte: Adaptação de (ZYMPECK, 2013)

Existem falhas provenientes do processo de fabricação e decorrentes do uso, que se caracterizam por falhas permanentes. Falhas intermitentes, que se caracterizam pela ocorrência temporária e repetida de um determinado comportamento podendo causar um defeito e dependem de alguma condição externa. Já as falhas transientes, ocorrem devido a fenômenos aleatórios como partículas radioativas (CONDESSA, 2009).

Além disso, falhas do tipo transiente podem ser provenientes do meio, como fonte de radiação ou influência eletromagnética e se caracterizam por provocarem uma alteração temporária em um dispositivo ou circuito, mas sem efeitos permanentes. As falhas que mais ocorrem em circuitos integrados operando no espaço e, mais recentemente, em circuitos integrados fabricados em tecnologia nanométrica operando também na Terra são do tipo transiente, chamados de *Single Event Effects* (SEE) ou *soft errors* (NICOLESCU, 2003).

Essas falhas são causadas por efeitos da radiação devido a interação de nêutrons com o silício, gerando partículas secundárias como a partícula alfa. Quando este efeito ocorre em um elemento de memória, é denominado *Single Event Upset* (SEU) e caracteriza-se pela inversão do valor armazenado no *flip-flop* (um *bit flip*). No entanto, quando esse efeito afeta uma porta lógica de um bloco combinacional, é denominado *Single Event Transient* (SET) e é observado como um pulso de tensão transiente (*glitch*) de duração variável conforme carga coletada (CHIELLE, 2012). A redução na escala de fabricação da tecnologia veio acompanhada de maiores frequências de operação, menores tensões de alimentação e margem de ruídos menores, tornando maior a sensibilidade do circuito a SET (GOMES, 2014) (DODD, 2010).

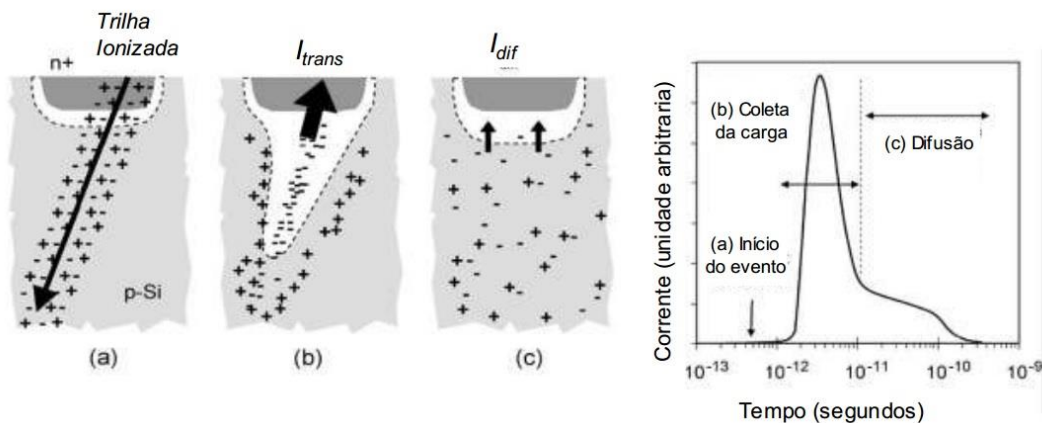
As falhas transientes que afetam circuitos integrados são causadas principalmente pela ação de radiação. No espaço, a atividade solar é a principal causa da radiação, a qual é constituída por partículas energéticas, tais como elétrons, prótons ou íons energéticos e também pela radiação eletromagnética, a qual pode ser constituída por raios X, raios gama ou luz ultravioleta (BAUMANN, 2001).

Ao nível do mar, três mecanismos diferentes geram as partículas energéticas responsáveis por induzir os *soft errors*: as partículas alfas, oriundas de impurezas presentes no encapsulamento e no próprio circuito integrado, nêutrons, gerados pela interação entre raios cósmicos de alta energia com átomos da atmosfera terrestre e raios cósmicos de baixa energia (NORMAND, 1996) (BAUMANN, 2005).

As junções P-N dos transistores são as regiões sensíveis dos transistores (BAUMANN, 2001). Quando um íon energético colide com uma destas regiões sensíveis, uma trilha cilíndrica de pares elétron-lacuna e uma elevada concentração de portadores são formados seguindo a passagem do íon, como mostra a Figura 2.2 (a). Quando a trilha de ionização resultante atravessa ou se aproxima da região de depleção (região vazia de cargas), portadores são coletados rapidamente pelo campo elétrico, que cria uma corrente/tensão transiente nesse nó (FRANK, 2011). Essa ionização gera uma deposição de carga que pode gerar um pulso de corrente transiente que, por conseguinte, pode ser interpretado por um sinal no circuito e causar uma mudança na lógica executada (KASTENSMIDT, 2006).

A região de depleção toma a forma de um funil que aumenta a eficiência da coleta de carga devido ao aumento da região de depleção dentro do substrato, como podemos observar na Figura 2.2 (b) (FRANK, 2011). Quanto mais distante da junção o evento ocorre, menor a quantidade de carga que será coletada, sendo menos provável que a colisão cause um *soft error* (BAUMANN, 2005). A Figura 2.2 (c) mostra a fase onde a difusão começa a dominar o processo de coleta. Já na Figura 2.2 (d), aparece a curva que representa o pulso transiente real gerado pela colisão da partícula energética com o dispositivo.

Figura 2.2 - Processo de coleta e difusão de carga



Fonte: (BAUMANN, 2005)

Analisar a propagação de uma falha do tipo *Single Event Transients* (SET) em circuitos combinacionais é importante para que seja possível identificar os pontos mais sensíveis de um circuito, permitindo localizar quais partes do sistema precisam ser protegidas contra radiação. (FRANK, 2011)

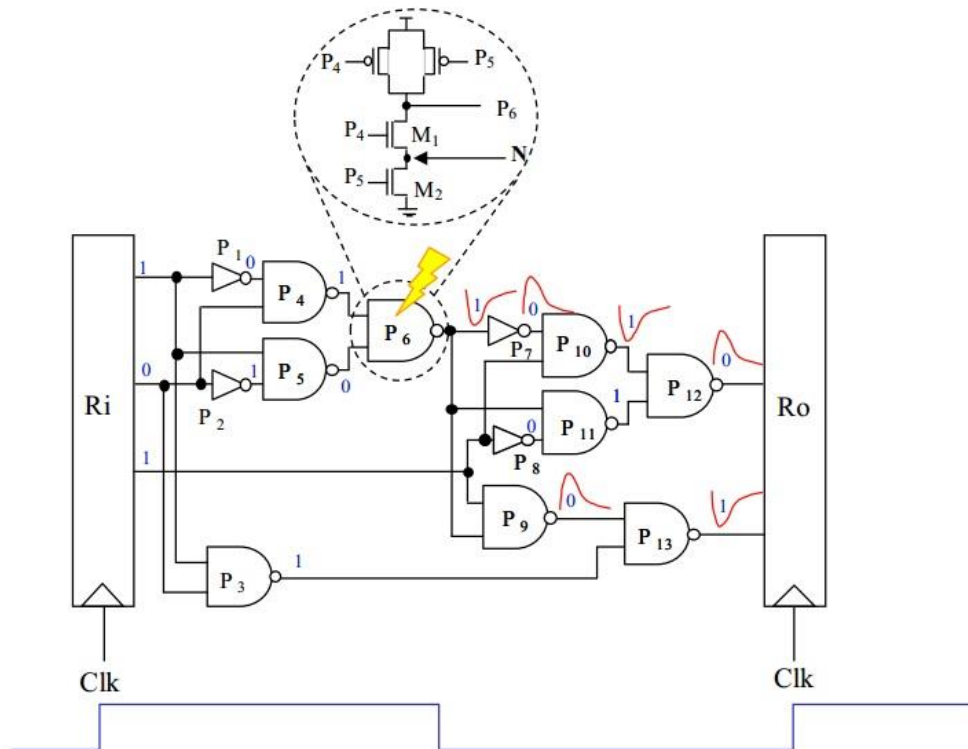
Quando uma falha do tipo SET ocorre em uma porta lógica que não está conectada diretamente a um elemento de memória é preciso avaliar se o pulso gerado irá se propagar ao longo do circuito até o próximo elemento de memória na saída, alterando o resultado final esperado.

Na Figura 2.3 é possível observar uma porta NAND denominada P6 onde existe um nó interno N. Para que seja observado neste nó os efeitos da colisão de uma partícula energética, o transistor M2 deve estar em estado *off*. E para propagar o pulso gerado, para a saída da porta P6, o transistor M1 deve estar em estado *on*. Esta condição pode ser satisfeita apenas quando as saídas das portas P4 e P5 forem 1 e 0, respectivamente, como ilustra a Figura 2.3 (FRANK, 2011).

Para saber a probabilidade do pulso gerado observado em N se propagar para a saída de P6 é necessário conhecer todos os vetores de entrada do circuito que fazem com que as regiões P-N vizinhas a N sejam sensíveis à colisão de uma partícula energética. Neste caso, os vetores —100 e —101 são os únicos vetores que tornam o dispositivo M2 sensível, gerando o pulso observado no nó N, que pode propagar-se para a saída da porta P6 (GILL et al., 2005) e através do circuito.

Assim, pode-se concluir que a probabilidade de um pulso se propagar até uma saída depende do número médio de níveis lógicos entre o ponto onde ocorreu o pulso até a saída. Quanto maior o número de portas lógicas que o pulso precisa percorrer, maior é a chance de que ele seja completamente mascarado lógica ou eletricamente (FRANK, 2011).

Figura 2.3 - Propagação de um SET em circuito combinacional



Fonte: (GILL et al., 2005)

2.2 Máscaramento

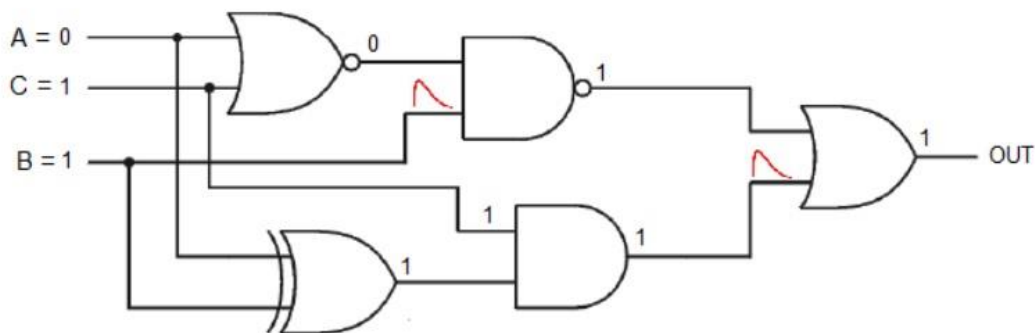
Pode-se dizer que uma falha foi mascarada em um circuito quando, mesmo na presença da falha, o resultado não é alterado, ou seja, não haverá a geração de erro no circuito e defeitos não serão visíveis ao usuário. Existem três principais tipos de mascaramento: o mascaramento lógico, o elétrico e o por janela de amostragem (SHIVAKUMAR, 2002).

2.4.1 Mascaramento lógico

O mascaramento lógico ocorre em um circuito quando uma falha atinge um nó que não é determinante para o resultado do circuito. Dessa forma, mesmo que ocorra a falha e esta altere o valor daquele nó, ela não irá se propagar porque o resultado depende apenas das entradas não atingidas por aquela falha. O mascaramento lógico está exemplificado na Figura 2.4. Observa-se que ocorre uma falha na entrada B da porta NAND2, porém essa falha não interfere no resultado da saída, visto que de acordo

com a Tabela Verdade da função NAND2, sempre que a entrada A for zero, a saída sempre será um, independente do valor da segunda entrada. Na porta OR2, observa-se um comportamento similar. Ela também possui a entrada B com falha. Neste caso, sempre que a entrada A da Tabela verdade da OR2 for um, a saída indicará como resultado o valor lógico um. Sendo assim, a saída final não é afetada pelas falhas que ocorreram em alguns nós do circuito.

Figura 2. 4 - Mascaramento lógico



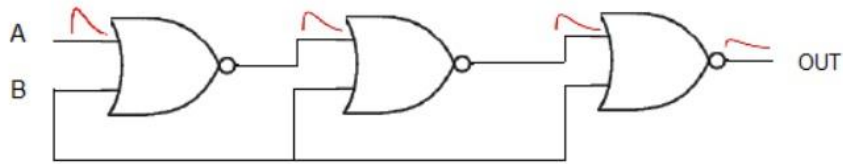
Fonte: ZYMPECK, 2013

2.4.2 Mascaramento elétrico

O mascaramento elétrico ocorre quando a propagação de um determinado pulso acontece de maneira atenuada ao longo do circuito. Ou seja, quando ao percorrer cada elemento do circuito a amplitude do pulso diminui até o momento em que não é mais significativa ao resultado do circuito. Esse efeito de atenuação ocorre com os transientes que possuem uma largura de banda maior que a frequência de corte da porta lógica (MUNTEANU, 2008). Nestes casos a amplitude do transiente pode ser reduzida e seus tempos de subida e descida aumentados e, eventualmente, o pulso pode desaparecer (GOMES, 2014).

