

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

Dissertação de Mestrado

**Análise da confiabilidade de circuitos combinacionais ao
proteger as portas críticas do circuito**

Marcio Oliveira da Rocha

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Paulo Francisco Butzen
Co-orientador: Prof. Dr. Denis Teixeira Franco

Rio Grande, 2020

Ficha Catalográfica

R672a Rocha, Marcio Oliveira da.

Análise da confiabilidade de circuitos combinacionais ao proteger as portas críticas do circuito / Marcio Oliveira da Rocha. – 2020.
68 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2020.

Orientador: Dr. Paulo Francisco Butzen.

Coorientador: Dr. Denis Teixeira Franco.

1. Confiabilidade 2. *Hardening* Seletivo 3. SPR 4. Circuitos Combinacionais I. Butzen, Paulo Francisco II. Franco, Denis Teixeira III. Título.

CDU 004.312.2

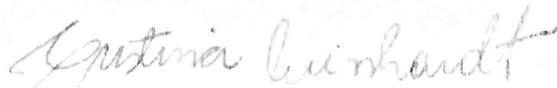
Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

DISSERTAÇÃO DE MESTRADO

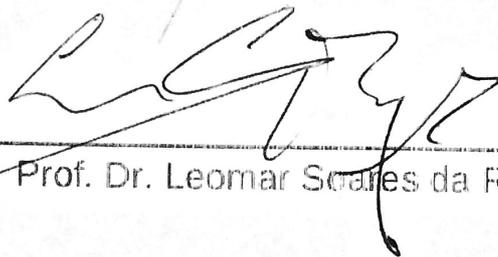
ANÁLISE DA CONFIABILIDADE DE CIRCUITOS COMBINACIONAIS AO
PROTEGER AS PORTAS CRÍTICAS DO CIRCUITO

Marcio Oliveira da Rocha

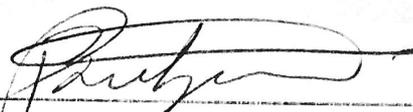
Banca examinadora:



Profa. Dra. Cristina Meinhardt



Prof. Dr. Leomar Soares da Rosa Junior



Prof. Dr. Paulo Francisco Butzen
Orientador



Prof. Dr. Denis Teixeira Franco
Coorientador

AGRADECIMENTOS

Em primeiro lugar gostaria de agradecer a Deus pela minha vida e por me dar força para encarar os momentos difíceis. Aos meus pais, que sempre se dedicaram e se sacrificaram para que eu estudasse e tivesse mais oportunidades, já que eles não as tiveram. À minha esposa e minhas filhas por compreender os períodos de minha ausência. Agradeço também aos meus orientadores, pelas orientações e dicas que são tão importantes para o orientando. E finalmente, aos meus colegas de grupo pelas dicas, material de estudo, circuitos, ferramentas.

*o que sabemos
é uma gota,
o que ignoramos
é um oceano*
— ISAAC NEWTON

RESUMO

ROCHA, Marcio Oliveira da. **Análise da confiabilidade de circuitos combinacionais ao proteger as portas críticas do circuito.** 2020. 68 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

A evolução da tecnologia CMOS melhorou a produtividade e o desempenho na indústria de semicondutores, produzindo circuitos mais rápidos e com menor consumo de energia. Essa conquista foi possível, devido à redução nas dimensões dos transistores, permitindo a integração de bilhões desses dispositivos em um circuito integrado. No entanto, a alta densidade dos transistores associada à baixa tensão de alimentação, resulta em sistemas com maior sensibilidade a falhas de radiação. Partículas alfa e nêutrons, presentes no ambiente, são as principais causas de falhas de radiação em circuitos nanométricos. Essa ocorrência de falhas tem um impacto negativo na confiabilidade do circuito. Portanto, a confiabilidade não pode ser negligenciada em circuitos nanométricos, e técnicas para análise e melhoria da confiabilidade do circuito são obrigatórias. As técnicas de análise normalmente, calculam somente a confiabilidade média do circuito e desconsideram uma análise do seu limite inferior, devido à complexidade temporal associada com a análise. Entretanto, a diferença entre ambas pode ser ordens de magnitude, e caso os valores estejam abaixo de limites aceitáveis, a confiabilidade deve ser melhorada. A melhoria da confiabilidade é alcançada com o uso de redundância. Todavia, esta não deve gerar custos excessivos em outros parâmetros de projeto como área e potência. Uma forma de atender esses requisitos, é identificando as partes mais sensíveis de um circuito e aplicando redundância somente nestas partes. Neste contexto, pretende-se analisar a confiabilidade dos circuitos, ao proteger as partes mais sensíveis do circuito. Além disso, para obtenção do limite inferior de confiabilidade, foi desenvolvido um algoritmo baseado em heurística. Os resultados do trabalho mostraram que é possível melhorar a confiabilidade média do circuito, com pouco esforço de computação. Entretanto, esta estratégia, não é a mais adequada, para melhorar o limite inferior de confiabilidade, que pode estar muito abaixo da média. Uma solução que avalia as saídas individualmente, apresentou resultados superiores a 200% para melhoria da confiabilidade crítica, quando comparado com a estratégia baseada na confiabilidade média.

Palavras-chave: Confiabilidade, *Hardening* seletivo, SPR, Circuitos Combinacionais.

ABSTRACT

ROCHA, Marcio Oliveira da. **Analysis of the reliability of combinational circuits by protecting critical circuit gates**. 2020. 68 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

The technology scaling of CMOS devices has improved productivity and performance in the semiconductor industry, producing circuits faster with lower power consumption. This achievement was possible due to the reduction in transistor dimensions, allowing the integration of billions of these devices in one integrated circuit. However, the high density of transistors associated with the low supply voltage result in systems with higher sensitivity to radiation faults. Alpha particles and neutrons, present in the environment, are the main causes of radiation faults in nanometer circuits. These faults occurrence have a negative impact on circuit reliability. Therefore, reliability cannot be neglected in nanometer circuits. Techniques and analysis to improve circuit reliability are mandatory. The analysis techniques usually only calculate the average reliability of the circuit and disregard an analysis of its lower limit, due to the temporal complexity associated with the analysis. The difference between the two can be orders of magnitude, and if the values are below acceptable limits, reliability must be improved. Reliability improvement is achieved with the use of redundancy. However, this should not generate excessive costs in other design parameters such as area and power. One way to meet these requirements is to identify the most sensitive parts of a circuit and apply redundancy only to those parts. In this context, it is intended to analyze the reliability of the circuits, by protecting the most sensitive parts of the circuit. In addition, to obtain the lower limit of reliability, an algorithm based on heuristics was developed. The results of the work showed that it is possible to improve the average reliability of the circuit, with little computing effort. However, this strategy is not the most suitable to improve the lower limit of reliability, which may be well below average. A solution that evaluates the outputs individually, presented results greater than 200 % for improvement of critical reliability, when compared with the strategy based on average reliability.

Keywords: Reliability, Selective Hardening, SPR, Combinational Circuits.

LISTA DE FIGURAS

Figura 1	Transistores em microprocessadores Intel adaptado de (YEAP; NISAR, 2018)	13
Figura 2	Taxonomia da Dependabilidade, adaptado de (AVIZIENIS et al., 2004)	17
Figura 3	Curva da banheira citado por (TEIXEIRA FRANCO, 2008) .	17
Figura 4	Tipos de mascaramentos adaptado de (MUNTEANU; AU-TRAN, 2008)	19
Figura 5	ITM e PTM da porta NAND	20
Figura 6	Operações PTM (a) serial (b) paralela (c) <i>fanout</i> (PATEL; MARKOV; HAYES, 2003)	21
Figura 7	Circuito com nível intermediário 4 com 5 entradas e 3 saídas .	21
Figura 8	Matriz PTM intermediária do circuito da Figura 7	22
Figura 9	Circuito constituído de quatro portas macro (XIAO et al., 2011)	23
Figura 10	Matriz de probabilidades de sinal de quatro estados	23
Figura 11	Matrizes de probabilidades para cálculo da confiabilidade média e para a confiabilidade associada ao vetor 10	24
Figura 12	Probabilidade do sinal em uma porta NAND	25
Figura 13	Probabilidade do sinal e confiabilidade em um circuito	26
Figura 14	Cálculo de confiabilidade em circuito sem <i>fanout</i> reconvergente (TEIXEIRA FRANCO, 2008)	26
Figura 15	Cálculo de confiabilidade em circuito com <i>fanout</i> reconvergente (TEIXEIRA FRANCO, 2008)	26
Figura 16	Diferença entre o pior e o médio PF_{OUT} ($Y = PF_{MAX} / PF_{AVG}$) para 298 sinais de saída ISCAS'85 (IBRAHIM, 2015)	29
Figura 17	Determinação de Consenso do CWRSA	30
Figura 18	O impacto do <i>gate sizing</i> sobre a geração e propagação de SET adaptado de (Raji; Sabet; Ghavami, 2019)	33
Figura 19	Esquema TMR	34
Figura 20	Esquema Duplicação com Comparação	34
Figura 21	Redundância Temporal adaptado de (Anghel; Alexandrescu; Nicolaidis, 2000)	35
Figura 22	Circuito com 50% de redução da SER através da duplicação de três portas	36
Figura 23	Análise de suscetibilidade de saída (WALI et al., 2017)	37
Figura 24	Fluxograma da análise de R_{Med} considerando diferentes ordenações das portas lógicas	40

Figura 25	Fluxograma da análise de R_{Med} e R_{Crit}	41
Figura 26	Dez vetores mais críticos do circuito C17	42
Figura 27	Sub-circuitos do C17	44
Figura 28	Ordenação das portas lógicas de cada sub-circuito	45
Figura 29	Análise de R_{Crit} considerando o circuito original	45
Figura 30	Análise do MTBF do C432 ao proteger as portas lógicas	48
Figura 31	Diferença de MTBF (caso médio versus pior caso) dos circuitos c499 e c1355	51
Figura 32	Diferença do MTBF (caso médio) ao proteger as portas lógicas considerando duas ordenações (ordem decrescente de R_{Med} e ordem decrescente de R_{Crit})	51
Figura 33	Análise do MTBF (pior caso) do circuito C432 ao proteger as portas lógicas considerando duas ordenações (ordem decrescente de R_{Med} e ordem decrescente de R_{Crit})	52
Figura 34	Diferença do MTBF (pior caso) dos circuitos C432 e c499 ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem considerando um único vetor crítico do circuito)	57
Figura 35	Diferença do MTBF (pior caso) dos circuitos C1355 e c499 ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem decrescente de R_{Med})	58
Figura 36	Biblioteca para mapeamento dos circuitos	68

LISTA DE TABELAS

Tabela 1	Comparação entre os métodos	28
Tabela 2	Técnicas para melhoria da confiabilidade	32
Tabela 3	Soma das criticalidades dos sub-circuitos	44
Tabela 4	Circuitos Benchmarks ISCAS'85	46
Tabela 5	MTBF dos circuitos analisados	47
Tabela 6	Diferença de MTBF ao considerar diferentes ordens das portas protegidas	48
Tabela 7	Custo Temporal ao proteger portas lógicas	49
Tabela 8	Comparação do MTBF (caso médio e pior caso)	49
Tabela 9	MTBF (caso médio e pior caso) ao proteger portas lógicas (ordem decrescente de R_{Med})	50
Tabela 10	Análise MTBF (caso médio) ao proteger portas lógicas (ordem decrescente de R_{Crit})	51
Tabela 11	Análise MTBF (pior caso) ao proteger portas lógicas (ordem decrescente de R_{Crit} [Ord1] versus ordem decrescente de R_{Med} [Ord2])	53
Tabela 12	Custo Temporal para identificar o vetor crítico	53
Tabela 13	Estrutura dos Sub-Circuitos	55
Tabela 14	Diferença do MTBF (caso médio versus pior caso) dos sub-circuitos	55
Tabela 15	Análise MTBF (pior caso) ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem considerando um único vetor crítico do circuito original)	56
Tabela 16	Comparação do Custo Temporal (análise circuito original versus análise sub-circuitos)	58

LISTA DE ABREVIATURAS E SIGLAS

ADD	<i>Algebraic Decision Diagram</i>
BC	<i>Bathtub Curve</i>
BN	<i>Bayesian Network</i>
CI	<i>Circuito Integrado</i>
CMOS	<i>Complementary Metal Oxide Semiconductor</i>
CNCA	<i>Critical Node Count Algorithm</i>
CS	<i>Criticality score</i>
CWRSA	<i>Consensus-based Worst Reliability Search Algorithm</i>
DWC	<i>Duplication with Comparison</i>
ECNCA	<i>Enhanced Critical Node Count Algorithm</i>
EMF	<i>Electrical Masking Factor</i>
FR	<i>Failure Rate</i>
ISCAS	<i>International Symposium on Circuits and Systems</i>
ITM	<i>Ideal Transfer Matrix</i>
MC	<i>Monte Carlo</i>
MTBF	<i>Mean time Between Failure</i>
PF	<i>Probability of failure</i>
PGM	<i>Probabilistic Gate Model</i>
PTM	<i>Probabilistic Transfer Matrix</i>
SEE	<i>Single-event-effects</i>
SER	<i>Soft Error Rate</i>
SET	<i>Single Event Transient</i>
SEU	<i>Single Event Upset</i>
SPR	<i>Signal Probability Reliability</i>
SPR-MP	<i>Signal Probability Reliability - Multi Pass</i>
TMR	<i>Triple Modular Redundancy</i>
WRV	<i>Worst Reliability Vector</i>

SUMÁRIO

1	INTRODUÇÃO	12
1.1	Objetivos	14
1.2	Organização do Trabalho	15
2	REFERENCIAL TEÓRICO	16
2.1	Confiabilidade	16
2.1.1	Conceitos	16
2.1.2	Falhas e Mascaramentos	18
2.2	Análise de Confiabilidade	19
2.2.1	Matriz de Transferência Probabilística - PTM	20
2.2.2	Confiabilidade pela Probabilidade do Sinal - SPR	23
2.2.3	SPR <i>Multi-pass</i>	25
2.2.4	Outros métodos	27
2.2.5	Comparação entre os métodos	28
2.2.6	Identificação do Vetor Crítico	28
2.3	Técnicas para melhoria da Confiabilidade	31
2.3.1	<i>Gate Sizing</i>	32
2.3.2	Redundância de Informação	32
2.3.3	Redundância Modular	32
2.3.4	Duplicação com Comparação	33
2.3.5	Redundância Temporal	34
2.3.6	<i>Hardening</i> Seletivo	35
3	METODOLOGIA	38
3.1	Análise de R_{Med} ao proteger portas lógicas, considerando diferentes ordenações	39
3.2	Análise de R_{Med} e R_{Crit} ao proteger portas lógicas	40
3.3	Análise de R_{Crit} ao proteger as portas críticas a partir da avaliação individual das saídas do circuito	43
4	RESULTADOS	46
4.1	Análise da Confiabilidade Média (R_{Med})	47
4.2	Análise da Confiabilidade (R_{Med} versus R_{Crit})	49
4.3	Análise R_{Crit} ao considerar uma avaliação individual das saídas do circuito	54
5	CONSIDERAÇÕES FINAIS	59
	REFERÊNCIAS	61

APÊNDICE A	PSEUDOCÓDIGO DO EXPERIMENTO 1	65
APÊNDICE B	PSEUDOCÓDIGO DO ALGORITMO PARA IDENTIFICAÇÃO DO VETOR CRÍTICO	66
APÊNDICE C	BIBLIOTECA PARA MAPEAMENTO DOS CIRCUITOS .	68

1 INTRODUÇÃO

A evolução da tecnologia CMOS tem permitido melhorias em produtos e serviços de diversas aplicações. Isso é possível devido à redução das dimensões dos transistores, o que permite alocar uma quantidade maior destes componentes em uma mesma área de um circuito integrado (CI). Esse acréscimo no número de transistores encapsulados em um chip se traduz em circuitos com mais funcionalidades, executando de forma cada vez mais rápida e com custo de fabricação mais baixo (FRANCO; NAVINER; NAVINER, 2006). A Figura 1, mostra a quantidade de transistores em microprocessadores Intel, desde a invenção do 4004. Quando ele foi criado na década de 70, este microprocessador possuía pouco mais de mil transistores. Já a partir de 2010, a quantidade de transistores nos microprocessadores era superior a um bilhão.

As melhorias podem ser percebidas em computadores, notebooks e *smartphones*, onde seus principais componentes de *hardware* (microprocessador e memória) tem seu poder de processamento e armazenamento, melhorados a cada nova geração. É possível citar ainda, os sistemas embarcados que são sistemas computacionais com propósito específico, e estão presentes em equipamentos eletrodomésticos, câmeras fotográficas, automóveis, televisão, entre tantos outros, permitindo uma melhor experiência do usuário.

Entretanto, conforme as dimensões dos transistores diminuem, o processo de fabricação de um circuito integrado torna-se mais complexo e o mesmo, também fica mais suscetível à ocorrência de falhas, devido a partículas alfa produzidas por traços de isótopos radioativos encontrados em materiais de encapsulamento e nêutrons atmosféricos criados pela interação de raios cósmicos com a atmosfera terrestre (NICOLAIDIS, 2005). A ocorrência dessas falhas em um circuito é conhecida como falha transiente, ou *soft error* e reflete, de forma negativa, na confiabilidade dos circuitos. Portanto, esta não pode ser negligenciada.

Existem diversas abordagens na literatura que analisam a confiabilidade de circuitos lógicos, utilizando métodos analíticos ou através de simulação (PATEL; MARKOV; HAYES, 2003), (FRANCO et al., 2008), (HAN et al., 2011), (HAN et al., 2012). O objetivo de tais técnicas é alcançar resultados mais precisos e com baixo custo computacional. Todavia, precisão e eficiência estão intimamente relacionadas às estruturas dos circuitos

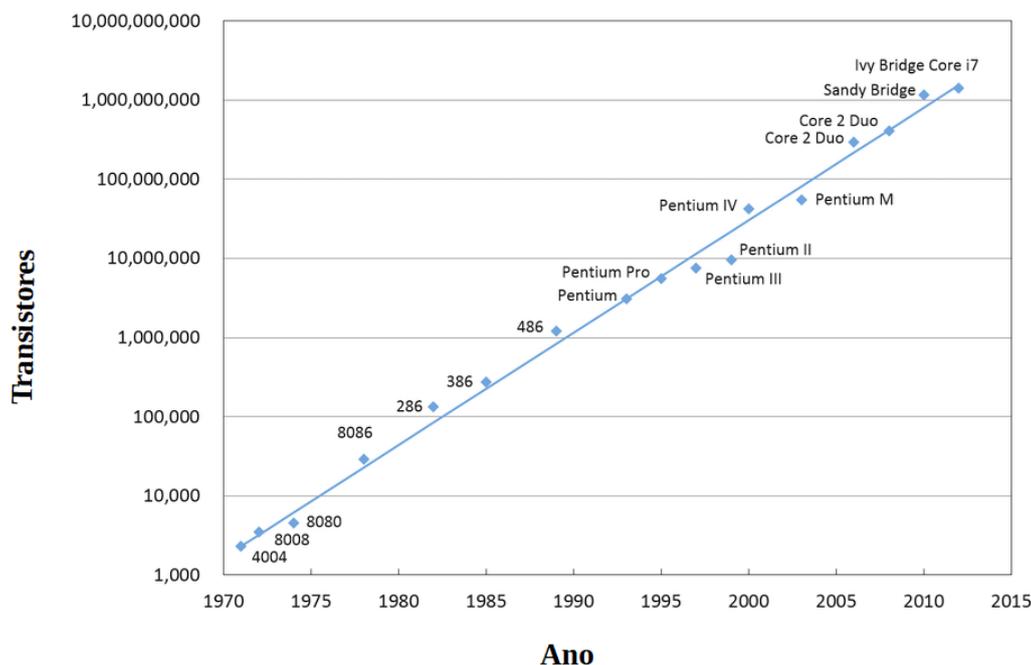


Figura 1: Transistores em microprocessadores Intel adaptado de (YEAP; NISAR, 2018)

e seus tamanhos e, na maioria das vezes, compensações entre ambas devem ser feitas (XIAO; CHEN, 2014). Por exemplo, o método *Probabilistic Transfer Matrix* (PTM) permite calcular a confiabilidade de forma exata. Entretanto, seu principal problema é a escalabilidade. Os métodos *Probabilistic Gate Model* (PGM) e *Signal Probability Reliability* (SPR) em suas versões simples e eficientes, mas não garantem precisão em circuitos com *fanouts* reconvergentes. Já suas versões precisas, entregam resultados exatos, todavia, a eficiência temporal das mesmas é comprometida. Para a técnica Monte Carlo (MC) ser mais precisa, ela depende da geração de mais números aleatórios. No entanto, isto significa perda de eficiência.

