

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

Dissertação de Mestrado

**Análise dos métodos PTM e SPR para a avaliação de
confiabilidade de circuitos combinacionais**

Matheus Ferreira Pontes

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

Rio Grande, 2019

Ficha catalográfica

P814a Pontes, Matheus Ferreira.

Análise dos métodos PTM e SPR para a avaliação de confiabilidade de circuitos combinacionais / Matheus Ferreira Pontes. – 2019.

99 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2019.

Orientador: Dr. Denis Teixeira Franco.

Coorientador: Dr. Paulo Francisco Butzen.

1. Circuitos Digitais 2. Confiabilidade 3. PTM 4. SPR 5. SPR-MP
I. Franco, Denis Teixeira II. Butzen, Paulo Francisco III. Título.

CDU 004.312.2

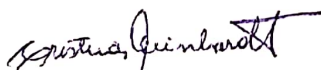
ATA DE SESSÃO DE DEFESA DE DISSERTAÇÃO DE MESTRADO

Ata nº 01 /2019

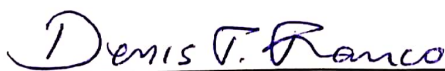
Na data de 06 de março de 2019, às 14 horas, ocorreu a Sessão de Defesa de Dissertação de Mestrado de Matheus Ferreira Pontes, que apresentou a dissertação intitulada Análise Dos Métodos PTM e SPR para Avaliação de Confiabilidade de Circuitos Combinacionais, realizada sob a orientação do Prof. Dr. Denis Teixeira Franco e coorientação do Prof. Dr. Paulo Francisco Butzen. A banca examinadora foi constituída pela Profa. Dra. Cristina Meinhardt (UFSC) e pelo Prof. Leomar Soares da Rosa Jr. (UFPEL), sob a presidência do orientador. Após a apresentação do trabalho, a banca arguiu o candidato e, a seguir, deliberou pela

- aprovação da Dissertação
- aprovação da Dissertação, sugerindo modificações no texto
- reprovação da Dissertação

Rio Grande, 06 de março de 2019



Profa. Dra. Cristina Meinhardt

Prof. Dr. Leomar Soares da Rosa Jr.

Prof. Dr. Denis Teixeira Franco
Orientador

Prof. Dr. Paulo Francisco Butzen
Coorientador

AGRADECIMENTOS

Agradeço a minha família pelo apoio incondicional na minha caminhada acadêmica. Agradeço também aos meus orientadores pela paciência e compreensão em vários momentos.

*O valor das metas futuras não
reside na imagem do futuro que
se cria na mente, mas sim na mudança
que elas provocam no presente!*

— DAVID ALLEN

RESUMO

PONTES, Matheus Ferreira. **Análise dos métodos PTM e SPR para a avaliação de confiabilidade de circuitos combinacionais**. 2019. 99 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

A confiabilidade de um circuito é uma informação importante principalmente quando o mesmo é projetado em tecnologias nanométricas. Este trabalho traz implementações de três métodos que estimam a confiabilidade de circuitos digitais: Matrizes de Transferência Probabilística (PTM), Análise de Confiabilidade pela Probabilidade de Sinais (SPR) e uma variação do SPR denominada SPR-Multipass (SPR-MP). São exploradas as vantagens e desvantagens de cada um, visando a aplicabilidade no processo de projeto de circuitos. A PTM já foi amplamente discutida na literatura, sendo considerada uma metodologia exata para estimar a confiabilidade de um circuito, apresentando, entretanto, restrições de escalabilidade. No presente trabalho, foi desenvolvida uma implementação alternativa, que sequencializa os cálculos envolvendo matrizes. Outro método que será abordado é o SPR, o qual não possui problemas com escalabilidade, pois sua complexidade é linear em relação ao número de portas. Porém, por não tratar os *fanouts* reconvergentes, os valores de confiabilidade encontrados com o SPR não são precisos. O SPR-MP é uma variação do SPR focada em lidar com a reconvergência de sinais e estimar a confiabilidade de um circuito de forma exata. O tempo de processamento do SPR-MP está diretamente relacionado à quantidade de *fanouts* do circuito, o que também pode inviabilizar a utilização do método para circuitos maiores. Assim, neste trabalho foram realizadas comparações entre os métodos, a fim de verificar as vantagens e desvantagens de cada um. O método PTM se mostrou impraticável para estimar a confiabilidade de circuitos com mais de duas dezenas de portas. Tendo em vista que o SPR-MP também gera resultados exatos de confiabilidade, quando todos os *fanouts* são considerados, a amostragem de circuitos pôde ser ampliada em termos de tamanho de circuito. Realizando comparações entre os métodos SPR e SPR-MP foi possível identificar certas tendências nos resultados obtidos pelo método SPR. Em termos numéricos, a diferença entre os valores de confiabilidade gerados pelo SPR e o valor exato é na sua grande maioria menor que 10%. Além disso, a redução no tempo de processamento, quando usado o SPR, pode chegar até 10^8 vezes o tempo necessário para extrair um valor exato. Estas características fazem com que o SPR seja uma excelente alternativa para uma rápida estimativa da confiabilidade de um circuito.

Palavras-chave: Circuitos Digitais, Confiabilidade, PTM, SPR, SPR-MP.

ABSTRACT

PONTES, Matheus Ferreira. **Combinational Circuit Reliability Analysis Using PTM and SPR**. 2019. 99 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

The reliability of a circuit is an important information mainly when it is designed in nanometric technologies. This work presents implementations of three methods that estimate the reliability of digital circuits: Probabilistic Transfer Matrices (PTM), Signal Probability Reliability Analysis (SPR) and a SPR variation called SPR-Multipass (SPR-MP). The advantages and disadvantages of each one will be explored, aiming at the applicability in the circuit design process. PTM has already been widely discussed in the literature, being considered an exact methodology to estimate the reliability of a circuit, presenting, however, scalability constraints. In the present work, an alternative implementation was developed that sequentially calculates matrices. Another method that will be presented is the SPR, which has no problems with scalability, because its complexity is linear in relation to the number of gates. However, since it does not handle reconvergent fanouts, the encountered reliabilities values with the SPR are not accurate. The SPR-MP is a variation of the SPR focused on dealing with the reconvergence of signals and accurately estimating the reliability of a circuit. The processing time of the SPR-MP is directly related to the amount of fanouts of the circuit, which may also make it unfeasible to use the method for larger circuits. Thus, in this work comparisons were made between the methods to verify the advantages and disadvantages of each one. The PTM method is impractical to estimate the reliability of circuits with more than two dozen ports. Since SPR-MP also generates accurate reliability results, when all fanouts are considered, circuit sampling could be expanded in terms of size. By making comparisons between the SPR and SPR-MP methods it was possible to identify certain trends in the results obtained by the SPR method. In numerical terms, the difference between the reliability values generated by the SPR and the exact value is, for the most part, less than 10 %. In addition, the reduction in processing time, when used the SPR, can reach up to 10^8 times the time needed to extract an exact value. These characteristics make the SPR an excellent alternative for a rapid estimation of the reliability of a circuit.

Keywords: Digital Circuits, Reliability, PTM, SPR, SPR-MP.

LISTA DE FIGURAS

Figura 1	Taxonomia de confiabilidade	13
Figura 2	Efeito dos mascaramentos Elétrico, Temporal e Lógico	14
Figura 3	Diagrama da Curva da Banheira	19
Figura 4	Relação entre Tabela Verdade e Matriz PTM	24
Figura 5	Exemplos de arranjos de fios condutores	24
Figura 6	Formando a matriz PTM de um circuito	26
Figura 7	Relação PTM e ITM	27
Figura 8	Matriz de probabilidade de sinal	28
Figura 9	Passos para estimar a confiabilidade do sinal de saída da <i>AND</i>	29
Figura 10	O problema do SPR	30
Figura 11	Exemplo de aplicação do algoritmo SPR-MP	31
Figura 12	Diagrama de classes da estrutura básica	32
Figura 13	Diagrama de classes da estrutura especializada	33
Figura 14	Ferramenta desenvolvida	34
Figura 15	Pseudo-código da PTM tradicional	37
Figura 16	Demonstração do Tensor de Kronecker	38
Figura 17	Processo Inverso da Aplicação do Tensor de Kronecker	38
Figura 18	Formação do elemento 5×4 da matriz resultante	39
Figura 19	Formação da Matriz PTM de um Circuito	40
Figura 20	Pseudo-código do Método SPR	41
Figura 21	Pseudo-código do Método SPR-MP	42
Figura 22	Convertendo um circuito sequencial em combinacional conforme a proposta de (CZUTRO, 2013)	51
Figura 23	Percentual de incremento no MTBF do circuito, melhores casos	60
Figura 24	Percentual de incremento no MTBF do circuito, piores casos	60
Figura 25	C17 - Versão 1	67
Figura 26	C17 - Versão 2	67
Figura 27	C17 - Versão 3	68
Figura 28	C17 - Versão 4	68
Figura 29	Multiplexador 4-bits	68
Figura 30	Somador Completo - Versão 1	69
Figura 31	Somador Completo - Versão 2	69
Figura 32	Somador Completo - Versão 3	69
Figura 33	C8	70

Figura 34	C9	70
Figura 35	C10	70
Figura 36	C11	71
Figura 37	C20	71
Figura 38	Descrição do s27	72
Figura 39	Descrição do s208	73
Figura 40	Descrição do s298 (saída n64)	73
Figura 41	Descrição do s344 (saída n61)	74
Figura 42	Descrição do s349 (saída n66)	74
Figura 43	Descrição do s382 (saída n69)	75
Figura 44	Descrição do s386 (saída n49)	75
Figura 45	Descrição do s400 (saída n64)	76
Figura 46	Descrição do s420 (saída Z)	77
Figura 47	Descrição do s444 (saída n109)	77
Figura 48	Descrição do s510 (saída n78)	78
Figura 49	Descrição do s641 (saída n178)	79
Figura 50	Descrição do s713 (saída n177)	80
Figura 51	Descrição do s820 (saída n95)	81
Figura 52	Descrição do s832 (saída n90)	82
Figura 53	Descrição do s838 (saída n215)	82
Figura 54	Descrição do s953 (saída n104)	83
Figura 55	Descrição do s1196 (saída G542)	84
Figura 56	Descrição do s1238 (saída n117)	85
Figura 57	Descrição do s1423 (saída n90)	86
Figura 58	Descrição do s1488 (saída n75)	87
Figura 59	Descrição do s1494 (saída n70)	88
Figura 60	Descrição do s5378 (saída n240)	89
Figura 61	Descrição do s9234 (saída n676)	90
Figura 62	Descrição do s13207 (saída n594)	91
Figura 63	Descrição do s15850 (saída n460)	92
Figura 64	Descrição do s38417 (saída n7962)	93
Figura 65	Descrição do s38584 (saída n7656)	94

LISTA DE TABELAS

Tabela 1	Memória estimada para armazenar as maiores matrizes	44
Tabela 2	Apresentação dos circuitos que serão utilizados nas comparações	45
Tabela 3	Comparação entre PTM e PTM Serial	46
Tabela 4	Aplicação do método SPR sobre os ISCAS85	47
Tabela 5	Comparação entre as confiabilidades obtidas pela PTM e SPR	48
Tabela 6	Comparação entre PTM e SPR-MP	49
Tabela 7	Comparação entre PTM Serial e SPR-MP	49
Tabela 8	ISCAS89 versão combinacional	52
Tabela 9	Nova amostragem de circuitos	53
Tabela 10	Apresentação de valores de confiabilidade em MTBF	54
Tabela 11	Diferença Percentual entre diferentes números de fanouts . . .	54
Tabela 12	Diferenças Percentuais entre os tempos de processamento . . .	55
Tabela 13	Diferenças entre SPR-MP e SPR	56
Tabela 14	Classificação dos circuitos mais confiáveis	57
Tabela 15	Portas que mais afetam a confiabilidade do s444	58
Tabela 16	Diferença percentual entre os valores e o valor de MTBF . . .	59
Tabela 17	Análise SPR-MP: Quantidade de <i>fanouts</i> analisados	96
Tabela 18	Análise SPR-MP: diferentes quantidades de fanouts	97
Tabela 19	Análise SPR-MP: Diferenças percentuais	98
Tabela 20	Análise SPR-MP: Tempos de processamento(ms)	99
Tabela 21	Análise SPR-MP: Tempos de processamento	99

LISTA DE ABREVIATURAS E SIGLAS

BC	<i>Bathtub Curve</i>
BN	<i>Bayesian Network</i>
CI	<i>Circuito Integrado</i>
CTMC	<i>Continuous-time Markov Chains</i>
DTMC	<i>Discrete time Markov Chains</i>
ESD	<i>Electrostatic discharge</i>
FIT	<i>Failure in Time</i>
ISCAS	<i>International Symposium on Circuits and Systems</i>
ITM	<i>Ideal Transfer Matrix</i>
MTBF	<i>Meantime Between Failure</i>
PGM	<i>Probabilistic Gate Model</i>
PMC	<i>Probabilistic Model Check</i>
PRISM	<i>Probabilistic Symbolic Model Checker</i>
PTM	<i>Probabilistic Transfer Matrix(ces)</i>
SET	<i>Single-Event Transient</i>
SPR	<i>Signal Probability Reliability</i>
SPR-MP	<i>Signal Probability Reliability - Multi Pass</i>

SUMÁRIO

1	INTRODUÇÃO	12
1.1	Falhas em Circuitos Digitais	13
1.2	Mascaramento	14
1.2.1	Mascaramento Elétrico	14
1.2.2	Mascaramento Temporal	15
1.2.3	Mascaramento Lógico	15
1.3	Objetivos	15
1.4	Organização do Trabalho	17
2	CONCEITOS E MÉTODOS EM ANÁLISE DE CONFIABILIDADE	18
2.1	Definições e Métricas	18
2.2	Simulação e Injeção Física de Falhas	20
2.3	Simulação de Monte Carlo	20
2.4	Abordagens Analíticas	21
3	OS MÉTODOS PTM E SPR	23
3.1	Matriz de Transferência Probabilística	23
3.2	Confiabilidade pela Probabilidade de Sinais	27
3.2.1	SPR Multi-Pass	30
4	FERRAMENTA DESENVOLVIDA	32
4.1	Descrição da Ferramenta	32
4.1.1	PTM	35
4.1.2	SPR	40
4.1.3	SPR-MP	41
5	RESULTADOS	43
5.1	Comparações entre os métodos	43
5.1.1	PTM e PTM-Serial	44
5.1.2	SPR	46
5.1.3	PTM e SPR	47
5.1.4	PTM e SPR-MP	48
5.1.5	PTM-Serial e SPR-MP	48
5.2	Expandindo as análises com o SPR-MP	50
5.2.1	Nova amostragem de circuitos	50
5.3	SPR como métrica de confiabilidade	55
6	CONSIDERAÇÕES FINAIS	61

REFERÊNCIAS	63
APÊNDICE A AMOSTRA DE CIRCUITOS DA PRIMEIRA ANÁLISE . .	67
APÊNDICE B DESCRIÇÃO VERILOG DOS CIRCUITOS DA SEGUNDA AMOSTRAGEM	72
APÊNDICE C ANÁLISE FANOUTS SPR-MP	95

1 INTRODUÇÃO

Equipamentos eletrônicos estão onipresentes no cotidiano das pessoas, desde o uso para entretenimento ao auxílio em processos industriais. Tais dispositivos são formados por sistemas digitais integrados ou circuitos integrados (CI) (BUTZEN, 2012). A evolução dos circuitos integrados levou-os a escalas nanométricas nos dias atuais, de acordo com a premissa da “Lei de Moore”, que diz que a cada 2 anos a densidade dos transistores dobra (MOORE, 1965). Grande parte dos circuitos integrados são projetados usando um fluxo automático auxiliado por computadores, onde diversos algoritmos de síntese são utilizados para gerar o leiaute circuito a ser fabricado. O bloco básico deste fluxo são as portas lógicas e estas por sua vez, são compostas por arranjos de transistores (KRAVETS, 2001).