É possível observar esse comportamento na Figura 2.5, onde a falha ocorre na entrada A da primeira NOR, porém o efeito dela é amenizado ao longo do circuito, e quando consegue alcançar a saída final do circuito, o pulso já não indica falha, visto que o resultado obtido é próximo ao resultado esperado.

Figura 2. 5 - Mascaramento elétrico



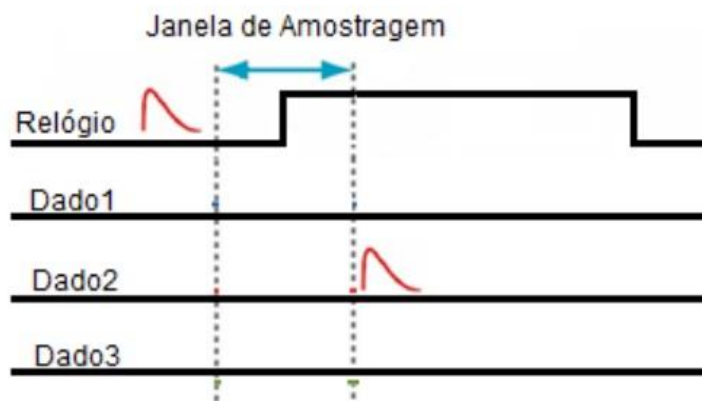
Fonte: ZYMPECK, 2013

2.4.3 Mascaramento temporal

O mascaramento temporal, também conhecido por janela de amostragem, pode ser visualizado na Figura 2.6. Ele ocorre quando um pulso resultante da colisão de uma partícula energética se propaga até um circuito sequencial, porém não atende os requisitos temporais (bordas relógio e tempos de *setup* e *hold*) necessários para ser capturado (GOMES, 2014).

Chama-se janela de amostragem o período de tempo em torno da transição do sinal de um relógio. Ela é responsável por controlar o armazenamento dos sinais das linhas de dados nos elementos de memória. Assim, um pulso de duração semelhante ao tempo de transição do sinal de relógio pode ser interpretado como um período de leitura/escrita pelos elementos de armazenamento, acarretando em armazenamento/leitura incorreta de dados.

Figura 2. 6 - Mascaramento temporal



Fonte: Adaptado de ZYMPECK, 2013

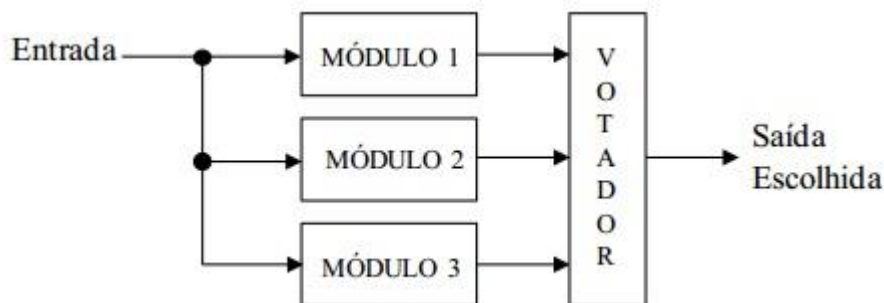
2.3 Tratamento de falhas em *hardware*

Em microeletrônica, as técnicas de tolerância a falhas podem ser descritas como técnicas usadas para detectar, mascarar e tolerar falhas em circuitos. O objetivo do uso dessas técnicas é garantir que o circuito continue funcionando corretamente mesmo na presença de falhas, tornando os possíveis defeitos invisíveis ao usuário.

A maior parte das técnicas de tolerância a falhas utiliza o princípio da redundância, seja através de *software* ou *hardware*. É necessário replicar componentes ou utilizar algoritmos específicos. Todo esse processo gera impacto tanto financeiro, quanto computacional influenciando na área, potência consumida e desempenho dos circuitos. Dessa forma, é importante mensurar todos esses aspectos e verificar se as técnicas aplicadas compensam os custos envolvidos no projeto.

A técnica mais utilizada para tolerar falhas em *hardware* é conhecida como Redundância Modular Tripla (*Triple Modular Redundancy*) que consiste em triplicar fisicamente um módulo que se deseja proteger e ligar sua saída a um votador, conforme pode ser visto na Figura 2.7. O votador retorna o valor lógico que for repassado por pelo menos duas de suas entradas. Nesse caso, mesmo que um dos módulos apresente alguma falha, os resultados dos outros dois módulos irão mascarar-la na saída do votador. A principal desvantagem desta técnica é o significativo acréscimo de *hardware*, o qual é normalmente estimado como sendo maior que 200% (FRANK, 2011).

Figura 2. 7 - Redundância modular tripla



Fonte: FRANK, 2011

Se uma falha do tipo SET atingir o votador, esta poderá ser propagada, causando uma falha em todo o sistema. Assim, o votador constitui um ponto único de falha. Além

disso, o votador ainda corre o risco de escolher o resultado errado, caso dois de seus módulos apresentem falha no resultado. Pensando em minimizar esses efeitos, o votador poderia ser triplicado ou a votação utilizar um sistema de controle por software. Porém, deve ser observado que a triplicação irá aumentar o número de componentes no sistema e quanto mais componentes, maior a possibilidade de falhas (WEBER, 2002). Outras alternativas exploram a construção de votadores majoritários mais robustos a falhas (LIEBL, 2015).

Muitos trabalhos abordam a redundância como uma técnica bastante confiável para tolerância a falhas. Dessa forma, existem muitos trabalhos que exploram a técnica Triple Modular Redundancy (TMR) e suas variações. Porém, a técnica TMR tradicional não é imune à *Common Mode Failures* (CMFs). As CMFs são falhas, que geralmente devido a uma causa comum, afetam mais de uma cópia em um esquema redundante ao mesmo tempo, da mesma maneira (AVIZIENIS E KELLY, 1984). Nesse caso, se mais de uma réplica for afetada pela mesma falha, o votador pode fazer a escolha errada. A diversidade nos projetos de redundância foi primeiramente proposta por (ELMENDORF, 1972) sob o nome de *N-Version Programming*, com o objetivo de proteger aplicativos redundantes baseados em *software* contra CMFs. Mais tarde, (AVIZIENIS E KELLY, 1984) ampliaram a definição de Design Diversity Redundancy ou DDR para projetos baseados em *hardware*.

De acordo com (RITER, 1995), o computador de bordo dos aviões Boeing 777 utiliza uma abordagem DDR através do uso de três processadores diferentes (Intel, AMD e Motorola). No artigo de (CHAU et al., 2001) foi proposto o uso de um esquema DDR para aumentar a tolerância a falhas das redes de aviões. Já em (ASHRAF et al., 2011) investigaram o uso de DDR para melhorar a tolerância a falhas de sistemas TMR em FPGAs. Em (TAMBARA et al, 2013), foi feito uma análise para avaliar a eficácia do método DTMR em experimentos utilizando FPGA sob a influência de neutrôns. Os resultados encontrados na simulação confirmam que os resultados obtidos com experimentos reais mostraram que DTMR e TMR apresentam o mesmo grau de robustez no mascaramento de falhas. Além disso, estudos recentes investigaram o uso do conceito DDR em plataformas de sinal misto (BORGES et al) (HIARI, SADEH; RAWASHDEH 2012).

Desta forma, usando DTMR na construção de uma solução semelhante a da Figura 2.7, poderia ser explorado três módulos diferentes, que ocupem áreas diferentes, executando uma mesma função. Assim, o Módulo 1 poderia ser implementado com um arranjo de transistores diferente do Módulo 2 e do Módulo 3, diversificando a implementação da função e reduzindo a possibilidade de falhas devido ao projeto ou a probabilidade de falhas SET afetarem o mesmo ponto do circuito, como acontece em caso de envelhecimento (DODD, 2010). Além disso, o DTMR pode ainda reduzir o consumo de energia, visto que pode operar com módulos diferentes e menores.

De acordo com (TAMBARA et al, 2013) e (HIARI, SADEH, RAWASHDEH, 2012), o uso da técnica de DTMR melhora o mascaramento de falhas em comparação com o uso da técnica de TMR. Em (CHIELLE, AZAMBUJA, BARTH, KASTENSMIDT, 2013), foi proposta uma ferramenta computacional baseada em técnicas de redundância para mascarar falhas de *Single Event Effects* utilizando redundância de software, também aplicada ao FPGA. Uma técnica de tolerância a falhas é aplicada e seus resultados são comparados no GPUS (GONÇALVES, SAQUETTI, KASTENSMIDT, 2017). Outros trabalhos que abordam o uso de redundância aplicada a tolerância a falhas são (BARTH, 1997), (KASTENSMIDT, CARRO, REIS, 2006) e (GOMES,2014).

Metodologia

O objetivo principal desta dissertação é apresentar uma comparação entre as técnicas de tolerância a falhas TMR e DTMR quanto a robustez no mascaramento de falhas do tipo SET e seus comportamentos elétricos. A avaliação dessas técnicas considera os resultados dos experimentos de inserção de falhas em circuitos somadores completos de 1 bit e em arranjos de portas lógicas XOR. Este trabalho considera a ocorrência de uma falha individualmente no ambiente. Os resultados serão analisados verificando se ocorreu a manifestação da falha no circuito e se a mesma se propagou. Também serão comparados os resultados de atrasos e potência em tecnologia preditiva de 32nm HP.

Para atingir este objetivo, este texto destaca a metodologia a ser adotada desde o início deste projeto até o final desta dissertação. Os passos adotados neste projeto são:

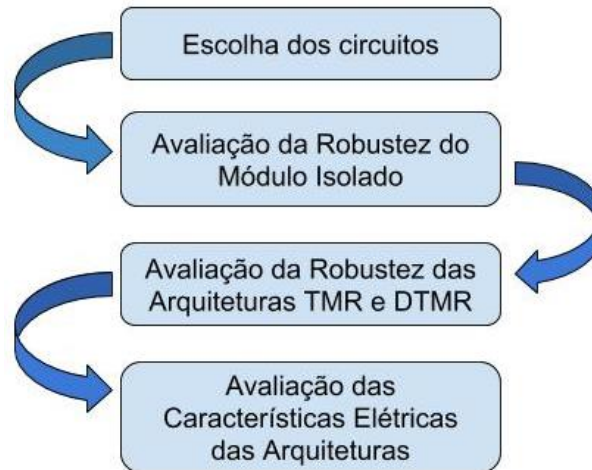
- 1) Revisão conceitual e do estado da arte;
- 2) Escolha dos circuitos utilizados;
- 3) Definição do ambiente de simulação;
- 4) Descrição, validação e caracterização elétrica de circuitos somadores de 1 bit – Versão módulo isolado, TMR e DTMR;
- 5) Injeção de falhas do tipo SET a nível arquitetural em somadores de 1 bit - Versão módulo isolado, TMR e DTMR;
- 6) Análise dos resultados obtidos com a caracterização elétrica e com a injeção de falhas em circuitos somadores de 1bit;
- 7) Desenvolvimento de módulo para automatizar injeção de falhas;
- 8) Descrição, validação e caracterização elétrica de diferentes implementações da porta lógica XOR. - Versão módulo isolado, TMR e DTMR
- 9) Injeção de falhas à nível arquitetural em portas lógicas XOR - Versão módulo isolado, TMR e DTMR
- 10) Análise dos resultados obtidos com a caracterização elétrica e com a injeção de falhas em portas lógicas XOR.

11) Elaboração da dissertação

O passo inicial deste projeto foi a revisão conceitual e também sobre outros trabalhos atualmente publicados abordando temas como técnicas de tolerância a falhas e injeção de falhas do tipo SET. A revisão conceitual possibilitou uma visão clara sobre as vantagens e desvantagens das técnicas TMR e DTMR e também sobre as falhas do tipo SET. Além disso, através da revisão de trabalhos atuais foi possível observar que a maioria dos trabalhos aplica as técnicas TMR e DTMR a nível de FPGA, diferente do que é proposto nessa dissertação, onde as análises são feitas a nível elétrico. Para a escolha das topologias dos circuitos utilizados nesse trabalho também ocorreu uma revisão na literatura tentando escolher os circuitos mais utilizados e também mais sensíveis à SET, afim de validar a robustez das técnicas TMR e DTMR. Mais detalhes sobre os circuitos utilizados e a motivação que levou a escolha de cada um deles pode ser visto na subseção 3.1. Por sua vez, o ambiente de simulação é descrito na subseção 3.2.