Além disso, devido à complexidade das análises, o vetor crítico costuma ser ignorado. O vetor crítico é o sinal de entrada associado com a confiabilidade crítica, ou seja, com o pior valor de confiabilidade. As ferramentas de análise, normalmente calculam a confiabilidade média das saídas do circuito e desconsideram uma análise individual das mesmas, assim como o efeito dos sinais de entrada (IBRAHIM, 2015). Entretanto, a diferença entre a maior probabilidade de falha e a média, pode chegar a ordens de magnitude (IBRAHIM; BEIU; AMER, 2009). A análise dos vetores de entrada não é tarefa trivial, e tem custo de processamento que cresce exponencialmente com o número de entradas, e se torna intratável para circuitos com muitas entradas. Após essa análise de confiabilidade, é possível determinar se a mesma está dentro de limites aceitáveis para o projeto. Caso esta não esteja, técnicas para melhorar a confiabilidade devem ser aplicadas.

A melhoria da confiabilidade, está relacionada ao uso de técnicas de prevenção ou

tolerância a falhas, e normalmente, se alcança através da redundância de recursos. Entretanto, estas técnicas introduzem ônus em outros parâmetros de projeto, que são tão importantes quanto a confiabilidade, como área, potência e atraso. Portanto, a aplicação das mesmas para proteção total dos circuitos atuais, pode inviabilizar a sua utilização, já que tem custo bastante elevado. Por exemplo, o *Triple Modular Redundancy* (TMR) apesar de ser uma das técnicas mais efetivas, tem custo de área e potência de mais de 200% e pode ser proibitivo para muitas aplicações (POLIAN; HAYES, 2010).

Uma técnica alternativa para aumentar a confiabilidade, sem onerar tanto outros parâmetros de projeto, é conhecida como *hardening* seletivo. Esta técnica consiste em identificar os blocos críticos (portas lógicas) do circuito, ou seja, aqueles que estão mais expostos à falhas, e então aplicar redundância somente nestes blocos (POLIAN; HAYES, 2010). Além disso, é possível identificar os blocos críticos associados aos vetores críticos, e aplicar as técnicas para garantir confiabilidade mínima para qualquer vetor de entrada.

Neste contexto, uma análise da confiabilidade, considerando tanto o caso médio quanto o pior caso, não pode ser desprezada e será o foco de pesquisa deste trabalho. Além disso, é preciso determinar os blocos críticos que permitam alcançar a máxima confiabilidade com o mínimo de impacto em outros parâmetros de projeto.

1.1 Objetivos

As técnicas mais efetivas para melhoria da confiabilidade, implicam em custo excessivo em outros parâmetros de projeto, como área, potência e performance. Em aplicações de missão crítica, em que é preciso garantir que os sistemas tenham alta confiabilidade, estes custos podem ser tolerados. Entretanto, não são aceitáveis em aplicações convencionais, onde custo e performance são os principais objetivos. Uma forma alternativa para lidar com este problema, é identificar partes de um circuito digital mais propensas à falhas e assim proteger seletivamente as mesmas. A determinação correta destas partes, que são consideradas críticas, permite maximizar a confiabilidade do circuito com o mínimo de impacto nos outros parâmetros.

Diante do contexto, o objetivo geral deste trabalho é analisar a confiabilidade de circuitos combinacionais ao proteger seletivamente as portas lógicas dos circuitos. Para isso, é necessário realizar a identificação das portas críticas a serem protegidas. A análise da confiabilidade, irá englobar tanto o caso médio quanto o pior caso de confiabilidade. Isso se mostrará relevante pois, em muitos casos, a diferença entre ambos é significativa. No escopo da análise do pior caso de confiabilidade, é necessário identificar o vetor crítico. Esta análise é impraticável para circuitos com muitos sinais de entrada, pois é necessário pesquisar exaustivamente todas as combinações de vetores de entrada. Sendo assim, foi implementado um algoritmo baseado em uma heurística para identificar o vetor crítico. Somente após a obtenção do vetor crítico é possível determinar o comprometimento da

confiabilidade nos vetores críticos quando comparada com o tradicional cálculo da confiabilidade média, bem como identificar as portas críticas associadas a estes vetores e efetuar a análise da confiabilidade ao proteger as mesmas. Por fim, as discussões também irão abranger a análise individual de cada saída do circuito com o intuito de fornecer uma análise de confiabilidade completa do sistema.

1.2 Organização do Trabalho

Os capítulos seguintes deste trabalho, estão organizados como segue: o Capítulo II apresenta conceitos básicos referentes à confiabilidade de circuitos lógicos. Em seguida, são introduzidos alguns métodos para análise de confiabilidade e uma comparação entre os mesmos é apresentada. A última seção deste Capítulo, apresenta uma visão geral sobre técnicas para melhoria da confiabilidade. O Capítulo III, apresenta a metodologia do trabalho, e esta se divide em três partes. Para a primeira parte, somente a confiabilidade média é considerada. Na segunda parte, além da confiabilidade média, o pior caso de confiabilidade é analisado e uma comparação entre as mesmas é mostrado. Na última parte, os circuitos são divididos em sub-circuitos, e uma análise do pior caso de confiabilidade é efetuada. O Capítulo IV, apresenta os resultados e por último, as considerações finais podem ser vistas no Capítulo V.

2 REFERENCIAL TEÓRICO

Neste capítulo, serão apresentados alguns conceitos relacionados a confiabilidade em sistemas digitais, e falhas que ocorrem nos mesmos, além do efeito de mascaramentos, que impede que uma falha se propague até uma saída primária do circuito. Como o avanço da tecnologia de semicondutores está impactando negativamente a confiabilidade, com circuitos mais expostos à falhas, a análise da confiabilidade não pode ser ignorada. A seção seguinte, aborda alguns métodos para análise da confiabilidade e identificação do vetor crítico. Como será mostrado, devido às estruturas dos circuitos, normalmente é necessário abdicar de uma análise precisa e obter resultados aproximados em benefício da eficiência. A última seção, apresenta uma visão geral de técnicas para melhoria da confiabilidade e finaliza com uma técnica que visa melhorar somente uma parte do circuito, para que outros parâmetros de projeto não sejam excessivamente penalizados.

2.1 Confiabilidade

2.1.1 Conceitos

Dependability ou dependabilidade é a capacidade de entregar um serviço que pode ser justificadamente confiável (AVIZIENIS et al., 2004). Trata-se de um conceito genérico que engloba diversos atributos representados na Figura 2. *Reliability* ou confiabilidade é um dos atributos e tem relação com a continuidade do serviço correto. Segundo (KAPUR; PECHT, 2014), confiabilidade é a capacidade de um produto ou sistema funcionar sem falha e dentro de limites de desempenho especificados, por um tempo determinado e sob certas condições.

Uma importante métrica para determinar a confiabilidade de sistemas digitais é a taxa de falhas, e seu comportamento pode ser expresso pela curva da banheira (BC), conforme a Figura 3 apresentada em (TEIXEIRA FRANCO, 2008). Taxa de falhas, ou $\lambda(t)$, pode ser definida como a frequência que um item falha. Durante o período de vida útil dos sistemas, a taxa de falhas pode ser considerada constante, ou seja, $\lambda(t) = \lambda$ e a confiabilidade (R) pode ser expressa como uma distribuição exponencial como descrita na Equação (1).

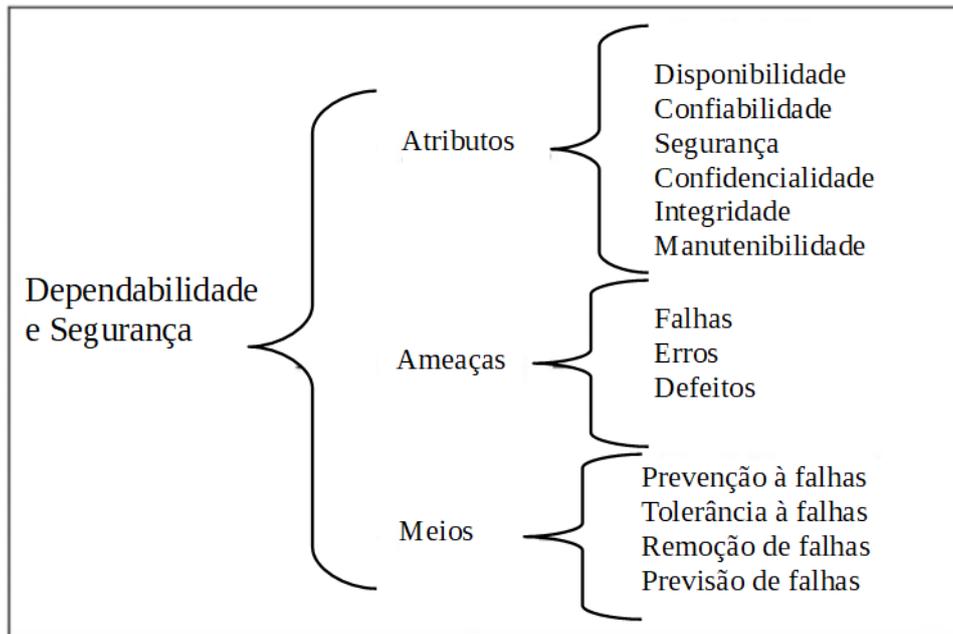


Figura 2: Taxonomia da Dependabilidade, adaptado de (AVIZIENIS et al., 2004)

$$R(t) = e^{-\lambda(t)} \quad (1)$$

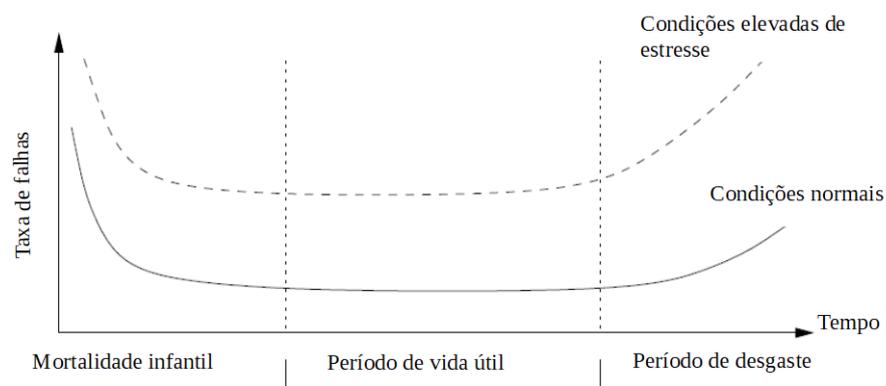


Figura 3: Curva da banheira citado por (TEIXEIRA FRANCO, 2008)

Como a confiabilidade é dependente do tempo de operação do sistema ou componente, não parece relevante observar seu valor abstrato. Por exemplo, aumentar a confiabilidade de 0,99999 para 0,999999, representa um aumento insignificante muito inferior a 1%. Entretanto, quando observamos a probabilidade de falha (PF), a diferença é 900%, conforme mostra a Equação (2).

$$Diff_{PF} = \frac{1 - 0,99999}{1 - 0,999999} = 10 \quad (2)$$

Este mesmo comportamento pode ser visto, ao utilizarmos a métrica *Mean time*

Between Failure (MTBF). O MTBF, ou tempo médio entre falhas, é inversamente proporcional à taxa de falhas e normalmente é apresentado em horas, como na Equação (3). Já a confiabilidade pode ser expressa em função do MTBF, como mostra a equação (4). Quando representados em função do MTBF, os valores de confiabilidade da Equação (2) são expressos como 99.999 e 999.999.

$$MTBF = \frac{1}{\lambda} \quad (3)$$

$$R(t) = e^{-\frac{t}{MTBF}} \quad (4)$$

2.1.2 Falhas e Mascaramentos

O avanço da tecnologia de semicondutores, que permitiu dispositivos com alta densidade, operando com baixa tensão e altas frequências, também proporcionou que os mesmos, se tornassem mais sensíveis à ocorrência de falhas. As falhas que ocorrem nestes dispositivos, tem impacto negativo na confiabilidade e podem ser divididas em três categorias principais (CONSTANTINESCU, 2003):

- Falhas permanentes, que produzem mudanças físicas irreversíveis no dispositivo;
- Falhas intermitentes, que são caracterizadas pela instabilidade do *hardware* e podem ser ativadas por mudanças ambientais, como alterações de temperatura e tensão. Muitas vezes, este tipo de falha, precede a ocorrência de falhas permanentes;
- Falhas transientes, também conhecidas como *soft errors*, que são causadas por condições ambientais temporárias, como partículas alfa e de nêutrons, interferência eletromagnética, descarga eletroestática, etc.

As falhas transientes podem ser caracterizadas pela interação de partículas com regiões sensíveis de um dispositivo, e que resulta em eventos chamados de *Single-event-effects* (SEE). Um SEE pode perturbar a operação do dispositivo, invertendo o dado de uma célula de memória, *latch* ou *flip-flop*. Neste caso, o evento é chamado de *Single Event Upset* (SEU). Quando a perturbação ocorre na lógica combinacional, o evento é definido como *Single Event Transient* (SET) (MUNTEANU; AUTRAN, 2008).

A taxa, na qual os *soft errors* ocorrem, é chamada de *Soft Errors Rate* (SER). Em circuitos lógicos combinacionais, ela tende a ser menor, devido aos mascaramentos de falhas, onde uma falha não é propagada até a saída do circuito, ou caso ela consiga alcançar, não é armazenada em um elemento de memória. Existem três tipos de mascaramentos, conforme a Figura 4.

O mascaramento lógico ocorre, quando uma falha alcança a entrada de uma porta lógica, porém a saída desta porta é determinada pelo valor de outra entrada. No exemplo

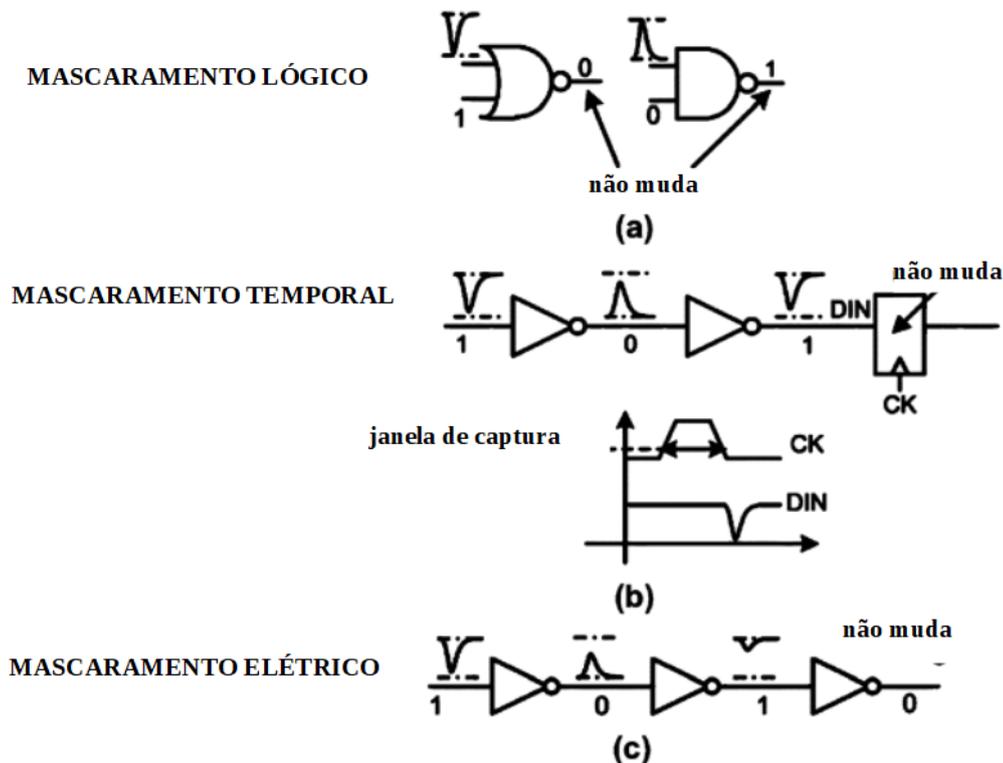


Figura 4: Tipos de mascaramentos adaptado de (MUNTEANU; AUTRAN, 2008)

da Figura 4(a), o valor 1 na entrada da porta NOR ou 0 na NAND, mascara qualquer falha que ocorra na outra entrada. O mascaramento temporal é caracterizado, quando a falha se propaga até um elemento de memória, porém não são atendidos requisitos temporais necessários para ser capturada (bordas de *clock* e tempos de *setup* e *hold*), conforme ilustrado na Figura 4(b). Finalmente, o mascaramento elétrico acontece quando a falha, ao ser propagada por outras portas lógicas, é atenuada devido ao tempo de transição destas ser maior que a duração da falha, como mostra a Figura 4(c).

2.2 Análise de Confiabilidade

Para o projeto de um circuito confiável, é necessário o desenvolvimento de ferramentas que possam avaliar precisa e eficientemente a confiabilidade do mesmo. Entretanto, a análise de confiabilidade não é uma tarefa trivial em circuitos grandes, assim como a complexidade da correlação de sinal e a propagação da probabilidade e confiabilidade dentro do circuito (XIAO; CHEN, 2014). Em circuitos livres de *fanouts* reconvergentes, a probabilidade de sinal e confiabilidade pode ser obtida com tempo linear ao número de portas lógicas, enquanto que, na presença de *fanouts* reconvergentes, resultados precisos exigem custos de computação que crescem exponencialmente com o número das entradas e saídas ou número de *fanouts* do circuito.

Diversos métodos para análise de confiabilidade tem sido propostos e alguns deles serão apresentados nas próximas seções. Um estudo detalhado dos métodos PTM, SPR e SPR-MP pode ser visto em (PONTES, 2019).

2.2.1 Matriz de Transferência Probabilística - PTM

Um formalismo conhecido como *Matriz de Transferência Probabilística* (PTM) foi proposto por Patel (PATEL; MARKOV; HAYES, 2003) para calcular a probabilidade de erro de um circuito inteiro, baseado nas probabilidades de erros de suas portas lógicas. Considerando um circuito lógico com m entradas e n saídas, onde cada entrada pode assumir o valor 0 ou 1, então, um vetor de entrada $(i, j)^{\text{th}}$ da matriz PTM representa a probabilidade do vetor de saída j dada a entrada i , ou seja, $p(j|i)$. Portanto, a PTM tem 2^m linhas e 2^n colunas.

Na Figura 5, são apresentadas as matrizes da porta lógica NAND. Na matriz PTM, q representa a probabilidade de um valor correto e p é a probabilidade de ocorrência de erro. Em uma *Matriz de Transferência Ideal* (ITM), q é igual a 1, pois, o circuito é considerado livre de falhas.

		saídas	
		0	1
entradas	00	0	1
	01	0	1
	10	0	1
	11	1	0

ITM

		saídas	
		0	1
entradas	00	p	q
	01	p	q
	10	p	q
	11	q	p

PTM

Figura 5: ITM e PTM da porta NAND

A PTM de um circuito é obtida através da combinação da PTM de blocos básicos internos (portas lógicas e fios) por meio de operações. Quando dois blocos são combinados em série, a PTM resultante será o produto entre as matrizes dos blocos, como mostra a Figura 6a. Em uma combinação em paralelo, a matriz resultante será o produto tensor entre as matrizes, conforme é ilustrado na Figura 6b. O *fanout* de uma porta lógica representa a quantidade de entradas que estão conectadas à saída desta porta. A representação deste tipo de conexão é feita eliminando as colunas da PTM do segundo bloco. Por exemplo, na Figura 6c, o bloco B tem duas entradas, portanto deve-se eliminar as colunas 1 e 2 que representam as combinações $\{01\}$ e $\{10\}$ respectivamente.

A abordagem PTM permite o cálculo exato das probabilidades de erro relativo a mas-

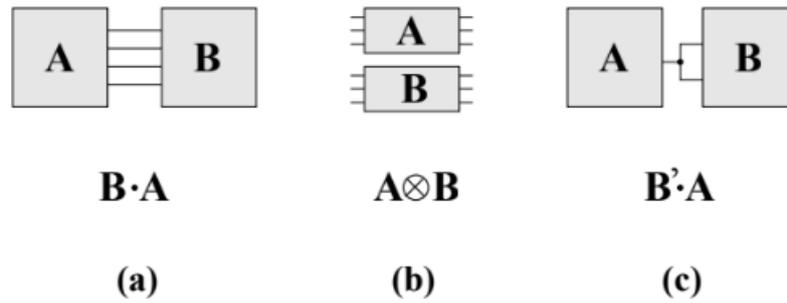


Figura 6: Operações PTM (a) serial (b) paralela (c) *fanout* (PATEL; MARKOV; HAYES, 2003)

caramento lógico. Dada a distribuição da probabilidade de entrada, a confiabilidade do circuito pode ser modelada usando PTM e ITM, de acordo com a Equação (5). Nesta expressão, cada $p(j|i)$ é um elemento da matriz PTM, ponderada pela probabilidade da correspondente entrada i . Os elementos (i, j) considerados são aqueles onde a respectiva matriz ITM tem valor 1. Isto significa que a confiabilidade é definida como a probabilidade de uma saída correta.

$$R_{(cir)} = \sum_{ITM_{cir}(i,j)} p(j|i) \quad (5)$$

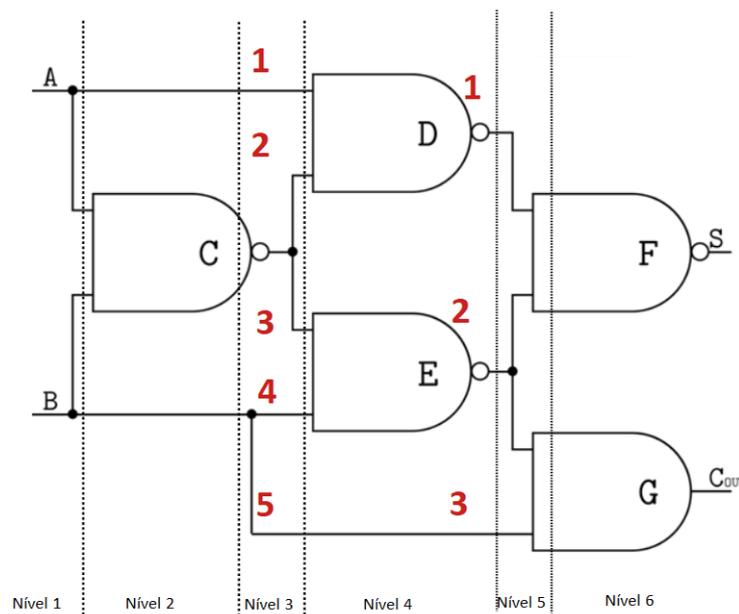


Figura 7: Circuito com nível intermediário 4 com 5 entradas e 3 saídas

O cálculo de confiabilidade PTM é exato. Porém, a demanda por memória associada aos cálculos da PTM, cresce exponencialmente com o número de entradas e saídas.