A concepção do transistor foi responsável pela revolução da microeletrônica e, consequentemente, da computação no último século. Esse dispositivo funciona de forma simplificada como uma chave “liga-desliga”, a qual implementa a linguagem binária dos computadores. Com transistores cada vez menores, os circuitos integrados passam a comportar uma grande quantidade desses dispositivos por área, o que possibilita um incremento nas funcionalidades dos sistemas. Nesta mesma linha, aspectos como velocidade e consumo de energia também são melhorados quando se entra na escala nanométrica da construção de circuitos. Entretanto, quanto menores são os transistores, maiores são as vulnerabilidades envolvidas (XIAO et al., 2017). Isso faz com que aumentem as chances de ocorrerem falhas ainda no processo de fabricação, falhas transitórias durante a utilização do circuito e também falhas por envelhecimento com o passar do tempo de utilização do sistema (CHOUDHURY; MOHANRAM, 2007).

Assim surge o termo “confiabilidade” que pode ser inserido em diversos escopos, pois não se restringe apenas à Ciência da Computação. No contexto deste trabalho, o conceito se insere dentro de uma das quatro propriedades fundamentais de sistemas eletrônicos: confiabilidade, funcionalidade, desempenho e custo (AVIZIENIS et al., 2001). Contudo, a tradução do termo para o português pode gerar ambiguidades, visto que em tradução direta, “*dependability*” equivale a confiabilidade, bem como “*reliability*” também equivale a confiabilidade. A ambiguidade pode ser visualizada na taxonomia do termo, demonstrado

na Fig. 1. Assim, a confiabilidade relacionada a “*dependability*” é uma característica mais ampla de um sistema, aquela que garante o correto funcionamento, ou seja, pode ser definido como a habilidade de um produto ou sistema de funcionar, dentro dos padrões e limites estabelecidos (KAPUR; PECHT, 2014). Já a confiabilidade oriunda de “*reliability*”, é definida como a probabilidade de um sistema executar uma determinada função, sobre certas condições, em um determinado período de tempo (BIROLINI, 2012).

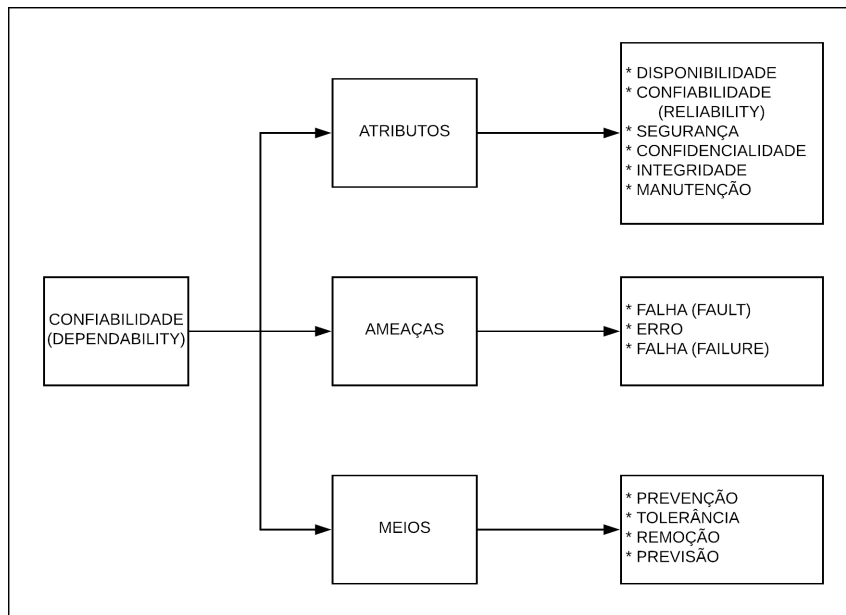


Figura 1: Taxonomia de confiabilidade adaptado de (AVIZIENIS et al., 2004)

1.1 Falhas em Circuitos Digitais

A miniaturização dos circuitos digitais possibilitou grandes avanços na indústria de semicondutores em termos de desempenho. Grande parte deste desempenho está relacionado com o tamanho e a baixa tensão dos transistores. Contudo, a mesma tecnologia que aumenta o desempenho dos arranjos de transistores, os torna menos confiáveis e mais sensíveis a falhas. Tendo em vista que todos os elementos padrões de um circuito digital são formados por redes de transistores, a confiabilidade de um transistor impacta diretamente a confiabilidade de todo o circuito. As falhas que afetam um circuito digital são destacadas conforme a sua fonte e duração e, segundo (MISKOV-ZIVANOV; MARCULESCU, 2006) podem ser classificadas em 3 tipos:

- **falhas permanentes** são aquelas que afetam de forma definitiva as características de um circuito, e que permanecem ativas até que um reparo seja feito (ex: *stuck-at-zero* e *stuck-at-one*);
- **falhas transientes** (*soft* ou *single-event upset*) são aquelas que ocorrem em um curto espaço de tempo e desaparecem, mas que podem causar uma mudança no

estado do circuito, sendo geralmente causadas por fenômenos físicos externos, tais como choques de partículas alpha e nêutrons, Descarga Eletrostática (ESD), dentre outros; e

- **falhas intermitentes** são aquelas que causam um comportamento errático no circuito em intervalos não padronizados, e que após a primeira ocorrência são grandes as chances de se tornar uma falha permanente.

1.2 Mascaramento

As propriedades de mascaramento dos circuitos digitais podem influenciar na percepção da confiabilidade final do circuito (PAGLIARINI, 2013). Tal característica se deve ao fato de que vários problemas em circuitos não são percebidos pois são “mascarados”. Assim serão abordados neste trabalho três dos principais tipos de mascaramento.

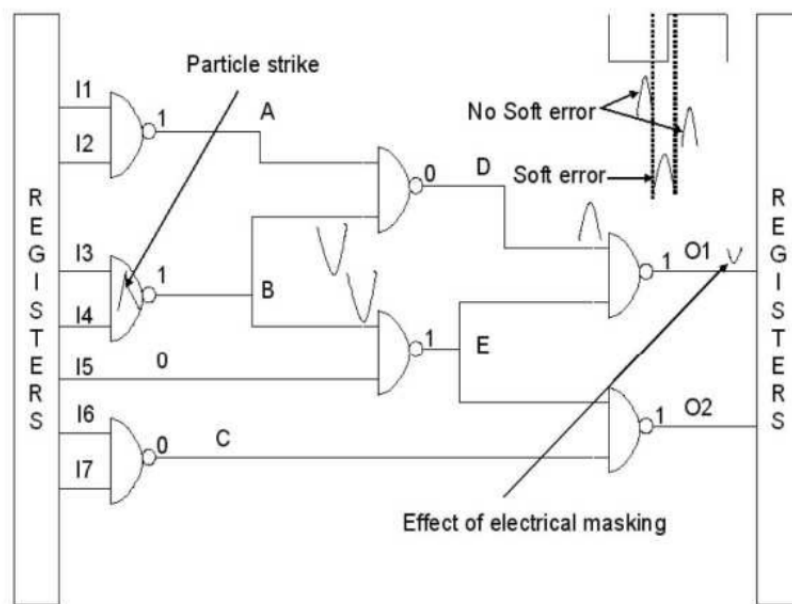


Figura 2: Efeito dos mascaramentos Elétrico, Temporal e Lógico causados por um choque de partícula (RAMANARAYANAN et al., 2009)

1.2.1 Mascaramento Elétrico

O mascaramento elétrico pode ser caracterizado pela capacidade de um circuito combinacional atenuar o efeito de um pulso elétrico, gerado por um choque de uma partícula com uma porta lógica, através das portas lógicas subsequentes (GEORGE; LACH, 2011). Como pode ser observado na Fig. 2, uma porta NAND2 foi atingida por uma partícula. Por conseguinte, o valor errado é propagado até a saída do circuito. Contudo, o pulso foi atenuado ao passar pelas portas subsequentes. Neste exemplo foi utilizado uma porta NAND2, mas o efeito pode ser observado em outros tipos de portas lógicas. A explicação

para o efeito se deve à composição de dois efeitos elétricos que reduzem a força do pulso, conforme este passa pelas portas lógicas. O atraso ou *delay* do circuito causado pelo tempo de troca dos transistor aumenta o tempo de subida e queda do pulso. Além disso, existe o fato de que a amplitude de um pulso em curto espaço de tempo, tende a cair, devido ao desligamento da porta antes de atingir toda a amplitude do pulso. Estes dois efeitos somados causam uma redução na duração do pulso e da amplitude, fazendo com que diminuam as chances de uma propagação ocorrer.

1.2.2 Mascaramento Temporal

Este tipo de mascaramento é observado quando uma falha é propagada até a saída de um circuito, gerando um valor inapropriado. Porém, se o momento de manifestação de tal falha está fora do ciclo de gravação de uma célula de memória ligada a saída deste circuito, a falha será mascarada, pois não será armazenada (KRISHNASWAMY; MARKOV; HAYES, 2008). Na Fig. 2, mais precisamente no lado superior direito, está representado o efeito do mascaramento temporal. É preciso destacar que a miniaturização dos circuitos propicia que maiores frequências sejam utilizadas, assim cada vez menos se pode contar com os efeitos do mascaramento temporal.

1.2.3 Mascaramento Lógico

O mascaramento lógico depende exclusivamente da topologia do circuito e não é afetado pela miniaturização da tecnologia (FRANCO, 2008). Assim, os dois tipos de mascaramentos anteriores irão se manifestar apenas se a falha não for mascarada logicamente. O efeito do mascaramento lógico, também é demonstrado na Fig. 2. A saída da NAND2 atingida por um partícula é um *fanout*, a qual é o sinal de entrada de outras duas portas NAND2. Observando de cima para abaixo, a segunda porta NAND2 recebe o valor errado "1". Contudo, independente do valor ser errado ou não, o valor do sinal de entrada **I5** é "0", o que determina o valor correto da saída desta porta lógica. Este trabalho irá explorar as propriedades do mascaramento lógico em circuitos combinacionais para estimar a confiabilidade.

1.3 Objetivos

A estimativa de confiabilidade é de suma importância para o desenvolvimento de circuitos mais robustos. Diversos métodos que estimam a confiabilidade de circuitos foram propostos. A partir da literatura correlata, pode ser inferido que métodos exatos exigem um alto poder computacional para obter resultados. Além disso, dependendo do tamanho do circuito, o processo de estimativa se torna inviável. O método PTM é um exemplo clássico desta característica, onde é possível lidar e obter a confiabilidade de circuitos pequenos. Porém, quando é necessário o tratamento de circuitos com mais de duas dezes-

nas de portas, o método se torna impraticável. Contudo, a exatidão do método o faz ser uma base de comparação no desenvolvimento de novos métodos. Nesse contexto, surge o método Confiabilidade pela Probabilidade de Sinais - SPR, sendo uma alternativa para a estimativa de circuitos baseado nas probabilidades de sinais. O SPR tem complexidade linear ao número de portas do circuito, caracterizando-o como um método sem problema de escalabilidade. Por outro lado, na presença de *fanouts* reconvergentes, o método considera o mesmo sinal, no mesmo instante, com valores distintos, ou seja, ele propaga valores errados, ocasionando um resultado de confiabilidade não exato. Assim, foi proposta uma variação deste método, denominada SPR *Multi-pass* (SPR-MP), a qual busca, além de lidar com o problema de *fanouts*, a precisão do resultado a ser obtido. Mais uma vez é demonstrado que a exatidão sempre tem um custo computacional elevado, e no caso do SPR-MP, é necessário percorrer diversas vezes o mesmo circuito, onde o número total de ciclos irá depender tanto do número de *fanouts*, quanto da posição onde estes estão localizados no circuito. Em relação à PTM, o SPR-MP é uma opção que não demanda grande quantidade de memória para estimar confiabilidade. Por outro lado, o tempo de processamento em circuitos grandes, considerando todos os *fanouts*, torna o SPR-MP tão impraticável quanto o método PTM.

Assim sendo, este trabalho tem como proposta explorar os métodos PTM, SPR e SPR-MP, no intuito de estabelecer seus limites de análise de confiabilidade. Para atingir este objetivo, é necessário entender e comparar os três métodos, verificando suas vantagens e desvantagens com relação aos custos de memória e de processamento. Os métodos PTM e SPR-MP geram resultados exatos de confiabilidade, o que os torna métodos equivalentes, tanto com relação à precisão dos resultados, quanto com relação aos problemas de escalabilidade. O problema de armazenamento do método PTM já foi amplamente explorado na bibliografia. Contudo, formas de trabalhar de forma sequencial com os cálculos da PTM, podem ser uma alternativa para ampliar a utilização deste método clássico. Essa alternativa também será explorada neste trabalho. Sobre o SPR-MP, considerar a totalidade das combinações entre os *fanouts* reconvergentes, o torna um método exato, porém impraticável dependendo do número de *fanouts*. Por outro lado, o método propõe alternativas para diminuir o número total de ciclos, ao custo da precisão do valor de confiabilidade a ser obtido. Além disso, cada *fanout* contribui de forma distinta na confiabilidade de todo o circuito. Esta característica também pode ser investigada, a fim de tornar seu uso viável. Por fim, levando em conta os três métodos, o SPR é o único que possui complexidade linear, o que possibilita a utilização em circuitos maiores que os outros dois métodos. Mesmo gerando valores de confiabilidade não exatos, é possível avaliar se o SPR pode se tornar uma métrica na estimativa de confiabilidade em circuitos digitais.