Os passos 4 ao 12 deste projeto definem a visão geral da metodologia adotada para possibilitar obtenção dos resultados de simulação capazes de gerar comparativos entre as técnicas TMR e DTMR. Dentre esses passos, está o desenvolvimento de módulos para automatização de injeção de falhas a nível elétrico em circuitos combinacionais. Esses módulos podem ser integrados futuramente e servirem como base para o desenvolvimento de uma ferramenta computacional mais robusta. Os resultados parciais do desenvolvimento deste trabalho foram publicados em congressos como o ICECS e o WCAS. O fluxograma mostrado na Figura 3.1 resume como os experimentos foram realizados e analisados ao longo do desenvolvimento da dissertação.

Figura 3.1 – Resumo dos experimentos realizados



Fonte: Elaborado pelo próprio autor.

3.1 Circuitos Somadores, XORs e Votadores

Circuitos somadores são unidades fundamentais em qualquer sistema computacional. Eles são o bloco principal da unidade lógica e aritmética, sendo utilizados para o processamento de praticamente todas as demais operações aritméticas. Estes circuitos são o foco de diversas pesquisas, pois fazem parte do caminho crítico da maioria dos sistemas computacionais (ALIOTO, 2002) (CHANG, GU, ZHANG, 2005) (PIGUET, 2006). Um somador completo é formado por três entradas e duas saídas (PEDRONI, 2010). A tabela verdade do somador completo é representada na Tabela 3.1, onde *Soma* é a saída que representa a soma de um bit e *Cout* representa o vai-um para o próximo dígito de um sistema de n bits.

Tabela 3.1 - Tabela verdade do somador completo de 1 bit

Entradas			Saídas	
<i>A</i>	<i>B</i>	<i>Cin</i>	<i>Soma</i>	<i>Cout</i>
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

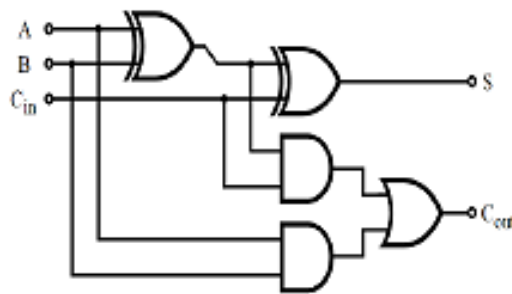
Fonte: Elaborada pelo próprio autor

Para construir um somador completo podemos realizar diferentes arranjos de transistores ou, realizar simplificações em suas funções lógicas, resultando em

diferentes arquiteturas de somadores. Na Figura 3.2 é possível observar a configuração padrão de um somador completo de 1 bit com portas lógicas.

Neste trabalho foram escolhidas quatro diferentes topologias de somadores completos de 1 bit, baseado nos trabalhos de (CHANG, GU, ZHANG, 2005) e de (NAVI et al, 2009), que apresentam uma comparação entre as arquiteturas de somadores completos, (FA – *full adders*): CMOS, CPL, híbrido, 14T, 10T, TGA e TFA. Neste trabalho, optou-se por utilizar quatro destas arquiteturas: CMOS, Híbrido, TGA e TFA.

Figura 3. 2 - Somador completo de 1 bit

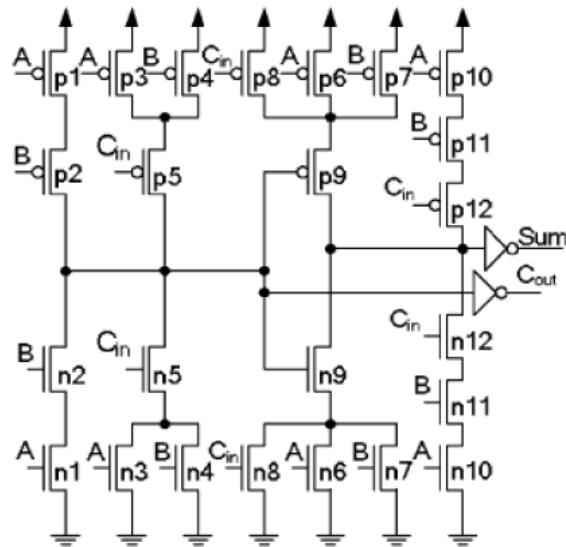


Fonte: (FLOYD, 2007)

O somador Mirror CMOS é uma topologia padrão e tradicional e conta com 28 transistores dispostos em redes *pull-up* e *pull-down* logicamente complementares. A principal vantagem dessa topologia é que ela fornece uma boa capacidade de condução, além de sua robustez, o que é muito bom quando se trabalha com tecnologias muito pequenas e tensões muito baixas. Porém, a principal desvantagem é sua capacitância de entrada alta, por serem conectadas a vários transistores, além de a rede *pull-up* deixar o circuito mais lento (NAVI, 2009). Na Figura 3.3 é possível observar o circuito somador Mirror CMOS construído com portas lógicas básicas. Para que o somador Mirror CMOS pudesse ser implementado seria necessário utilizar a lógica negada.

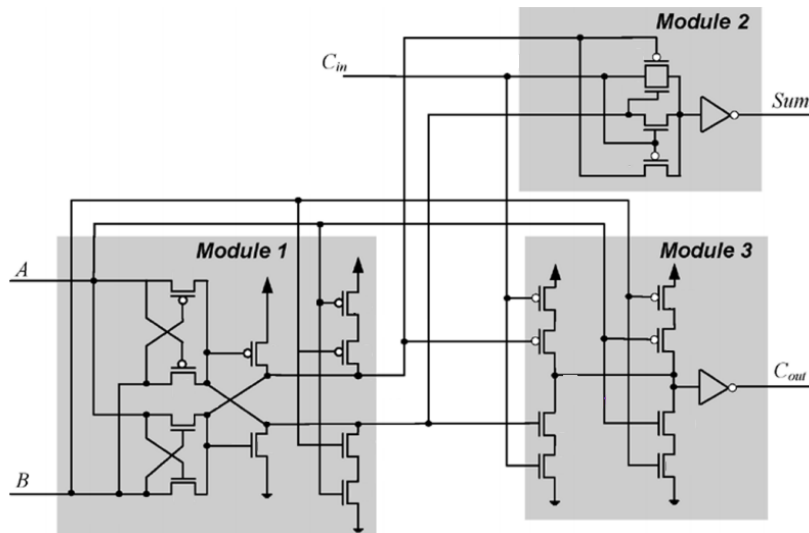
A principal característica do somador híbrido é a união das famílias lógicas CMOS e CPL para otimizar o seu desempenho e potência, utilizando 26 transistores (CHANG, GU, ZHANG, 2005). A principal vantagem desta arquitetura é apresentar um sinal forte na saída e trabalhar bem em baixas tensões. Entretanto, esta arquitetura apresenta uma alta capacitância de entrada para alguns dos vetores de entrada (NAVI, 2009). Na Figura 3.4 é possível observar o circuito somador Híbrido.

Figura 3.3 - Somador completo CMOS



Fonte: (NAVI et al, 2009)

Figura 3.4 - Somador completo híbrido



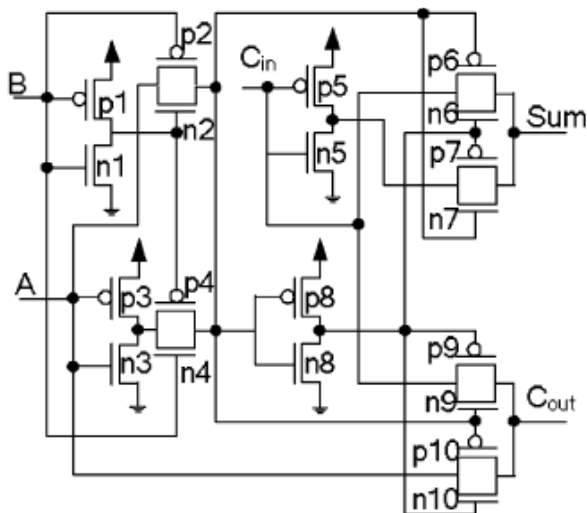
Fonte: (CHANG, GU, ZHANG, 2005)

A arquitetura de somador *Transmission Gate Full Adder* (TGA) possui 20 transistores e é baseada na teoria dos *transmission gates*, que são um tipo particular de porta de passagem e consiste basicamente de um transistor PMOS e um NMOS conectados em paralelo (CHANG, GU, ZHANG, 2005). A principal vantagem desse somador está em não apresentar problemas quando operando em baixas tensões (NAVI, 2009). Na Figura 3.5 é possível observar o circuito somador TGA.

O *Transistor Function Full Adder* (TFA) é uma arquitetura de somador baseada na teoria da função de transmissão e conta com 16 transistores. Para construir os inversores

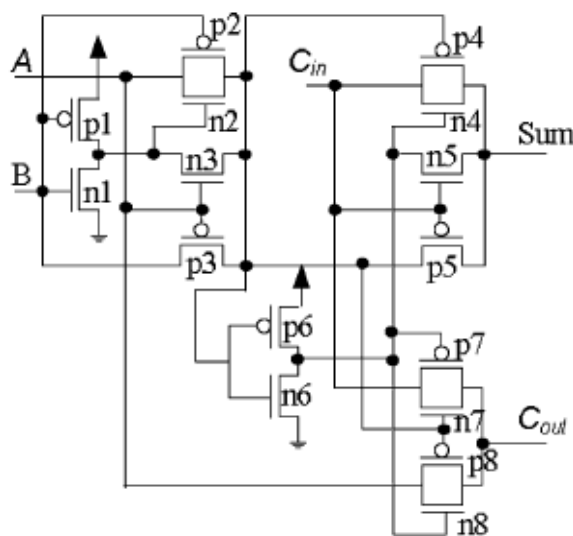
utiliza caminhos *pull-up* e *pull-down*, e para o restante da lógica do somador são utilizados *transmission gates*. A principal desvantagem está na falta de capacidade de condução e uma grande degradação no desempenho quando é cascateado, sendo necessário, quase que na maioria das vezes, *buffers* adicionais na saída desse tipo de somador (NAVI, 2009). Na Figura 3.6 é possível observar o circuito somador TFA.

Figura 3. 5 - Somador completo TGA



Fonte: (CHANG, GU, ZHANG, 2005)

Figura 3. 6 - Somador completo TFA



Fonte: (CHANG, GU, ZHANG, 2005)

A Tabela 3.2 resume as principais características das quatro arquiteturas de somadores que serão avaliadas neste trabalho.

Tabela 3. 2 - Principais características das quatro arquiteturas de somador CMOS

Somador	Número de transistores	Principais características
Mirror CMOS	28	Boa capacidade de condução Rede <i>pull-up</i> deixa o circuito mais lento
Híbrido	26	Trabalha bem em baixas voltagens Alta capacitância para alguns dos vetores de entrada
TGA	20	Falta de capacidade de condução Degradação no desempenho quando é cascateado.
TFA	16	Eficientes implementações para XORs e XNORs Degradação no desempenho quando é cascateado.