Como exemplo, considere o circuito da Figura 7. Embora o mesmo tenha apenas 5 portas lógicas, uma operação entre matrizes no nível intermediário 4, produz a matriz PTM $2^5 \times 2^3$, apresentada na Figura 8. Portanto, esta técnica é bastante custosa e impraticável para grandes circuitos.

	000	001	010	011	100	101	110	111
00000	p^2	0	pq	0	pq	0	q^2	0
00001	0	p^2	0	pq	0	pq	0	q^2
00010	p^2	0	pq	0	pq	0	q^2	0
00011	0	p^2	0	pq	0	pq	0	q^2
00100	p^2	0	pq	0	pq	0	q^2	0
00101	0	p^2	0	pq	0	pq	0	q^2
00110	pq	0	p^2	0	q^2	0	pq	0
00111	0	pq	0	p^2	0	q^2	0	pq
01000	p^2	0	pq	0	pq	0	q^2	0
01001	0	p^2	0	pq	0	pq	0	q^2
01010	p^2	0	pq	0	pq	0	q^2	0
01011	0	p^2	0	pq	0	pq	0	q^2
01100	p^2	0	pq	0	pq	0	q^2	0
01101	0	p^2	0	pq	0	pq	0	q^2
01110	pq	0	p^2	0	q^2	0	pq	0
01111	0	pq	0	p^2	0	q^2	0	pq
10000	p^2	0	pq	0	pq	0	q^2	0
10001	0	p^2	0	pq	0	pq	0	q^2
10010	p^2	0	pq	0	pq	0	q^2	0
10011	0	p^2	0	pq	0	pq	0	q^2
10100	p^2	0	pq	0	pq	0	q^2	0
10101	0	p^2	0	pq	0	pq	0	q^2
10110	pq	0	p^2	0	q^2	0	pq	0
10111	0	pq	0	p^2	0	q^2	0	pq
11000	pq	0	q^2	0	p^2	0	pq	0
11001	0	pq	0	q^2	0	p^2	0	pq
11010	pq	0	q^2	0	p^2	0	pq	0
11011	0	pq	0	q^2	0	p^2	0	pq
11100	pq	0	q^2	0	p^2	0	pq	0
11101	0	pq	0	q^2	0	p^2	0	pq
11110	q^2	0	pq	0	pq	0	p^2	0
11111	0	q^2	0	pq	0	pq	0	p^2

Figura 8: Matriz PTM intermediária do circuito da Figura 7

Diversos trabalhos têm sido apresentados ao longo dos anos buscando otimizar o método PTM. Em (KRISHNASWAMY et al., 2005), os autores propuseram melhorar a eficiência das operações PTM utilizando *Diagramas de Decisão Algébrica* (ADDs) para comprimir as matrizes aumentando a escalabilidade, mas não o suficiente para lidar com circuitos grandes.

Em (XIAO et al., 2011), é proposto um modelo PTM iterativo que reduz a complexidade de tempo em comparação ao modelo PTM original. Este modelo é baseado no conceito de porta macro, que pode ser definida como uma porta elementar ou grupo de portas

a partir de um *fanout* até um ponto de convergência (porta lógica ou saídas primárias do circuito), como visto na Figura 9.

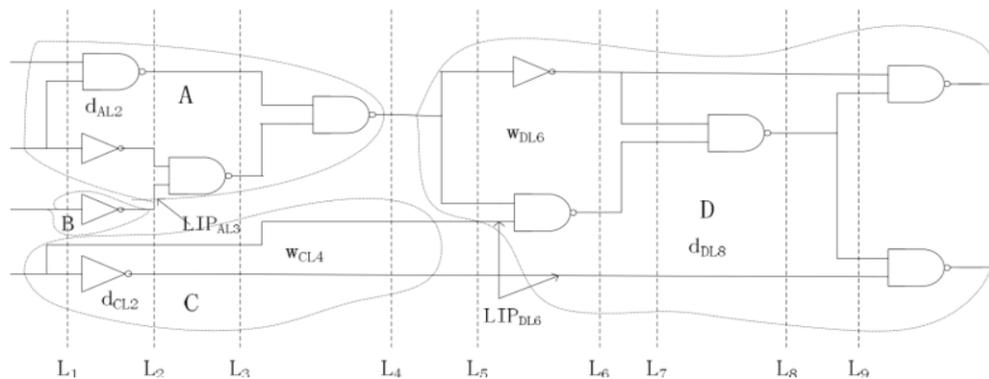


Figura 9: Circuito constituído de quatro portas macro (XIAO et al., 2011)

2.2.2 Confiabilidade pela Probabilidade do Sinal - SPR

O algoritmo SPR (*Signal Probability Reliability*) descrito em (FRANCO et al., 2008), leva em conta a probabilidade de falha de portas e a estrutura topológica do circuito para determinar a corretude dos sinais. A confiabilidade de um circuito pode ser calculada com base no efeito cumulativo de erros nos sinais do circuito, sendo que este efeito, incorpora a contribuição de múltiplas falhas simultâneas para a confiabilidade do circuito.

Este método difere de outras abordagens de probabilidade de sinal utilizando uma representação de quatro estados para o sinal binário propenso à falha. Além da probabilidade de ocorrência de 0 e 1 corretos, a representação de probabilidade de quatro estados inclui a probabilidade de ocorrência de 0 e 1 incorretos. Um 0 incorreto é um valor que deveria ser 1 em uma condição livre de falha e 1 incorreto é um valor que deveria ser 0. A figura 10 mostra a representação de matriz.

$$P_{\text{sinal}} = \begin{pmatrix} P_{\text{sinal}} = 0 \text{ correto} & P_{\text{sinal}} = 1 \text{ incorreto} \\ P_{\text{sinal}} = 0 \text{ incorreto} & P_{\text{sinal}} = 1 \text{ correto} \end{pmatrix}$$

Figura 10: Matriz de probabilidades de sinal de quatro estados

A Figura 11 mostra as matrizes de probabilidades para o cálculo da confiabilidade média e para um vetor específico. Ao calcular a confiabilidade média, deve ser informado o valor 0,5 nas posições 0 e 1 corretos da matriz, que significa 50% de chance de ocorrer 0

ou 1. Para o cálculo de um vetor específico, deve ser informado o valor 1 que corresponde a 100% de chance de ocorrência do 0 ou 1, na posição referente ao 0 correto ou 1 correto, conforme o exemplo.

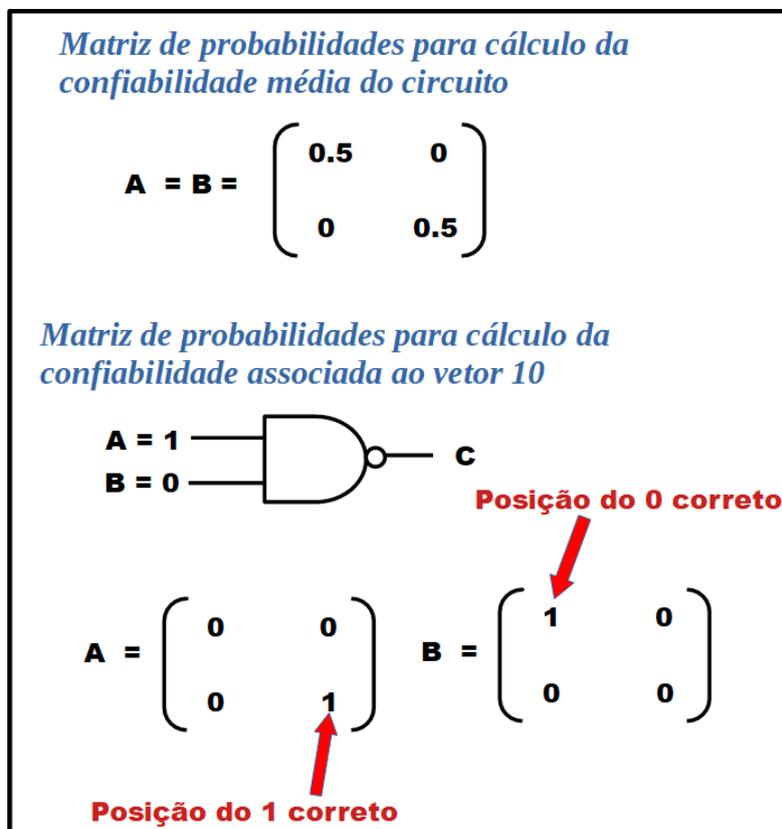


Figura 11: Matrizes de probabilidades para cálculo da confiabilidade média e para a confiabilidade associada ao vetor 10

As Figuras 12 e 13, ilustram exemplos do cálculo de confiabilidade usando SPR em uma porta *NAND* e em um circuito, respectivamente, considerando $q = 0,9$.

A probabilidade de sinais para os nós em um circuito pode ser obtida pela propagação das probabilidades do sinal de entrada através das portas do circuito. O processo de propagação usa a probabilidade de falha de determinada porta e sua probabilidade de sinal de entrada para calcular as probabilidades do sinal de saída. A probabilidade de falha da porta é representada pela PTM e sua função livre de falha é representada por sua ITM. As probabilidades de sinal de entrada (I) de uma porta correspondem ao produto tensor das probabilidades conjuntas (produto Kronecker) dos sinais de entrada, conforme o Passo 1 da Figura 12. As probabilidades de sinais de saída podem ser determinadas pela multiplicação das probabilidades de sinal de entrada pela PTM da porta, de acordo com a Equação (6) e apresentado no Passo 2 da Figura 12.

$$P(S) = I_{gate} \times PTM_{gate} \quad (6)$$

A probabilidade de saída 2×2 é calculada pelo reagrupamento das probabilidades

de saída, de acordo com a função lógica da porta, representada por sua ITM, como é apresentado no Passo 3 da Figura 12. A probabilidade de sinal de quatro estados incorpora a informação de confiabilidade do sinal definido pelas probabilidades de 0 e 1 corretos, ou seja, $R_{sinal} = P_{sinal\ 0\ correto} + P_{sinal\ 1\ correto}$, conforme é mostrado na Figura 13 (portas F e G). A confiabilidade do circuito R_{Circ} pode ser obtida pela multiplicação da confiabilidade individual dos sinais de saída, conforme é apresentado na Figura 13. O método SPR garante precisão na confiabilidade de circuitos onde não existam *fanouts* reconvergentes.

$$\begin{array}{l}
 \text{Passo 1} \\
 \mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} \otimes \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} = \begin{pmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0.25 \end{pmatrix} \\
 \hline
 \text{Passo 2} \\
 \mathbf{C} = (\mathbf{A} \otimes \mathbf{B}) * \mathbf{PTM}_{NAND} = \begin{pmatrix} 0.25 & 0 & 0 & 0 \\ 0 & 0.25 & 0 & 0 \\ 0 & 0 & 0.25 & 0 \\ 0 & 0 & 0 & 0.25 \end{pmatrix} * \begin{pmatrix} 0.1 & 0.9 \\ 0.1 & 0.9 \\ 0.1 & 0.9 \\ 0.9 & 0.1 \end{pmatrix} \\
 \hline
 \text{Passo 3} \\
 \mathbf{C} = \begin{pmatrix} 0.025 & 0.225 \\ 0.025 & 0.225 \\ 0.025 & 0.225 \\ 0.225 & 0.025 \end{pmatrix} = \begin{pmatrix} 0.225 & 0.025 \\ 0.075 & 0.675 \end{pmatrix}
 \end{array}$$

Figura 12: Probabilidade do sinal em uma porta NAND

2.2.3 SPR Multi-pass

O algoritmo *multi-pass* considera que cada *fanout* contribui diferentemente para a probabilidade dos sinais e confiabilidade do circuito, enquanto que o método SPR trata diferentes ramificações do mesmo sinal de forma independente, resultando em probabilidades inconsistentes e redundantes (TEIXEIRA FRANCO, 2008). Para ilustrar o problema da dependência dos sinais no cálculo de confiabilidade do método SPR, a Figura 14 apresenta como este cálculo é realizado, considerando circuitos livres de *fanouts* reconvergentes.

A Figura 15 considera a presença de sinais correlacionados e a expressão de confiabilidade é composta por polinômios com probabilidades inconsistentes, que dependem de estados diferentes do mesmo sinal (a_0a_3 , a_3a_0) e probabilidades redundantes, que aparecem duas vezes (a_0^2 , a_3^2). Probabilidades inconsistentes deveriam resultar em valores nulos e probabilidades redundantes, deveriam considerar somente uma ocorrência de cada evento, levando ao resultado correto de $R_{circuit} = a_3q^2 + a_0q^2$.

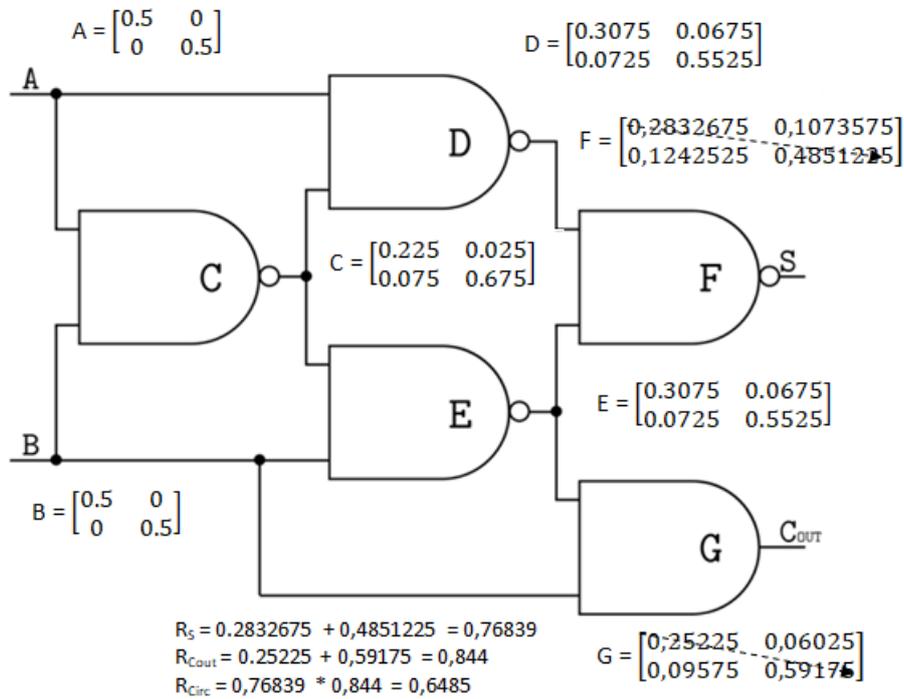


Figura 13: Probabilidade do sinal e confiabilidade em um circuito

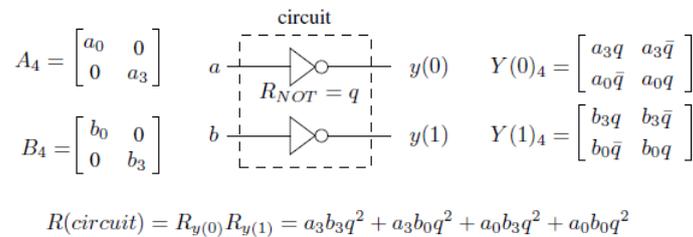


Figura 14: Cálculo de confiabilidade em circuito sem *fanout* reconvergente (TEIXEIRA FRANCO, 2008)

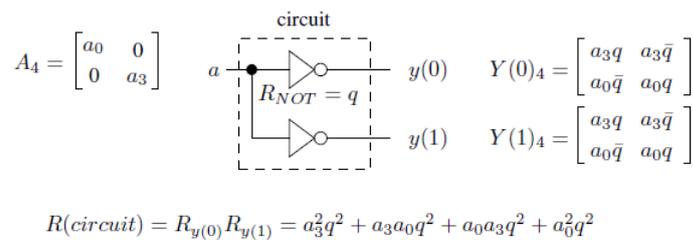


Figura 15: Cálculo de confiabilidade em circuito com *fanout* reconvergente (TEIXEIRA FRANCO, 2008)

Com o processamento exaustivo de um conjunto relevante de probabilidades de sinal *fanout*, é possível reduzir a complexidade do problema e alcançar um bom nível de precisão. O conjunto de *fanouts* relevantes a ser considerado na análise tem relação direta com o tempo de processamento, e o projetista pode determinar o compromisso adequado entre tempo de processamento e precisão através da seleção do número de *fanouts*. A solução permite uma análise precisa, pela escolha de todos os *fanouts*, ou uma análise rápida e menos exata, selecionando a opção sem *fanout*. Depois de definido o número de *fanouts*, o algoritmo *multi-pass* itera por todas as combinações de um estado *fanout*, seguindo a ordem topológica dos *fanouts*. A cada iteração, um novo valor de confiabilidade do circuito é calculado de acordo com a Equação (7), onde F é o número de *fanouts* relevantes.

$$R(\text{circuit}) = \sum_{f=1}^{4^F} R(\text{circuit}, f) \quad (7)$$

A complexidade do algoritmo *multi-pass* cresce exponencialmente com o número de *fanouts* e uma redução da mesma depende da quantidade de *fanouts* selecionados para o cálculo de confiabilidade.

No trabalho de (PAGLIARINI et al., 2013), foi proposta uma abordagem baseada em heurística para avaliação da confiabilidade de circuitos lógicos que promete estimativas melhores que o algoritmo SPR. Uma heurística chamada de contribuições positivas/negativas introduz menos erro médio com tempo de execução acessível.

2.2.4 Outros métodos

O *Modelo de Portas Probabilísticas* (PGM) é um método probabilístico cuja versão simples do algoritmo, assume que os sinais de entrada de cada porta no circuito são independentes e a saída pode ser obtida, usando as probabilidades do sinal de entrada e a taxa de erro da porta (HAN et al., 2011). A complexidade deste algoritmo aumenta linearmente com o número de portas do circuito, entretanto, o fato de considerar independência de sinais, acarreta em imprecisão no cálculo de confiabilidade em circuitos com *fanouts* reconvergentes.

Para lidar com esta questão, uma versão precisa do algoritmo PGM foi proposta. Para cada *fanout* no circuito, a probabilidade de saída é calculada considerando dois circuitos auxiliares. Apesar do algoritmo levar a resultados exatos, a complexidade computacional é função exponencial do número de *fanouts* reconvergentes e tamanho do circuito $O(N_g \cdot 2^{N_{in} + N_f})$, impossibilitando seu uso em circuitos grandes.

A abordagem Monte Carlo (MC), diferentemente das abordagens vistas até aqui, é baseada em injeção de falhas e simulação de padrões aleatórios e um grande número de combinações deve ser simulado. A confiabilidade do circuito é baseada em resultado estatístico e a eficiência do MC é limitada ao total de números pseudo-aleatórios gerados e

ao número de simulações necessárias para se obter resultados estáveis (HAN et al., 2012).

2.2.5 Comparação entre os métodos

A Tabela 1 apresenta um resumo dos métodos apresentados anteriormente. Nenhum dos métodos atende a todos os requisitos elencados. Por exemplo, para o método MC ser mais preciso, ele necessita executar mais simulações. Todavia, isto tem impacto na eficiência do mesmo. Os métodos PTM, SPR-MP e PGM-V2 conseguem fornecer resultados de confiabilidade precisos. Entretanto, suas deficiências estão relacionadas a custo temporal ou de memória. Portanto, nenhum destes métodos é escalável. Já os métodos SPR e PGM tem complexidade linear ao número de portas lógicas e fornecem resultados aproximados com custo temporal aceitável.