1.4 Organização do Trabalho

As próximas seções são organizadas como segue. A Seção II trará uma visão geral dos modelos de análise de confiabilidade existentes. A Seção III será dedicada à parte teórica dos modelos PTM, SPR e SPR-MP. Na Seção IV será descrita a forma como foi implementada e desenvolvida a ferramenta. Já na Seção V são demonstrados os resultados obtidos com as implementações dos três métodos, utilizando tanto os circuitos do *benchmark* ISCAS85 (BRGLEZ; FUJIWARA, 1985), quanto uma amostragem de circuitos menores. Por fim, na Seção VI as conclusões do trabalho são apresentadas.

2 CONCEITOS E MÉTODOS EM ANÁLISE DE CONFIABILIDADE

O incremento na complexidade dos circuitos integrados, diretamente relacionado com escalas nanométricas, exige que seja possível analisar, antes de sua fabricação, o seu comportamento na presença de possíveis falhas (HASAN; PATEL; TAHAR, 2011). A fim de guiar o processo de *design* de projeto desses componentes, buscando obter as saídas lógicas para as quais o projeto do circuito foi idealizado, é necessário que sejam desenvolvidas ferramentas que possam avaliar, com eficiência e acurácia, a confiabilidade dos mesmos. Contudo, o processo de avaliar a confiabilidade exata de um circuito integrado, envolve ou metodologias exaustivas ou métodos com complexidade exponencial, ou seja, diretamente relacionada com o tamanho do circuito a ser analisado. Para estimar a confiabilidade de um circuito combinacional qualquer, utilizando uma metodologia exaustiva, a complexidade total estará relacionada com a Equação 1, onde N_g é o número de portas lógicas e N_{in} refere-se ao número de entradas do circuito (XIAO; CHEN, 2014). Isso quer dizer que, dependendo do tamanho do circuito a ser analisado, o poder computacional exigido pode inviabilizar o processo de análise. A seguir serão apresentados diversos conceitos e métodos de análise relacionados à confiabilidade em circuitos digitais.

$$O(N_g \cdot 2^{(N_{in} + N_g)}) \quad (1)$$

2.1 Definições e Métricas

Considerando todas as informações descritas até agora, é possível determinar que conhecer a confiabilidade de um circuito pode auxiliar no processo de concepção do projeto destes dispositivos. Quando se trata de confiabilidade em sistemas, no geral, são utilizados os termos R e $R(t)$, onde a confiabilidade será a probabilidade de um sistema funcionar corretamente no intervalo $[0, t]$, onde $R(0) = 1$ (LALA, 2001). A métrica utilizada para a confiabilidade em sistemas eletrônicos é a taxa de falhas, a qual é representada pelo símbolos λ ou $\lambda(t)$ e, nada mais é do que a frequência que o sistema falha em um determinado período de tempo (FINKELSTEIN, 2008). Contudo, dado o número de fatores

que podem influenciar a taxa de falhas de um componente eletrônico, um valor preciso é difícil de ser determinado, o que leva à aplicação de métodos probabilísticos, onde são consideradas informações de diferentes fontes, tais como simulação e testes (FRANCO, 2008). Além disso, a taxa de falhas de sistemas eletrônicos é variável, desde o período de utilização inicial até o de desgaste (*wear-out*), e este comportamento é modelado como diagrama da “curva da banheira” (*bathtub curve - BC*), a qual está representado na Fig. 3. Neste diagrama, a taxa de falhas inicial é alta devido ao grande número de componentes que estão sendo usados pela primeira vez, e o termo utilizado para definir este período é “mortalidade infantil”. Em seguida existe um período com uma taxa de falhas quase constante, ao longo da “vida útil” do sistema. Por último, ocorre o período de “desgaste” do sistema, onde a taxa de erros sobe. Isso se deve aos fatores de fadiga e de degradação dos componentes do sistema (FINKELSTEIN, 2008).

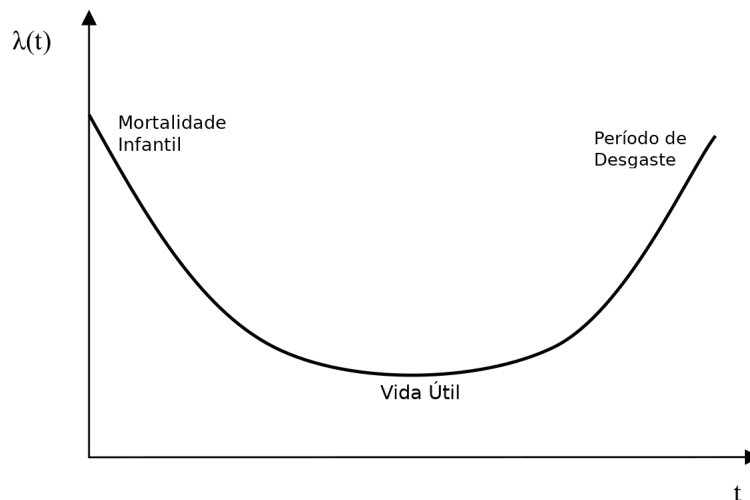


Figura 3: Diagrama da Curva da Banheira, adaptado de (FINKELSTEIN, 2008)

Considerando somente o período de vida útil do sistema, a taxa de falhas será constante, ou seja, $\lambda = \lambda(t)$. Assim, a confiabilidade do sistema pode ser definida pela Equação 2 (FRANCO, 2008).

$$R(t) = e^{-\lambda t} \quad (2)$$

Por outro lado, uma métrica mais utilizada para fazer comparações entre confiabilidades de diferentes sistemas é o **MTBF** (*Mean-Time Between Failure* (LIENIG; BRUEMMER, 2017), o qual tem uma relação recíproca com a taxa de falhas e é expressa em horas. Nas Equações 3 e 4, estão descritas, respectivamente, como obter o MTBF a partir da taxa de falhas e a equação de confiabilidade adaptada para o MTBF.

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

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

Outra métrica utilizada na indústria de semicondutores é o **FIT** (*Failure in Time*), o qual significa a quantidade de falhas em 10^9 horas (LIENIG; BRUEMMER, 2017). O FIT pode ser obtido a partir do MTBF, conforme a Equação 5 (ICHINOMIYA et al., 2010).

$$FIT = \frac{10^9}{MTBF} \quad (5)$$

2.2 Simulação e Injeção Física de Falhas

Simulação e injeção física de falhas podem ser divididas em duas técnicas baseadas na necessidade de fabricação do circuito. A Simulação de Falhas é uma técnica simples e intuitiva para se estimar a confiabilidade de um circuito. O processo consiste em selecionar um nodo, que dependendo da granularidade utilizada na análise, pode ser desde um bloco no circuito até um transistor individual (HSUEH; TSAI; IYER, 1997). Selecionado o nodo, a análise consiste na observação dos valores de saída do circuito em determinado período de tempo. Como base de comparação, geralmente se utiliza, em paralelo, uma segunda versão do circuito livre de falhas. Assim a simulação verifica quais nodos estão provocando desvios nos valores de saída. É necessário destacar que o tempo consumido em simulações de injeção de falhas é alto. Isso se deve à simulação ter de lidar com todos cenários possíveis, incluindo a combinação de falhas e todas os valores de entrada possíveis do circuito. Assim, dependendo do tamanho do circuito, esta abordagem se torna inviável.

Técnicas de injeção necessitam de uma amostra do circuito, ou seja, é necessário que o circuito seja fabricado. O procedimento de estimar a confiabilidade por meio desta técnica é expor o circuito a fatores que podem causar falhas. Como é necessário um processo de fabricação e construção de um ambiente fisicamente simulado, o custo desta técnica é elevado e não trivial. Um grande exemplo do uso deste tipo de técnica pode ser encontrado em (PARTRIDGE; HALL; HANLEY, 1965), onde é descrito como os circuitos da missão Apollo da NASA foram submetidos a testes e simulações.

2.3 Simulação de Monte Carlo

A ideia de utilizar a aleatoriedade para determinar algo foi revolucionária. É possível rastrear cientistas que fizeram uso desde o século 18. Contudo, o método de Monte Carlo como é conhecido e reconhecido atualmente, foi utilizado na época da Segunda Guerra Mundial, no projeto Manhattan, por John Von Neumann e Stanislaw Ulam para o desenvolvimento de armamento nuclear (HARRISON, 2010). Desde então, a técnica é utilizada por diversos profissionais de várias áreas, como finanças, gerenciamento de

projetos, energia, indústrias, engenharia, pesquisa e desenvolvimento, seguros, petróleo e gás, transportes e meio ambiente. A simulação de Monte Carlo é uma técnica computacional que possibilita levar em conta o risco em análises quantitativas e tomadas de decisão fornecendo uma gama de resultados possíveis e as probabilidades de ocorrências desses resultados de acordo com a ação escolhida como decisão. As desvantagens que envolvem o método são o seu alto custo computacional e a inexatidão dos resultados, pois estes são baseados no modelo de entrada do método.

Com relação à estimativa de confiabilidade de circuitos lógicos, a simulação de Monte Carlo pode lidar com parâmetros randômicos, o que pode ser uma grande vantagem perante outros métodos, conforme pode ser visto em (LIU; CAI, 2017), o qual propõe um modelo para obter a confiabilidade de um circuito combinacional na presença de falhas do tipo SET. Existem diversas formas de conceber um modelo de simulação Monte Carlo, mas para fins de confiabilidade de circuitos, todas soluções irão convergir na expressão representada na Equação 6, onde N_o é o número total de amostragem e N_e é o número de resultados não esperados.

$$R_{circuit} = 1 - \frac{N_e}{N_o} \quad (6)$$

2.4 Abordagens Analíticas

Existem diversos métodos que estimam a confiabilidade de um circuito por meio de abordagens analíticas. A maioria deles leva em conta dados probabilísticos (FRANCO, 2008). A probabilidade é uma forma de se lidar e modelar grande parte das falhas em circuitos lógicos. Assim sendo, o método “Modelo de Portas Probabilísticas - PGM” é baseado análise de confiabilidade por meio do tratamento probabilístico de sinais (HAN et al., 2014). A probabilidade de um sinal de entrada ou saída, geralmente, é definida pela probabilidade de o sinal ser um “1” lógico. Levando em conta tais premissas, o método PGM foi proposto por (HAN et al., 2011) onde é utilizada a Lógica Probabilística para modelar e estimar a confiabilidade do circuito lógico.

O método PTM (Matrizes de Transferência Probabilística) visa estimar a confiabilidade de um circuito de forma precisa, tendo sido proposto por (PATEL; HAYES; MARKOV, 2003). É um método que tem um problema de escalabilidade, sendo assim, é um bom modelo para analisar circuitos pequenos, mas se torna computacionalmente inviável quando se quer trabalhar com circuitos com mais de 2 dezenas de portas lógicas. Mesmo assim, este método foi o escolhido para ser analisado e implementado neste trabalho, pois quando se trabalha com escalas nanométricas, a precisão pode ser um grande diferencial.

Outro método que pode ser utilizado para estimar a confiabilidade de um circuito é o método “*Probabilistic Model Checking - PMC*”, o qual é um procedimento de verificação se uma certa probabilidade satisfaz uma probabilidade especificada. O PMC foi utilizado

para a análise de confiabilidade em (BHADURI et al., 2007). Neste método os circuitos são descritos em cadeias de Markov em tempo discreto ou DTMC, o que segundo (SI-EWIOREK; SWARZ, 2017) é a estrutura mais apropriada para modelar sistemas digitais. A confiabilidade de um circuito é obtida calculando a probabilidade de atingir estados de DTMC específicos, onde tais estados representam os valores booleanos corretos nas saídas do circuito para uma certa distribuição de probabilidade nas entradas. Em 1999 um grupo de pesquisas da Universidade de Birmingham criou o PRISM, acrônimo para *Probabilistic Symbolic Model Checker*. O software utiliza o método PMC para dar suporte à análise de três tipos de modelos probabilísticos (KWIATKOWSKA; NORMAN; PARKER, 2004): DTMC, CTMC(*continuous-time Markov chains*) e MDP(*Markov decision processes*).

Por fim, o método “*Signal Probability Reliability - SPR*” é um modelo baseado na probabilidade de um sinal assumir valores corretos e incorretos. A representação dos sinais se dá por quatro estados: 0 correto e incorreto e 1 correto e incorreto. Os quatro estados são representados por uma matriz de ordem 2×2 . O método não trata as correlações de sinais, ou seja, na presença de caminhos reconvergentes (*fanouts*), sinais iguais são tratados de formas diferente ao mesmo tempo, invalidando o resultado final de confiabilidade (FRANCO et al., 2008).

A PTM foi um dos métodos escolhidos para este trabalho. Por ser um método exato de estimativa de confiabilidade, pode ser usado para avaliar outros métodos. Assim, outro método escolhido foi o SPR que além de ter sido desenvolvido baseado nas características da PTM, é um método com complexidade linear. Esta característica do SPR pode tornar possível a sua utilização para estimar a confiabilidade e lidar com problemas de escalabilidade em circuitos grandes.

3 OS MÉTODOS PTM E SPR

Neste capítulo serão descritos, com maior aprofundamento teórico, métodos que estimam a confiabilidade de circuitos digitais por abordagem analítica. Além disso, esses métodos foram utilizados para atingir os objetivos deste trabalho.

A análise de confiabilidade em componentes eletrônicos pode ser dividida em dois aspectos: predição e avaliação (PAGLIARINI, 2013). Os dois aspectos são igualmente importantes no processo de design de circuitos, pois a avaliação da confiabilidade permite a validação e refinamentos dos modelos de confiabilidade preditivos. O contexto deste trabalho será o estudo de confiabilidade pelo aspecto da predição.

3.1 Matriz de Transferência Probabilística

A abordagem do método Matriz de Transferência Probabilística (*Probabilistic Transfer Matrix* - PTM) é de estimar a confiabilidade de um circuito de forma exata, pois representa, por meio de matrizes, todas as possibilidades de entrada, saída e operação de um circuito. O método foi proposto por Patel et al (PATEL; HAYES; MARKOV, 2003), o qual foi amplamente explorado por Krishnaswamy et al (KRISHNASWAMY et al., 2005). A ideia central da PTM, é correlacionar as entradas e saídas de um circuito, considerando sua topologia e confiabilidade individual de cada porta (FRANCO, 2008).

Em uma matriz PTM, a relação entre linhas e colunas corresponde à probabilidade de uma combinação de entrada (linha) gerar uma combinação de saída (coluna). Nesse sentido, é possível inferir que ao somar todos os elementos de uma linha, o resultado deverá ser 1. Tendo em vista que a matriz é formada por todas as combinações de entrada e de saída de uma porta ou de todo um circuito, a ordem da matriz resultante é dada pela fórmula $2^x \times 2^y$, onde x é o número de entradas e y é o número de saídas.