Fonte: Elaborado pelo próprio autor

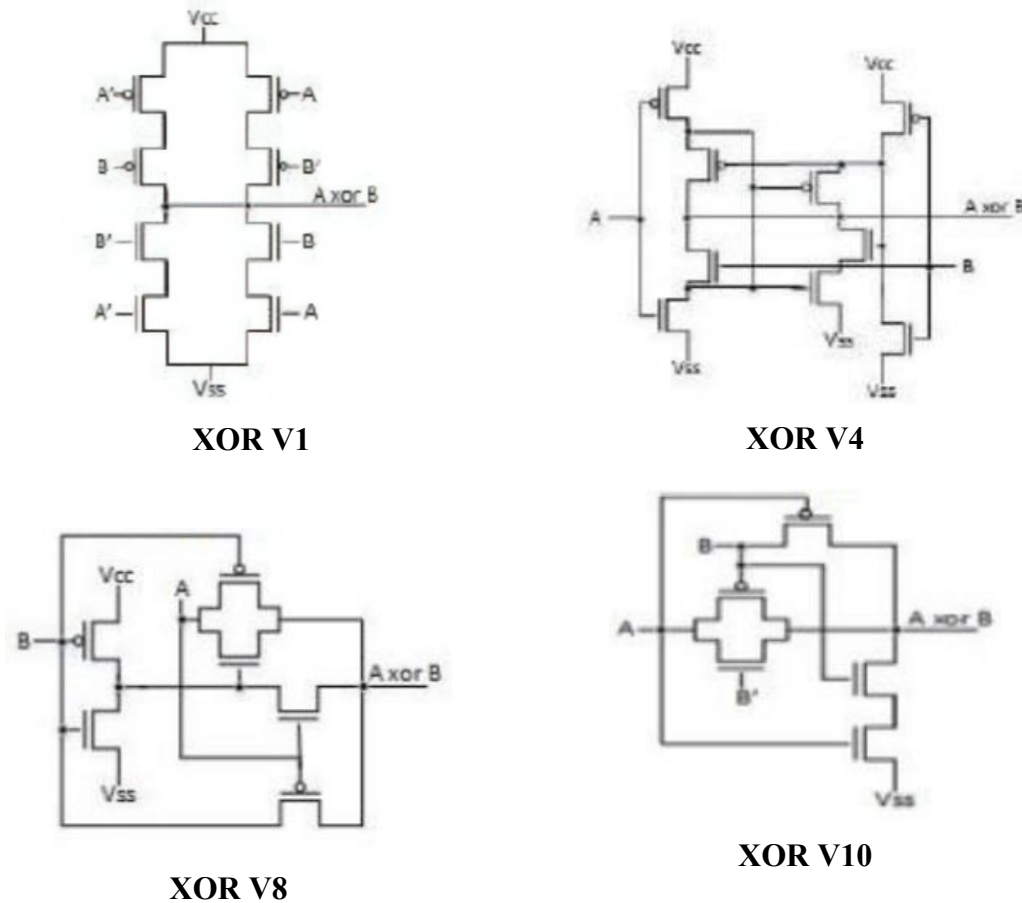
Assim, como o somador, a função XOR é fundamental em sistemas computacionais, fazendo parte dos principais circuitos aritméticos, e dos blocos operacionais de sistemas digitais. A função XOR é descrita pela Tabela 3.3, e de forma geral, pode-se descrever ela como verdadeira quando somente uma das entradas for verdadeira. Além disso, circuitos de XOR de 2 entradas são uma alternativa de poucos transistores para implementar multiplexadores, aumentando a sua utilização em projetos. Para sistemas confiáveis, deve-se reconsiderar a implementação do XOR para aumentar a robustez quanto à radiação. Os circuitos da porta lógica XOR utilizados neste trabalho foram escolhidos baseados na pesquisa apresentada por (AGUIAR, 2017), onde o autor descreve um mapa de nodos sensíveis a falhas do tipo SET. De acordo com (AGUIAR, 2017), o nó de saída não é necessariamente o nó mais sensível de um circuito. Assim, as topologias escolhidas para aplicação neste trabalho foram as descritas como as mais sensíveis a falhas do tipo SET: XOR_V1, XOR_V4, XOR_V8 e XOR_V10, e são apresentadas na Figura 3.7.

Tabela 3. 3 - Tabela verdade da XOR

Entradas		Saída
<i>A</i>	<i>Cin</i>	<i>XOR</i>
0	0	0
0	1	1
1	0	1
1	1	0

Fonte: Elaborada pelo próprio autor

Figura 3. 7 - Topologia porta lógica XOR



Fonte: (SILVA; BUTZEN; MEINHARDT, 2016)

A XOR_V1 é uma implementação clássica da função exclusiva ou lógica. Isto é, tem cinco nós sensíveis, enquanto as topologias alternativas possuem apenas três ou dois. Além da sobrecarga de área, a implementação XOR_V1 fornece um número maior de nós sensíveis, aumentando a probabilidade de ocorrência de SET. Esta comparação destaca que a implementação XOR normalmente encontrada em bibliotecas de células não pode ser a melhor opção quanto à robustez aos efeitos de radiação. A XOR_V4 é uma alternativa que adota também a família lógica CMOS na sua construção, com planos *pull-up* and *pull-down* complementares. As versões V8 e V10 exploram a lógica de transistor de passagem (PTL), reduzindo o número de transistores, mas sendo mais sensíveis a ruídos nos sinais e podendo apresentar sinais de saída com degradação. O circuito XOR_V8 é menor e apresenta nós menos sensíveis (AGUIAR, 2017).

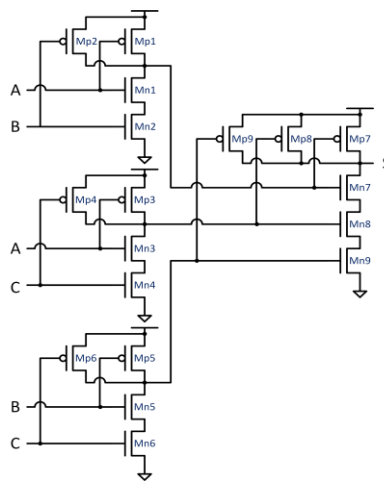
Além dos módulos a serem replicados, para a implementação das técnicas TMR e DTMR é necessário um votador majoritário. O votador majoritário é um circuito

responsável por selecionar a saída correta em um sistema TMR. Ele recebe como entradas as saídas dos módulos replicantes e retorna a que teve a maioria dos votos (FLOYD, 2007). Para realizar essa escolha ele tem como base a Equação (1):

$$V = AB + AC + BC \quad (1)$$

Neste trabalho escolhemos trabalhar com o votador NAND que é uma implementação do votador majoritário clássico utilizando portas NAND. Esse votador conta com 18 transistores. A topologia do votador majoritário NAND pode ser vista na Figura 3.8.

Figura 3. 8 - Votador NAND

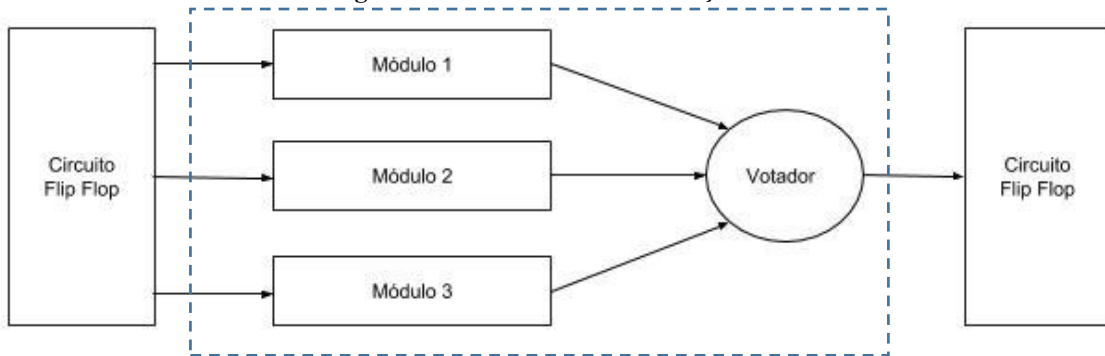


Fonte: (FLOYD, 2007)

3.2 Ambiente de simulação

Foi definido como ambiente de teste o esquemático apresentado na Figura 3.9. Os módulos redundantes estão inseridos entre barreiras de *flip-flops*, como normalmente são projetados em sistemas digitais, como por exemplo estruturas de *pipeline* em arquiteturas de computadores com paralelismo de instruções. As versões que avaliam os módulos isolados implementam cada circuito adotado sem replicação, para determinar a cobertura de falhas dos circuitos sem a adoção de técnicas de tolerância a falhas TMR e DTMR.

Figura 3. 9 - Ambiente de simulação.



Fonte: Elaborado pelo próprio autor

Em todos os experimentos foi utilizado o votador majoritário NAND, cujo circuito é apresentado na Figura 3.8. Para simular as barreiras de pipeline, foi utilizado um *Flip Flop* do tipo D ativo na borda de subida e frequência de *clock* igual a 1GHz. A área destaca pelo pontilhado na Figura 3.9 faz parte da configuração interna da topologia estudada, que pode ser do tipo TMR quando todos os módulos replicados são iguais, ou DTMR, quando os módulos replicados possuem configuração distinta. A determinação da potência total considera o consumo de energia destes quatro módulos. O atraso da arquitetura considera o tempo de propagação das transições dos sinais nas entradas dos módulos até a mudança no estado da saída do votador.

Com o objetivo de comparar as técnicas TMR e DTMR, este trabalho modela topologias TMR com três circuitos idênticos. Já para a técnica DTMR, foram testadas diferentes combinações de topologias. Como módulos, este trabalho explora circuitos somadores de 1 bit CMOS, TFA, TGA, Híbrido, e circuitos XORs em quatro versões apresentados na Seção 3.1. Estes circuitos são apresentados nas Figuras 3.3 até a Figura 3.7 e os arranjos explorados como TMR e DTMR são apresentados na Tabela 3.4.

Tabela 3. 4 – Relação de topologias que foram utilizadas neste trabalho

Técnica	Topologia	Módulo 1	Módulo 2	Módulo 3
TMR	T_FA_CMOS	CMOS	CMOS	CMOS
DTMR	D_FA_CMOS_TGA_TFA	CMOS	TGA	TFA
	D_FA_CMOS_HIB_TFA	CMOS	Híbrido	TFA
TMR	T_XOR_V1	XOR_V1	XOR_V1	XOR_V1
	T_XOR_V4	XOR_V4	XOR_V4	XOR_V4
	T_XOR_V8	XOR_V8	XOR_V8	XOR_V8

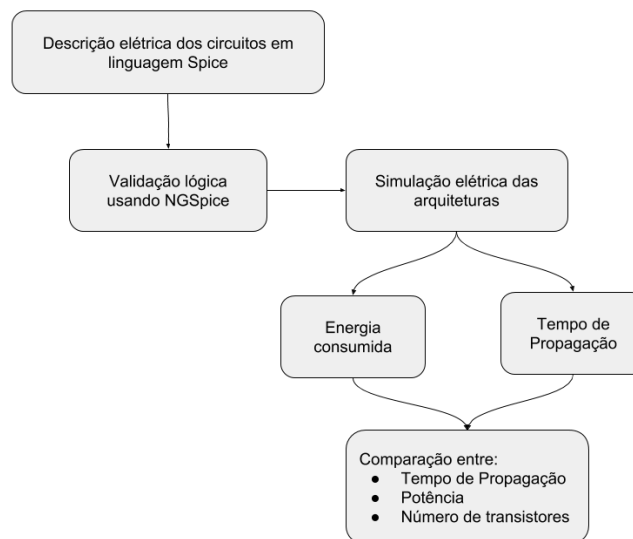
	T_XOR_V10	XOR_V10	XOR_V10	XOR_V10
DTMR	D_XOR_V1_V4_V8	XOR_V1	XOR_V4	XOR_V8
	D_XOR_V1_V4_V10	XOR_V1	XOR_V4	XOR_V10
	D_XOR_V1_V8_V10	XOR_V1	XOR_V8	XOR_V10
	D_XOR_V4_V8_V10	XOR_V4	XOR_V8	XOR_V10

Fonte: Elaboração do próprio autor

3.3 Avaliação das características elétricas das arquiteturas

Após, foi realizada a descrição elétrica destes circuitos em linguagem SPICE. Para isso, utilizamos a tecnologia preditiva de alto desempenho de 32nm *High Performance* (PTM, 2017). Após, realizamos a validação lógica dos circuitos utilizando a ferramenta NGSpice. (NGSPICE, 2017). A segunda etapa consistiu em realizar as medidas dos tempos de propagação e energia consumida por cada um dos arranjos de circuitos, através das simulações elétricas das três diferentes topologias, com o objetivo de realizar um comparativo entre os tempos de propagação, potência e número de dispositivos dessas topologias. A Figura 3.10 apresenta o fluxo de desenvolvimento das atividades desta etapa, de avaliação das arquiteturas TMR e DTMR quanto as características elétricas.

Figura 3. 10 – Fluxo de desenvolvimento das atividades.

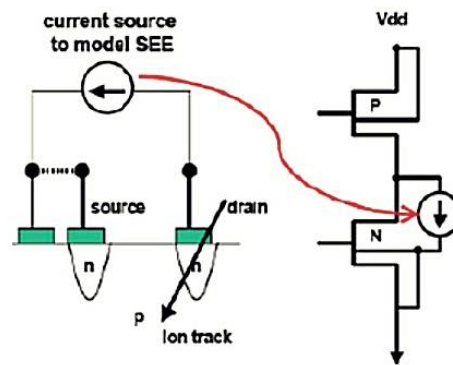


Fonte: Elaborado pelo próprio autor

3.4 Modelagem e simulação de falhas transientes

O modelo matemático para um pulso transiente SET é definido como uma fonte de corrente transiente com forma típica de uma dupla exponencial (MESSENGER, 1982). Dessa forma, para simular a ação de uma falha do tipo SET atingindo um circuito combinacional, deve-se inserir uma fonte de corrente no nodo onde deseja-se injetar a falha, assim como está indicando a Figura 3.11.

Figura 3.11 - Formação do pulso de corrente



Fonte: (WEATHERFORD, 2002)

No simulador NGSpice uma fonte de corrente deve ser declarada da seguinte maneira:

```
I{nome} nodo1 nodo2 EXP (I1 I2 TD1 TAU1 TD2 TAU2)
```

Onde:

Nome: Define o nome da fonte de corrente

Nodo1 e **Nodo2:** são os nodos de conexão da fonte de circuito

I1: é o valor inicial da corrente.