Tabela 1: Comparação entre os métodos

	PTM	SPR	SPR-MP	PGM	PGM-V2	MC
Resultados Exatos	Sim	Não	Sim	Não	Sim	Não
Custo Temporal	Não	Não	Sim	Não	Sim	Sim
Custo de memória	Sim	Não	Não	Não	Não	Não
Escalabilidade	Não	Sim	Não	Sim	Não	Não

2.2.6 Identificação do Vetor Crítico

Os métodos apresentados nas seções anteriores se preocupam apenas em estimar a probabilidade de falha média do circuito (PF_{AVG}) ou a probabilidade de falha associada a um vetor específico. Entretanto, nenhum deles tenta identificar o vetor crítico, isto é, o vetor associado com a maior probabilidade de falha (PF_{MAX}) do circuito. Todavia, esta análise não pode ser ignorada, já que PF_{MAX} pode ser ordens de magnitude superior a PF_{AVG} (IBRAHIM; BEIU; AMER, 2009). A Figura 16, apresenta um histograma para as diferenças entre PF_{MAX} e PF_{AVG} para 298 saídas de circuitos *benchmark* ISCAS'85 (BRGLEZ; FUJIWARA, 1985). Portanto, é preciso garantir que a confiabilidade esteja acima da margem aceitável para o projeto, mesmo quando o vetor crítico é aplicado nas entradas primárias do circuito. Caso o valor esteja abaixo do esperado, uma técnica para melhoria da confiabilidade deve ser introduzida.

A identificação do vetor crítico é uma tarefa cuja complexidade cresce exponencialmente com o número de entradas do circuito. Então, uma pesquisa exaustiva de todas as combinações de vetores de entrada, se torna impossível em circuitos com muitas entradas. Por exemplo, considerando que o tempo para calcular a probabilidade de falha para qualquer vetor de entrada fosse 100ms, uma pesquisa exaustiva em um circuito com 32 sinais de entradas, levaria aproximadamente 13 anos. Neste contexto, dividir um circuito a partir de suas saídas primárias, pode ser uma alternativa melhor. Apesar de haver

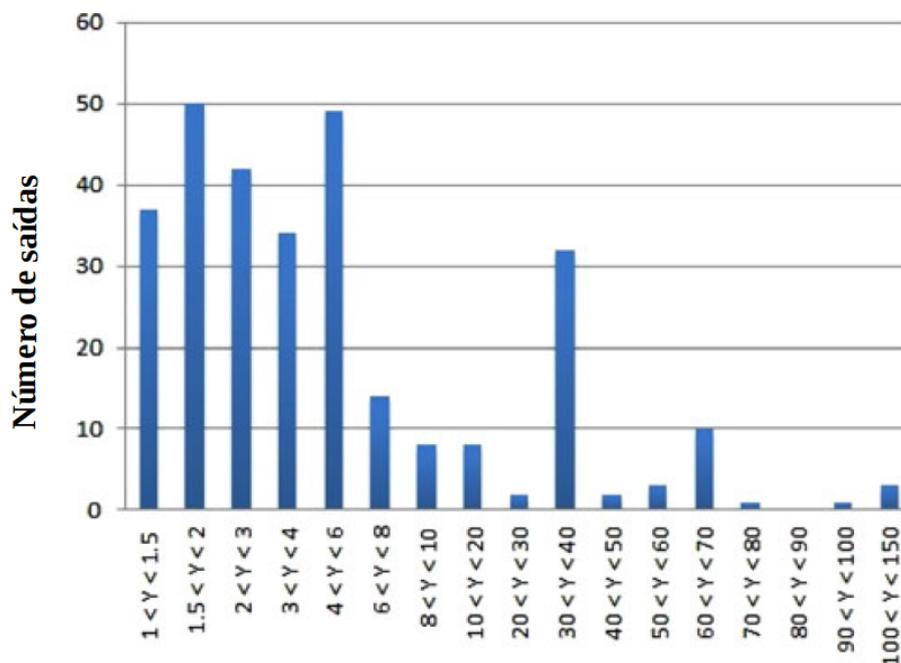


Figura 16: Diferença entre o pior e o médio PF_{OUT} ($Y = PF_{MAX} / PF_{AVG}$) para 298 sinais de saída ISCAS'85 (IBRAHIM, 2015)

mais circuitos para análise, após a fragmentação, normalmente, eles tem menos portas e sinais de entrada que o circuito original, melhorando a eficiência dos algoritmos de pesquisa. Supondo que, ao particionar o circuito do exemplo anterior, fosse gerado 5 novos circuitos com 16 sinais de entrada cada. Agora, uma pesquisa exhaustiva para todos os 5 circuitos, levaria aproximadamente 9 horas, considerando o tempo de 100ms para busca de um vetor.

Criticality Score (CS) é uma métrica usada para classificar os vetores de entrada e que foi definida em (IBRAHIM; SHOUSA; CHINNECK, 2014). Esta métrica foi usada em (IBRAHIM, 2015), com intuito de identificar o *Worst Reliability Vector* (WRV), que é o vetor associado com a maior probabilidade de falha do circuito (PF_{MAX}). A solução proposta consiste de duas partes. A primeira parte, utiliza um algoritmo de pesquisa da pior confiabilidade baseado em consenso (CWRSA) para identificação do vetor crítico. Na segunda parte, um algoritmo recursivo é usado, para encontrar as portas críticas associadas com os WRVs e que sejam comuns, para a maioria dos sinais de saída.

O algoritmo CWRSA, inicia com a geração aleatória de n vetores. Para cada vetor gerado, seu score é calculado usando o CS e uma busca entre os vetores vizinhos é realizada. Um vetor é vizinho de outro, quando eles diferem em apenas um bit, por exemplo, os vetores 100 e 101. Sempre que encontrar uma opção melhor entre os vizinhos, o vetor anterior é substituído e recomeça uma nova busca entre os vizinhos para o novo vetor. Durante todo este processo de pesquisa de vetores, uma lista com os dez mais críticos (maior score) vai sendo atualizada. Esta lista serve para determinação de consenso entre

os bits. Além disso, ela armazena o WRV.

Após a geração dos n vetores, o consenso é executado. Inicialmente, ele é configurado em 0,9. A Figura 17 mostra como o consenso é determinado pelo algoritmo. A partir da lista de vetores mais críticos, uma posição é selecionada por vez. No exemplo, a posição 2 está sendo considerada. Então, todos os vetores da lista, tem o bit da posição 2 configurado para 1. Em seguida, seus escores são calculados e a soma dos mesmos é obtida ($soma_{um}$). $Soma_{zero}$ é obtida ao configurar o bit da posição 2 em 0. O consenso é determinado, quando o cálculo realizado por C_2 for igual ou superior ao valor limite. Neste caso, foi determinado ou descoberto o bit 1 para a posição 2. Portanto, nas próximas execuções de consenso, a posição 2 não será mais analisada, pois já foi descoberto o bit para a mesma. Além disso, durante a geração aleatória de vetores, o algoritmo também leva em conta que a posição 2 já está fixada em 1. Caso nenhum bit seja descoberto, o valor limite do consenso é reduzido. Todo este procedimento, é repetido por várias vezes, até que o número de bits descobertos ultrapasse um valor pré-definido ou o consenso fique abaixo de 0,7.

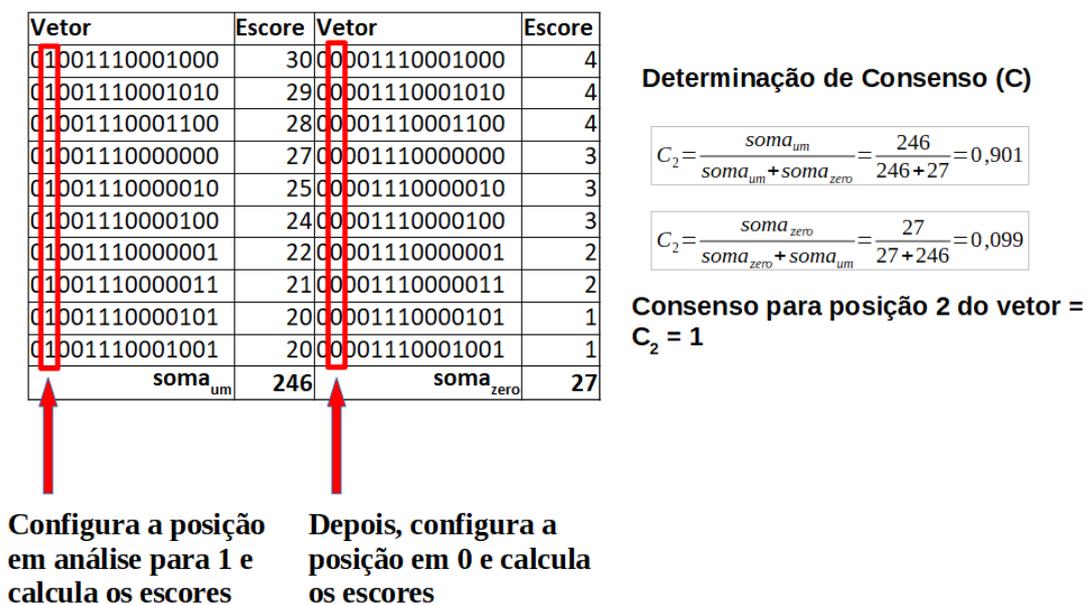


Figura 17: Determinação de Consenso do CWRSA

Na etapa seguinte, o algoritmo tenta encontrar um vetor com escore melhor que o vetor crítico identificado na etapa anterior. Para isso, ele efetua uma pesquisa sequencial até $2^{\text{bits_não_descobertos}}$ ou, caso o número de bits não descobertos seja maior que o limite (configurado em 16 bits), a busca é $2^{\text{limite_bits}}$. Por exemplo, considerando que o algoritmo obteve êxito e encontrou 17 bits em um circuito com 32 sinais de entrada. Neste caso, a pesquisa sequencial será de 2^{15} . Caso, ele tivesse descoberto apenas 12 bits, a pesquisa seria 2^{16} .

Apesar do CS possuir complexidade linear as portas do circuito, a sua precisão fica

comprometida na presença de *fanouts* reconvergentes. Além disso, o cálculo do CS para todos os nós, independentemente de sua criticalidade, afeta sua eficiência. O CNCA introduzido em (IBRAHIM; AMER, 2016), resolve algumas deficiências do CS. Todavia, ainda permite que nós não críticos sejam marcados como críticos e vice-versa. O ECNCA (IBRAHIM; AMER, 2016) consegue superar este problema, entretanto, tem sua eficiência prejudicada, já que precisa atualizar os valores lógicos dos nós subsequentes, sempre que encontrar um nó *fanout*.

Já (XIAO; LOU; JIANG, 2019), propõe um método baseado nos modelos PTM e MC para, rapidamente, quantificar os efeitos dos vetores de entrada sobre a confiabilidade do circuito e promete acurácia e pouca complexidade de tempo e espaço, superando as desvantagens do método apresentado em (IBRAHIM, 2015).

2.3 Técnicas para melhoria da Confiabilidade

A confiabilidade de um circuito, pode ser melhorada com o uso de técnicas de prevenção ou tolerância a falhas. Prevenção a falhas depende do aprimoramento de materiais, processos de fabricação e projeto de circuito. Tolerância a falhas está relacionada à redundância de recursos e é implementada a nível de circuito ou sistema (CONSTANTINESCU, 2003).

As técnicas de tolerância a falhas podem ser divididas em três classes: detecção de falhas, redundância estática e redundância dinâmica (MAXION; SIEWIOREK; ELKIND, 1987). Técnicas de detecção de falhas reconhecem que uma eventual falha irá ocorrer, não importando quão bem o sistema foi projetado. Portanto, considera o uso de recursos extras para operação normal do sistema. A ação que se segue após a detecção de uma falha, pode variar desde ignorar as falhas até tentar novamente a operação. Redundância estática, ou mascaramento de falhas, usa redundância para tolerar falhas, isolando ou corrigindo as mesmas, antes que os resultados defeituosos atinjam as saídas dos módulos. A técnica é considerada estática porque, uma vez que as cópias redundantes de um elemento são conectadas no circuito, elas permanecem fixas. Os erros resultantes de componentes defeituosos são mascarados pela presença de outras cópias desses componentes. Técnicas de redundância dinâmica envolvem a reconfiguração dos componentes do sistema em resposta a falhas. A reconfiguração evita que as falhas contribuam com seus efeitos para a operação do sistema. Em muitos instantes, a reconfiguração equivale a desconectar as unidades danificadas do sistema. A reconfiguração é acionada pela detecção de erros em sua saída. Há três problemas principais envolvidos na implantação da detecção de falhas em um sistema reconfigurável. O primeiro é o confinamento dos efeitos da falha antes que ocorram danos irreversíveis. O segundo é a detecção de falhas e o terceiro é o diagnóstico correto do local da falha, para que a unidade defeituosa seja marcada para remoção ou substituição.

Nas próximas seções, será apresentado uma breve descrição de algumas técnicas e este trabalho não pretende esgotar todas as opções. As técnicas podem ser vistas na Tabela 2, além de sua classificação.

Tabela 2: Técnicas para melhoria da confiabilidade

Classificação	Técnica
Prevenção a falhas	<i>Gate Sizing</i>
Detecção de falhas	Duplicação com Comparação Códigos de Paridade Redundância Temporal
Redundância estática	Redundância Modular Tripla (TMR) Código de Correção de Erros (ECC)
Redundância dinâmica	Redundância Modular Reconfigurável

2.3.1 *Gate Sizing*

Gate sizing é uma técnica efetiva para otimização de parâmetros do circuito, como desempenho, potência e desafios de confiabilidade, como *soft errors*. As portas lógicas são protegidas com o aumento de capacitância (para aumentar a carga crítica) ou *drive* (para dissipar a carga depositada) ou ambos, através da alteração da relação entre largura e comprimento (W/L) dos transistores (Quming Zhou; Mohanram, 2006). O aumento das dimensões de uma porta lógica, aumenta a capacitância de saída da mesma, e com isso, reduz a chance de um *soft error* ser gerado pela porta. Já a redução das dimensões de uma porta aumenta a atenuação elétrica de um SET durante a propagação do mesmo pelas portas lógicas do circuito. Estes dois comportamentos podem ser visualizados na Figura 18.

2.3.2 Redundância de Informação

Códigos de paridade e ECCs são exemplos de redundância de informação e consistem de bits ou sinais extras que são armazenados ou transmitidos junto com o dado. Códigos de paridade adicionam um bit a mais na palavra de memória e permitem a detecção de falhas simples, ou seja, falhas que afetam apenas um bit da palavra. ECCs são usados exaustivamente em memórias e permitem a detecção e correção de erros. Entretanto, é necessário adicionar um número maior de bits extras.

2.3.3 Redundância Modular

Redundância Modular é uma técnica clássica para projetar circuitos tolerantes a falha e se baseia em replicar um circuito várias vezes. *Triple Modular Redundancy* (TMR) é um caso específico desta técnica, e consiste da triplicação do circuito, onde todos os módulos

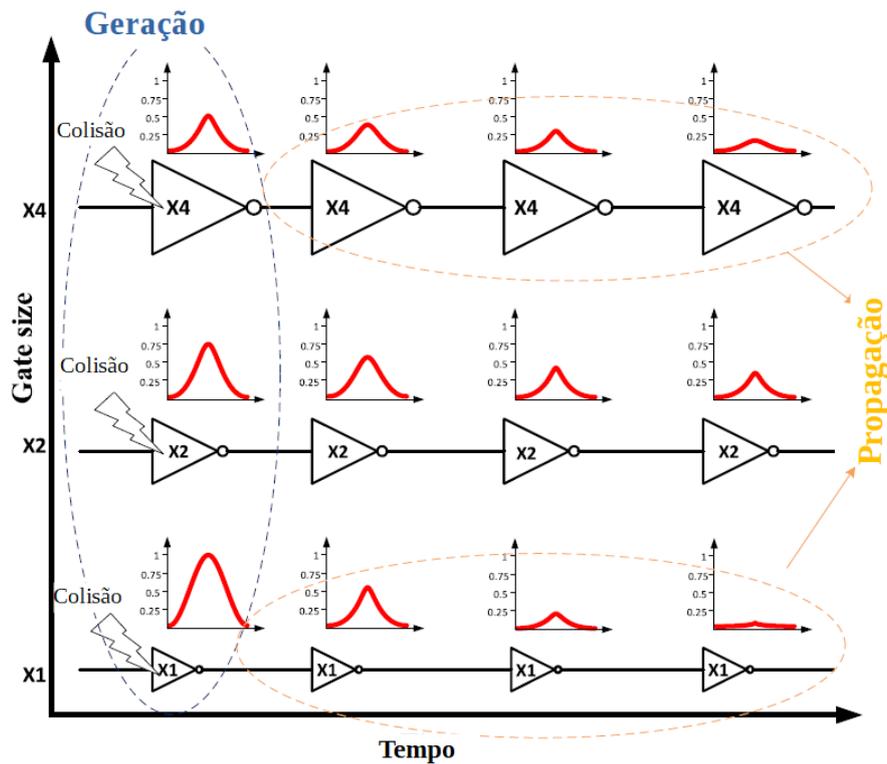


Figura 18: O impacto do *gate sizing* sobre a geração e propagação de SET adaptado de (Raji; Sabet; Ghavami, 2019)

funcionam em paralelo, possuindo as mesmas entradas, mas com saídas distintas. Um votador então, compara o valor das saídas dos módulos, e decide por maioria. Um exemplo de um esquema TMR, pode ser visto na Figura 19. O TMR é bastante eficiente, quando se considera a ocorrência de falhas únicas em um dos módulos. Sua principal desvantagem, está relacionada aos custos de área e energia decorrentes da replicação dos módulos.

Uma das desvantagens da redundância modular com votação é que a capacidade de mascarar falhas se deteriora à medida que mais cópias falham. Os módulos defeituosos acabam superando os módulos corretos. No entanto, um sistema com redundância modular pode continuar funcionando se os módulos defeituosos conhecidos puderem ser descontados na votação. Dois métodos de reconfiguração realizam essa função. Um deles é chamado de redundância híbrida, que substitui os módulos com falha por peças de reposição anteriormente não utilizadas. O outro, modifica o processo de votação dinamicamente à medida que o sistema se deteriora.

2.3.4 Duplicação com Comparação

Duplicação com Comparação (DWC) consiste em dois módulos iguais que realizam a mesma computação, recebem os mesmos dados de entrada e comparam os resultados de saída, como é apresentado na Figura 20. Caso os valores de saída sejam diferentes, é informado a ocorrência de erro. Diferentemente do TMR, a técnica DWC não mascara o

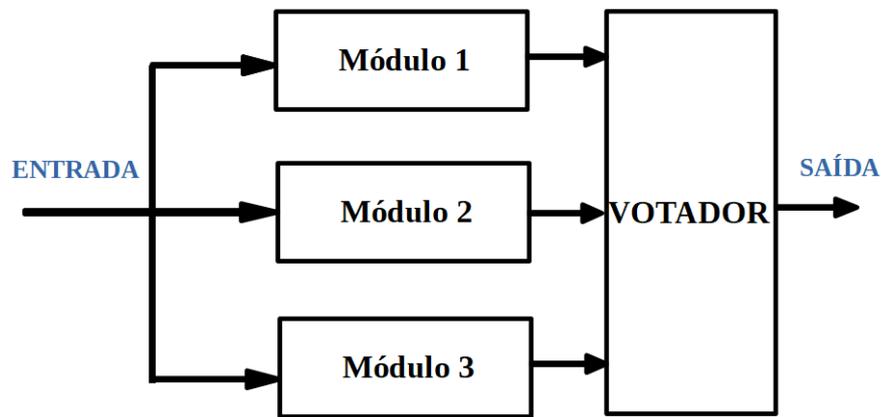


Figura 19: Esquema TMR

erro, apenas o detecta. Apesar desta técnica apresentar menor *overhead* quando comparada ao TMR, ainda continua com custos proibitivos para o uso em aplicações comerciais.

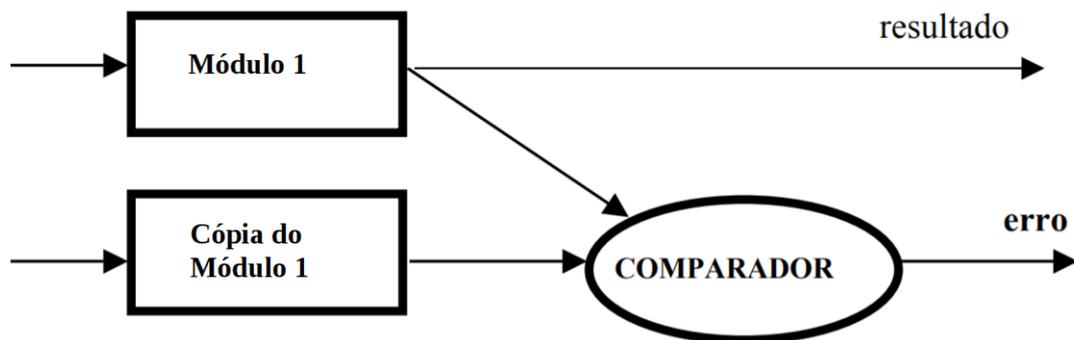


Figura 20: Esquema Duplicação com Comparação

2.3.5 Redundância Temporal

Neste tipo de técnica, o sinal de saída de um circuito combinacional é observado em diferentes momentos de tempo. Em seguida, um elemento que preserva o estado de cada sinal, faz a comparação entre os sinais (Anghel; Alexandrescu; Nicolaidis, 2000). Um esquema simples de redundância temporal pode ser visualizado na Figura 21. Este esquema reduz custos associados ao hardware, tendo apenas custo com o módulo comparador. Entretanto, necessita de mais tempo para cada computação e é usado em sistemas onde o custo com tempo não é crítico, ou quando o processador tem capacidade de computação ociosa.