A matriz PTM individual de cada porta está diretamente relacionada com a respectiva tabela verdade. Enquanto na tabela verdade são demonstrados os valores gerados pela função lógica da porta, na PTM são demonstradas as probabilidades da porta gerar os valores 0 e 1 na sua saída, considerando possíveis falhas de operação da porta. No método a matriz PTM de uma porta lógica possui duas variáveis: “ q ” que representa a

confiabilidade da porta, e “ $1 - q$ ” que, como pode ser observado, é o complemento da confiabilidade, ou seja, a probabilidade da porta gerar um valor errado. Outro ponto importante do método é a existência de Matriz de Transferência Ideal (*Ideal Transfer Matrix* - ITM), a qual representa o comportamento ideal da porta lógica ou do circuito, ou seja, sem a presença de falhas. Na Figura 4 estão representadas uma tabela verdade, uma matriz ITM e uma matriz PTM de uma porta lógica *AND* de duas entradas, as quais demonstram a forma como são obtidos os elementos para os futuros cálculos.

Input	Input	Output
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

(a)

	0	1
00	1	0
01	1	0
10	1	0
11	0	1

(b)

	0	1
00	q	$1-q$
01	q	$1-q$
10	q	$1-q$
11	$1-q$	q

(c)

Figura 4: Relação entre Tabela Verdade e Matriz PTM: (a)Tabela Verdade e Matrizes (b)ITM e (c)PTM de uma Porta *AND*

Outro elemento básico a ser levado em consideração pelo método são os fios condutores entre as portas lógicas. Esses elementos também são representados na forma de matrizes. Contudo, o método considera que a probabilidade de falha entre interconexões é nula, ou seja, é considerada apenas a propriedade de condução dos valores lógicos. A construção da matriz segue as mesmas características da matriz ITM anteriormente introduzida. Alguns arranjos de interconexões e suas respectivas matrizes estão representados na Fig. 5.

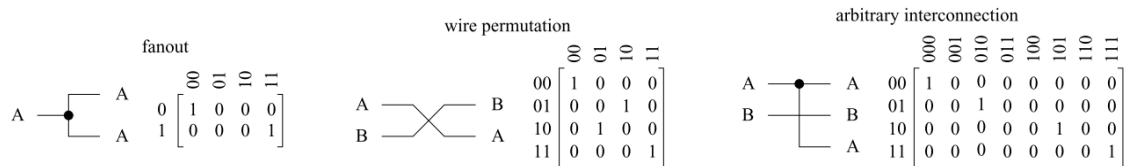


Figura 5: Exemplos de arranjos de fios condutores e suas respectivas matrizes (FRANCO, 2008)

No método PTM, as principais operações entre as matrizes são o tensor de Kronecker e a multiplicação de matrizes, onde a aplicação de uma ou de outra dependerá da disposição entre as portas do circuito. Em termos gerais, quando se tem uma disposição em paralelo é utilizado o tensor de Kronecker, pois este expande as combinações entre as matrizes. No caso de disposições em série, é utilizada a multiplicação de matrizes.

É importante destacar a necessidade da caracterização topológica do circuito, visto que a abordagem PTM considera a divisão do circuito em dois tipos de níveis: o de portas

e o de interconexões (fios condutores). No nível de portas, como o próprio nome já sugere, são consideradas as portas que possuem a mesma profundidade lógica. Já o nível de interconexões está relacionado com a topologia dos fios condutores entre a entrada e o primeiro nível de portas e entre os demais níveis de portas subsequentes.

Quando se trata de nível de portas, vale destacar que, uma interconexão poderá ser considerada uma porta. Isto ocorre, por exemplo, quando duas portas estão separadas por um ou mais níveis de profundidade. Em um nível de porta, todas as portas estão em uma disposição paralela. Sendo assim, a matriz PTM dos níveis de portas é obtida por meio da aplicação do tensor de Kronecker entre todas as portas do respectivo nível, respeitando a ordem vertical das portas.

Todos os níveis de portas são interligados por níveis de interconexão. Porém, nem todos os níveis de interconexão são levados em consideração nos cálculos. Como exemplo, pode ser citado um arranjo de quatro condutores de entrada **abcd**, onde cada letra é um rótulo de um respectivo fio, formando um nível de interconexão. Se for considerado que na entrada e na saída deste nível a topologia permanece a mesma, ou seja, o nível possui a entrada **abcd** e saída **abcd** (uma disposição paralela), este nível de interconexão não causará efeito nos cálculos de confiabilidade. Contudo, se existe um nível de interconexão com a entrada **abcd** e saída **abcbd**, isso significa que ocorreu uma mudança topológica do condutor **b**, o qual se dividiu em um *fanout*, e neste caso em específico, a matriz gerada por este nível de interconexão deve ser considerada para se obter a confiabilidade exata de todo o circuito. A matriz gerada no nível também terá uma ordem dada pela fórmula $2^x \times 2^y$, onde x é o número de entradas, que neste caso são quatro, e y é o número de saídas, ou seja, cinco (abcbd) no exemplo citado. Os elementos desta matriz serão os valores “0” ou “1”, os quais estarão relacionados com as combinações de entrada e saída corretas. Nesse sentido, a primeira linha desta matriz, a qual tem o endereço “0000” ($a = 0, b = 0, c = 0, d = 0$), a única coluna correta será a “00000” ($a = 0, b = 0, b = 0, c = 0, d = 0$). Portanto o elemento 0×0 desta matriz de interconexão terá o valor “1” e todos os outros elementos desta linha serão “0”.

Já foi visto que quando há uma disposição paralela de elementos no circuito, aplica-se o tensor de Kronecker. Quando os elementos estão em série, a multiplicação de matrizes é aplicada. Sendo assim, para gerar a matriz PTM de todo o circuito, é necessário multiplicar as matrizes tanto de portas quanto de interconexões entre si.

Apresentados os componentes básicos do método, é possível seguir com a forma de determinar a confiabilidade de um circuito. Para tanto, o primeiro passo é determinar quais são os níveis do circuito a ser analisado. A Figura 6 demonstra todos os passos para se obter a matriz PTM de um circuito. Após a divisão do circuito em níveis (6a), as portas são vinculadas às suas respectivas matrizes PTM (6b). Pode ser visto que, dada a topologia do circuito, as duas matrizes de interconexão (1 - entrada do circuito; 2 - entre os dois níveis de porta) serão desconsideradas nos cálculos, visto que não apresentam

alteração das disposições dos fios condutores. Identificados os dois níveis de porta, é possível descrever a equação que formará a matriz PTM de todo o circuito (6c). A equação deve respeitar a ordem dos sinais no circuito. Assim, primeiro se obtém a matriz do primeiro nível, aplicando o tensor de Kronecker (6d), em seguida se identifica o próximo nível, o qual nesse exemplo é composto por apenas uma porta lógica e, por fim, multiplica-se as matrizes dos dois níveis, obtendo, neste caso, a matriz de todo o circuito (6e).

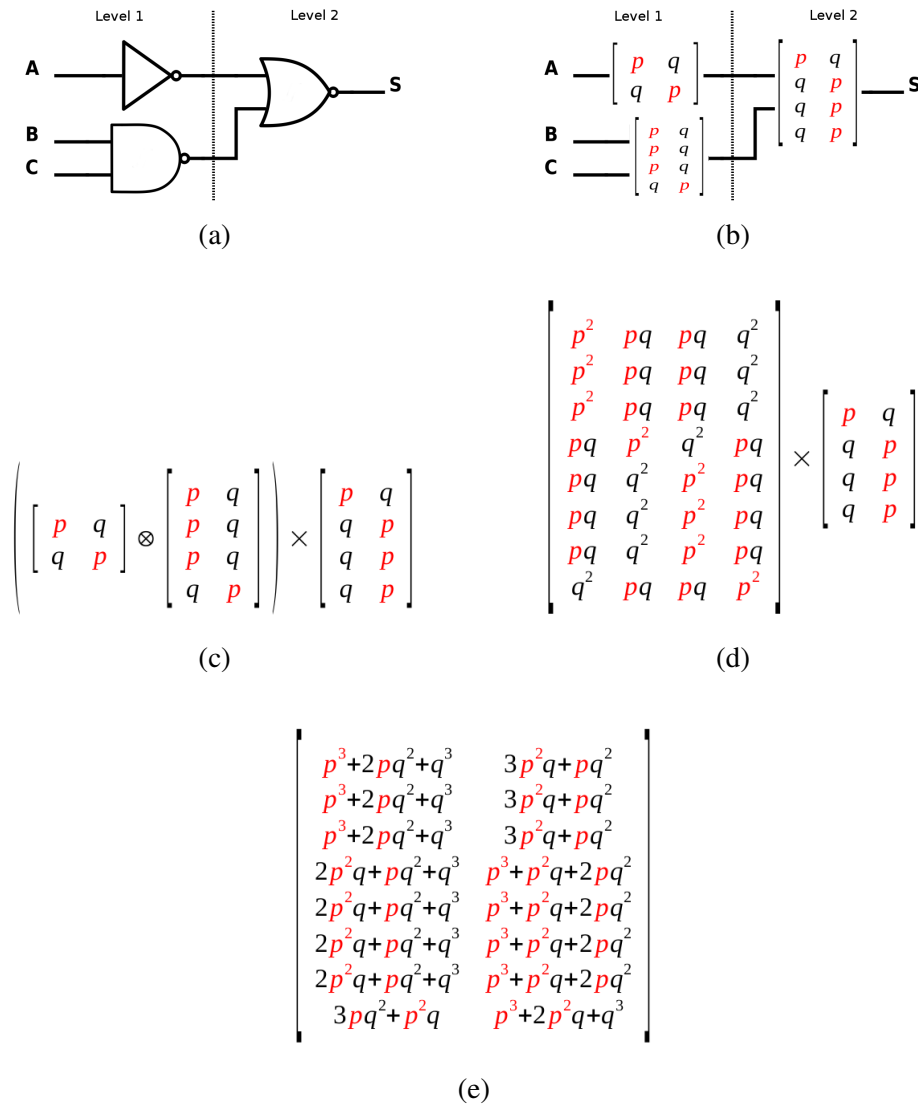


Figura 6: Formando a matriz PTM de um circuito: (a)divisão em níveis, (b)matrizes PTM das portas, (c)operações do método, (d)resolução do tensor de Kronecker e (e)matriz PTM do circuito gerada pela multiplicação de todas as matrizes dos níveis

Tendo a matriz PTM de todo o circuito, o próximo passo é a forma de se obter o valor de confiabilidade. Para tanto, é necessário obter os valores da matriz ITM, a qual terá as mesmas dimensões da matriz PTM de todo o circuito. Por outro lado, a ITM refletirá o comportamento do circuito livre de qualquer falha. Assim, são definidos duas equações que podem ser utilizadas para determinar a confiabilidade do circuito. Caso as probabilidades dos sinais sejam distintas entre si, a Expressão (7) deve ser utilizada.

Geralmente, as entradas de um circuito possuem as mesmas probabilidades. Além disso, também são consideradas livres de falhas. Assim, se este for o caso, é possível utilizar a Expressão (8) para obter a confiabilidade do circuito.

$$R_{PTM} = \sum_{ITM_c(i,j)=1} p(j|i)p(i) \quad (7)$$

$$R_{PTM} = \frac{1}{2^n} \sum_{ITM_c(i,j)=1} p(j|i) \quad (8)$$

O acumulador presente em ambas expressões refere-se aos elementos da matriz PTM que tem uma relação com os elementos iguais a 1 na matriz ITM. Esta relação pode ser vista na Fig. 7, a qual é a representação da matriz PTM e ITM do circuito presente na Fig. 6. O método PTM é um modo exato de estimar a confiabilidade de um circuito, mas sua complexidade é exponencial ao número de portas em um mesmo nível, o que o torna impraticável de ser utilizado em sua forma original.

$$ITM_c = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad PTM_c = \begin{bmatrix} p^3 + 2pq^2 + q^3 & 3p^2 + pq^2 \\ p^3 + 2pq^2 + q^3 & 3p^2 + pq^2 \\ p^3 + 2pq^2 + q^3 & 3p^2 + pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ 3pq^2 + p^2q & p^3 + 2p^2q + q^3 \end{bmatrix}$$

Figura 7: Relação PTM e ITM

3.2 Confiabilidade pela Probabilidade de Sinais

O método SPR e todas as suas variantes se baseiam nas matrizes do tipo PTM. Por outro lado, o SPR visa reduzir o problema com o tamanho das matrizes. Isso o torna escalável, ou seja, pode ser aplicado mesmo em circuitos grandes, diferente do método PTM. A premissa do SPR é de que é possível estimar a confiabilidade de um circuito por meio da propagação das probabilidades dos sinais da entrada até a saída. A metodologia considera que a ocorrência de uma saída correta pode ser determinada pela computação cumulativa dos efeitos de múltiplas falhas nos sinais do circuito. Levando em consideração todas as possíveis interações dos sinais e portas propensos a falhas, o método de probabilidade de sinais os modela obtendo as probabilidades dos sinais subsequentes e, por fim, a confiabilidade de todo o circuito.

Assim como na PTM, operações com matrizes são utilizadas para representar as probabilidades dos sinais. Cada matriz de sinal possui quatro estados: 0-correto, 0-incorreto,

1-correto e 1-incorreto (FRANCO et al., 2008). Uma representação da matriz e as respectivas posições de cada estado, podem ser observadas na Fig. 8.

$$SPR_{signal} = \begin{bmatrix} P("0" \text{ correto}) & P("1" \text{ incorreto}) \\ P("0" \text{ incorreto}) & P("1" \text{ correto}) \end{bmatrix}$$

Figura 8: Matriz de probabilidade de sinal

O processo de estimativa de confiabilidade pelo método SPR, diferente do método PTM, se dá de forma sequencial, ou seja, propagando e calculando as probabilidades dos sinais desde as entradas até as saídas do circuito. Assim, é importante demonstrar como os sinais se propagam por uma porta lógica. Como exemplo, o procedimento é ilustrado na Fig. 9, onde são consideradas duas entradas primárias em uma porta do tipo *AND* lógica.