I2: é o valor referente a amplitude do pulso do sinal.

TD1: é o tempo de atraso de subida do pulso.

TAU1: é a constante de tempo de subida.

TD2: é o tempo de atraso de descida do pulso.

TAU2: é a constante de tempo de descida.

Os parâmetros do pulso transiente de corrente gerado pela ionização de uma partícula energética no silício dependem diretamente de características como a energia da partícula incidente, o local que atingiu o dispositivo, da tecnologia e da tensão de alimentação do dispositivo e a sua capacitância de saída (WANG AND AGRAWAL,

2008). O modelo analítico proposto em (MESSENGER, 1982) é amplamente utilizado e propõe uma fonte de corrente cujo comportamento obedece a uma dupla exponencial. A modelagem dessa corrente transiente segundo (MESSENGER, 1982) é descrito pela Equação 2 e Equação 3.

$$I(t) = \frac{Q_{coll}}{\tau_{\alpha} - \tau_{\beta}} \left(e^{-\frac{t}{\tau_{\alpha}}} - e^{-\frac{t}{\tau_{\beta}}} \right) \quad (2)$$

$$Q_{coll} = 10.8 \times L \times LET \quad (3)$$

Onde:

Q_{coll} é a carga coletada na junção;

τ_{α} é a constante de tempo de coleção de carga;

τ_{β} é a constante de tempo para estabelecer a trilha do íon pesado;

L é a profundidade da coleção de carga.

Os valores típicos utilizados para simulações e experimentos no silício são 1.64×10^{-10} segundos para τ_{α} e 5×10^{-11} segundos para τ_{β} (CHOI et al., 1990). O L possui um valor típico de $2\mu m$ para cada LET de $1\text{MeV} \cdot \text{cm}^2/\text{mg}$, e a constante 10.8fC corresponde a carga que uma partícula ionizante deposita para cada $1\mu m$ (WANG AND AGRAWAL, 2008).

Para a simulação dos experimentos, utilizamos um LET com valor fixo igual a $2,73\text{MeV} \cdot \text{cm}^2/\text{mg}$, valor inicial igual a 0 e valor de amplitude igual a 190μ . O valor de constante de tempo de subida igual à 10p e constante de tempo de descida igual a 320p . O pulso de corrente que foi inserido nas simulações foi igual à:

$$\text{Iexp gnd } \mathbf{nodo} \text{ exp (0 190u } \mathbf{10n} \text{ 10p } \mathbf{10.0001n} \text{ 320p)}$$

Os parâmetros alterados a cada simulação constam em negrito, que são o nodo onde será inserida a falha e o tempo em que ela será inserida. Dessa forma, é possível observar o comportamento do circuito para diferentes combinações de entrada.

As simulações de inserção de falhas foram projetadas em duas etapas: arquitetural ou à nível de transistor. Nas inserções a nível arquitetural, as falhas foram inseridas no nodo soma de cada um dos somadores, nas arquiteturas TMR e DTMR. Já na etapa a nível de transistor, foram inseridas falhas nos nodos internos de cada um dos

somadores que compõem o arranjo TMR e DTMR. As falhas foram inseridas de forma exaustiva e uma por vez em cada nodo, ao longo do tempo. O modelo de falha inserido foi o 010, assim, quando o sinal estiver alto no trecho onde a falha poderia se manifestar, a mesma será mascarada. Ela vai se manifestar quando o sinal que deveria estar em zero e for para nível lógico 1 devido a incidência da falha.

O objetivo é verificar se essas falhas sensibilizam o circuito e se elas se propagam para outros módulos, até chegar a um elemento de memória. Também foi observado como o circuito se comportou quando se tratava de um arranjo TMR ou DTMR na presença de falhas, para avaliar qual deles era o mais robusto quanto ao mascaramento de falhas.

Como são muitas simulações, foi desenvolvido um programa na linguagem Java, e através dele foi possível inserir falhas em todos os nodos de forma automática. Para análise dos dados obtidos com as simulações, foi desenvolvido um programa que faz a comparação entre o valor esperado do circuito sem falhas, com o valor obtido na simulação com inserção de falhas. Esse código de análise falhas compara os valores de tensão e informa se a falha se manifestou ou não em determinado estágio do circuito.

4 RESULTADOS

Este Capítulo apresenta inicialmente os resultados obtidos referentes as simulações de inserção de falhas do tipo SET nos módulos e nas arquiteturas TMR e DTMR. Esses resultados estão divididos em duas etapas: Arquitetural e à nível de transistor. Após, são apresentados os resultados das simulações elétricas onde foram comparados tempo de propagação, potência e número de transistores das arquiteturas TMR e DTMR.

4.1 Robustez dos módulos

Nesta Seção são apresentados os resultados obtidos através da simulação isolada dos módulos para verificar sua robustez individual quanto a falhas do tipo SET. Os circuitos combinacionais utilizados para testes de robustez de módulo isolado foram os somadores completos de 1 bit CMOS, TFA e TGA. Também foi testada a robustez do módulo isolado para portas XOR na versão XOR_V1, XOR_V4, XOR_V8 e XOR_V10. Os primeiros resultados mostrados são das topologias da porta lógica XOR e em seguida são exibidos os resultados dos módulos isolados dos somadores.

Os resultados obtidos para os módulos isolados, nas quatro topologias estudadas neste trabalho, da porta lógica XOR estão relacionados na Tabela 4.1, onde é exibida a relação de mascaramento de falhas, ou seja, a quantidade de falhas inseridas pela quantidade de falhas detectadas. A cada simulação foi inserida uma falha em cada nodo para cada combinação de entrada, logo a quantidade de combinações de entrada multiplicada pelo número de nodos do circuito é a quantidade total de falhas inseridas. De acordo com o que é mostrado na Tabela 4.1 a XOR que apresentou mais sensibilidade quando inseridas falhas em módulos isolados foi a XOR_V1, onde foram detectadas 13 falhas dentre as 44 injeções realizadas. Através desses resultados também

foi possível observar quais nodos se mostraram mais sensíveis e quais combinações de entradas deixavam a porta lógica XOR mais vulnerável a ocorrência de falhas.

Tabela 4. 1 - Relação de *fault masking* dos módulos isolados da XOR

Módulos isolados	XOR_V1	XOR_V4	XOR_V8	XOR_V10
Total de falhas inseridas	44	48	36	40
Total de falhas detectadas	13	5	4	4
<i>Fault masking</i>	32%	10%	11%	10%

Fonte: Elaborado pelo próprio autor.

A porta lógica de topologia XOR_V10 foi a que mais apresentou o maior grau de robustez na presença de falha do tipo SET, comparada às demais topologias. Os nodos atingidos e seus respectivos valores de entrada foram listados na Tabela 4.2 para que os nodos e valores de entrada pudessem ser comparados com as demais topologias. A partir dos dados exibidos na Tabela 4.2 é possível observar que os nodos de entrada q1, q2 e os nos de saída s são os nodos mais sensíveis. Quanto a configuração de entrada, a combinação de entradas de valor lógico 11, para XOR de duas entradas, torna os nodos mais sensíveis a ocorrência de falhas.

Tabela 4. 2 - Nodos e entradas com maior sensibilidade a SET por topologia de XOR

Nodos	Entradas	XOR_V1	XOR_V4	XOR_V8	XOR_V10
A	11	x			
B	11	x			
NA	11	x			
NB	11	x			
Q1	00	x	x	x	x
Q1	01		x		
Q1	11	x			
Q2	00	x	x	x	x
Q2	11	x			
S	00	x	x	x	x
S	11	x	x	x	x
VCCBLOCO	11	x			
VSSBLOCO	11	x			
XOR	11	x			

Fonte: Elaborado pelo próprio autor

Os resultados obtidos com os experimentos utilizando os módulos isolados dos somadores CMOS, TFA e TGA são exibidos também levando em conta o número de falhas detectadas pelo número de falhas inseridas. Foram inseridas uma falha por

simulação em cada nodo do somador para todas as entradas, da mesma forma que foram simuladas as topologias da porta lógica XOR. A Tabela 4.3 mostra os resultados obtidos com os experimentos.

Tabela 4.3 - Relação do *fault masking* dos módulos isolados dos somadores

Módulos isolados	FA CMOS	FA TGA	FA TFA
Falhas inseridas	176	122	104
Falhas detectadas	12	16	14
<i>fault masking</i>	7%	13%	13%

Fonte: Elaborado pelo próprio autor

O somador Mirror CMOS foi que apresentou maior robustez, mesmo apresentando maior número de nodos internos, o que aumenta o número de falhas inseridas, este somador mascara os efeitos das falhas nestes nodos internos, apresentando praticamente o dobro da robustez dos somadores TGA e TFA. No caso dos somadores também foi possível identificar os nodos e as combinações de entrada mais sensíveis a ocorrência de SET. No entanto, não houve destaque para nenhuma combinação de entrada que tornasse mais sensível a ocorrência de falha no nodo. Porém, os nodos que apresentaram maior sensibilidade foram os nodos das entradas e da saída dos somadores.

4.2 Robustez das Arquiteturas TMR e DTMR

A Tabela 4.4 apresenta os resultados da inserção de falhas nas arquiteturas TMR e DTMR avaliadas. Para todas elas, foi verificado se a saída em cada um dos estágios estava correta (V), indeterminada (X) ou incorreta (I). É definido como indeterminado sempre que a saída naquele estágio estiver em um nível elétrico entre 40% e 60% de VDD, sendo verificado no próximo estágio se o sinal se propagou corretamente ou a falha se manifestou. Os resultados foram sintetizados por todos os experimentos terem demonstrado o comportamento esperado quanto ao mascaramento de falhas inseridas em um módulo individualmente, confirmando a robustez das técnicas. A tabela apresenta as saídas verificadas após a inserção de uma falha em um dos módulos, sendo observado o comportamento após a saída do Votador, após a escrita do Flip-Flop e após o inversor de carga conectado na saída do flip-flop.

Os resultados apresentados na Tabela 4.4 mostram que tanto a técnica TMR quanto a técnica DTMR mostram o mesmo grau de confiabilidade para o tratamento de falhas tipo SET. Em todas as experiências, o comportamento se mostrou conforme o esperado. Ou seja, a falha se manifestou apenas no nó onde foi inserido, mas não houve propagação dessa falha para os próximos elementos do circuito.

Tabela 4. 4 - Resultados falhas do tipo SET inseridas a nível arquitetural

TMR	Votador NAND	Flip Flop	Inversor
V	100%	100%	100%
X	0	0	0
I	0	0	0
DTMR	Votador NAND	Flip Flop	Inversor
V	100%	100%	100%
X	0	0	0
I	0	0	0

Fonte: Elaborado pelo próprio autor.

Os resultados obtidos com os experimentos mostram que tanto a técnica TMR quanto a técnica DTMR apresentam o mesmo grau de confiabilidade para o tratamento de falhas tipo SET. Em todas as experiências, o comportamento se mostrou conforme o esperado, sendo observado 100% de cobertura de falhas em todos os testes. Ou seja, a falha se manifestou apenas no nó onde foi inserida, mas não houve propagação dessa falha para os próximos elementos do circuito. Os experimentos realizados levaram em conta a inserção de uma falha por simulação em todos os nodos das réplicas dos somadores e réplicas das XORs para todas as combinações de entrada, tanto para TMR quanto para DTMR.

4.3 Tempos de propagação, potência e área

A Tabela 4.5 apresenta os atrasos parciais dos módulos somadores que compõem cada uma das arquiteturas. Comparando os módulos da arquitetura TMR, é possível observar que os valores da média, desvio padrão e máximo são praticamente os mesmos. No entanto, pode-se observar uma pequena oscilação dos valores mínimos de atraso. Já na arquitetura DTMR_1, o módulo somador híbrido foi o que apresentou o maior atraso médio e também o maior atraso máximo, isso deve-se ao fato de o somador híbrido apresentar uma topologia mais complexa tornando alguns caminhos mais

longos. O módulo somador Mirror CMOS apresentou o menor atraso máximo nesta configuração. Por outro lado, o módulo somador TFA apresentou o menor atraso médio e o menor atraso mínimo. Na arquitetura DTMR_2, o módulo somador TFA continuou apresentando a menor média de atrasos e, também, o menor atraso mínimo, porém, apresentou também o maior atraso máximo. O TGA apresentou o menor atraso máximo. Já o módulo somador Mirror CMOS, nesta configuração, apresentou o maior atraso médio e o maior atraso mínimo.