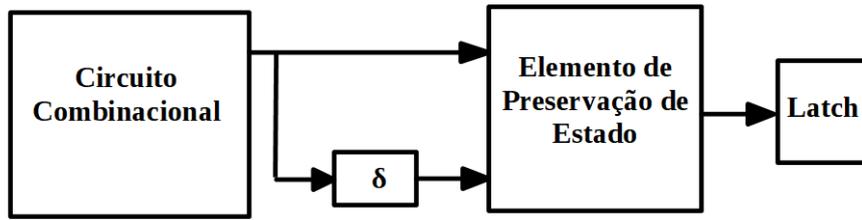


Figura 21: Redundância Temporal adaptado de (Anghel; Alexandrescu; Nicolaidis, 2000)

2.3.6 *Hardening* Seletivo

Os custos excessivos relacionados aos esquemas de redundância mencionados anteriormente, podem inviabilizar o projeto de muitas aplicações. Para lidar com este problema, uma técnica conhecida como *hardening* seletivo, pode ser utilizada (POLIAN; HAYES, 2010). O *hardening* seletivo consiste em aplicar redundância somente em partes do circuito, que estão particularmente expostas à falhas ou que são críticas para o funcionamento correto do sistema. Como resultado, é obtido um misto de componentes que são protegidos contra falhas (parte robusta do circuito) e componentes desprotegidos (parte não robusta) (POLIAN; HAYES, 2010). Para (Quming Zhou; Mohanram, 2006), a ideia principal desta técnica, é explorar as probabilidades de mascaramento lógico assimétrico das portas lógicas, protegendo as que tem menor probabilidade de mascaramento, para obter melhor *tradeoff* entre *overhead* e redução da SER.

Diversos trabalhos pretendem encontrar melhores compromissos entre confiabilidade e outros parâmetros de projeto como custo com o aumento de área, potência ou atraso. (NIEUWLAND; JASAREVIC; JERIN, 2006) implementou uma ferramenta de software para executar um novo método para melhorar a robustez de circuitos lógicos com baixo impacto na performance. O método gera uma estrutura com as portas lógicas a partir da leitura de um arquivo verilog e armazena a taxa de soft error (SER) de cada porta. Foi implementada uma heurística para substituir as portas críticas desta estrutura por portas duplicadas, as quais tem menor contribuição da SER. No exemplo da Figura 22, três portas lógicas do circuito foram duplicadas, resultando em acréscimo de área de 30% e redução de 50% da SER. A SER do circuito foi calculada como mostra a equação 8. *Failure Rate* (FR) é definida como uma função de parâmetros relacionados a topologia do circuito e propriedades básicas como a SER de uma célula padrão. FR foi calculada como em 9 e seus termos são descritos como segue:

$$SER_{circuit} = \sum_n FR_{gate(n)} \quad (8)$$

$$FR_{gate} = Gate_{SER} \times Glitch_{observability} \times EMF \quad (9)$$

- $Gate_{SER}$ é obtida somando o resultado da multiplicação das probabilidades de todos

os vetores de entrada com as correspondentes taxas de falhas.

- *Glitch_{observability}* - uma falha na entrada de uma porta lógica pode ser observada em sua saída, quando as outras entradas não tem um valor de controle, ou seja, o valor que força a saída para determinado valor. No caso de uma porta AND ou NAND, o valor de controle é 0, e para as portas OR e NOR, o valor é 1.
- Fator de mascaramento elétrico (EMF) - uma falha ocorrida em um local do circuito vai ser atenuada caso ela não tenha amplitude e duração suficientemente grandes. A probabilidade de atenuação depende do tamanho do caminho e dos tipos de portas no caminho. Quanto mais distante da saída, maior é a probabilidade da falha ser atenuada. Além disso, diferentes portas tem diferentes propriedades de atenuação, devido as diferenças de capacitâncias parasitas. Por exemplo, um inversor tem baixa capacidade de atenuar uma falha.

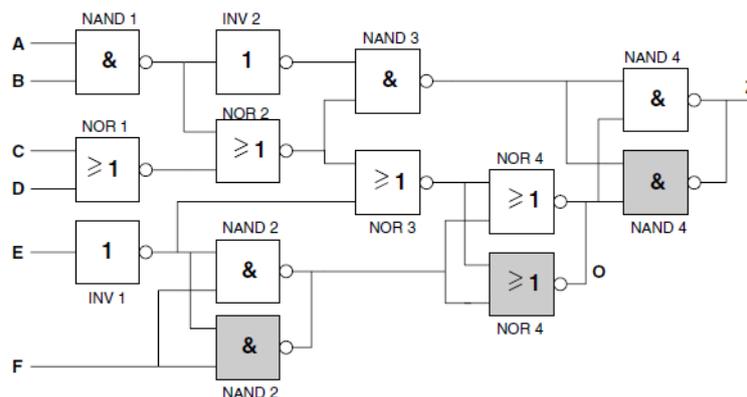


Figura 22: Circuito com 50% de redução da SER através da duplicação de três portas

O trabalho proposto por (Quming Zhou; Mohanram, 2006) utiliza o *gate sizing* para redução da taxa de *soft error*, através do redimensionamento seletivo das portas mais sensíveis a falhas do circuito. A técnica apresenta *overhead* médio de 38,3% em área, 27,1% em potência e 3,8% em atraso.

Em (MARQUES; BARROS NAVINER; NAVINER, 2010), foi proposta uma ferramenta que auxilia o projetista a selecionar uma implementação mais adequada, para determinada aplicação, conforme os requisitos de projeto. A ferramenta gera diferentes versões do projeto original, baseado em TMR e as classifica, de acordo com métricas fornecidas pelo projetista.

No trabalho de (NAVINER et al., 2011), foi apresentado dois novos métodos de classificação que se baseiam em conceitos de sensibilidade e elegibilidade. O primeiro conceito, tem a ver com a sensibilidade da confiabilidade R de um circuito, com relação à confiabilidade de um bloco, e significa uma métrica que fornece o impacto em R ao

alterar a confiabilidade deste bloco. Já a elegibilidade de um bloco, é definida como uma métrica, que expressa como a melhoria de confiabilidade deste bloco é significativa.

(PAGLIARINI; NAVINER; NAVINER, 2012) apresentou uma metodologia para o *hardening* seletivo, baseada no algoritmo SPR para o cálculo de confiabilidade e que fornece uma lista de blocos combinacionais candidatos, de acordo com sua importância relativa, com relação à confiabilidade do circuito. A determinação de um parâmetro de afinidade de *hardening* permite estabelecer compromissos entre o ganho de confiabilidade com os custos associados ao *hardening*.

Em (BAN; JUNIOR, 2017), foi apresentado um método para aplicar o *hardening* seletivo, com uma abordagem diferente para identificar os blocos lógicos críticos em circuitos aritméticos. A ideia principal, é que erros diferentes podem ter consequências diversas para diferentes aplicações digitais. Por exemplo, em uma palavra binária de saída, erros localizados nos bits mais significativos, tendem a ser mais críticos do que nos bits menos significativos.

(WALI et al., 2017) apresentou uma metodologia de análise da suscetibilidade de saída, baseada no fato que, nem todas as saídas de um circuito combinacional, tem a mesma suscetibilidade a *Single Event Transient* (SET). A suscetibilidade é determinada em função do número de entradas e saídas das portas lógicas, presentes em seus cones lógicos *fan-in*, como mostra a Figura 23. De acordo com a figura, é possível inferir que a saída O_1 é mais suscetível a SET que O_2 . Com base nesta metodologia, foi apresentada uma técnica de *hardening* seletivo, que emprega duplicação e comparação para detectar falhas e um esquema de *rollback* de ciclo único para correção.

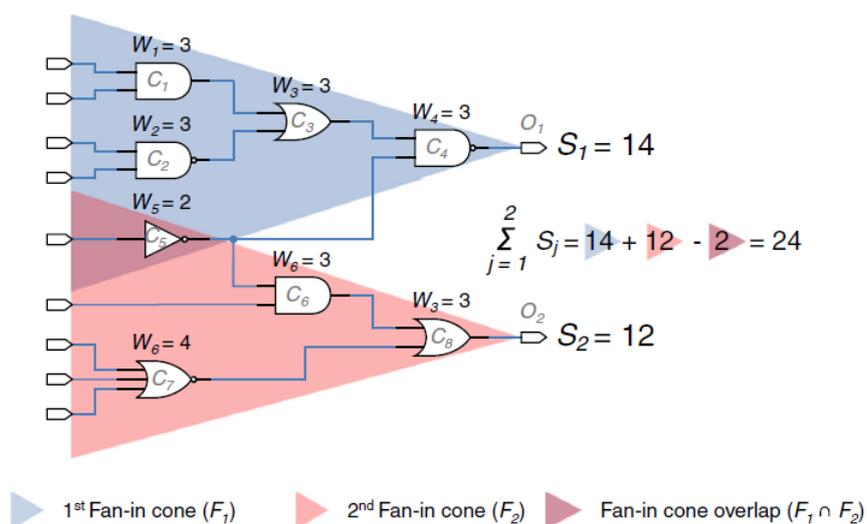


Figura 23: Análise de suscetibilidade de saída (WALI et al., 2017)

3 METODOLOGIA

Para atender os objetivos propostos neste trabalho, foram realizados três experimentos que serão explicados nas próximas seções. Com intuito de facilitar o entendimento de nossos testes, a definição de R_{Med} e R_{Crit} será apresentada a seguir:

- R_{Med} , ou confiabilidade média, é a média das contribuições individuais dos vetores de entrada de um circuito.
- R_{Crit} , ou confiabilidade crítica, é o pior valor de confiabilidade associado a um vetor de entrada específico.

Com exceção das rotinas *Ler um circuito* e *Calcular a confiabilidade R*, diversos métodos foram desenvolvidos para a realização dos experimentos e serão listados a seguir. Mais detalhes sobre os mesmos, serão apresentados na medida que forem utilizados.

1. Calcular a criticidade das portas lógicas
2. Ordenar as portas lógicas
3. Melhorar a confiabilidade ao proteger as portas lógicas
4. Algoritmo para identificação do vetor crítico
5. Identificação dos vetores vizinhos
6. Criação de sub-circuitos

Para o cálculo de confiabilidade R_{Med} e R_{Crit} , foi utilizada uma ferramenta que implementa o método SPR (PONTES et al., 2018). Embora o método SPR apresente valores imprecisos, o trabalho de (PONTES, 2019), comprovou que o mesmo, pode ser usado como uma métrica de confiabilidade e identificar os pontos mais sensíveis de um circuito, assim como, o incremento ou decremento da confiabilidade, quando ocorre alterações nestes pontos. Além disso, em (PAGLIARINI; NAVINER; NAVINER, 2012), este método já havia sido utilizado para identificar os blocos mais críticos dos circuitos.

Para o cálculo SPR, foi definido o valor individual de confiabilidade de uma porta lógica $q=0,99999$. Quando nos referirmos a proteger uma porta, processo que poderia ser feito com a utilização de alguma das técnicas de melhoria apresentadas no Capítulo 2, significa que estamos considerando hipoteticamente $q=0,999999$.

3.1 Análise de R_{Med} ao proteger portas lógicas, considerando diferentes ordenações

Neste experimento, foram realizadas três análises de R_{Med} , considerando os seguintes cenários de ordenação das portas lógicas:

- Ordenação das portas lógicas, conforme o grau de criticalidade das mesmas para o circuito, ou seja, da porta mais crítica até a porta menos crítica.
- Ordenação inversa ao primeiro cenário, ou seja, da porta menos crítica para a porta mais crítica.
- Ordenação aleatória das portas lógicas.

O objetivo deste experimento, é analisar as diferenças de R_{Med} , ao proteger as portas lógicas de um circuito, considerando os cenários descritos acima. Por exemplo, poderemos determinar qual a diferença entre R_{Med} ao proteger $x\%$ das portas mais críticas versus $x\%$ das portas menos críticas.

A figura 24 apresenta o fluxograma para esta análise. Um pseudocódigo pode ser visto no Apêndice A. As etapas do fluxograma são descritas como segue:

1. **Ler Circuito** - inicializa um circuito através da leitura de um arquivo verilog e obtém uma lista de todas as portas lógicas do mesmo (linha 5 do Pseudocódigo A).
2. **Calcular a criticalidade das portas lógicas** - recebe a lista de portas da etapa anterior. Uma porta é selecionada por vez e tem sua confiabilidade configurada para $0,999999$ e o restante das portas em $0,99999$ e R_{Med} é calculada. Depois de repetir este procedimento para todas as portas, é gerada uma lista contendo o grau de criticalidade de cada porta (linha 6 à 10 do Pseudocódigo A).
3. **Ordenar as portas lógicas** - ordena as portas, conforme o grau de criticalidade das mesmas, ou seja, começando da porta mais crítica até a menos crítica para o circuito (ordem decrescente de R_{Med}). Além disso, são criadas mais duas listas com as seguintes ordenações: ordem crescente de R_{Med} e ordem aleatória (linha 11 do pseudocódigo A).
4. **Melhorar a confiabilidade de $x\%$ de portas lógicas** - para cada lista de portas ordenadas, configura a confiabilidade de $x\%$ das portas da lista para $0,999999$ e o restante em $0,99999$ (linha 13 do pseudocódigo A).

5. **Calcular R_{Med}** - para cada lista, R_{Med} é calculada (linha 14 do pseudocódigo A).

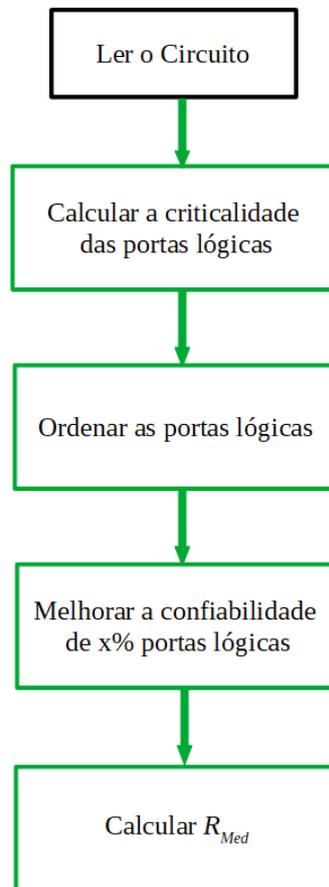


Figura 24: Fluxograma da análise de R_{Med} considerando diferentes ordenações das portas lógicas

3.2 Análise de R_{Med} e R_{Crit} ao proteger portas lógicas

Um dos objetivos deste experimento, é avaliar a confiabilidade do circuito ao identificar o vetor crítico R_{Crit} . Além disso, analisar qual impacto para R_{Med} e R_{Crit} , ao proteger as portas lógicas, considerando sua criticidade para o circuito geral e criticidade para o vetor crítico. Basicamente, com esta simulação, nós queremos responder as seguintes questões:

1. Qual é a diferença entre R_{Med} e R_{Crit} ?
2. Quanto é a melhoria de R_{Med} e R_{Crit} , ao proteger as portas críticas de um circuito?
3. Quanto é a melhoria de R_{Med} e R_{Crit} , ao proteger as portas críticas para o vetor crítico?
4. Qual é o custo para a identificação do vetor crítico?

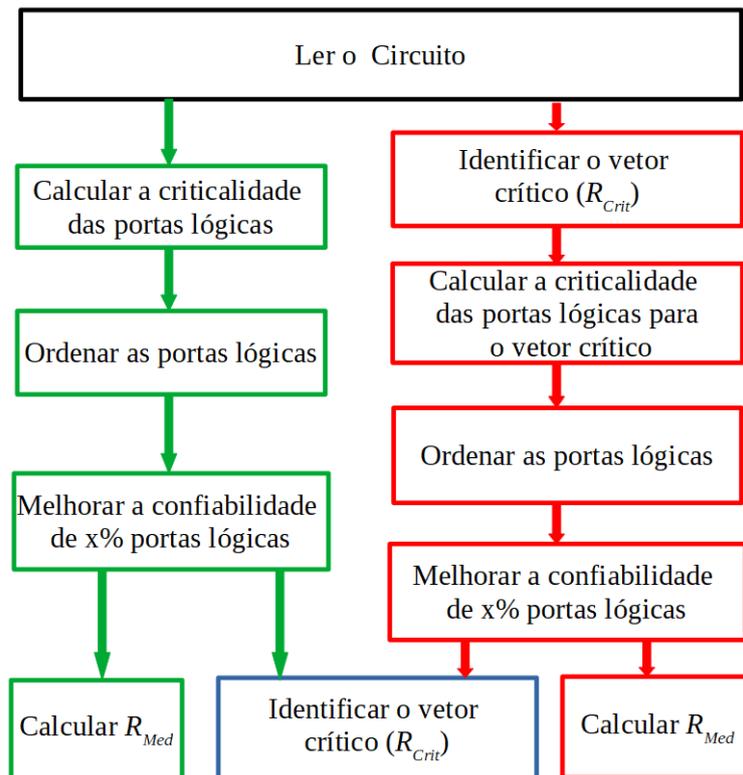


Figura 25: Fluxograma da análise de R_{Med} e R_{Crit}

A Figura 25 mostra o fluxograma para o experimento. Os blocos na cor verde, são idênticos aos blocos do experimento anterior. Entretanto, somente a melhor ordem de portas vai ser gerada (ordem decrescente de R_{Med}). Os blocos **Identificar o vetor crítico** (R_{Crit}) (cores vermelha e azul), executam o mesmo algoritmo. A diferença é que para o bloco vermelho, além da informação de confiabilidade, estamos interessados também no valor binário do vetor, ex: 10011. Este binário é repassado para o método **Calcular a criticidade das portas lógicas para o vetor crítico**. A única diferença deste método, para o **Calcular a criticidade das portas lógicas** é o vetor crítico que é passado para o cálculo do SPR. Neste caso, temos uma lista com a criticidade das portas para o vetor crítico.

Para a execução do bloco **Identificar o vetor crítico**, foi implementado um algoritmo em Java, baseado na heurística apresentada em (IBRAHIM, 2015). Para o cálculo dos escores dos vetores, nós utilizamos o método SPR. Mais detalhes sobre o algoritmo pode ser encontrado na Seção 2.2.3 e um pseudocódigo pode ser visualizado no Apêndice B. A heurística consiste em determinar bits nos vetores de entrada que potencializam uma maior probabilidade de falha do circuito. A cada bit descoberto, se reduz o espaço de pesquisa pela metade. A Figura 26, apresenta um exemplo, em que foram obtidos os dez vetores mais críticos do circuito C17. É possível notar que, o valor de in1 é 1 em nove dos dez vetores.

Existem algumas variáveis que determinam a precisão e eficiência do algoritmo. De-

Vetor					Confiabilidade
in5	in4	in3	in2	in1	
0	1	1	1	1	0,9999400017
0	1	1	1	0	0,9999400018
0	0	1	1	1	0,9999400019
1	1	1	1	1	0,9999400019
0	0	0	0	1	0,999940002
0	0	0	1	1	0,999940002
0	0	1	0	1	0,999940002
1	0	0	0	1	0,999940002
1	0	0	1	1	0,999940002
1	0	1	1	1	0,999940002

Figura 26: Dez vetores mais críticos do circuito C17

vido às características topológicas dos circuitos, é interessante ajustar dinamicamente seus valores. Por exemplo, quando o método de consenso é executado, e nenhum bit é descoberto, então o limite de consenso é reduzido, ou seja, ele começa em 0,9 e pode chegar até 0,7. Esta redução melhora a eficiência, mas diminui a precisão do algoritmo. Além disso, n vetores aleatórios são gerados e tem seus escores melhorados com uma pesquisa entre os vetores vizinhos. Melhorar os escores, significa encontrar um vetor que tem confiabilidade menor que o vetor em análise. Os vetores que tiverem os piores valores de confiabilidade, irão compor uma lista de vetores candidatos. Esta lista será utilizada pelo método de consenso. Aumentar n , melhora a precisão, entretanto, há perda de eficiência. Portanto, n pode ser reduzido, quando um determinado número de bits for descoberto ou, quando o limite de consenso ultrapassar determinado valor.