A Fig. 9b demonstra como é o processo inicial para se obter a confiabilidade do sinal de saída de uma porta *AND* lógica, a qual é representada na Fig. 9a. Analisando as duas imagens em conjunto, fica claro qual é a relação entre a topologia da porta com as matrizes do método. Assim, o primeiro cálculo a ser feito é obter uma matriz de probabilidade entre os dois sinais de entrada. Como na PTM, tendo em vista que os dois sinais, obviamente, estão em paralelo, o tensor de Kronecker deve ser aplicado, e esse passo pode ser observado na Fig. 9c. A matriz PTM da porta é utilizada para modelar como as probabilidades irão se propagar pela mesma. Mais uma vez, fazendo alusão à PTM, já que os sinais estão em série com a porta, a operação de matriz a ser utilizada é a multiplicação, e a representação se encontra na Fig. 9d. A matriz resultante será uma matriz $2^x \times 2^y$, onde X e Y são respectivamente o número de entradas e saídas da porta. Contudo, a ordem de uma matriz de probabilidade de sinal, no método SPR, sempre será 2×2 . Portanto, a consolidação da matriz do sinal de saída é feita utilizando a matriz ITM da porta lógica. Esta relação está demonstrada na Fig. 9e. Por meio da ITM, é possível agrupar quais combinações de entrada resultarão em cada um dos quatro estados do sinal. Após agrupar, os elementos são somados e posicionados conforme a definição de matriz de probabilidade de sinal do método.

No exemplo utilizado, após obter a matriz do sinal de saída da porta, a confiabilidade do sinal será a soma das probabilidades dos estados corretos, ou seja, a soma entre os elementos das posições 1×1 e 2×2 . Para se estimar a confiabilidade de todo o circuito, além de fazer a sequência de passos descritos para cada uma das portas, é necessário fazer uma junção entre as confiabilidades dos sinais de saída do circuito, conforme é expressado na Equação 9, onde R_j está relacionado à j^{th} saída do circuito a ser estimado.

$$R_{SPR} = \prod_{j=0}^{m-1} R_j \quad (9)$$

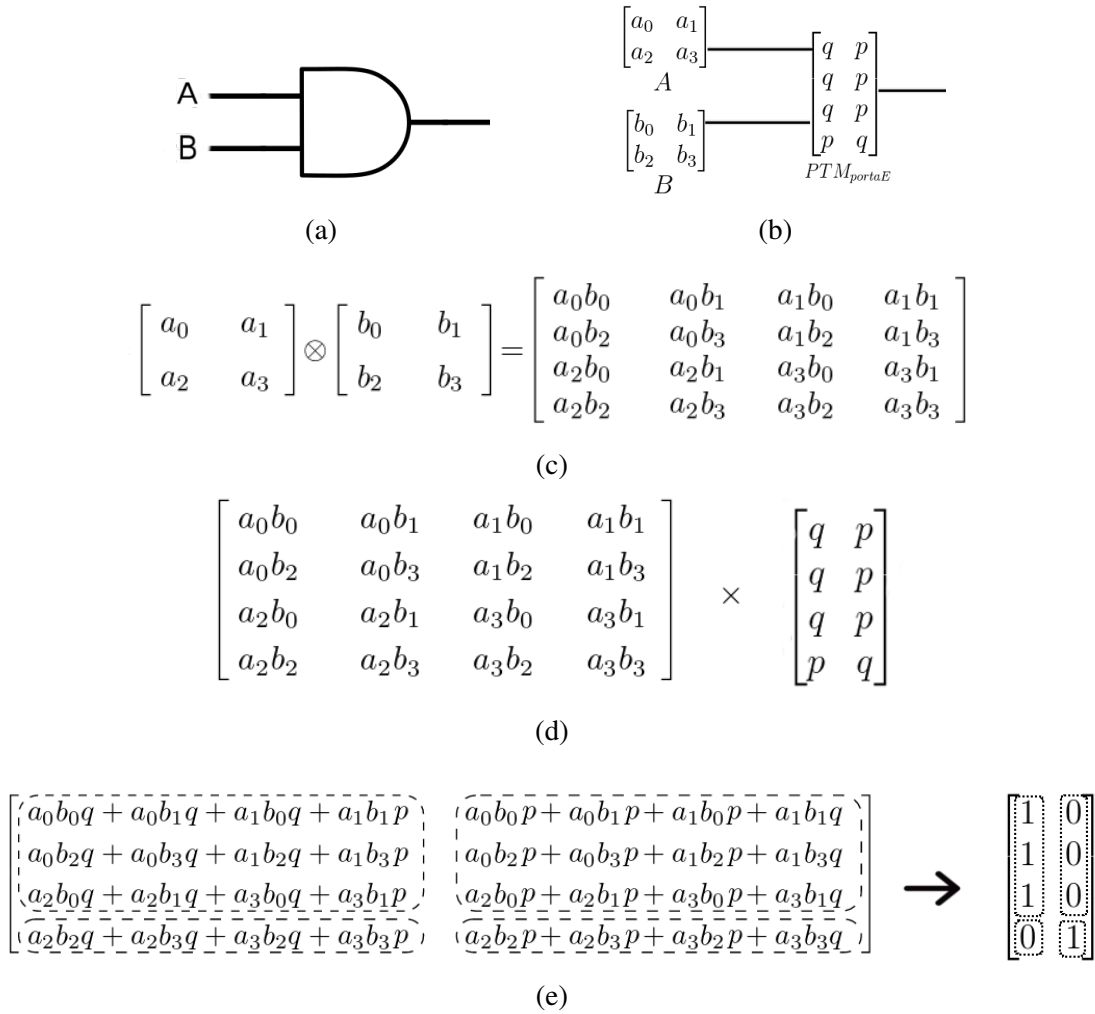


Figura 9: Passos para estimar a confiabilidade do sinal de saída de uma porta *AND* com o método SPR: (a) porta E com duas entradas primárias, (b) matrizes das entradas e matriz PTM da porta, (c) aplicação do tensor de Kronecker nas entradas da porta, (d) multiplicação da matriz referente às entradas com a matriz PTM da porta e (e) matriz resultante da multiplicação com os valores que irão compor a matriz do sinal de saída

O modelo SPR é um método rápido para estimar a confiabilidade de um circuito, pois sua complexidade é linear ao número total de portas lógicas do circuito, ou seja, $O(G)$. Os valores gerados pela abordagem são exatos desde que o circuito não possua *fanouts* reconvergentes. Essa característica é a que inviabiliza o método, pois as derivações em fios condutores é comum em circuitos digitais.

A fim de demonstrar como o SPR se comporta na presença de *fanouts* reconvergentes, foi elaborada a Fig. 10. Nesta imagem estão presentes duas formas distintas de duas portas lógicas do tipo inversor, onde à esquerda as entradas das portas compartilham o mesmo sinal de entrada, ou seja, existe um *fanout*. Na Figura à direita, as portas recebem entradas independentes. No intuito de explicitar o problema do SPR, foi utilizada a PTM como base exata de comparação. Analisando a imagem, é possível inferir que o método SPR processa o sinal de *fanout* como se o mesmo representasse dois sinais distintos.

Contudo, isso faz com que sejam considerados valores redundantes de probabilidade, o que ao final irá gerar valores de confiabilidade nulos.

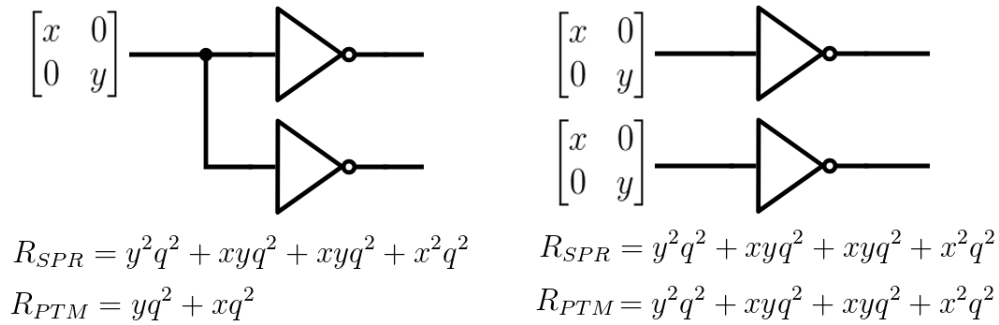


Figura 10: O problema do SPR

3.2.1 SPR Multi-Pass

Tendo em vista a forma como o método SPR lida com *fanouts* reconvergentes, uma nova abordagem foi proposta por (FRANCO et al., 2008). Essa abordagem foi denominada *SPR Multi-pass* ou simplesmente “SPR-MP”. Esta variação do SPR foi baseada no fato de que cada *fanout* contribui de forma diferente nas probabilidades dos sinais e na confiabilidade final do circuito. Assim, o algoritmo do SPR-MP corrige as probabilidades nos sinais, propagando probabilidades distintas em múltiplas iterações. Contudo, um resultado exato somente é obtido, se todos os estados de todos os *fanouts* forem considerados.

$$R_c = \sum_{f=1}^{4^F} R_{c(f)} \quad (10)$$

A forma de cálculo da confiabilidade do circuito de acordo com o SPR-MP pode ser observada na Expressão 10, onde 4^F está relacionado aos quatro estados de uma matriz de sinais sobre o número de *fanouts*; f está relacionado ao estado de *fanout* corrente. Na Fig. 11 está representado o comportamento do algoritmo do SPR-MP. É possível observar que cada estado do *fanout* representa um ciclo em todo o circuito. No final, a confiabilidade de cada ciclo é somada. Vale ressaltar que o expressão encontrada é a mesma da PTM na Fig. 10.

A complexidade do método SPR-MP é exponencial em relação ao número total de *fanouts*. Isso pode tornar o método inviável em circuitos reais. Assim, outra forma de racionalizar a quantidade de ciclos do algoritmo, pode ser utilizar um limite de valores de probabilidade próximas a 0. Isso significa que passadas com probabilidades de estados de *fanouts* que sejam iguais ou menores ao limite definido, não serão executadas pelo algoritmo.

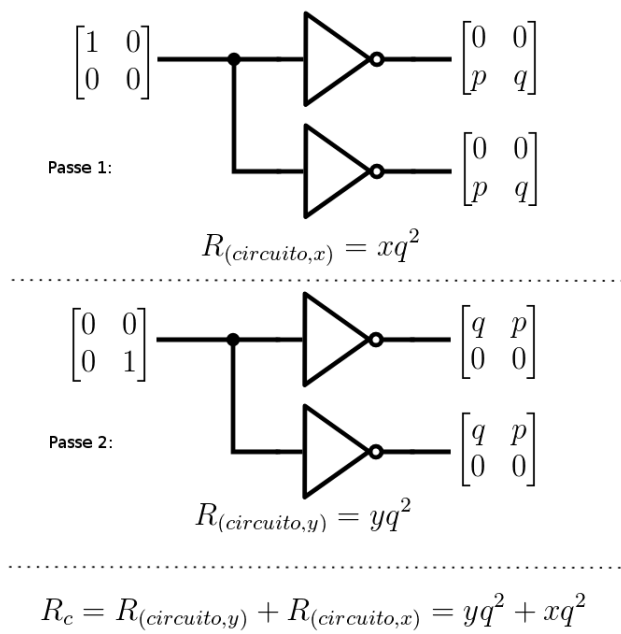


Figura 11: Exemplo de aplicação do algoritmo SPR-MP

4 FERRAMENTA DESENVOLVIDA

4.1 Descrição da Ferramenta

A fim de atingir os objetos propostos neste trabalho, foi desenvolvida uma ferramenta que implementa os métodos PTM, SPR e SPR-MP. A linguagem de programação escolhida no desenvolvimento foi JAVA, devido a familiaridade do autor com a linguagem e a utilização desta linguagem em outros projetos no grupo de pesquisas. O circuito é caracterizado na ferramenta através de uma descrição no padrão HDL (*Hardware Description Language*) - Verilog. A partir da descrição HDL do circuito, uma estrutura de dados é criada, modelando características básicas do circuito, tais como: quantidade total de portas e sinais, tipos de portas utilizadas no circuito, níveis lógicos, dentre outros.

Um dos principais princípios do paradigma da programação orientada a objetos é o foco no reuso dos códigos gerados (BRUEGGE; DUTOIT, 2004). Com isso em mente, a estrutura de dados que representa o circuito foi desenvolvida para ser a mais genérica possível, ou seja, que se adaptasse a outras demandas que envolvam circuitos lógicos. Nesse sentido, na Fig. 12 está ilustrado o diagrama de classes desta estrutura básica.

Conforme o diagrama, um objeto “Circuito”(Circuit) é composto por objetos “Porta”(Gate) e “Sinal”(Signal). A relação entre sinais e portas lógicas se deu aplicando o

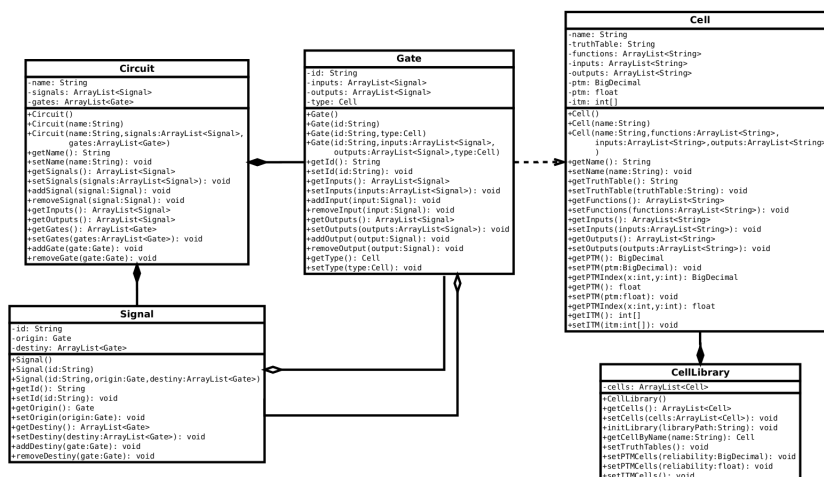


Figura 12: Diagrama de classes da estrutura básica que representa um circuito lógico

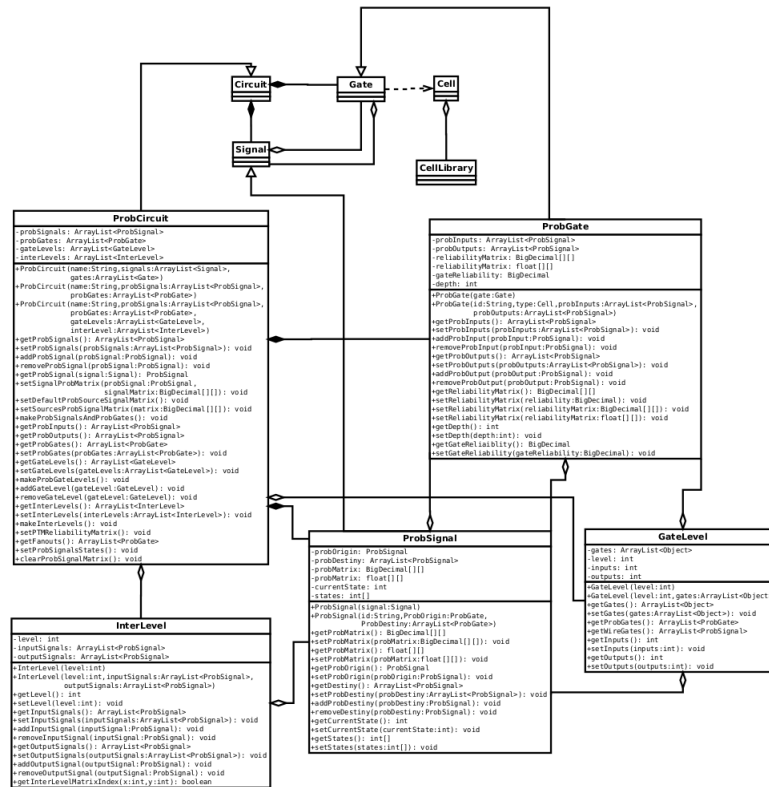


Figura 13: Diagrama de classes da estrutura especializada para os métodos PTM e SPR

conceito de “composição” entre as classes “Porta” e “Sinal”, e assim foi possível modelar as entradas e saídas de uma porta lógica e, por sua vez, a origem e destino de um sinal lógico. Ainda sobre a classe “Porta” foi modelada uma relação de dependência com a classe “Célula”(Cell), a qual contém os atributos do tipo da porta lógica. Por fim, a classe “Biblioteca de Células”(CellLibrary) é utilizada para lidar com as células criadas a partir do arquivo externo que conterá todos os tipos de células.