Tabela 4. 5 - Tempos de propagação dos módulos internos das arquiteturas

Tempos de Propagação dos módulos internos das arquiteturas (ps)					
Arquiteturas		Média	Desvio padrão	Máximo	Mínimo
TMR	CMOS 1	43,40	17,90	70,10	15,30
	CMOS 2	43,50	18,00	70,10	14,80
	CMOS 3	43,40	18,10	70,00	14,40
	TFA 1	47,4	35,8	133	6,3
	TFA 2	46,4	36,2	133	6,3
	TFA 3	45,7	36,9	133	6,3
DTMR_1	CMOS	41,70	15,60	74,90	16,70
	HIBRIDO	79,00	108,00	376,00	7,18
	TFA	35,60	32,10	103,00	2,50
DTMR_2	CMOS	41,30	14,20	76,00	15,60
	TGA	32,20	17,10	64,60	6,46
	TFA	29,40	19,90	79,30	5,58

Fonte: Elaborado pelo próprio autor.

Para mensurar o quanto de área cada arquitetura ocupa utilizamos a soma do número de transistores de cada módulo somador mais a quantidade de transistores do módulo votador. Essas informações podem ser vistas claramente na Tabela 4.6.

Tabela 4. 6 - Relação entre os atrasos, potência e número de transistores de somadores.

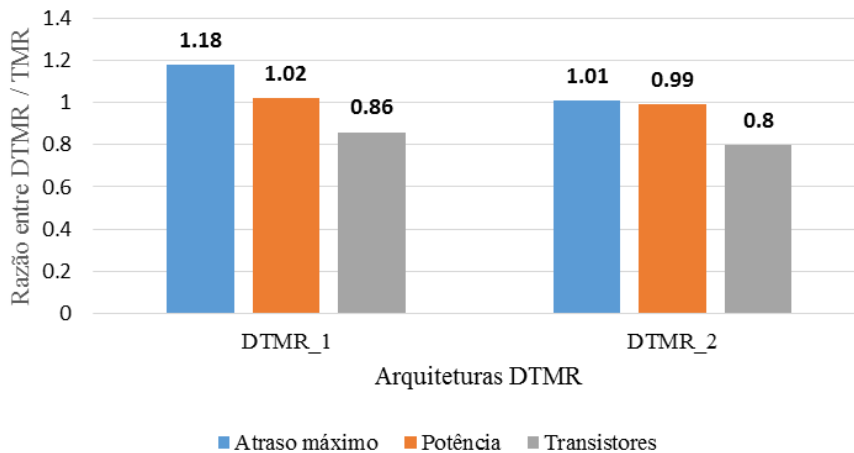
Arquitetura	Atrasos (ps)				Potência (μW)	Nº de transistores
	Média	Desvio padrão	Máximo	Mínimo		
TMR_CMOS	43,40	19,70	85,90	31,10	76,5	102
TMR_TFA	51,9	37,2	174	4,5	53,3	66
DTMR_1	58,20	26,40	101,53	22,90	77,7	88
DTMR_2	55,40	17,90	86,40	27,50	75,6	82

Fonte: Elaborado pelo próprio autor

Através dos dados expostos na Tabela 4.6 podemos observar que o menor atraso médio entre as arquiteturas estudadas foi o da técnica TMR_CMOS. Isso provavelmente ocorreu porque, nessa primeira fase de testes, estamos utilizando o dimensionamento mínimo de para todos os circuitos. No entanto, o maior atraso foi observado na arquitetura DTMR_1, cerca de 20% superior a TMR, devido aos atrasos do somador híbrido. A arquitetura TMR_TFA alcançou o maior atraso máximo e o menor atraso mínimo entre as arquiteturas estudadas. Com relação à área, a arquitetura TMR_TFA foi a que obteve maior ganho de área, ficando com 36 transistores a menos que a arquitetura TMR_CMOS.

Os resultados obtidos em atraso máximo, potência e área nas arquiteturas DTMR_1 e DTMR_2 foram normalizados com base na arquitetura TMR_CMOS e obtivemos o gráfico exposto na Figura 4.1. É possível observar a redução de 14% e 20% no número de transistores, para as arquiteturas DTMR_1 e 2, respectivamente. E que embora a arquitetura DTMR_1 tenha inserido uma penalidade de 18% no atraso, a potência continuou semelhante a TMR clássica.

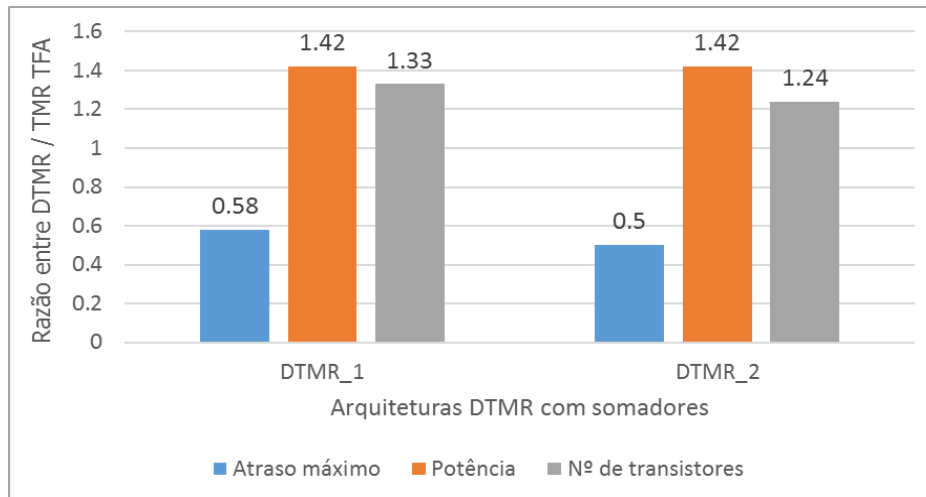
Figura 4.1 - Resultados normalizados de atraso máximo, potência e área com base no TMR_CMOS



Fonte: Elaborado pelo próprio autor

O mesmo processo de normalização foi aplicado a DTMR_1 e DTMR_2 com base nos resultados de TMR_TFA e os resultados são exibidos no gráfico da Figura 4.2. Através da análise dos dados foi possível concluir que houve redução de 50% nos atrasos e acréscimo de 42% em potência. Também houve acréscimo significativo em número de transistores.

Figura 4. 2 - Resultados normalizados de atraso máximo, potência e área com base no TMR_TFA



Fonte: Elaborado pelo próprio autor

A Tabela 4.7 apresenta os resultados dos tempos de propagação dos módulos internos dos arranjos TMR da porta lógica XOR. O arranjo TMR que apresentou os menores valores de atraso máximo foi o TMR composto por réplicas XOR_V4. Com relação a pior performance entre os arranjos é possível destacar o TMR XOR_V10 que contou com os maiores valores de atraso máximo.

Os mesmos experimentos foram aplicados as simulações DTMR para portas lógicas XOR, onde foram medidos os tempos de propagação dos módulos internos que fazem parte do arranjo. Os resultados obtidos estão disponíveis na Tabela 4.8. Com base nos resultados o arranjo DTMR_C composto por réplicas da XOR_V1, XOR_V8 e XOR_V10 apresentou os melhores valores de atrasos máximos e mínimos. Também foi possível verificar que a XOR_V4 quando em arranjo com portas lógicas XOR de outras topologias acaba apresentando valores piores com relação aos atrasos. O atraso máximo fica mais elevado devido a capacitância de saída.

Tabela 4. 7 - Tempos de propagação dos módulos internos da TMR XOR

Tempo de propagação dos módulos internos de TMR XOR					
Arquiteturas		Média (ps)	Desvio padrão (ps)	Máximo (ps)	Mínimo (ps)
XOR_V1	Replica 1	19,6	6,38	29,4	9,1
	Replica 2	19,5	6,39	29,3	8,9
	Replica 3	19,4	6,40	29,3	8,8
	Votador	36,3	1,10	49,3	17,5
XOR_V4	Replica 1	17,4	4,69	25,8	11,9
	Replica 2	17,2	4,88	25,8	11,1
	Replica 3	17,0	5,11	25,7	10,2
	Votador	39,0	8,45	47,9	25,8
XOR_V8	Replica 1	22,9	8,43	35,9	13,4
	Replica 2	22,8	8,44	35,9	13,3
	Replica 3	22,7	8,46	35,9	13,2
	Votador	42,1	14,3	63,1	20,0
XOR_V10	Replica 1	22,8	9,38	37,5	12,7
	Replica 2	22,8	9,41	37,5	12,6
	Replica 3	22,7	9,44	37,4	12,5
	Votador	39,2	11,6	60,3	19,0

Fonte: Elaborado pelo próprio autor

Tabela 4. 8 - Tempos de propagação dos módulos internos de DTMR XOR

Tempo de propagação dos módulos internos de DTMR XOR					
Arquiteturas		Média	Desvio padrão	Máximo	Mínimo
DTMR A	XOR_V1	19,3	3,17	22,8	13,3
	XOR_V4	23,4	6,67	35,7	15,1
	XOR_V8	17,9	5,62	27,0	9,4
	Votador	38,3	8,43	48,7	25,1
DTMR B	XOR_V1	19,5	4,04	24,8	11,5
	XOR_V4	23,1	6,41	35,8	15,3
	XOR_V10	17,3	5,61	24,5	7,14
	Votador	40,1	7,55	48,7	25,3
DTMR C	XOR_V1	20,9	5,55	29,0	11,6
	XOR_V8	21,2	7,01	28,0	11,1
	XOR_V10	20,1	7,45	29,3	10,6
	Votador	38,0	9,44	48,5	17,4
DTMR D	XOR_V4	24,0	5,99	34,6	15,5
	XOR_V8	20,4	6,39	27,7	11,1
	XOR_V10	19,4	6,98	28,7	7,1
	Votador	41,5	7,03	50,0	30,1

Fonte: Elaborado pelo próprio autor.

Pensando em relacionar os resultados obtidos de tempos de propagação, para que fosse possível mensurar o impacto dos tempos de propagação em cada técnica, foi feita a média dos resultados para cada arranjo, TMR ou DTMR. Os resultados são exibidos na Tabela 4.9. Assim como nas avaliações das técnicas TMR e DTMR com somadores, a potência se manteve semelhante para todas as arquiteturas avaliadas.

A implementação TMR_XOR_V4 apresentou o melhor resultado para atrasos, sendo aproximadamente 40% melhor do que a arquitetura TMR_XOR_V8 e TMR_XOR_V10, considerando o pior atraso. Em relação as arquiteturas T_XOR_V1, DTMR A, DTMR B e DTMR C, a arquitetura TMR_XOR_V4 apresenta um atraso crítico no mínimo 10% inferior. Considerando a diversidade de projeto, das alternativas avaliadas as versões DTMR B e C apresentam bons resultados, diferindo somente em 10% nos atrasos, e com potências similares.

Com relação ao número de transistores, a última coluna exibe a soma da quantidade de transistores do módulo mais a soma de transistores do somador. O objetivo é ter uma ideia de quanto de área pode ser ocupada utilizando uma ou outra configuração, de arranjos ou de técnica de tolerância a falhas utilizada. Foi possível observar que o menor atraso máximo, igual a 25,5 ps ocorreu em TMR_XOR_V4, que foi o arranjo que apresentou a maior área, contando com 45 transistores. Já no arranjo TMR_XOR_V10 foi observado o maior atraso máximo igual à 35,5 ps e a menor área aproximada igual a 33 transistores.

Com relação a número de transistores em arranjos TMR, o TMR_XOR_V4 foi o que apresentou o maior número de transistores enquanto que o TMR_XOR_V10 foi o que apresentou menor número de transistores no arranjo, contando ambos com o número de transistores do módulo votador. Já com relação a quantidade de transistores apenas para módulos DTMR, o que apresentou o maior número de transistores foi a configuração do DTMR A, e a menor configuração de transistores pode ser vista em DTMR C, contando com apenas 37 transistores.

Tabela 4. 9 - Relação entre os atrasos, potência e número de transistores das XORs

Tempo de Propagação (ps)						
Arquitetura	Média	Desvio padrão	Máximo	Mínimo	Potência (μW)	N ° de transistores
T_XOR_V1	26,5	6,91	29,4	16,5	16,65	42
T_XOR_V4	17,2	6,52	25,5	15,9	16,38	45
T_XOR_V8	26,4	7,96	35,3	13,0	16,83	36
T_XOR_V10	27,1	6,80	35,5	16,1	16,83	33
DTMR A	20,2	5,15	28,5	12,6	16,38	41
DTMR B	19,9	5,35	28,4	11,3	16,60	40
DTMR C	20,7	6,67	28,8	11,1	17,01	37
DTMR D	21,3	6,45	30,3	11,2	16,83	38

Fonte: Elaborado pelo próprio autor.