Em nossa implementação do algoritmo, foram introduzidas três otimizações que melhoram a eficiência do mesmo e serão explicadas a seguir:

1. Uma lista com o valor decimal dos vetores pesquisados foi armazenada. Então, antes de calcular o escore de um vetor, a lista é consultada. Portanto, um vetor não é calculado mais de uma vez. Pode parecer incomum que, ao gerar um vetor aleatoriamente, em um universo enorme de possibilidades, o mesmo irá se repetir. Entretanto, isto ocorre, principalmente quando os vetores vizinhos são consultados. Esta otimização melhora a eficiência do método, pois se evita cálculos desnecessários do SPR.
2. Em alguns casos, ao analisar a lista de vetores candidatos, todos tem o mesmo valor de confiabilidade associado. Portanto, foram criadas outras listas temporárias, com outros vetores, desde que o valor de confiabilidade associado aos mesmos fosse igual ao da lista principal. Então, o consenso é executado para todas as listas, e a lista onde o maior número de bits for descoberto, passa a ser a principal e as outras listas são descartadas. Esta otimização teve efeito prático apenas para o

circuito C499, em função da enorme quantidade de vetores com o mesmo valor de confiabilidade associado.

- Quando o consenso é executado, independentemente de haver ou não descoberta de bits, nós armazenamos o valor do limite de consenso mais próximo para descoberta de bits. Por exemplo, considerando que o limite de consenso atual é 0,89 e o valor mais próximo ao calcular o consenso é 0,85. Então, caso a lista de vetores candidatos não se altere nas próximas iterações do algoritmo, o consenso não precisa ser executado novamente, até que o limite de consenso tenha sido reduzido para 0,85.

3.3 Análise de R_{Crit} ao proteger as portas críticas a partir da avaliação individual das saídas do circuito

O último experimento pretende investigar R_{Crit} , a partir da proteção de circuitos fracionados, que chamamos de sub-circuitos. Neste caso, a análise é feita considerando individualmente as saídas do circuito. Em seguida, será realizado um estudo comparativo entre R_{Crit} obtido ao proteger as portas dos sub-circuitos e o R_{Crit} do experimento anterior (circuito original).

Um sub-circuito é construído, considerando os caminhos lógicos existentes entre uma saída primária até as entradas primárias. Para cada sub-circuito gerado, uma descrição Verilog do mesmo é armazenada em arquivo. A Figura 27 mostra os dois novos sub-circuitos, ao considerar o circuito C17. A criação dos sub-circuitos, permite uma análise da probabilidade de falha máxima de cada saída. Conforme mencionado anteriormente, esta pode ser ordens de magnitude maior que a probabilidade de falha média (IBRAHIM, 2015). Neste caso, poderia se proteger as portas para uma saída específica. Além disso, ao dividir os circuitos, em muitos casos, os sub-circuitos resultantes, tem menos sinais de entrada e portas lógicas que o circuito original, e isto facilita a análise de R_{Crit} . Considere uma pesquisa exaustiva da confiabilidade para o circuito C17 do exemplo (5 sinais de entrada), e os sub-circuitos (4 sinais de entrada cada). Em ambos os casos, o total de vetores pesquisados é 32 ($2^4 + 2^4 = 32$, para os sub-circuitos). Ainda assim, a análise dos sub-circuitos é mais rápida, pois estes possuem 4 portas, enquanto o circuito original possui 6. Ao considerar circuitos maiores, estas diferenças aumentam, e isso será mostrado nos resultados.

Após a geração dos sub-circuitos, o próximo passo é identificar o vetor crítico para cada sub-circuito e, em seguida, ordenar as portas lógicas de acordo com a criticalidade das mesmas para o vetor crítico. Um fluxograma desses passos é apresentado na Figura 28. Os seus blocos já foram explicados nos experimentos anteriores. Depois de obter as portas ordenadas, de acordo com sua criticalidade para o vetor crítico de cada sub-circuito, nós vamos somar as criticalidades das portas para cada sub-circuito. No exemplo

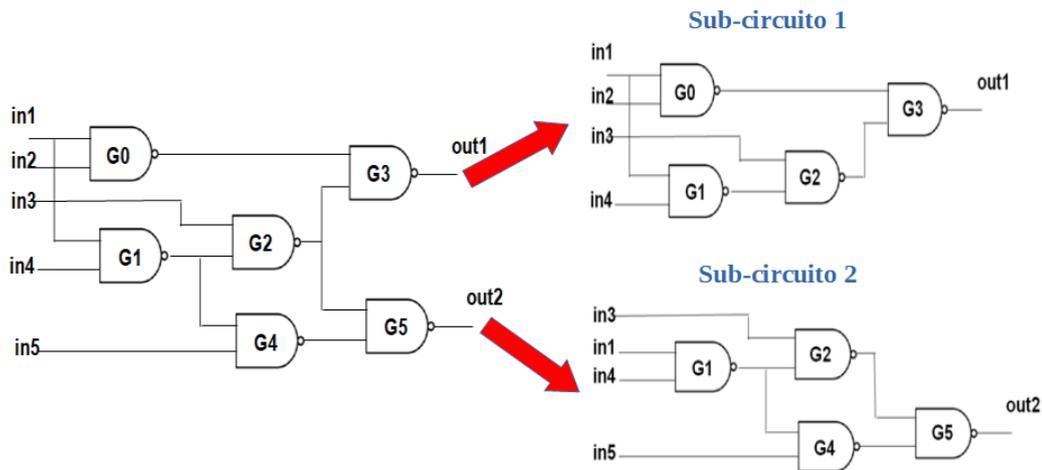


Figura 27: Sub-circuitos do C17

Tabela 3: Soma das criticidades dos sub-circuitos

	Porta	Peso	Porta	Peso	Porta	Peso	Porta	Peso	Porta	Peso
Sub 1	g10	1,056	g4	1,056	g2	1,055	g13	1,055	g16	1,054
Sub 2	g10	1,088	g2	1,045	g16	1,045	g9	1,045	g8	1,044
Sub 3	g1	1,06	g3	1,059	g14	1,058	g16	1,058	g2	1,057
Sub 4	g14	1,15	g16	1,15	g15	1,15	g2	1,09	g8	1,03
Sub 5	g3	1,07	g2	1,05	g15	1,05	g4	1,04	g14	1,04
Total	g2	5,297	g16	4,307	g14	3,248	g15	2,2	g10	2,144

apresentado na Tabela 3, nós consideramos as 5 portas mais críticas para cada vetor crítico dos 5 sub-circuitos analisados. Juntamente com a porta, tem a informação da criticidade da mesma (coluna Peso). Este valor é obtido ao dividir a confiabilidade associada ao vetor crítico ao proteger uma porta específica, pela confiabilidade associada ao vetor crítico, quando nenhuma porta é protegida. A última linha da Tabela é obtida, ao efetuar a soma dos pesos de cada porta. A última parte do experimento é executada no circuito original. Com a nova ordem das portas, estas são protegidas e R_{Crit} é calculado, como apresentado na Figura 29.

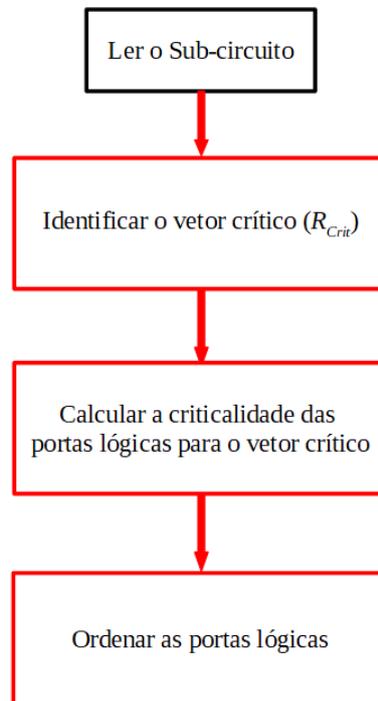


Figura 28: Ordenação das portas lógicas de cada sub-circuito

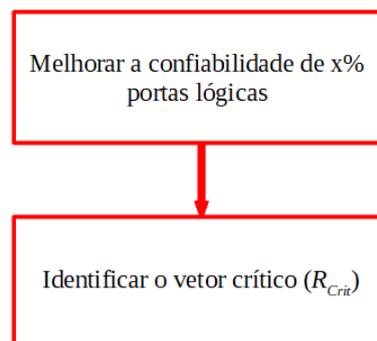


Figura 29: Análise de R_{Crit} considerando o circuito original

4 RESULTADOS

Neste Capítulo, serão apresentados os resultados obtidos ao utilizar a metodologia definida anteriormente. Os experimentos foram executados em um computador com processador Intel Core i7 de 1.80GHz, 4 núcleos e 16 GB de RAM. Os circuitos analisados fazem parte dos *benchmarks ISCAS'85* (BRGLEZ; FUJIWARA, 1985) e foram mapeados usando a ferramenta ABC (BERKELEY, 2009) com uma biblioteca contendo as seguintes portas lógicas: *INV*, *BUF*, *NOR*, *OR*, *NAND*, *AND*, *OAI*, *AOI* e *XOR*. A descrição da biblioteca pode ser vista no Apêndice C.

A Tabela 4 mostra as características dos circuitos. Nela, são apresentadas a quantidade de portas e níveis lógicos dos mesmos. A quantidade de portas pode ser relacionada com a área dos circuitos e a quantidade de níveis lógicos com o atraso. No escopo da estimativa de confiabilidade usando o método SPR, a quantidade de portas lógicas é o fator mais importante. Ao se avaliar o vetor crítico, a quantidade de sinais de entrada passa a ser preponderante em função do total de combinações possíveis. Neste sentido, observa-se que os circuitos possuem relevante variação em suas características, sendo adequados para as análises a serem apresentadas. Por fim, todos os resultados apresentados nas próximas seções, serão expressos em função do MTBF. A escolha deste, se dá pelo fato do mesmo ser uma métrica de fácil compreensão, pois ele reflete diretamente o tempo entre as falhas de um circuito.

Tabela 4: Circuitos Benchmarks ISCAS'85

Circuito	Entradas	Saídas	Portas	Nível
c432	36	7	136	14
c499	41	32	188	9
c880	60	26	229	11
c1355	41	32	546	24
c1908	33	25	234	14
c2670	233	139	543	11
c3540	50	22	673	18
c5315	178	123	1026	18

Tabela 5: MTBF dos circuitos analisados

Circuito	MTBF	
	Original	Proteção total
c432	884	8840
c499	812	8120
c880	763	7630
c1355	449	4490
c1908	587	5870
c2670	246	2460
c3540	136	1360
c5315	152	1520

4.1 Análise da Confiabilidade Média (R_{Med})

Nesta seção, a análise da confiabilidade média dos circuitos é apresentada. Ela segue diretamente a metodologia definida na Seção 3.1. Além da análise da confiabilidade média, é apresentado o custo temporal da mesma.

A Tabela 5 apresenta o MTBF, ao considerar os circuitos sem portas protegidas ($q=0,99999$) e quando todas as portas são protegidas ($q=0,999999$). Como mostra a Tabela, a proteção total das portas, aumenta 10 vezes o valor do MTBF.

O gráfico da Figura 30 apresenta os valores MTBF, ao proteger as portas lógicas do circuito c432, considerando as três ordenações de portas que foram introduzidas na metodologia. A legenda *Ordem Mais Críticas* se refere a ordenação decrescente de R_{Med} , ou seja, da porta mais crítica para a menos crítica. *Ordem Menos Críticas* se refere a ordem crescente de R_{Med} . E por fim, *Ordem Aleatória* é obtida sem levar em conta R_{Med} . Para qualquer número de portas protegidas, a *Ordem Mais Críticas* sempre é a melhor escolha. Ao comparar a diferença de MTBF considerando *Ordem Mais Críticas* e *Ordem Menos Críticas*, esta aumenta até certo limite, conforme o número de portas protegidas aumenta. A diferença média é 179%, enquanto a maior diferença é 346%. O valor médio é obtido ao considerar todas as diferenças com a proteção começando com uma porta até o total de portas do circuito. Já quando o MTBF é comparado, considerando *Ordem Mais Críticas* e *Ordem Aleatória*, a diferença máxima é 127%. Esta simulação nos mostra, o quanto é importante escolher as portas corretas para proteção. Por exemplo, para obter um MTBF igual a 2000, ao escolher a *Ordem Mais Críticas*, seria necessário proteger aproximadamente 40 portas. Caso outra ordem fosse escolhida, seria preciso no mínimo o dobro de portas.

A Tabela 6 apresenta as diferenças para todos os circuitos analisados. Como pode ser visto, o comportamento é semelhante ao c432. O circuito c499 apresentou as menores diferenças, provavelmente em função do número de portas XOR, que representa aproxi-

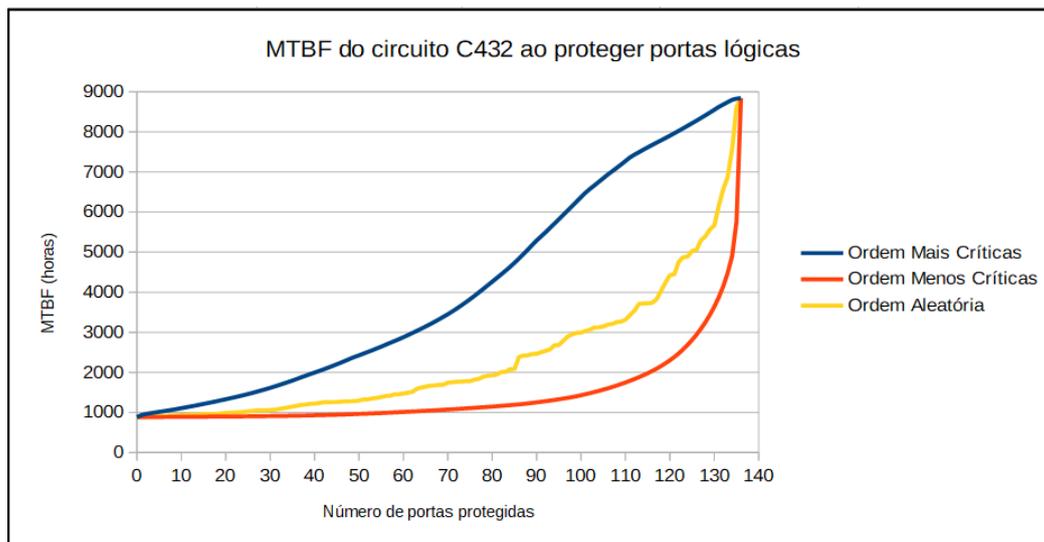


Figura 30: Análise do MTBF do C432 ao proteger as portas lógicas

madamente 65% do total de portas do circuito. A porta *XOR* é considerada crítica, pois não consegue mascarar logicamente uma falha em suas entradas. O circuito c3540 apresenta a maior diferença (maior que 6x). O mesmo possui aproximadamente 8% de portas *XOR*, além de ser um dos maiores da análise, em termos de portas lógicas.

Tabela 6: Diferença de MTBF ao considerar diferentes ordens das portas protegidas

Circuito	Mais Críticas x Menos Críticas		Mais Críticas x Aleatórias	
	Dif. Média (%)	Maior Dif. (%)	Dif. Média (%)	Maior Dif. (%)
c432	179	346	76	127
c499	83	151	42	82
c880	143	272	72	136
c1355	184	324	80	129
c1908	267	494	119	239
c2670	237	450	118	231
c3540	309	574	116	211
c5315	208	362	88	144
Média	201	371	89	162

O custo temporal para obtenção das portas ordenadas para os circuitos utilizados, pode ser visualizado na Tabela 7. Conforme é apresentado, o custo para a análise da confiabilidade média usando o SPR é baixo. Basicamente, o tempo para o cálculo de confiabilidade é linear ao número de portas lógicas, e isto se confirmou para a maioria dos circuitos. Entretanto, uma porta com mais entradas demanda mais tempo para o cálculo do que outra que tenha menos entradas. Este comportamento pode ser observado para os circuitos c1355 e c2670, cujo circuito com mais portas, executou em menos tempo. Para o c1355, excetuando-se portas de 2 entradas e inversores, o circuito tem 18 portas com

mais de 2 entradas, enquanto o c2670 tem 142 portas.

Tabela 7: Custo Temporal ao proteger portas lógicas

Circuito	Portas	Tempo (ms)
c432	136	440
c499	188	645
c880	229	788
c1908	234	963
c1355	546	2116
c2670	543	2930
c3540	673	8068
c5315	1026	10747

4.2 Análise da Confiabilidade (R_{Med} versus R_{Crit})

Os resultados que serão apresentados a seguir, foram obtidos ao utilizar a metodologia apresentada na Seção 3.2. O MTBF é referenciado como caso médio e pior caso, para distinguir entre R_{Med} e R_{Crit} , respectivamente. Inicialmente, a análise apresenta os valores absolutos de ambos e suas diferenças. Na sequência, a análise do MTBF considera a proteção das portas lógicas, com diferentes ordenações das mesmas. Por fim, o custo temporal para identificação do vetor crítico é apresentado.

Tabela 8: Comparação do MTBF (caso médio e pior caso)

Circuito	MTBF		Diferença (x)
	Caso Médio	Pior Caso	
c432	884	100	8,84
c499	812	429	1,89
c880	763	148	5,16
c1355	449	40	11,51
c1908	587	191	3,07
c2670	246	64	3,84
c3540	136	11	12,36
c5315	152	31	4,90

A Tabela 8 apresenta os valores MTBF (caso médio e pior caso) dos circuitos analisados, e a diferença entre eles. Novamente, a menor diferença é do circuito c499, provavelmente devido ao número de portas XOR. O circuito c1908, que é a segunda menor diferença, tem 35% de portas XOR. Já os circuitos c1355 e c3540, que tem as maiores diferenças, possuem 0% e 8% de portas XOR, respectivamente. Além disso, um único

vetor crítico não vai ser crítico para todas as saídas do circuito. Neste caso, existem sinais de entrada que são críticos para algumas saídas e para outras, não. Devido a estes conflitos, acontece compensações entre as saídas. Por isso, as diferenças são menores em uma análise geral do circuito, como pode ser visto com os circuitos c2670 e c5315, que são os que possuem as maiores quantidades de sinais de entrada e saída.

Tabela 9: MTBF (caso médio e pior caso) ao proteger portas lógicas (ordem decrescente de R_{Med})

Circuito	20% Portas		40% Portas		60% Portas		80% Portas	
	MTBF		MTBF		MTBF		MTBF	
	Médio	Pior	Médio	Pior	Médio	Pior	Médio	Pior
c432	1518	128	2597	244	4354	389	7101	656
c499	1114	501	1793	682	2966	1236	5115	2062
c880	1184	190	2016	235	3395	309	5906	413
c1355	838	41	1502	45	2179	67	3258	188
c1908	1144	232	2047	259	4159	366	5476	759
c2670	443	101	784	127	1578	185	2375	297
c3540	294	22	540	33	994	52	1314	105
c5315	282	48	497	76	803	108	1214	171

A Tabela 9 apresenta o MTBF (caso médio e pior caso), ao considerar a proteção das portas lógicas (ordem decrescente de R_{Med}). Os valores 20%, 40%, 60% e 80% se referem a quantidade de portas protegidas. Com a análise destes valores, é possível inferir que as diferenças entre ambos aumentam em comparação com a diferença inicial (sem portas protegidas). Este cenário pode ser visto no gráfico da Figura 31. Como foi comentado anteriormente, devido às capacidades de mascarar falhas, o circuito c1355 apresentou as maiores diferenças, chegando a 3x, com 40% de proteção, enquanto o circuito c499 tem as menores diferenças.

Com esta análise, é possível concluir que, a proteção das portas mais críticas para o circuito, não melhora o pior caso do MTBF na mesma proporção que o caso médio. Portanto, esta estratégia não é a mais adequada, caso o objetivo seja aumentar o limite inferior de confiabilidade.

A Tabela 10 apresenta os valores de MTBF (caso médio), obtido com a proteção de portas com ordem decrescente de R_{Crit} . Todos os valores são menores, quando comparados com os valores da Tabela 9. Portanto, é possível concluir que, a proteção de portas com ordem decrescente de R_{Med} , é mais adequada para melhorar a confiabilidade média de um circuito. O gráfico da Figura 32, mostra as diferenças dos circuitos c1355 e c499, os quais apresentaram a maior e a menor diferença, respectivamente. O comportamento de ambos foi parecido. As diferenças aumentaram até 40% de proteção e depois foram sendo reduzidas.