Contudo, para o cálculo da confiabilidade, utilizando os métodos propostos, foi necessário especializar e criar classes que modelassem as peculiaridades de cada um. Assim, chegou-se ao diagrama de classes representado na Fig. 13. O conceito de herança foi aplicado às três classes básicas: “Circuito”, “Porta” e “Sinal”. Foram introduzidos atributos que pudessem lidar com as probabilidades individuais de portas e sinais. No tocante à nova classe que representa uma porta lógica, pode parecer que há redundância das probabilidades com a classe “Célula”.

Com a possibilidade de cada porta poder possuir uma probabilidade individual, experimentos que envolvam a análise individual de cada porta são possíveis. A justificativa pode ser a mesma com relação à nova classe que representam os sinais do circuito. As classes “Nível de Gate”(GateLevel) e “Nível de Interconexão”(InterLevel) representam a topologia do circuito, a qual é uma característica importante, principalmente para o método PTM. Em um objeto “Nível de Porta” estão contidos as portas e sinais relacionados a um determinado nível lógico do circuito. Já um objeto “Nível de Interconexão”

modela os sinais que estão entre os níveis de portas de um circuito. Este objeto facilita a verificação do comportamento do sinais, principalmente com relação às derivações e cruzamentos entre os mesmos.

Uma característica que é possível destacar sobre a ferramenta é a não necessidade de *softwares* externos para os cálculos dos métodos. Isto geraria uma dificuldade maior para o desenvolvimento, pois existem *softwares* consolidados para fazer cálculos, tais como *Matlab* e *Scilab*. Todavia, a flexibilidade em customizar as funcionalidades se torna um diferencial nos estudos.

Um diagrama de funcionamento da ferramenta é demonstrado na Figura 14a. Como é possível visualizar, houve uma preocupação em utilizar diferentes tipos de dados a fim de verificar a influência de cada um no desempenho final dos métodos principais. Os parâmetros básicos necessários para se estimar a confiabilidade de um circuito, pela ferramenta são: (1) descrição do circuito em Verilog, contendo entradas, saídas e ligações entre as portas; (2) biblioteca de células na qual o circuito foi sintetizado; (3) a confiabilidade da tecnologia, ou seja, o valor de confiabilidade que será considerado em todas as portas lógicas do circuito. A Fig. 14b mostra o terminal da ferramenta sendo utilizado.

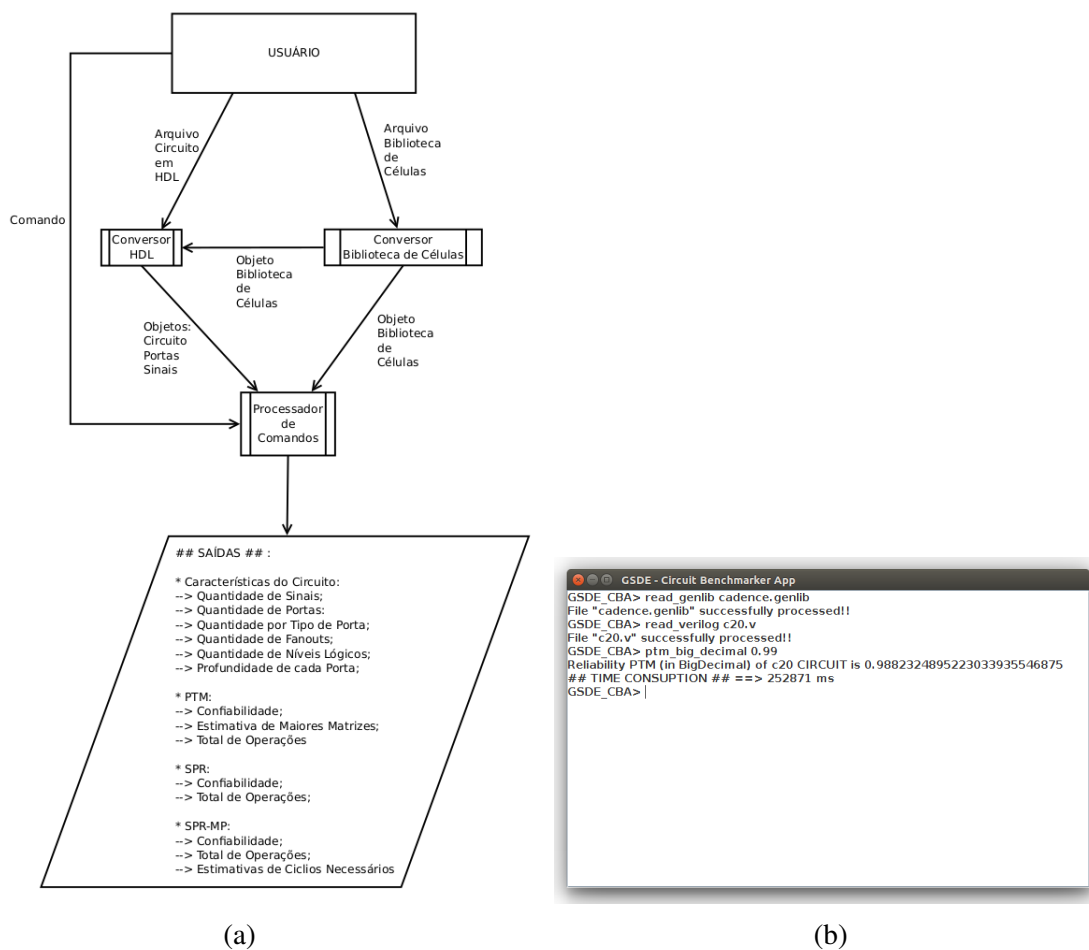


Figura 14: Ferramenta desenvolvida: (a)Funcionamento básico da ferramenta desenvolvida e (b)Print screen do terminal gerado pela ferramenta

Segundo (KRISHNASWAMY et al., 2005), o método PTM extrai a confiabilidade exata de um circuito. Por isso, a PTM será utilizada como base nas comparações. Na primeira implementação do método foi utilizada a biblioteca EJML, a qual utiliza em sua estrutura de dados valores do tipo “double” em JAVA. Contudo, para utilização da ferramenta em um trabalho correlato (SCHVITZ et al., 2018), foi necessário implementar a PTM utilizando métodos que lidassem com o tipo de dado “BigDecimal”. Este tipo de dado nativo da linguagem JAVA visa a precisão. Então, ao incluir funcionalidades que pudessem lidar com BigDecimal, foi possível notar que os valores de confiabilidade gerados para os circuitos não eram iguais, em termos de precisão. Assim, o próximo passo foi desenvolver funcionalidades que lidassem com diferentes tipos de dados, neste caso o BigDecimal e o *float*. Não foram desenvolvidas apenas funcionalidades que estimam a confiabilidade de circuitos. Visto que, conforme a bibliografia correlata, o grande problema de estimar a confiabilidade são os problemas de escalabilidade. Assim, foram desenvolvidas aplicações que estimam o número de operações necessárias para o cálculo de confiabilidade, ou até mesmo, no caso da PTM, que estimam o tamanho das maiores matrizes a serem criadas. Na sequência serão relatados, de forma individual, como foram implementados os métodos PTM, SPR e SPR-MP.

4.1.1 PTM

O método PTM foi o primeiro a ser desenvolvido e inserido na ferramenta. Como descrito nas seções anteriores, a PTM é baseada em operações com matrizes. Assim, nesta etapa inicial, optou-se por buscar uma biblioteca para apoiar nas representações e cálculos destas matrizes. Dentre todas as bibliotecas encontradas, a biblioteca EJML (EJML, 2017) mostrou-se mais apta a ser aplicada no escopo do trabalho, pois era do tipo *open source*, tinha ampla documentação e seu código era de simples compreensão e aplicabilidade.

Os primeiros valores de confiabilidade foram gerados com o método, a partir de circuitos equivalentes, em termos de tamanho, ao C17 do ISCAS85 *Benchmark* (BRGLEZ; FUJIWARA, 1985). O grande problema da PTM pôde ser confirmado, comprovando as limitações descritas na literatura sobre o método. Assim, foram seguidas duas linhas de implementação, uma denominada “tradicional” e outra “serializada”, as quais são descritas a seguir.

4.1.1.1 Implementação Tradicional

A implementação tradicional segue os mesmos procedimentos mostrados na teoria da PTM (Seção 3.1). A estrutura central nesta primeira etapa do desenvolvimento foi a representação das matrizes em sua totalidade, ou seja, as matrizes e todos os seus respectivos elementos eram criados e alocados em memória, sem qualquer tipo de compactação. Com relação às operações de multiplicação e de tensor de Kronecker, no mínimo três ma-

trizes precisavam ser alocadas, ao mesmo tempo, em memória: as duas matrizes operando e a matriz resultante.

O passo inicial para preparar o circuito para análise é organizá-lo em níveis de interconexão e níveis de portas, tal qual abordado no referencial teórico. Neste sentido, a ferramenta identifica todos os níveis de interconexão e de portas do circuito, tomando como base a profundidade lógica das portas.

Após a identificação e divisão do circuito em níveis, a ferramenta inicia o procedimento de estimativa calculando a matriz PTM dos níveis, seguindo progressivamente das entradas às saídas do circuito. Nos níveis de porta, a matriz é obtida aplicando o tensor de Kronecker em todas as portas deste nível. Já nos níveis de interconexão, primeiro a ferramenta verifica se existe alguma diferença na topologia, ou seja, é feita uma comparação entre os sinais que entram e saem deste nível. Caso sejam idênticos, tanto em quantidade, como na ordem, esta matriz PTM de nível não será necessária para a estimativa. Porém, na presença de mudanças entre os sinais, a ferramenta gera a matriz de interconexão combinando os valores de entrada com os de saída, através de um laço de repetição que vai de 0 até o número total de linhas (2^x , onde x é o total de sinais de entrada). Tendo em vista que as matrizes de interconexão são matrizes esparsas contendo apenas 0's e 1's, e que no método PTM, a soma dos elementos de uma mesma linha será sempre 1, é necessário identificar apenas em qual coluna será posicionado o elemento 1. Nesse sentido, a coluna corresponderá ao mapeamento entre as posições de entrada e saída do nível de interconexão.

A ferramenta irá calcular a matriz resultante entre duas matrizes de níveis até que a matriz PTM de todo o circuito seja definida. Este procedimento exige o armazenamento de, pelo menos, três matrizes, o que torna a estimativa viável apenas para circuitos que gerem matrizes de tamanho compatível com a estrutura de dados de vetores em JAVA.

O procedimento que resulta na matriz ITM do circuito, é o mesmo que o da PTM, com a diferença que a confiabilidade informada à ferramenta é a ideal, ou seja, a confiabilidade de todas as portas será 1 ($q = 1$).

Após carregar circuitos de diferentes tamanhos e estruturas, as limitações como memória e velocidade de processamento foram conformadas. Com relação à memória, as características da biblioteca EJML se mostraram ineficientes para lidar com o método PTM, isso por que os objetos gerados pelas classes da biblioteca, até o momento que esta dissertação foi elaborada, tem a limitação de número total de elementos da matriz a ser representada, diretamente relacionado ao limite máximo de representação do tipo “int” na linguagem JAVA, a qual é $2^{32} - 1$. Tal fato já limita o tamanho de circuitos a serem analisados. Com relação ao processamento, notou-se que, se fosse possível representar os objetos que representam as matrizes de níveis, o tempo não seria um problema. Como forma de ilustrar o algoritmo desenvolvido baseado no método tradicional da PTM, foi elaborado um pseudo-código representado na Fig. 15.

```

1 public class PTM {
2     public static void main(String args[]) {
3
4         dividirCircuitoEmNiveis();
5         for(x=0; x<NIVEIS.length; x++) {
6             if(NIVEIS.get(x) == "NÍVEL DE PORTA") {
7                 //Aplica Tensor de Kronecker em todas as portas do nível
8                 matrizCorrente = CriaPTMNivelPorta();
9             } else {
10                //Faz o mapeamento das interconexões
11                matrizCorrente = CriaPTMNivelInterconexão();
12            }
13            matrizPTM = multiplicaMatriz(matrizPTM, matrizCorrente);
14        }
15        matrizITM = getMatrizIdeal();
16        acumulador = 0;
17
18        for(y=0; y<matrizITM.length; y++) {
19            if(matrizITM.get(y) == '1') {
20                acumulador = acumulador + matrizPTM.get(y);
21            }
22        }
23
24        CONFIABILIDADE = (acumulador / (2^ENTRADAS));
25    }
26 }

```

Figura 15: Pseudo-código da PTM tradicional

4.1.1.2 Implementação Serializada

O método PTM é baseado em matrizes, mas não é estipulada a forma como as operações com matrizes podem ser realizadas. Não obstante, o grande problema do método PTM, como já foi exposto, é o crescimento exponencial no consumo de memória para a representação das matrizes e seus respectivos elementos. Sendo assim, a próxima etapa da pesquisa foi investigar uma outra maneira de realizar os cálculos, minimizando ao máximo o uso de memória.

Em um contexto geral, a matriz PTM de todo o circuito é formada por cálculos e iterações entre as diversas portas lógicas que a compõe. Nesse sentido, é possível obter qualquer elemento da matriz PTM do circuito, de forma independente dos demais elementos, fazendo o uso de memória apenas para armazenar as informações da topologia do circuito e suas portas lógicas. Sendo assim, o objetivo nesta nova etapa seria tentar minimizar o problema exponencial de memória, transferindo a demanda ao processamento.