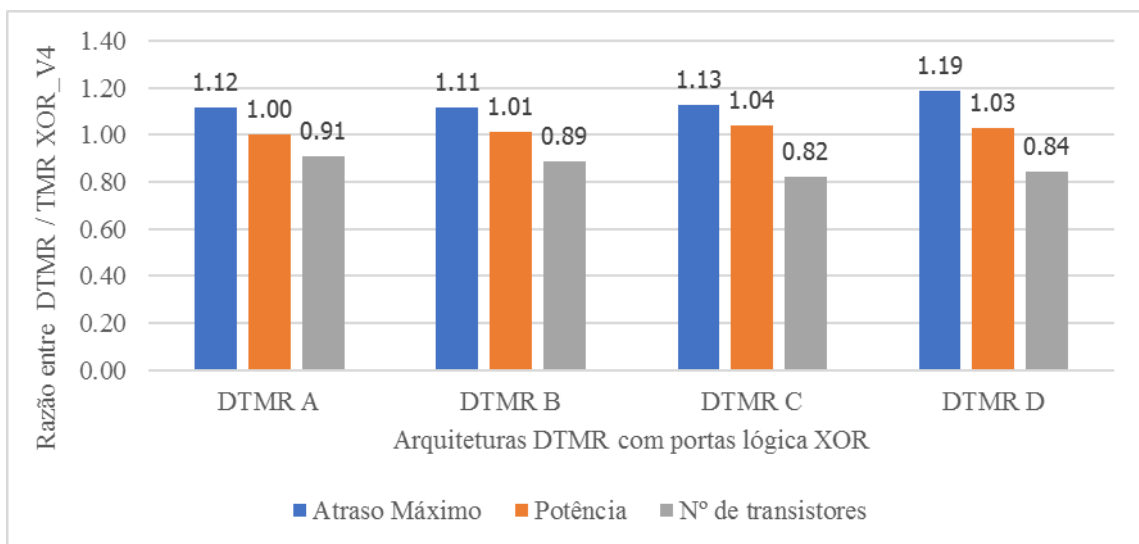
De acordo com os dados apresentados foi possível observar que o melhor tempo de propagação em TMR foi apresentado no arranjo TMR com réplicas da XOR_V4 com atraso máximo igual a 25 ps. No entanto, o pior tempo de propagação dos arranjos TMR foi observado no TMR com réplicas da XOR_V10 que apresentou atraso máximo de 35 ps. Com relação ao DTMR o melhor tempo de propagação de um arranjo DTMR foi o apresentado pelo DTMR_C composto por réplicas de XOR_V1, XOR_V8 e XOR_V10, onde foram encontrados 29 ps, 28 ps e 29,3 ps. Ainda foi possível analisando os resultados concluir que no melhor tempo de um arranjo DTMR não conta com um circuito XOR_V4, que foi o melhor tempo encontrado nos arranjos TMR. Outro fator relevante é ressaltar que quando o circuito XOR_V4 é combinado com outros circuitos no DTMR, ele tende a apresentar atrasos maiores devido a capacitância da saída. Dessa forma, o arranjo DTMR_C apresenta atrasos equivalentes aos atrasos encontrados para o arranjo TMR composto por réplicas XOR_V4. Também foi possível identificar que o pior arranjo DTMR, ou seja, aquele que apresentou os maiores tempos máximos de atraso foi o DTMR_D, composto pelas topologias XOR_V4, XOR_V8 e XOR_V10.

Analisando os resultados obtidos para portas lógica XOR comparando atrasos, potência e número de transistores foi possível identificar que além de apresentar ganho em atrasos, o arranjo TMR XOR_V4 ainda apresenta ganho em potência, mas perde em área. Apresentando a maior área equivalente entre os arranjos TMR_XOR. Quando a análise é estendida para DTMR é possível observar que DTMR_A apresenta um comportamento semelhante, obtendo ganho em atraso e potência, porém com perda em relação à área equivalente. Já o arranjo DTMR_C apresenta um atraso e potência um pouco maior e ganho significativo com relação à área equivalente. Quando o projetista

tem acesso a essas informações, então ele precisa mensurar o que será mais importante para o projeto, consumo de energia ou área ocupada.

Da mesma forma que os resultados de arranjos TMR e DTMR aplicados aos circuitos somadores foram normalizados pelo arranjo TMR que apresentou o melhor resultado com relação aos atrasos, o mesmo foi aplicado aos experimentos realizados com portas lógicas XOR. Na Figura 4.3 são exibidos os resultados obtidos pela normalização dos arranjos DTMR de portas lógica XOR pelo TMR_XOR_V4.

Figura 4.3 - Resultados normalizados de DTMR por TMR XOR_V4



Fonte: Elaborado pelo próprio autor.

Na Figura 4.3 o melhor resultado foi apresentado por DTMR B, onde a potência se manteve equivalente, houve um crescimento em atrasos de 11% e uma redução de 11% em área. O segundo melhor resultado obtido foi o DTMR_C que obteve uma perda de 18% em área e um acréscimo de 13% em atrasos, mantendo a potência equivalente. Já o pior caso ficou com o DTMR_D, onde apresentou um acréscimo de 19% em atrasos.

CONCLUSÕES

Com o aumento na capacidade de integração dos circuitos foi possível desenvolver dispositivos cada vez menores e mais complexos. Dessa forma, surgiram inúmeros desafios aos projetistas de sistemas digitais e embarcados, entre eles, circuitos mais sensíveis a falhas. O uso da redundância é uma das técnicas mais utilizadas para tolerar falhas em circuitos integrados, sendo a técnica TMR uma das mais utilizadas devido a sua comprovada robustez. No entanto, apesar de ser uma técnica robusta, apresenta alguns pontos negativos, como gerar um grande acréscimo de área do circuito integrado e por sua vez aumento no consumo de energia. Nesse cenário, vários estudos foram feitos a respeito da técnica DTMR ou TMR diversificado, que nada mais é que utilizar três módulos diferentes, mas que executam a mesma função. A grande maioria das pesquisas que exploram as técnicas TMR e DTMR são aplicadas a FPGA. Este trabalho tem como objetivo comparar o comportamento da técnica TMR e DTMR, a nível elétrico, sob a influência de falhas do tipo SET.

O desenvolvimento deste trabalho foi dividido em seis etapas: 1) Simulação elétrica para comparar atrasos, potência e número de transistores de somadores completos de 1 bit. 2) Simulação elétrica para comparar atrasos, potência e número de transistores de topologias da porta lógica XOR. 3) Implementação de módulos capazes de automatizar a injeção de falhas em circuitos combinacionais. 4) Simulação elétrica de inserção de falha do tipo SET em somadores completos de 1 bit. 5) Simulação elétrica de inserção de falha do tipo SET em topologias de XORs mais sensíveis a SET. 6) Análise de resultados obtidos. Para a continuidade desse trabalho foi pensado integrar esses módulos automatizados e desenvolver uma ferramenta de apoio a decisão.

Através dos resultados encontrados podemos concluir que a grande vantagem da utilização do DTMR com relação ao TMR está no uso de diversidade de projeto sem causar um grande impacto nas medidas de atrasos e potência. Com relação à área não

podemos afirmar que haverá ganho, pois dependerá do arranjo escolhido. O que pode acontecer é que exista um ganho quando a área complementar a escolhe onde outros requisitos são os principais, como ganhos em potência, desempenho e robustez.

As simulações com a inserção de falhas do tipo SET mostraram que o DTMR é tão robusto quanto o TMR para mascamento de falhas. Sendo o circuito votador o ponto sensível destas técnicas.

O próximo passo é uma análise mais detalhada dos resultados e o desenvolvimento da ferramenta computacional de apoio a decisão, que compara TMR e DTMR. Mais detalhes dessa ferramenta podem ser vistos na Seção 5.1 sobre trabalhos futuros. Resultados parciais deste trabalho foram publicados no congresso ICECS 2017 e no WCAS2018.

5.1 Trabalhos Futuros

Uma possibilidade de trabalho futuro é a conclusão do desenvolvimento de uma ferramenta para comparar TMR e DTMR quanto a sua robustez na presença de falhas do tipo SET. Partes desta ferramenta foram desenvolvidas para automatizar os experimentos apresentados nesta dissertação. Além disso, este trabalho também realizou o levantamento de requisitos para a implementação. Já temos algumas informações de parâmetros que poderão ser alterados utilizando a ferramenta. Através dela será possível selecionar o tipo de técnica que será aplicada (TMR ou DTMR), os circuitos somadores, a tecnologia preditiva, frequência de clock, nodos onde serão inseridas as falhas. Também será possível escolher outros modelos de votador.

O primeiro passo para o desenvolvimento da ferramenta é a análise de requisitos. A priorização de requisitos identifica que a ferramenta deve ser prática e fácil de usar. Além disso, também deve ser leve e consumir pouco processamento, para que seja possível realizar um grande número de simulações consecutivas. Foi definido que o desenvolvedor pode escolher qual técnica aplicar, quais circuitos simular, nós, onde falhas transitórias de evento único (SET) podem ser injetadas, valor lógico das entradas, a tecnologia, entre outras configurações de dados.

Como saída, a ferramenta retorna um relatório contendo uma avaliação comparativa. As métricas de relatório indicam qual técnica foi capaz de mascarar mais falhas, a energia consumida, os atrasos máximos e o número de transistores requeridos por cada arranjo de circuito e técnica de replicação. Com base nesses resultados, é possível definir qual técnica é a mais robusta como o mascaramento de falhas do SET.

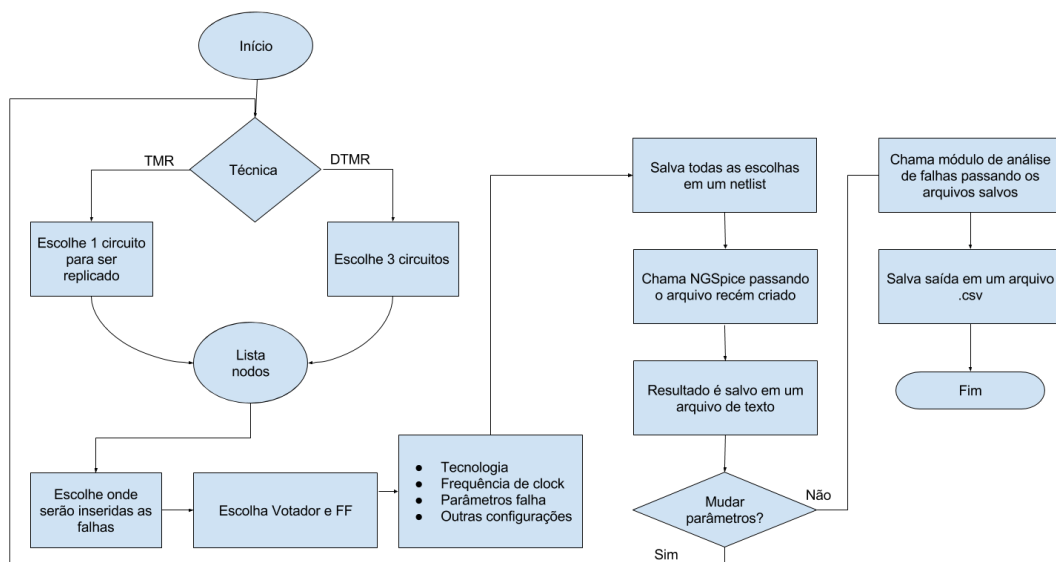
De acordo com os requisitos selecionados, a ferramenta recebe como entrada um arquivo de configuração em formato de texto contendo os principais campos configuráveis. Os principais parâmetros, entre outros, disponíveis no arquivo de configuração, são: a) a técnica de tolerância a falhas a ser aplicada; b) o (s) circuito (s) original (ais); c) a tensão da fonte; d) os nomes das entradas para cada circuito; e e) a tecnologia adotada

De acordo com o fluxograma, o primeiro passo é a escolha da técnica de tolerância a falhas. Depois de escolher a técnica desejada, o próximo passo é inserir a entrada dos circuitos. O projetista pode selecionar um único circuito que pode ser replicado no caso do TMR ou três circuitos diferentes no caso do DTMR. Nesse estágio, a ferramenta lê o arquivo de configuração e carrega os parâmetros de entrada. Após a configuração inicial, os parâmetros de simulação de falha SET são definidos. O projetista pode escolher simular primeiro o circuito com falha (ou circuitos ao executar o modo DTMR), quais nós serão afetados por falhas, o número de simulações e tais parâmetros SET como o pulso de corrente, o começo e fim do pulso transiente, intensidade e duração. Automaticamente, os nós internos dos circuitos são identificados. O número selecionado de falhas é inserido por simulação nos nós internos da disposição dos dispositivos do circuito, considerando todas as possibilidades de entrada. A partir da leitura do arquivo de configuração, uma descrição elétrica do arranjo a ser simulado é criada usando a linguagem SPICE, com os circuitos especificados nele. A partir da leitura do arquivo de configuração, uma descrição elétrica do arranjo a ser simulado é criada usando a linguagem SPICE, com os circuitos especificados nele. As configurações de pulso atuais para a simulação de falha do tipo SET também serão aplicadas. Após gerar a descrição elétrica do arranjo, a ferramenta chama o simulador elétrico de fonte aberta NGSpice, passando a descrição do circuito. Assim, as simulações ocorrem sucessivamente percorrendo os nós especificados no arquivo de configuração.