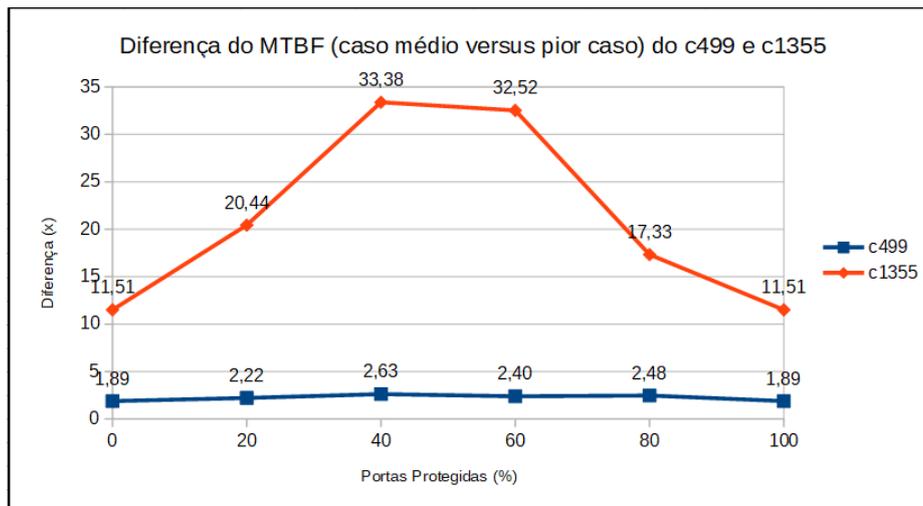


Figura 31: Diferença de MTBF (caso médio versus pior caso) dos circuitos c499 e c1355

Tabela 10: Análise MTBF (caso médio) ao proteger portas lógicas (ordem decrescente de R_{Crit})

Circuito	MTBF (caso médio)			
	20%	40%	60%	80%
c432	1256	1547	2220	3203
c499	1035	1372	2525	4368
c880	971	1472	2152	3373
c1355	491	535	832	2266
c1908	845	1340	2000	4551
c2670	393	488	629	1300
c3540	217	333	511	552
c5315	208	294	486	858

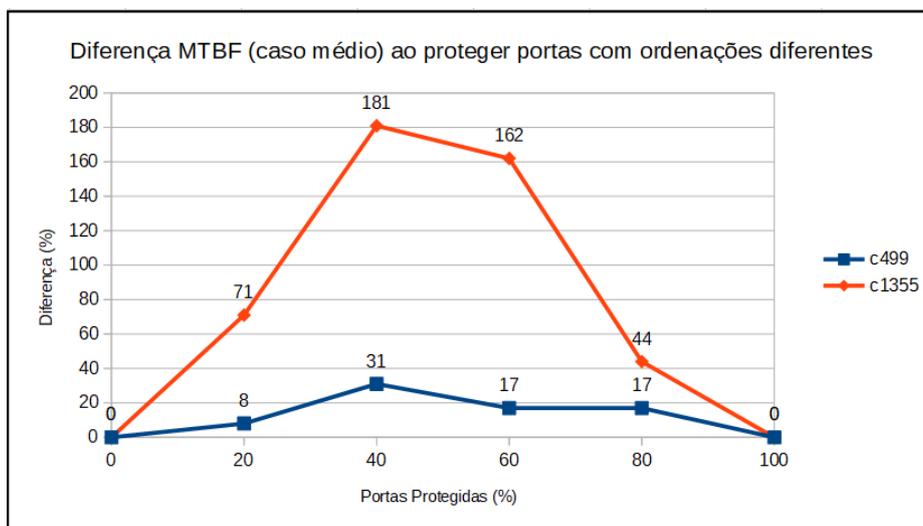


Figura 32: Diferença do MTBF (caso médio) ao proteger as portas lógicas considerando duas ordenações (ordem decrescente de R_{Med} e ordem decrescente de R_{Crit})

A análise seguinte considera o pior caso do MTBF ao proteger as portas lógicas considerando diferentes ordenações, como mostra o gráfico da Figura 33. A legenda *Ordem das portas críticas para o vetor crítico* é obtida com o cálculo da criticalidade das portas, considerando R_{Crit} . Já para a legenda *Ordem das portas críticas para o circuito*, o cálculo da criticalidade considera R_{Med} . A primeira ordenação é mais vantajosa, quando consideramos a proteção de 20% e 40% das portas. Nestes dois pontos de proteção, o valor do MTBF duplica e depois triplica. Entretanto, a partir de 40% até 80%, há um leve aumento do MTBF, e a outra ordenação produz os melhores resultados. Isto acontece, porque um outro vetor passa a ser o vetor crítico e as portas protegidas nos pontos citados não são adequadas para melhorar a confiabilidade associada a este vetor.

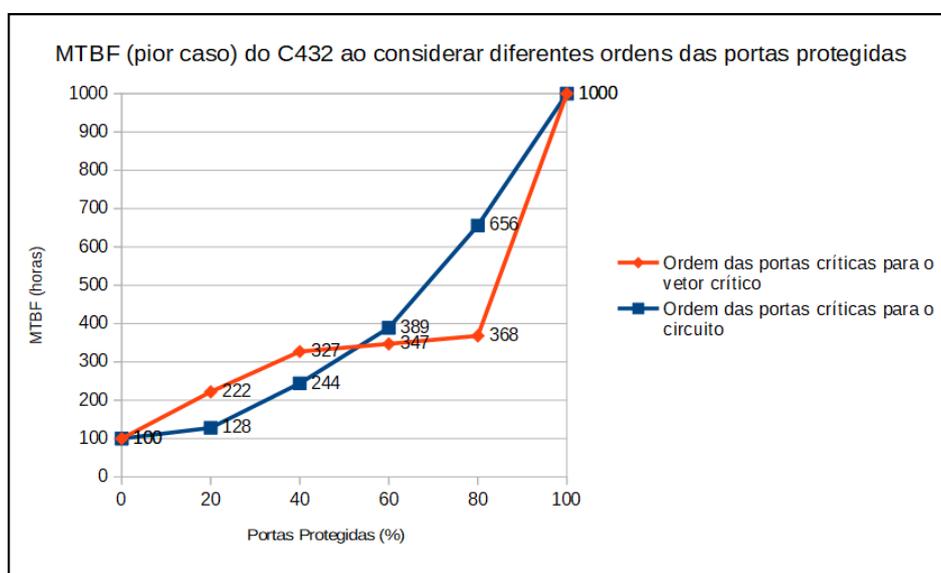


Figura 33: Análise do MTBF (pior caso) do circuito C432 ao proteger as portas lógicas considerando duas ordenações (ordem decrescente de R_{Med} e ordem decrescente de R_{Crit})

A Tabela 11 mostra uma análise do MTBF (pior caso) para todos os circuitos. As colunas *Ord1* representam o MTBF obtido com a proteção de portas com a *Ordem das portas críticas para o vetor crítico*. Já as colunas *Ord2*, apresentam o pior caso do MTBF, obtido com a proteção das portas com *Ordem das portas críticas para o circuito*. Com exceção do circuito c880, todos os outros circuitos mantiveram o comportamento parecido com c432, ou seja, após determinado percentual de proteção, a *Ordem das portas críticas para o vetor crítico* deixa de ser adequada.

A última análise desta Seção, apresenta o custo temporal para identificação do vetor crítico, conforme a Tabela 12. Este custo está relacionado, principalmente, com o número de portas lógicas e sinais de entrada do circuito. Por exemplo, os circuitos c880 e c1908 tem praticamente o mesmo número de portas, assim como os circuitos c1355 e c2670. Entretanto, a diferença entre os sinais de entrada é aproximadamente 2x para o primeiro caso e 5x para o segundo. Portanto, os circuitos com mais sinais, levaram mais tempo para identificar o vetor crítico. Já ao comparar os circuitos c499 e c1355, eles tem a mesma

Tabela 11: Análise MTBF (pior caso) ao proteger portas lógicas (ordem decrescente de R_{Crit} [Ord1] versus ordem decrescente de R_{Med} [Ord2])

Circuito	20% Portas		40% Portas		60% Portas		80% Portas	
	Ord1	Ord2	Ord1	Ord2	Ord1	Ord2	Ord1	Ord2
	MTBF (pior caso)							
c432	222	128	327	244	347	389	368	656
c499	639	501	869	682	1223	1236	1887	2062
c880	274	190	322	235	358	309	627	413
c1355	79	41	122	45	144	67	162	188
c1908	253	232	342	259	444	366	560	759
c2670	107	101	136	127	178	185	234	297
c3540	24	22	29	33	44	52	51	105
c5315	64	48	93	76	105	108	160	171

Tabela 12: Custo Temporal para identificar o vetor crítico

Circuito	Entradas	Portas	Tempo (s)
c432	36	136	1219
c499	41	188	329
c880	60	229	4781
c1908	33	234	1208
c1355	41	546	692
c2670	233	543	17217
c3540	50	673	5908
c5315	178	1026	47096

quantidade de sinais de entrada. Neste caso, o número maior de portas do circuito c1355, foi responsável pelo tempo maior. Além de portas e sinais de entrada, existem outros fatores que influenciam o custo temporal. Por exemplo, quando um vetor é gerado, ele efetua diversas consultas entre os vetores vizinhos. Se encontrar um vetor vizinho com melhor escore, este vetor é selecionado e novamente, são executadas mais consultas entre os vetores vizinhos deste novo vetor. Este processo se repete, até que não se encontre um vetor vizinho com melhor escore. Normalmente, um circuito com mais entradas, tende a executar mais consultas entre os vetores vizinhos. Entretanto, isto nem sempre acontece, pois, depende do grau de profundidade da busca. Quando a pesquisa não encontra melhor opção entre os vetores vizinhos, o tempo pode ser menor, quando comparado com uma busca, que está sempre encontrando um vetor melhor. Este comportamento pode ser observado, ao comparar os circuitos c2670 e c5315. Além do total de portas, o número de consultas entre os vetores vizinhos, é o que determina a diferença de tempo entre eles. Apesar do circuito c5315 possuir 55 sinais de entrada a menos, ele executou aproximadamente, 37% mais consultas que o c2670. Esta análise, também se aplica ao compararmos os circuitos c432 e c499. Apesar do c499 apresentar maior número de portas e sinais de entrada, ele executou com menor tempo que o c432. Como o c499 tem muitas portas *XOR*, a confiabilidade calculada para grande parte dos vetores de entrada é igual. Portanto, a pesquisa por melhor escore entre os vizinhos é mais rápida, pois em muitos casos, um vizinho com melhor escore não é encontrado.

4.3 Análise R_{Crit} ao considerar uma avaliação individual das saídas do circuito

As próximas análises consideram a divisão dos circuitos, a partir de suas saídas, de acordo com a metodologia definida na Seção 3.3. Primeiramente, uma tabela contendo as características dos sub-circuitos é apresentada, e na sequência, uma análise geral das diferenças do MTBF (caso médio e pior caso). Em seguida, a melhoria do MTBF (pior caso) é comparada com duas outras análises da seção anterior. No primeiro caso, ela é comparada com a análise que considera um único vetor crítico (circuito original sem divisão). No segundo, a comparação é com a melhoria do MTBF (pior caso) ao considerar a ordem decrescente de R_{Med} . Por fim, o custo temporal da análise individual das saídas é apresentado.

A divisão de todos os circuitos, resultou em 406 novos sub-circuitos. Foram descartados, os sub-circuitos com menos de 10 sinais de entrada ou portas lógicas. Então, um total de 193 sub-circuitos foram analisados. Em circuitos com menos de 10 sinais de entrada, o vetor crítico poderia ser encontrado rapidamente através de uma busca exaustiva. Já os circuitos com poucas portas lógicas, vão apresentar pequenas diferenças entre a confiabilidade média e a crítica. Portanto, eles não são o foco de nossa análise.

As características dos sub-circuitos podem ser vistas na Tabela 13. Além do total de sub-circuitos para cada circuito, as quantidades mínima, máxima e média dos sinais de entrada e portas lógicas por saída são apresentadas. Como mencionado anteriormente, a divisão dos circuitos vai acarretar mais sub-circuitos para análise. Entretanto, na maioria dos casos, os tamanhos dos mesmos são menores, tornando a identificação do vetor crítico mais eficiente. Por exemplo, o circuito c5315 em sua estrutura original, tem 178 sinais de entrada e 1026 portas. A partir da divisão, o sub-circuito com mais sinais de entrada tem 67 e com mais portas tem 373. O mesmo não acontece com o circuito c3540, que continua tendo pelo menos, um sub-circuito com a mesma quantidade de sinais de entrada e um sub-circuito com quase a mesma quantidade de portas.

Tabela 13: Estrutura dos Sub-Circuitos

Circuito	Sub-circuitos	Sinais de entrada por saída			Portas lógicas por saída		
		Menor	Maior	Média	Menor	Maior	Média
c432	7	18	36	32	18	121	86
c499	32	41	41	41	103	103	103
c880	9	29	45	37	39	107	76
c1355	32	41	41	41	322	322	322
c1908	25	32	33	32	96	173	118
c2670	21	10	119	39	15	267	85
c3540	18	19	50	38	20	630	321
c5315	49	15	67	39	26	373	100
Geral	193	10	119	37	15	630	151

Tabela 14: Diferença do MTBF (caso médio versus pior caso) dos sub-circuitos

Circuito	Dif. MTBF (médio versus pior) (x)		
	Menor	Maior	Média
c432	4,41	12,46	9,31
c499	33,00	33,00	33,00
c880	6,35	12,43	9,20
c1355	276,00	276,00	276,00
c1908	3,99	51,30	33,95
c2670	1,40	188,67	20,34
c3540	1,72	419,52	47,72
c5315	1,39	126,20	13,71
Geral	1,39	419,52	55,40

A Tabela 14, apresenta as diferenças (máxima, mínima e média) do MTBF (caso médio e pior caso), considerando a análise individual das saídas. Quando comparado com a análise geral do circuito, que foi mostrada na Tabela 8, na qual apenas um vetor

crítico é considerado, podemos perceber que as diferenças aumentaram. Quando uma análise completa do circuito é realizada, é pouco provável que um único vetor, vai ser crítico para todas as saídas do circuito. Portanto, uma saída compensa outra e neste caso, as diferenças são menores. Isto fica evidente ao olharmos para o circuito c499, que é um circuito regular em que todas as suas 32 saídas estão conectadas a todos os 41 sinais de entrada. Como vimos, com a análise geral, a diferença entre o MTBF (caso médio e pior caso) foi 1,89x. Entretanto, em uma análise das saídas, a diferença é 33x para qualquer saída. Ao analisar as diferenças máximas, percebe-se que nos maiores circuitos, elas são mais expressivas, como é o caso do c2670, c3540 e c5315. Com esta análise, técnicas para melhorar a confiabilidade especificamente destas saídas, poderiam ser aplicadas.

A próxima análise considera a proteção das portas críticas dos sub-circuitos. A Tabela 15 apresenta os valores MTBF (pior caso). Neste caso, as portas foram identificadas, considerando o vetor crítico de cada sub-circuito (análise individual das saídas). Além disso, é possível visualizar o MTBF (pior caso) que foi apresentado na Tabela 11, cujas portas críticas foram identificadas, ao considerar um único vetor crítico (análise do circuito original). Em todos os casos, o MTBF foi maior ao considerar a análise individual das saídas. A maior e a menor diferença entre ambos, podem ser vistas na Figura 34.

Tabela 15: Análise MTBF (pior caso) ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem considerando um único vetor crítico do circuito original)

Circuito	Análise individual das saídas				Análise do circuito original			
	MTBF (pior caso)				MTBF (pior caso)			
	20%	40%	60%	80%	20%	40%	60%	80%
c432	258	485	698	895	222	327	347	368
c499	654	935	1283	2103	639	869	1223	1887
c880	328	556	744	901	274	322	358	627
c1355	97	148	242	302	79	122	144	162
c1908	318	425	593	1133	253	342	444	560
c2670	114	177	262	399	107	136	178	234
c3540	29	52	78	101	24	29	44	51
c5315	81	113	133	183	64	93	105	160

O circuito c499 apresentou a menor diferença, e os motivos já foram explicados nas seções anteriores. O circuito c432 apresentou a maior diferença, chegando próximo a 2.5x ao proteger 80% das portas, e isto pode estar associado ao fato do mesmo, ter a menor quantidade de saídas. Esta análise permite concluir que, proteger as portas lógicas, que são críticas para os vetores críticos de cada sub-circuito (saída), produz melhor efeito, quando comparado com a proteção de portas críticas para um único vetor crítico, em uma análise do circuito completo. Como mencionado anteriormente, em uma análise

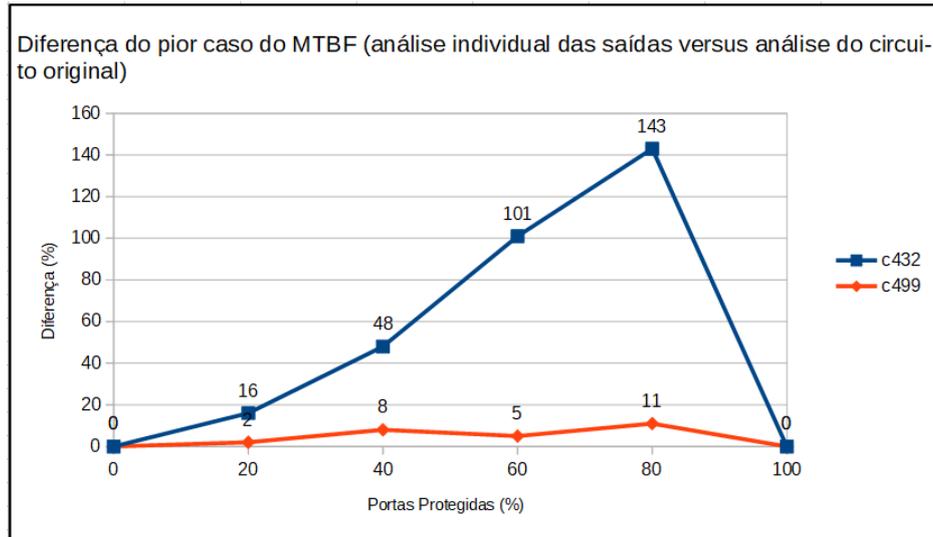


Figura 34: Diferença do MTBF (pior caso) dos circuitos C432 e c499 ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem considerando um único vetor crítico do circuito)

do circuito inteiro, um único vetor crítico não maximiza as probabilidades de falha de todas as saídas. Isto ocorre, porque alguns sinais de entrada são compartilhados por mais de uma saída e em alguns casos, vão estar em conflito entre estas. Portanto, a análise que considera as saídas individualmente, é mais adequada para melhorar a confiabilidade crítica.

A Figura 35 mostra os circuitos com a maior e a menor diferença do MTBF (pior caso) considerando a análise individual das saídas, que já foi apresentada na Tabela 15, e o MTBF (pior caso) obtido ao considerar a ordem decrescente de R_{Med} , e que foi apresentado na Tabela 9. Utilizar a ordem decrescente de R_{Med} , é a melhor estratégia para melhoria da confiabilidade média. Entretanto, não é a melhor opção para melhoria da confiabilidade crítica. É possível obter ganho superior a 3x no melhor caso, ao considerar a análise individual das saídas, como é mostrado no gráfico para o circuito c1355.

Nossa última análise considera o custo temporal, e pode ser vista na Tabela 16. Além do custo para análise individual das saídas, também é apresentado o custo ao considerar o circuito original (que já foi apresentado na seção anterior) e a diferença entre os mesmos. As diferenças podem ser melhor compreendidas, ao analisar as características dos circuitos e sub-circuitos. Como já foi mencionado anteriormente, o custo para identificação do vetor crítico, tem relação principalmente, com a quantidade de sinais de entrada e portas lógicas. Por exemplo, os 2 piores casos, são o c499 e o c1355. Estes circuitos foram particionados em 32 sub-circuitos, e todos continuaram com a mesma quantidade de sinais de entrada do circuito original. Houve uma redução do tempo de cada sub-circuito, devido à redução do número de portas. Entretanto, a soma dos tempos de todos os sub-circuitos, é maior que o tempo do circuito original. Já os circuitos c2670 e c5315, representam os me-

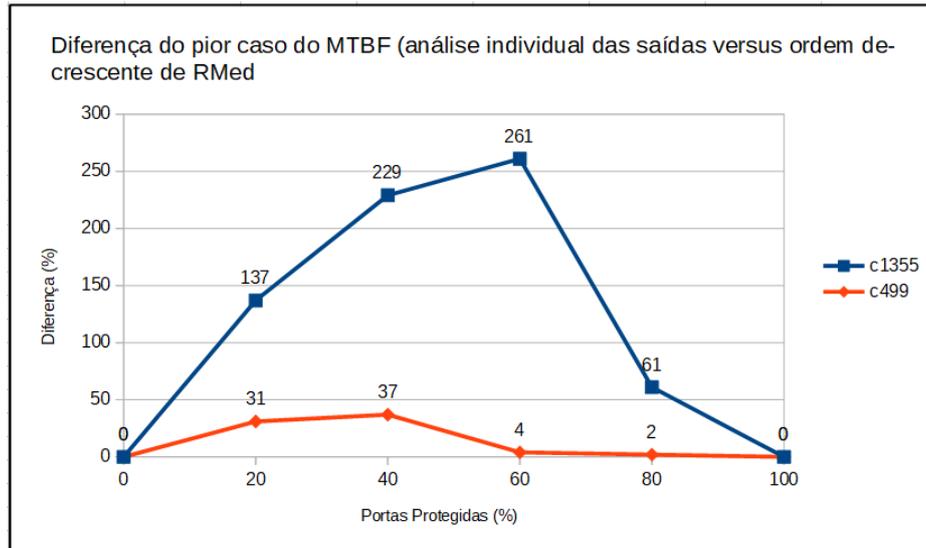


Figura 35: Diferença do MTBF (pior caso) dos circuitos C1355 e c499 ao proteger as portas lógicas (ordem considerando os vetores críticos dos sub-circuitos versus ordem decrescente de R_{Med})

lhores casos. Seus tempos foram reduzidos em 7% e 35%, respectivamente. Para o c2670, o sub-circuito com mais sinais de entrada tem 119, e o que tem mais portas lógicas possui 267. Isto, representa aproximadamente a metade de entradas e portas do circuito original. Com relação ao c5315, apesar de haver mais sub-circuitos analisados, as diferenças são ainda maiores. Houve uma redução de 2.65x nos sinais de entrada e 2.75x na quantidade de portas lógicas.