Assim foram desenvolvidos métodos na ferramenta que buscam obter apenas um elemento de uma matriz resultante, seja por multiplicação seja por aplicação do tensor de Kronecker. Tendo em vista a forma como o método PTM utiliza as operações de matrizes, a adaptação desta nova abordagem passa por uma análise de como cada elemento lógico no circuito contribui para a matriz resultante.

Com relação a matriz de um nível de portas, o número de operandos envolvidos na

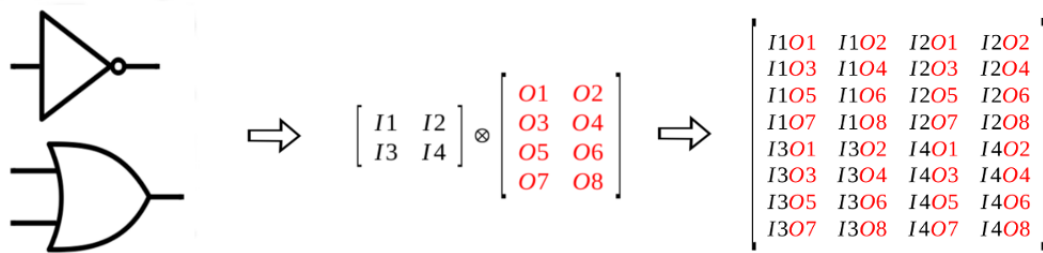


Figura 16: Demonstração do Tensor de Kronecker em Duas Portas Lógicas em Paralelo

aplicação do tensor Kronecker está diretamente relacionado com o número de portas no nível, ou seja, cada elemento da matriz resultante será composto de uma multiplicação entre apenas um elemento de cada matriz PTM das portas envolvidas, conforme demonstrado na Fig. 16. Sendo assim, ao saber quais portas e seus respectivos posicionamentos dentro do nível em questão, é possível identificar quais elementos deverão ser multiplicados para obter um elemento específico na matriz PTM do nível. A fim de ilustrar o raciocínio, a Fig. 17 traz o processo inverso da aplicação de um tensor de Kronecker, no qual é possível observar a relação entre matrizes e seus elementos. Na Fig. 17, os números apresentados correspondem ao cálculo do tensor de Kronecker entre 3 matrizes.

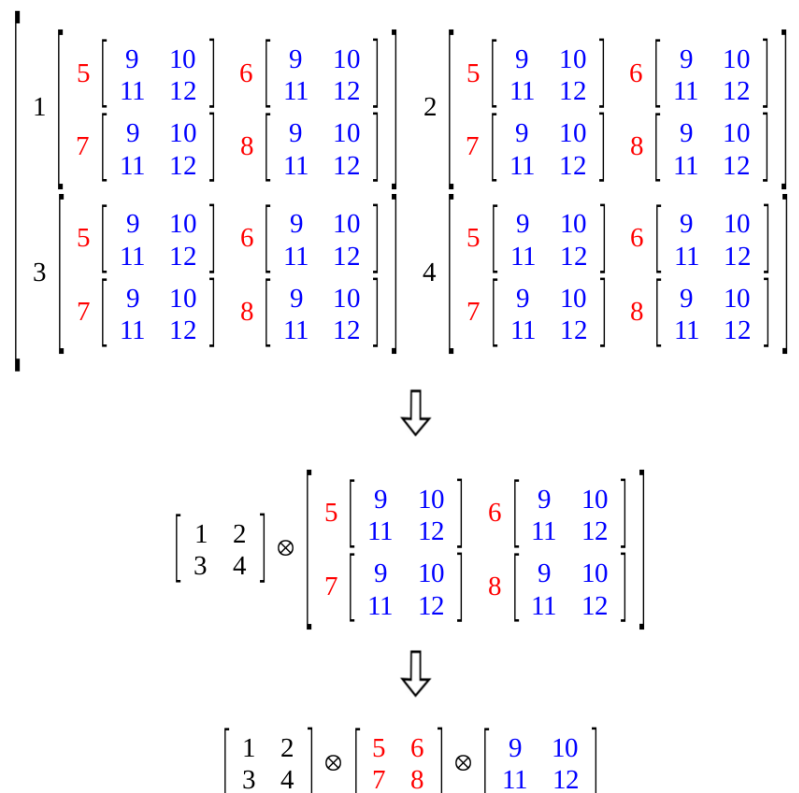


Figura 17: Processo Inverso da Aplicação do Tensor de Kronecker

	1	2	3	4	5	6	7	8
1	???	???	???	???	???	???	???	???
2	???	???	???	???	???	???	???	???
3	???	???	???	???	???	???	???	???
4	???	???	???	???	???	???	???	???
5	???	???	???	???	???	???	???	???
6	???	???	???	???	???	???	???	???
7	???	???	???	???	???	???	???	???
8	???	???	???	???	???	???	???	???

Figura 18: Formação do elemento 5×4 da matriz resultante (Fig. 17) na abordagem sequencial

O procedimento para obter um elemento de uma matriz de nível de interconexão, nesta nova abordagem, consiste em comparar o X da coordenada, a qual equivale a combinação de entrada, com o Y , que está relacionado com a combinação de saída. Utilizando a relação entre as posições de entrada e de saída do nível de interconexão e, convertendo os valores decimais para binário, o próximo passo é comparar se a combinação de entrada gera uma combinação de saída idêntica ao valor de Y convertido em binário. Assim, se a comparação é igual, o elemento da matriz será 1, caso contrário será 0.

Com relação à serialização dos cálculos em uma multiplicação entre matrizes, o processo consiste em identificar quais linha, coluna e elementos formarão o elemento a ser obtido da matriz resultante. Assim, por meio da utilização de recursividade entre os níveis de portas e de interconexão, métodos foram desenvolvidos para obter o valor do elemento de uma determinada coordenada de uma matriz. Um exemplo dos cálculos envolvidos na formação de uma matriz PTM resultante da multiplicação entre dois níveis, está representado na Fig. 19.

Assim, foram desenvolvidos métodos na ferramenta para obter um elemento de uma matriz de nível sem armazenar qualquer elemento anterior em memória. Vale destacar que, até o momento, neste novo modo de calcular a confiabilidade de um circuito pelo método PTM, a matriz ITM foi considerada com um parâmetro de entrada. Além disso, visando obter resultados que comparem o desempenho desta nova abordagem, foram consideradas apenas probabilidades de sinais de entrada iguais. Assim sendo, o valor final da confiabilidade é dada por um somatório de elementos relacionados aos elementos “1” da matriz ITM do circuito e, o resultado, assim como na implementação clássica da PTM, é dividido por 2^n , onde n é igual ao número de entradas do circuito.

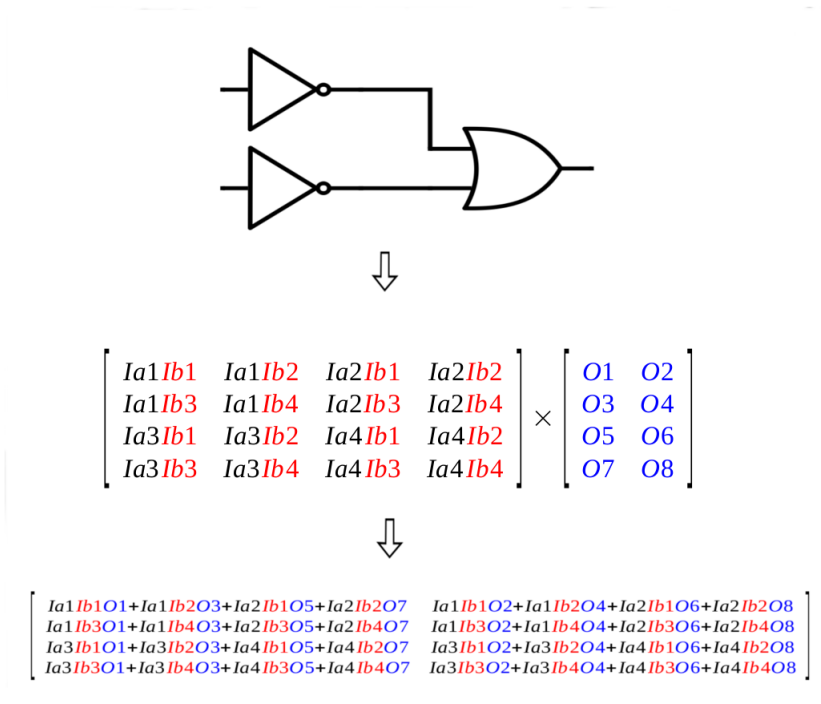


Figura 19: Formação da Matriz PTM de um Circuito

4.1.2 SPR

Seguindo a ordem de implementação, o próximo método a ser incluído na ferramenta foi o método SPR. Algumas características facilitaram o desenvolvimento do método, já que o mesmo utiliza, da mesma forma que o método PTM, operações de multiplicação e tensor de Kronecker nas matrizes de sinais e nas matrizes PTM e ITM das portas lógicas, o que tornou possível que vários métodos fossem reaproveitados. Métodos como o de representação das características do circuito, e das matrizes PTM das portas, foram os mais importantes no desenvolvimento do novo método.

O método SPR consiste em percorrer todo o circuito acumulando as probabilidades nas matrizes de sinais. A adaptação desenvolvida foi a inclusão de novos atributos na classe que representa os sinais de um circuito. Assim sendo, obter a confiabilidade do circuito, por meio do método SPR, consistiu em apenas percorrer todos os objetos de sinais, a partir dos de entrada, calculando e relacionando com as matrizes PTM e ITM das portas. O processo é finalizado quando as matrizes dos sinais de saída são determinadas. Finalmente, com todas as matrizes de probabilidade dos sinais de saída, é possível obter a confiabilidade de todo o circuito, aplicando a fórmula do SPR demonstrada na Expressão 9 localizada na Seção 3. A Fig. 20 ilustra a implementação do método por meio de um pseudo-código.

Em termos de limitações computacionais, haja vista que, conforme já explicado, a complexidade é linear ao número de portas lógicas, foi possível obter valores de confiabilidade de todos os circuitos que compõem o ISCAS 85. Apesar desta possibilidade, levando

```

1 public class SPR {
2     public static void main(String args[]) {
3
4         for(x=0; x<PORTAS.length; x++) {
5
6             aplicaTensorEntradas(PORTAS.get(x));
7             multiplicaMatriz(matrizSinaisEntrada, PORTAS.get(x).gePTM());
8             matrizSinalSaida = decodificaMatriz(matrizResultante, PORTAS.get(x).getITM());
9         }
10
11         CONFIABILIDADE = 0;
12
13         for (y=0; y<SINAIS_SAIDA.length; y++) {
14             sinaisCorretos = SINAIS_SAIDA.get(y).get0correto +
15                             SINAIS_SAIDA.get(y).get1correto;
16             CONFIABILIDADE = CONFIABILIDADE * sinaisCorretos;
17         }
18     }
19 }

```

Figura 20: Pseudo-código do Método SPR

em consideração a teoria exposta, os valores são obtidos desconsiderando os *fanouts* re-convergentes, gerando valores inconsistente de confiabilidade.

4.1.3 SPR-MP

O último método a ser implementado foi a variação do método SPR, o SPR-MP. Este método se propõe a ter uma flexibilidade em sua forma de obter a confiabilidade de um circuito. Com ele, é possível obter desde resultados tão exatos quanto o método PTM, mas com custo de processamento compatível, até resultados rápidos e aproximados como os obtidos com o SPR. Da mesma forma que no método SPR, grande parte das funcionalidades já desenvolvidas foram utilizadas e aplicadas no SPR-MP. É importante frisar que cada “passada” do método nada mais é do que a aplicação do método SPR. O que determina a quantidade de ciclos no método é a quantidade de *fanouts*. Além disso, a flexibilização do método está diretamente relacionada à quantidade de *fanouts* consideradas. Assim sendo, após a caracterização do circuito na mesma estrutura de dados utilizada nos métodos anteriores, são identificados quais dos sinais do circuito são *fanouts*. A partir disso, os sinais que forem identificados são separados em um vetor. Caso todos os sinais deste vetor forem considerados, o método irá gerar um valor exato de confiabilidade do circuito. Porém, vale lembrar, que considerar todos os *fanouts*, no método SPR-MP, pode ser um problema computacional, visto que o número de iterações será 4^x , onde x é o número de *fanouts*. Todos os ciclos de cálculo do método são controlados pelo vetor de *fanouts*, assim os sinais removidos deste vetor não serão considerados. Tendo em vista esta característica, é possível gerar resultados de confiabilidade baseados em qualquer número de *fanouts* de um circuito, o que torna viável a análise de quais sinais influenciam na confiabilidade final do circuito. Na Figura 21 é apresentado um pseudo-código do método SPR-MP.

```
1 public class SPR-MP {  
2     public getMultiPass(FANOUTS, CONFIABILIDADE) {  
3  
4         if(FANOUTS.length > 0) {  
5             for (ESTADOS FANOUTS[0]) {  
6                 ProbEstado = ESTADO;  
7                 ESTADO = 1;  
8                 RESTO_ESTADOS = 0;  
9                 REMOVE_FANOUTS[0] de FANOUTS;  
10                CONFIABILIDADE = CONFIABILIDADE + (getMultiPass(FANOUTS) * ProbEstado);  
11            }  
12        } else {  
13            PARCIAL_CONF = EXECUTA_SPR();  
14            return PARCIAL_CONF;  
15        }  
16    }  
17 }
```

Figura 21: Pseudo-código do Método SPR-MP

5 RESULTADOS

Esta seção descreve os resultados obtidos com a implementação da ferramenta e dos métodos PTM, SPR e SPR-MP. Todos os resultados que serão apresentados neste trabalho foram obtidos em um computador pessoal com as seguintes configurações de *hardware* e *software*:

- Sistema operacional Ubuntu 18.04 x64;
- Processador Intel Core i5-7200U 2.50GHz;
- Memória RAM total: 8GiB;
- Disco rígido: Samsung SSD 850 - 250GiB.

A apresentação dos resultados será feita conforme a ordem dos objetivos: na primeira etapa serão demonstradas as características individuais de cada método e as comparações entre si. A seguir, discussões referentes ao método SPR-MP e análises considerando os *fanouts* dos circuitos serão expostas. Por fim será discutido se o método SPR pode ser uma métrica, válida, de estimativa de confiabilidade em circuitos com *fanouts*.