Ao final de toda a rodada de simulação, um relatório final é gerado com dados sobre falhas e as características elétricas são analisadas. Além disso, uma breve noção sobre o número de transistores de cada arranjo é informada. O relatório final auxilia na identificação de quais configurações representam um ganho de área. Outras informações disponíveis neste relatório são o número de falhas mascaradas por cada técnica, em quais nós as falhas foram manifestadas e em quais situações a falha é propagada. Eles auxiliam os projetistas na verificação de qual técnica é a mais robusta e na escolha de qual tipo de técnica melhor atenderá aos requisitos do projeto

A seguir, na Figura 5.1 é possível observar uma proposta de fluxograma que descreve o funcionamento da ferramenta.

Figura 5.1 - Fluxograma da ferramenta de comparação TMR e DTMR



Fonte: Elaboração do próprio autor

REFERÊNCIAS

AGUIAR, Q. A. **Radiation Robustness of XOR and Majority Voter Circuits at FinFET Technology under Variability**. Dissertação. Mestrado em Microeletronica. Universidade Federal do Rio Grande do Sul, Porto Alegre, 2017.

ALIOTO, M.; PALUMBO, G. Analysis and comparison on full adder block in submicron technology. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, [S.l.], v. 10, n. 6, p.806-823, dez. 2002.

ASHRAF, R. A. et al., Design-for-Diversity for Improved Fault-Tolerance of TMR Systems on FPGAs, in **International Conference on Reconfigurable Computing, Cancun**, 2011, pp. 99-104.

AVIZIENIS, a.; Kelly, j. P. J. **Fault tolerance by design diversity:concepts and experiments**. 67–80p. v.17, n.8, 1984.

AVIZIENIS, a; The N-Version Approach to Fault-Tolerant Software, in **IEEE Transactions on Software Engineering**, vol. SE-11, no. 12, pp. 1491-1501, Dec. 1985. doi: 10.1109/TSE.1985.231893

BARTH, J. Applying Computer Simulation Tools to Radiation Effects Problems. In: **IEEE NUCLEAR SPACE RADIATION EFFECTS CONFERENCE, NSREC**, 1997. Anais. . . [S.l.: s.n.], 1997.

BAUMANN, R. C. Soft Errors in Advanced Semiconductor Devices - Parte I: The Three Radiation Sources. **IEEE Transactions on Device and Materials Reliability**, v. 1, n. 1, Mar 2001.

BAUMANN, R. C. Radiation-Induced Soft Errors in Advanced Semiconductor Technologies. **IEEE Transactions on Devices and Materials Reliability**, New York, v.5, n.3, p.305-316, Set, 2005.

BECKETT, P. A fine-grained reconfigurable logic array based on double gate transistors. **IEEE International Conference on Field-Programmable Technology (FPT)**, [S.l.], p.260-267, dez. 2002.

BORGES, G. M. et al., Diversity TMR: Proof of Concept in a Mixed-Signal Case, in **IEEE Latin American Test Workshop**, Pule del Este, 2010, pp. 1-6.

BORKAR, S. Designing reliable systems from unreliable components: the challenges of transistor variability and degradation. **Micro, IEEE**, v. 25, n. 6, p. 10-16, 2005.

BORKAR, S. et al. Parameter variations and impact on circuits and microarchitecture. **Design Automation Conference**, 2003. 338-342.

CHOI, G. S.; IYER, R. K. and CARRENO V. A. , "Simulated fault injection: a methodology to evaluate fault tolerant microprocessor architectures," in *IEEE Transactions on Reliability*, vol. 39, no. 4, pp. 486-491, Oct 1990. doi: 10.1109/24.58726

CHANG, C. H.; GU, J.; ZHANG, M. A review of 0.18 μ m full adder performances for tree structured arithmetic circuits. **IEEE Trans. on VLSI Systems**, v. 13, n. 6, p.686-695, jun. 2005.

CHAU, S. N.; J. SMITH and T. TAI, A design-diversity based fault-tolerant COTS avionics bus network, **Pacific Rim International Symposium on Dependable Computing**, Seoul, 2001, pp. 35-42. doi: 10.1109/PRDC.2001.992677

CHIELLE, E. **CFT-tool: ferramenta configurável para aplicação de técnicas de detecção de falhas em processadores por software**. 2012.98f. Dissertação (Mestrado em Computação) - Instituto de Informática, UFRGS, Porto Alegre.

CHIELLE, E.; AZAMBUJA, J. R.; BARTH, R. S.; KASTENSMIDT, F. L. Improving error detection with selective redundancy in software-based techniques. In: 2013 **14th Latin American Test Workshop LATW**, Cordoba. 2013

CONDESSA, M. S. **Uma análise de como a diversidade do hardware afeta a susceptibilidade a SEU e SET em circuitos tolerantes à falhas**.2009. 67f. Trabalho de Conclusão de Curso. Instituto de Informática, UFRGS, Porto Alegre.

DODD, P. E.; SHANEYFELT, M. R.; SCHWANK, J.R.; FELIX; J.A. Current and Future Challenges in Radiation Effects on CMOS Electronics. **IEEE Transactions on Nuclear Science**, v. 57, n 4, p. 1747-1763, ago. 2010.

ELMENDORF, W. R. Fault-tolerant programming. In: **International Symposium on Fault-tolerant Computing (FTCS-2)**, 2., 1972. Proceedings.. [S.l.: s.n.], 1972. v.31, p.79-83.

FRANCO, D. T. **Fiabilité Du Signal des Circuits Logiques Combinatoires sous Fautes Simultanées Multiples**, Doctoral Thesis, l'École Nationale Supérieure des Télécommunications, França, 2008.

FLOYD, T. L. **Sistemas Digitais Fundamentos e Aplicações**. 9. ed. Porto Alegre: Bookman, 2007

FRANK, H. de S. **Avaliação de Atraso, Consumo e proteção de somadores tolerantes à falhas**. 2011. 165f. Dissertação (Mestrado em Microeletrônica). Instituto de Informática, UFRGS, Porto Alegre.

GILL, B.; PAPACHRISTOU, C.; WOLFF, F.; SEIFERT, N. Node sensitivity analysis for soft errors in CMOS logic. In: **TEST CONFERENCE**, 2005. PROCEEDINGS. ITC 2005. IEEE INTERNATIONAL, 2005. Anais. . . [S.l.: s.n.], 2005. p.9 pp. –972.

GOMES, I. A. C. **Uso de Redundância Modular Tripla Aproximada para Tolerância a falhas em circuitos digitais**. 2014. 91f. Dissertação (Mestrado em Computação). Instituto de Informática, UFRGS, Porto Alegre.

GOMES, I. A. C. and KASTENSMIDT F. G. L., Reducing TMR overhead by combining approximate circuit, transistor topology and input permutation approaches, **2013 26th Symposium on Integrated Circuits and Systems Design (SBCCI)**, Curitiba, 2013, pp. 1-6. doi: 10.1109/SBCCI.2013.6644856

GONÇALVES, M.; SAQUETTI, M.; KASTENSMIDT, F.; AZAMBUJA, J. R. A low-level software-based fault tolerance approach to detect SEUs in **GPUs' register files**. **Microelectronics Reliability**, v. 1, p. 1, 2017.

HIARI, O; SADEH, W; RAWASHDEH, O. Towards single-chip diversity TMR for automotive applications", 2012 **IEEE Internacional Conference on Electro/Information Technology (EIT)**, 2012.

JOHNSON, B.W. **An Introduction to the design and analysis of Fault-Tolerant System**, Prentice Hall, New Jersey, 1996.

KASTENSMIDT, F.; CARRO, L.; REIS, R. **Fault-Tolerance Techniques for SRAM-based FPGA**. Netherlands. Springer, 2006. 184p.

KOREN, I.; Krishna, C. M. **Fault Tolerant Systems**. [S.l.]: Morgan Kaufmann, 2010. 378p.

LALA, P. K. Self-Checking and Fault-Tolerant Digital Design. [S.l.: s.n.], 2001.

LIEBEL, E. **Avaliação da Robustez de diferentes topologias de circuitos votadores**. 2016. 62 f. Dissertação. Mestrado em Engenharia de Computação. Universidade Federal do Rio Grande, Rio Grande, 2016.

MESSENGER, G. C. Collection of Charge on Junction Nodes from Ion Tracks. **IEEE Transactions on Nuclear Science**, [S.l.], v.29, p.2024–2031, 1982.

MUNTEANU, D.; AUTRAN, J.L. Modeling and simulation of single-event effects in digital devices and ICs. **IEEE Transactions on Nuclear Science**, [S.l.:s.n.], v.55, n.4, p.1854-1878, 2008

NAVI, K. et al. A novel low-power full-adder cell for low voltage. Integration, **The VLSI Journal**, [S.l.], v. 42, n. 4, p.457-467, set. 2009. Elsevier BV.

NGSPICE. <<http://ngspice.sourceforge.net/>> acessado em 04 de agosto de 2017.

NICOLESCU, B.; VELAZCO, R. Detection soft errors by a purely software approach: method, tools and experimental results. **Design, Automation and Test in Europe Conference and Exhibition**, p57-62, 2003.

NORMAND, E. Single event upset at ground level. Nuclear Science, **IEEE Transactions on**, [S.l.], v.43, n.6, p.2742 –2750, dec. 1996

PEDRONI, V. A. **Eletrônica Digital Moderna e VHDL**. Rio de Janeiro: Campus, 2010.

PIGUET, C. **Low-Power CMOS Circuits – Technology, Logic Design and CAD Tools**. Neuchatel, Switzerland: Taylor & Francis Group, 2006.

PTM. Predictive Technology Model. Disponível em: <<http://ptm.asu.edu>>. Acesso em: 03 ago. 2017.

RITER, R. Modeling and testing a critical fault-tolerant multi-process system, **Twenty-Fifth International Symposium on Fault-Tolerant Computing. Digest of Papers**, Pasadena, CA, USA, 1995, pp. 516-521. Doi: 10.1109/FTCS.1995.466946

SAMAR K. SAHAR. Compact MOSFET modeling for process variability-aware VLSI circuit design. **IEEE Access**, 2, 2014. 104-115.

SHIVAKUMAR, P.; KISTLER, M.; KECKLER, S.; BURGER, D.; ALVISI, L. Modeling the effect of technology trends on the soft error rate of combinational logic. PROCEEDINGS. INTERNATIONAL CONFERENCE ON, 2002. Anais. . . [S.l.: s.n.], 2002. p.389 – 398. .In: **DEPENDABLE SYSTEMS AND NETWORKS**, 2002. DSN 2002.

SILVA, F. G. R. G.; BUTZEN, P. F.; MEINHARDT, C. Portas Lógicas XOR: Impacto da Variabilidade PVT no Desempenho para Tecnologia de 32nm - XOR Logic Gates: PVT Variability Impact on Performance at 32nm. **Revista Junior de Iniciação Científica em Ciências Exatas e Engenharia**, v. 1, p. 29-35, 2016

TAMBARA, L. A, et al. Evaluating the effectiveness of a diversity TMR scheme under neutrons. In: **Europeanco Nference On Radiation And Its Effects On Components And Systems (RADECS)**, 2013.

THILAK, K. R. and GAYATHRI, S. "Fault coverage analysis using fault model and functional testing for DPM reduction," *2015 International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT)*, Mandya, 2015, pp. 76-81. doi: 10.1109/ERECT.2015.7498991

WEATHERFORD, T.; A review of see charge collection processes. **In IEEE Nuclear and Space Radiation Effects Conference Short Course**, Phoenix, AZ, 2002.

UZNANSK, S., Gasiot, G., Roche, P., Tavernier, C. and J.-L. Autran, "Single event upset and multiple cell upset modeling in commercial bulk 65-nm cmos srams and flip-flops," **IEEE Transactions on Nuclear Science**, vol. 57, no. 4, pp. 1876–1883, 2010.

ZIMPECK, A. L. **Falhas stuck-open e stuck-on: Analise do comportamento em nanotecnologias e desenvolvimento de uma ferramenta de simulação de falhas em portas logicas CMOS**. Trabalho de Conclusão de Curso em Engenharia de Computação. 99p. Universidade Federal do Rio Grande. Rio Grande, 2013.

WANG, F and AGRAWAL, V. D. Soft Error Rate Determination for Nanometer CMOS VLSI Logic, **2008 40th Southeastern Symposium on System Theory (SSST)**, New Orleans, LA, 2008, pp. 324-328. doi: 10.1109/SSST.2008.4480247

WEBER, T. S. Um roteiro para exploração dos conceitos básicos de tolerância a falhas, **Textos didáticos em tolerância a falhas**, Porto Alegre, 2002.