Tabela 16: Comparação do Custo Temporal (análise circuito original versus análise sub-circuitos)

Circuito	Circuito Original Tempo (s)	Sub-circuitos Tempo (s)	Dif. (x)
c432	1219	6887	5,65
c499	329	4645	14,12
c880	4781	8318	1,74
c1908	1208	5073	4,20
c1355	692	9252	13,37
c2670	17217	16011	0,93
c3540	5908	48977	8,29
c5315	47096	30612	0,65

5 CONSIDERAÇÕES FINAIS

O avanço da tecnologia CMOS, que permite a fabricação de dispositivos semicondutores com dimensões nanométricas, tem contribuído para a melhoria de produtos e serviços das aplicações. Entretanto, tais dispositivos ficaram mais suscetíveis à ocorrência de falhas, devido à interferência externa ou de radiação. Portanto, a confiabilidade passou a ser um importante parâmetro do projeto de sistemas digitais, que não pode ser desprezado. Além disso, é preciso garantir que seu limite inferior esteja acima de valores aceitáveis. Conforme foi apresentado nos resultados, as diferenças entre a confiabilidade média e a confiabilidade crítica é bem expressiva. Nos piores casos, ela é 12 vezes ao considerar uma análise do circuito original, e 419 vezes em uma análise individual das saídas. Quando os valores de confiabilidade, estão abaixo do esperado, é necessário o uso de técnicas para melhorar a confiabilidade. Porém, é preciso garantir também, que outros parâmetros de projeto não sejam tão sobrecarregados. Uma forma de atender esses requisitos, é identificar as portas do circuito, com base na sua criticalidade, e então, melhorar a confiabilidade somente das portas mais críticas.

Neste sentido, a contribuição deste trabalho, foi efetuar uma análise da confiabilidade ao proteger seletivamente as portas lógicas do circuito. Para tanto, foi necessário identificar as portas críticas e vetores críticos dos circuitos. Para identificação dos vetores críticos, um algoritmo baseado em heurística foi desenvolvido. A primeira parte do trabalho, considerou apenas a confiabilidade média. O objetivo foi demonstrar que é possível, rapidamente, melhorar a confiabilidade do circuito ao determinar as portas críticas do mesmo, e que quaisquer outras escolhas de portas, não produzirá melhores resultados. Conforme foi apresentado na análise, a maior diferença é superior a 6 vezes e tem relação com as estruturas dos circuitos. Entretanto, a confiabilidade média, não diz nada sobre o pior caso de confiabilidade. E quando falamos em determinar a confiabilidade crítica, ou encontrar o vetor crítico, não devemos imaginar que exista apenas um vetor crítico. Existem muitos vetores críticos, ou vetores cuja confiabilidade associada se aproxima muito da confiabilidade crítica, e que, ao melhorá-la, se melhora a confiabilidade associada a todos esses vetores.

A melhor opção para aumentar a confiabilidade média, demonstrada na análise ante-

rior, não melhora na mesma proporção a confiabilidade crítica. Por exemplo, a diferença do MTBF (caso médio e pior caso) que era 11 vezes, para o circuito c1355 sem proteção, passou para 33 vezes, com a proteção de 40% das portas. Já a proteção das portas críticas para o vetor crítico, até determinado percentual de proteção, apresenta um resultado melhor para a confiabilidade crítica. Entretanto, o tempo para determinar o vetor crítico, é bem maior, quando comparado com o tempo da análise anterior.

Por fim, em uma análise do circuito inteiro, os sinais de entrada são compartilhados por muitas saídas e pode haver conflito entre estas. Neste caso, a determinação da criticidade das portas fica prejudicada. É por este motivo, que a análise individual das saídas, permite determinar as portas mais adequadas a serem protegidas, apresentando diferenças superiores a 100%. Ao comparar com a estratégia da primeira análise (ordem decrescente R_{Med}), as diferenças são maiores que 3x. Quanto ao tempo para uma análise individual das saídas, este tende a ser menor, em circuitos com mais entradas. Por exemplo, é o caso c2670 e do c5315, cujos sub-circuitos com mais entradas tem a metade ou menos entradas que o circuito original, além da redução do número de portas lógicas.

Existem algumas hipóteses para melhorar a complexidade temporal e que não foram exploradas neste trabalho. Por exemplo, não é necessário analisar todas as saídas do circuito. Em alguns casos, as portas críticas são as mesmas para mais de uma saída. Com relação ao algoritmo para identificação do vetor crítico, é possível introduzir otimizações, por exemplo, com processamento paralelo para determinar os escores dos vetores. Além disso, explorar o método SPR na tentativa de melhorar sua eficiência, ou até mesmo, utilizar outro método mais rápido, para o cálculo dos escores.

REFERÊNCIAS

Anghel, L.; Alexandrescu, D.; Nicolaidis, M. Evaluation of a soft error tolerance technique based on time and/or space redundancy. In: SYMPOSIUM ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN (CAT. NO.PR00843), 13., 2000. **Proceedings...** [S.l.: s.n.], 2000. p.237–242.

AVIZIENIS, A.; LAPRIE, J.-C.; RANDELL, B.; LANDWEHR, C. Basic concepts and taxonomy of dependable and secure computing. **IEEE transactions on dependable and secure computing**, [S.l.], v.1, n.1, p.11–33, 2004.

BAN, T.; JUNIOR, G. G. Critical gates identification for fault-tolerant design in math circuits. **Journal of Electrical and Computer Engineering**, [S.l.], v.2017, 2017.

BERKELEY, A. **A system for sequential synthesis and verification.**

BRGLEZ, F.; FUJIWARA, H. A neutral netlist of 10 combinational benchmark circuits and a target translator in Fortran. **IEEE International Symposium on Circuits and Systems (ISCAS-85)**, [S.l.], n.June, p.663–698, 1985.

CONSTANTINESCU, C. Trends and challenges in VLSI circuit reliability. **IEEE micro**, [S.l.], v.23, n.4, p.14–19, 2003.

FRANCO, D. T.; NAVINER, J.-F.; NAVINER, L. Yield and reliability issues in nanoelectronic technologies. In: ANNALES DES TÉLÉCOMMUNICATIONS, 2006. ... [S.l.: s.n.], 2006. v.61, n.11-12, p.1422–1457.

FRANCO, D. T.; VASCONCELOS, M. C.; NAVINER, L.; NAVINER, J.-F. Reliability analysis of logic circuits based on signal probability. In: ELECTRONICS, CIRCUITS AND SYSTEMS, 2008. ICECS 2008. 15TH IEEE INTERNATIONAL CONFERENCE ON, 2008. ... [S.l.: s.n.], 2008. p.670–673.

HAN, J.; CHEN, H.; BOYKIN, E.; FORTES, J. Reliability evaluation of logic circuits using probabilistic gate models. **Microelectronics Reliability**, [S.l.], v.51, n.2, p.468–476, 2011.

HAN, J.; CHEN, H.; LIANG, J.; ZHU, P.; YANG, Z.; LOMBARDI, F. A Stochastic Computational Approach for Accurate and Efficient Reliability Evaluation. **IEEE Transactions on Computers**, [S.l.], n.1, p.1, 2012.

IBRAHIM, W. Identifying the worst reliability input vectors and the associated critical logic gates. **IEEE Transactions on Computers**, [S.l.], v.65, n.6, p.1748–1760, 2015.

IBRAHIM, W.; AMER, H. Critical Nodes Count Algorithm for Accurate Input Vectors Reliability Ranking. In: SUMMER COMPUTER SIMULATION CONFERENCE, 2016, San Diego, CA, USA. **Proceedings...** Society for Computer Simulation International, 2016. p.19:1–19:7. (SCSC '16).

IBRAHIM, W.; BEIU, V.; AMER, H. How much input vectors affect nano-circuit's reliability estimates. In: SOCIETY FOR COMPUTER SIMULATION INTERNATIONAL, 2009. ... [S.l.: s.n.], 2009. p.699 – 702.

IBRAHIM, W.; SHOUSHA, M.; CHINNECK, J. W. Accurate and efficient estimation of logic circuits reliability bounds. **IEEE Transactions on Computers**, [S.l.], v.64, n.5, p.1217–1229, 2014.

KAPUR, K.; PECHT, M. **Reliability Engineering**. [S.l.]: Wiley, 2014. (Wiley Series in Systems Engineering and Management).

KRISHNASWAMY, S.; VIAMONTES, G. F.; MARKOV, I. L.; HAYES, J. P. Accurate reliability evaluation and enhancement via probabilistic transfer matrices. In: DESIGN, AUTOMATION AND TEST IN EUROPE-VOLUME 1, 2005. **Proceedings...** [S.l.: s.n.], 2005. p.282–287.

MARQUES, E. C.; BARROS NAVINER, L. A. de; NAVINER, J.-F. An efficient tool for reliability improvement based on TMR. **Microelectronics Reliability**, [S.l.], v.50, n.9-11, p.1247–1250, 2010.

MAXION, R. A.; SIEWIOREK, D. P.; ELKIND, S. A. Techniques and architectures for fault-tolerant computing. **Annual Review of Computer Science**, [S.l.], v.2, n.1, p.469–520, 1987.

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

NAVINER, L. A. d. B.; NAVINER, J.-F.; BAN, T.; GUTEMBERG, G. Reliability analysis based on significance. In: ARGENTINE SCHOOL OF MICRO-NANOELECTRONICS, TECHNOLOGY AND APPLICATIONS, 2011., 2011. ... [S.l.: s.n.], 2011. p.1–7.

NICOLAIDIS, M. Design for soft error mitigation. **IEEE Transactions on Device and Materials Reliability**, [S.l.], v.5, n.3, p.405–418, 2005.

NIEUWLAND, A. K.; JASAREVIC, S.; JERIN, G. Combinational logic soft error analysis and protection. In: IEEE INTERNATIONAL ON-LINE TESTING SYMPOSIUM (IOLTS'06), 12., 2006. ... [S.l.: s.n.], 2006. p.6–pp.

PAGLIARINI, S. N.; BAN, T.; NAVINER, L. A. d. B.; NAVINER, J.-F. Reliability assessment of combinational logic using first-order-only fanout reconvergence analysis. In: IEEE 56TH INTERNATIONAL MIDWEST SYMPOSIUM ON CIRCUITS AND SYSTEMS (MWSCAS), 2013., 2013. ... [S.l.: s.n.], 2013. p.113–116.

PAGLIARINI, S. N.; NAVINER, L. A. d. B.; NAVINER, J.-F. Selective hardening methodology for combinational logic. In: AMERICAN TEST WORKSHOP (LATW), 2012., 2012. ... [S.l.: s.n.], 2012. p.1–6.

PATEL, K. N.; MARKOV, I. L.; HAYES, J. P. Evaluating circuit reliability under probabilistic gate-level fault models. In: INTERNATIONAL WORKSHOP ON LOGIC AND SYNTHESIS, 2003. **Proceedings...** [S.l.: s.n.], 2003. p.59–64.

POLIAN, I.; HAYES, J. P. Selective hardening: Toward cost-effective error tolerance. **IEEE Design & Test of Computers**, [S.l.], v.28, n.3, p.54–63, 2010.

PONTES, M. F. **Análise dos Métodos PTM e SPR para avaliação de confiabilidade de circuitos combinacionais**. 2019. Dissertação (Mestrado em Ciência da Computação) — Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande. (99 f).

PONTES, M. F.; BUTZEN, P. F.; SCHVITZ, R. B.; ROSA, S. L.; FRANCO, D. T. The Suitability of the SPR-MP Method to Evaluate the Reliability of Logic Circuits. In: IEEE INTERNATIONAL CONFERENCE ON ELECTRONICS, CIRCUITS AND SYSTEMS (ICECS), 2018., 2018. ... [S.l.: s.n.], 2018. p.433–436.

Quming Zhou; Mohanram, K. Gate sizing to radiation harden combinational logic. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, [S.l.], v.25, n.1, p.155–166, Jan 2006.

Raji, M.; Sabet, M. A.; Ghavami, B. Soft Error Reliability Improvement of Digital Circuits by Exploiting a Fast Gate Sizing Scheme. **IEEE Access**, [S.l.], v.7, p.66485–66495, 2019.

TEIXEIRA FRANCO, D. **Fiabilité du signal des circuits logiques combinatoires sous fautes simultanées multiples**. 2008. Theses — Télécom ParisTech.

WALI, I.; DEVEAUTOUR, B.; VIRAZEL, A.; BOSIO, A.; GIRARD, P.; REORDA, M. S. A Low-Cost Reliability vs. Cost Trade-Off Methodology to Selectively Harden Logic Circuits. **Journal of Electronic Testing**, [S.l.], v.33, n.1, p.25–36, 2017.

XIAO, J.; JIANG, J.; ZHU, X.; OUYANG, C. A method of gate-level circuit reliability estimation based on iterative PTM model. In: **DEPENDABLE COMPUTING (PRDC), 2011 IEEE 17TH PACIFIC RIM INTERNATIONAL SYMPOSIUM ON, 2011. ...** [S.l.: s.n.], 2011. p.276–277.

XIAO, J.; LOU, J.; JIANG, J. A Fast and Effective Sensitivity Calculation Method for Circuit Input Vectors. **IEEE Transactions on Reliability**, [S.l.], 2019.

XIAO, R.; CHEN, C. Gate-level circuit reliability analysis: A survey. **VLSI Design**, [S.l.], v.2014, p.4, 2014.

YEAP, K. H.; NISAR, H. **Very-Large-Scale Integration**. [S.l.]: BoD–Books on Demand, 2018.

APÊNDICE A PSEUDOCÓDIGO DO EXPERIMENTO 1

```
1: declare  $q$  = // confiabilidade da porta lógica
2: declare  $gl$  = // lista de portas do circuito
3: declare  $rl$  = // lista das confiabilidades do circuito ao melhorar o  $q$  de uma porta
4: declare  $R$  = // lista das confiabilidades do circuito ao melhorar o  $q$  de x% de portas
5: inicialize  $q \leftarrow 0,99999$  para todas as portas do circuito
6: for  $i = 1, 2, \dots, gl$  do
7:    $q_i \leftarrow 0,999999$ 
8:    $rl_i \leftarrow$  confiabilidade do circuito com o SPR
9:    $q_i \leftarrow 0,99999$ 
10: end for
11:  $lista\_ordenada \leftarrow$  lista de portas em ordem descendente de  $rl$ 
12: for  $i = 1, 2, \dots, lista\_ordenada$  do
13:    $q_i \leftarrow 0,999999$ 
14:    $R_i \leftarrow$  confiabilidade do circuito com o SPR
15: end for
16: return  $R$ 
```

APÊNDICE B PSEUDOCÓDIGO DO ALGORITMO PARA IDENTIFICAÇÃO DO VETOR CRÍTICO

```
1: declare tamanho_entrada = // número de sinais de entrada do circuito
2: declare tamanho_lista_vet = 10 // total de vetores candidatos
3: declare lista_vet = // lista de vetores candidatos
4: declare n = 1000 // total de vetores aleatórios
5: declare limite_bits = 17 // limite de bits para pesquisa exaustiva
6: declare bits_determinados = // armazena o valor e a posição do bit
7: declare bits_indeterminados = tamanho_entrada // bits não determinados
8: declare wrv = // pior vetor de confiabilidade
9: declare limite_consensos = 0.9 // limite de consenso inicial
10: declare limite_consensos_min = 0.7 // limite de consenso mínimo
11: declare loop_sem_alteracao = 0 // número de iterações sem descoberta de bits quando
    o consenso for igual consensos_min
12: declare max_consultas = 0.2 // percentual máximo de consultas de vetores
13: leia o arquivo verilog do circuito
14: while bits_indeterminados > limite_bits e loop_sem_alteracao < 5 do
15:     for i = 1, 2, ..., n do
16:         for j = 1, 2, ..., tamanho_entrada do
17:             if j ∈ bits_determinados then
18:                 vetor[j] ← bits_determinados[j]
19:             else
20:                 define vetor[j] aleatoriamente para 0 ou 1
21:             end if
22:         end for
23:         calcula o escore do vetor com o SPR
24:         melhora vetor pesquisando vetores vizinhos
25:         if escore_vetor > menor_escore_lista_vet then
26:             adiciona vetor na lista_vet substituindo
27:             o de menor escore
```

```

28:     end if
29: end for
30: executa o consenso para lista_vet
31: atualiza bits_determinados
32: if bits_determinados nao muda then
33:     if limite_consenso > limite_consenso_min then
34:         limite_consenso ← limite_consenso – 0.01
35:         if limite_consenso ≤ 0.82 e limite_consenso > 0.75 then
36:             tamanho lista_vet ← 20
37:         end if
38:         if limite_consenso ≤ 0.75 then
39:             tamanho lista_vet ← 30
40:         end if
41:     else
42:         incrementa loop_sem_alteracao
43:     end if
44: end if
45: if bits_determinados ≥ 8 ou limite_consenso ≤ 0.88 then
46:     n ← 10 × bits_indeterminados
47: end if
48: end while
49: wrv ← maior escore lista_vet
50: if bits_indeterminados < limite_bits then
51:     faz busca sequencial até  $2^{\text{bits\_indeterminados}}$ 
52:     atualiza wrv caso encontre um vetor com melhor escore
53: else
54:     faz busca sequencial até  $2^{\text{limite\_bits}}$ 
55:     atualiza wrv caso encontre um vetor com melhor escore
56: end if
57: return wrv

```

APÊNDICE C BIBLIOTECA PARA MAPEAMENTO DOS CIRCUITOS

```

# 180 nm Generic Library
# Download from http://create.cadence.com
# Copyright 2003, Cadence Design Systems - All Rights Reserved
# (Single-output logic gates converted to GENLIB by Alan Mishchenko.)
GATE ZERO 1 Y=CONST0;
GATE ONE 1 Y=CONST1;
GATE INVX1 1 Y=!A; PIN * INV 1 999 1 0 1 0
GATE INVX2 1 Y=!A; PIN * INV 1 999 1 0 1 0
GATE INVX4 1 Y=!A; PIN * INV 1 999 1 0 1 0
GATE INVX8 1 Y=!A; PIN * INV 1 999 1 0 1 0
GATE BUF3 1 Y=A; PIN * NONINV 1 999 1 0 1 0
GATE BUF3 1 Y=A; PIN * NONINV 1 999 1 0 1 0
GATE CLKBUF3 1 Y=A; PIN * NONINV 1 999 1 0 1 0
GATE CLKBUF3 1 Y=A; PIN * NONINV 1 999 1 0 1 0
GATE NOR2X1 1 Y=!(A+B); PIN * INV 1 999 1 0 1 0
GATE NOR3X1 1 Y=!(A+B+C); PIN * INV 1 999 1 0 1 0
GATE NOR4X1 1 Y=!(A+B+C+D); PIN * INV 1 999 1 0 1 0
GATE NAND2X1 1 Y=!(A*B); PIN * INV 1 999 1 0 1 0
GATE NAND2X2 1 Y=!(A*B); PIN * INV 1 999 1 0 1 0
GATE NAND3X1 1 Y=!(A*B*C); PIN * INV 1 999 1 0 1 0
GATE NAND4X1 1 Y=!(A*B*C*D); PIN * INV 1 999 1 0 1 0
GATE OR2X1 1 Y=A+B; PIN * NONINV 1 999 1 0 1 0
GATE OR4X1 1 Y=A+B+C+D; PIN * NONINV 1 999 1 0 1 0
GATE AND2X1 1 Y=A*B; PIN * NONINV 1 999 1 0 1 0
GATE OAI21X1 1 Y=!((A0+A1)*B0); PIN * INV 1 999 1 0 1 0
GATE OAI22X1 1 Y=!((A0+A1)*(B0+B1)); PIN * INV 1 999 1 0 1 0
GATE OAI33X1 1 Y=!((A0+A1+A2)*(B0+B1+B2)); PIN * INV 1 999 1 0 1 0
GATE AOI21X1 1 Y=!(A0*A1+B0); PIN * INV 1 999 1 0 1 0
GATE AOI22X1 1 Y=!(A0*A1+B0*B1); PIN * INV 1 999 1 0 1 0
GATE MX2X1 1 Y=(A*B)+(S0*B)+(!S0*A); PIN * UNKNOWN 1 999 1 0 1 0
GATE XOR2X1 1 Y=(A*B)+(!A*B); PIN * UNKNOWN 1 999 1 0 1 0

```

Figura 36: Biblioteca para mapeamento dos circuitos