5.1 Comparações entre os métodos

Como apontado anteriormente, o grande desafio do método PTM é analisar circuitos grandes. Dependendo do tamanho do circuito, o método, em sua forma original, gera matrizes impossíveis de serem armazenadas com a tecnologia disponível nos dias de hoje. O primeiro circuito analisado pela ferramenta foi uma versão do C17, um dos 11 circuitos do *benchmark* ISCAS85 (BRGLEZ; FUJIWARA, 1985). Tendo em vista que este circuito possuía apenas 6 portas “NAND” (Apêndice A - Fig. 27), não foi possível visualizar os problemas com memória. Contudo, ao seguir para o próximo circuito *benchmark*, no caso o C432, um circuito de 136 portas, os problemas de armazenamento de memória já puderam ser observados. Para ilustrar esta dificuldade, na Tabela 1 foram destacadas quais são as maiores matrizes encontradas nos circuitos *benchmark* ISCAS85 e, baseado no tamanho do tipos primitivos em Java (BLOCH, 2016), a memória necessária estimada para

Tabela 1: Memória estimada para armazenar as maiores matrizes dos circuitos *benchmarks* ISCAS85 pela PTM

Circuito	Portas	Níveis	Maior Matriz	Memória(GB)
C17	6	2	$2^6 \times 2^4$	1.0e−6
C432	136	14	$2^{85} \times 2^{82}$	6.9e+41
C499	188	9	$2^{136} \times 2^{130}$	4.4e+71
C880	229	11	$2^{175} \times 2^{126}$	1.5e+82
C1355	188	9	$2^{136} \times 2^{130}$	4.4e+71
C1908	234	14	$2^{144} \times 2^{128}$	2.8e+73
C2670	544	11	$2^{421} \times 2^{299}$	2.0e+208
C3540	673	18	$2^{563} \times 2^{353}$	2.0e+267
C5315	1026	18	$2^{833} \times 2^{452}$	2.4e+378
C6288	1418	45	$2^{512} \times 2^{482}$	6.2e+290
C7552	1151	15	$2^{734} \times 2^{531}$	2.3e+372

o armazenamento destas matrizes. Fica clara a impossibilidade da representação das matrizes e todos os seus elementos. A quantidade de memória necessária para representação ultrapassa as tecnologias de armazenamento atuais. Nesse sentido, também fica exposta a impraticabilidade do método, visto que o ISCAS85 é um conjunto de circuitos pequenos se comparados a circuitos de *benchmarks* mais recentes, como é o caso do circuito “vga_lcd” do *benchmark OpenCores*, o qual é um circuito que possui 124031 portas (ALBRECHT, 2005).

Com o problema de escalabilidade posto, foi necessário buscar outras opções de circuitos para aplicar o método PTM. Assim, a amostragem de circuitos utilizados em (FRITZ, 2017) foi selecionada para ser a base de análise das comparações entre os três métodos. As características são apresentados na Tabela 2, já a representação dos circuitos podem ser encontrados no Apêndice A.

5.1.1 PTM e PTM-Serial

Conforme descrito anteriormente, foi considerada uma outra forma de calcular a confiabilidade, utilizando uma alternativa sequencial de realização das operações. Nesta abordagem, apenas as matrizes relacionadas a cada porta são armazenadas em memória. Apesar de consumir mais tempo de execução, pois transfere todo o problema de memória para processamento, pode ser totalmente paralelizada, já que não existem dependências entre os elementos da matriz PTM do circuito. Todo o processo consiste em buscar o valor dos elementos relacionados à ITM do circuito, a qual é obtida por meio de uma funcionalidade que calcula os vetores de saída do circuitos baseado em todas as combinações de entrada.

Tabela 2: Apresentação dos circuitos que serão utilizados nas comparações

Circuito	E / S	Portas	Níveis	fanouts	Maior Matriz
C17-v1	5 / 2	9	3	3	$2^9 \times 2^6$
C17-v2	5 / 2	6	3	3	$2^7 \times 2^4$
C17-v3	5 / 2	6	3	3	$2^6 \times 2^4$
C17-v4	5 / 2	7	3	3	$2^7 \times 2^4$
Multiplex	6 / 1	7	3	4	$2^{10} \times 2^8$
SC-v1	3 / 2	6	2	3	$2^9 \times 2^5$
SC-v2	3 / 2	9	6	6	$2^6 \times 2^4$
SC-v3	3 / 2	12	3	6	$2^{15} \times 2^{10}$
C8	4 / 3	8	3	7	$2^8 \times 2^5$
C9	4 / 1	9	4	4	$2^9 \times 2^6$
C10	8 / 1	10	4	1	$2^9 \times 2^6$
C11	10 / 1	5	3	3	$2^{12} \times 2^7$
C20	16 / 1	20	7	3	$2^{16} \times 2^9$

Na Tabela 3 é realizada uma comparação entre as duas formas de cálculo, considerando circuitos de teste gerados para a avaliação das mesmas. Os tempos registrados na tabela foram obtidos ao se calcular a confiabilidade de cada circuito com a confiabilidade sendo $q = 0.99$. É tácito que ao se representar as matrizes em sua totalidade, o resultado é obtido mais rapidamente, visto que no método sequencial vários índices, principalmente os dos primeiros níveis do circuito, são buscados repetidas vezes. Tendo em vista que os tempos de execução dos maiores circuitos demonstraram grande diferença com relação à PTM com matrizes, decidiu-se incluir a quantidade de operações de soma e de produto de cada método. Até o C10, o número de operações do método sequencial foi menor que o método original.

Contudo, vale ressaltar que o método sequencial não executa apenas operações matemáticas, pois as funcionalidades foram implementadas utilizando a recursividade, ou seja, além das operações de soma e de multiplicação, são executadas várias chamadas a métodos de busca, principalmente, nas matrizes de portas lógicas. Por outro lado, foram incluídas características de matrizes esparsas nos cálculos envolvendo matrizes de interconexão. Uma matriz de interconexão, por ser um mapeamento entre os sinais de entrada e de saída do nível de interconexão, possuirá, no máximo, um “1” lógico por linha. Levando esta característica em consideração, foi possível racionalizar a quantidade de operações, aplicando propriedades de matrizes esparsas, pois àquelas multiplicações que envolvam o “0” lógico não são realizadas.

O método sequencial obteve os mesmos valores de confiabilidade que o método PTM original. Porém, mesmo com a otimização com relação à memória, a estimativa de tempo

Tabela 3: Comparação entre PTM e PTM Serial

Circuito	PTM Normal			PTM Serial		
	tempo(ms)	somas	produtos	tempo(ms)	somas	produtos
C17v1	2.94	2.16e+6	2.25e+6	47.04	3.27e+4	2.95e+5
C17v2	0.38	1.76e+5	1.87e+5	9.44	4.09e+3	2.50e+4
C17v3	0.49	1.22e+5	1.30e+5	7.94	4.09e+3	2.50e+4
C17v4	0.28	1.78e+5	1.87e+5	13.38	4.09e+3	2.50e+4
Multiplex	2.93	6.90e+7	6.98e+7	1297.10	2.62e+5	2.88e+6
SCv1	0.19	2.63e+5	3.01e+5	0.53	2.55e+2	1.53e+3
SCv2	0.33	7.67e+4	8.72e+4	3868.34	2.09e+6	1.51e+7
SCv3	195.34	6.75e+8	7.52e+8	4959.20	2.62e+5	3.67e+6
C8	0.35	4.03e+5	4.34e+5	12.82	8.19e+3	6.63e+4
C9	1.36	2.35e+6	2.48e+6	2886.07	5.24e+5	5.79e+6
C10	1.42	1.72e+7	1.74e+7	1194.97	1.04e+6	9.47e+6
C11	79.26	1.17e+9	1.17e+9	1307323.76*	5.36e+8	6.45e+9
C20	4431.45	5.08e+9	5.20e+9	107726635008**	1.23e+12	9.85e+14

* 21 minutos e 47 segundos

** 3 anos 5 meses e 15 dias

para se obter a confiabilidade do circuito C20 demonstrou a impraticabilidade desta abordagem sequencial. O tempo de processamento passa a ser a característica que inviabiliza o método. Ainda nesta linha de raciocínio, os circuitos representados na tabela são menores do que os do ISCAS85, contribuindo para a afirmação de impraticabilidade anterior. Por outro lado, o método serial abre possibilidades na área da paralelização, pois os índices das matrizes podem ser obtidos de forma independente. Assim, cria-se uma alternativa para tornar viável a aplicação do método PTM em sua forma original.

5.1.2 SPR

O SPR é um método que estima a confiabilidade por meio das probabilidades dos sinais. A complexidade é linear ao número de portas do circuito a ser aplicado. Esse desempenho, em comparação à PTM que possui complexidade exponencial ao número de entradas, de saídas e da quantidade de portas lógicas, pode ser notado quando se aplica o SPR no conjunto de circuitos do ISCAS85. Pelo método PTM, foi possível apenas extrair a confiabilidade do circuito C17. Já no caso do SPR, conforme a Tabela 4, são gerados valores de confiabilidade em poucos milissegundos.

A tabela traz os valores extraídos por meio de duas implementações com diferentes tipos de dados. Conforme apontado anteriormente, a utilização de ponto flutuante nos cálculos gera um maior desempenho, porém os valores gerados possuem pequenas diferenças em relação aos valores exatos. Todavia, até o momento, a expectativa com relação às diferenças eram na ordem de várias casas decimais. Contudo, a diferença dos resultados das confiabilidades do C6288 extrapolaram qualquer hipótese. A característica que destaca este circuito dos outros é a sua profundidade lógica. Assim sendo, foi realizado um estudo individual da evolução das diferenças entre “float” e “Bigdecimal” neste

Tabela 4: Aplicação do método SPR sobre os ISCAS85

Circuito	SPR Float		SPR Bigdecimal	
	tempo(ms)	Confiabilidade	tempo(ms)	Confiabilidade
C17	0.07	0.9456	0.16	0.9456
C432	0.31	0.5316	1.81	0.5321
C499	0.43	0.3754	2.24	0.3754
C880	0.62	0.3104	2.45	0.3105
C1355	0.75	0.3754	2.89	0.3754
C1908	0.71	0.2173	3.09	0.2174
C2670	0.94	0.0840	4.29	0.0840
C3540	1.20	0.0988	9.03	0.1012
C5315	1.34	0.0144	8.65	0.0144
C6288	1.63	5.6e-44	8.94	5.9e-6
C7552	1.88	0.0006	12.14	0.0006

circuito. Tomando uma das portas do nível 29 (em um total de 45 níveis) do circuito verificou-se que a diferença entre as probabilidades do sinal de saída da porta era 31%, diferente dos 560.000% notados em um dos sinais de saída do mesmo circuito. Assim, é possível concluir que, conforme o erro vai se propagando pelas portas lógicas, o erro de precisão vai se acumulando de forma exponencial.

Levando em consideração essa diferença entre os tipos de dados, a partir deste momento, os resultados apresentados neste trabalho terão sido gerados, exclusivamente, através do tipo de dado “Bigdecimal”.

5.1.3 PTM e SPR

A possibilidade de analisar circuitos maiores é uma das vantagens do método SPR. Porém, conforme destacado nas seções anteriores, o método não lida com os *fanouts* reconvergentes, o que significa que não é possível considerar a confiabilidade extraída pelo SPR como um valor exato. Para exemplificar isso, a Tabela 5 mostra as diferenças entre os resultados de confiabilidade obtidos com a PTM e o SPR. Os resultados da tabela demonstram as diferenças nos tempos de processamento entre os dois métodos. É visível a grande diferença entre um método com complexidade exponencial em relação ao número de portas e entradas de um circuito, e outro método que possui complexidade linear ao número de portas. No circuito com o maior número de portas, desta amostragem, o método SPR consegue reduzir em 34441 vez o tempo em relação à PTM. Com relação aos valores de confiabilidade, o método SPR gera valores diferentes da PTM. Nesses valores, não é possível traçar um padrão, pois a grande maioria dos valores de SPR estão abaixo dos valores obtido com a PTM. Por outro lado, com alguns circuitos, os valores de confiabilidade do SPR ficaram acima dos valores da PTM.

Tabela 5: Comparação entre as confiabilidades obtidas pela PTM e SPR

Circuito	BIGDECIMAL					
	confiabilidade (0.99)			tempo (ms)		
	PTM	SPR	δ %	PTM	SPR	$ \delta $
C17v1	0.94219	0.94218	-0.001%	3.71	0.15	25
C17v2	0.95252	0.94908	-0.361%	1.38	0.13	11
C17v3	0.95193	0.94565	-0.660%	1.32	0.11	12
C17v4	0.94675	0.94392	-0.299%	1.04	0.09	12
Multiplex	0.95900	0.95796	-0.108%	36.11	0.13	278
SCv1	0.95260	0.95438	0.186%	1.21	0.09	14
SCv2	0.92510	0.91832	-0.733%	1.11	0.08	14
SCv3	0.92526	0.92972	0.483%	1197.56	0.24	4990
C8	0.94303	0.93668	-0.674%	2.29	0.22	11
C9	0.97575	0.97201	-0.384%	14.67	0.13	113
C10	0.97716	0.97969	0.258%	23.87	0.16	150
C11	0.97521	0.97417	-0.106%	530.89	0.12	4425
C20	0.98823	0.98471	-0.356%	89546.13	2.60	34441

5.1.4 PTM e SPR-MP

O SPR-MP é uma abordagem que visa a correção das probabilidades dos sinais, na presença de *fanouts*, a cada iteração do algoritmo. Este método, assim, como o SPR e a PTM-Serial, utiliza a memória somente para representação do circuito e seus componentes. A primeira verificação após a implementação do método foi a equivalência, em termos de valores de confiabilidade, com o método PTM. Assim, na Tabela 6 estão presentes os valores de confiabilidade obtidos e o tempo de processamento decorrido para a geração dos resultados. O leiaute da tabela é semelhante da Tabela 5. O intuito foi frisar que o SPR-MP consegue lidar com *fanouts* reconvergentes, e gera valores exatos como o método PTM. Além da equivalência nos valores de confiabilidade, o tempo exigido para se conseguir a confiabilidade dos circuitos pelo método SPR-MP é outra importante informação, já que demonstra que é possível avaliar a confiabilidade de circuitos maiores do que a PTM. Para confirma disso, é possível observar o ganho em processamento temporal do C20, o tempo para extrair a confiabilidade pelo método PTM demorou 16491 vezes a mais que o SPR-MP.

5.1.5 PTM-Serial e SPR-MP

Tendo em vista que, da mesma forma que a PTM-Serial, o SPR-MP é um método que não tem problema com memória mas sim com tempo de processamento, é natural uma comparação entre os dois, a fim de verificar qual seria a melhor opção a ser utilizada. Assim sendo, a Tabela 7 traz o tempos exigidos em cada método e a quantidade de operações necessárias para se chegar a um resultado de confiabilidade exato. Com relação ao PTM-

Tabela 6: Comparação entre as confiabilidades obtidas pela PTM e SPR-MP

Circuito	BIGDECIMAL					
	confiabilidade (0.99)			tempo (ms)		
	PTM	SPR-MP	δ %	PTM	SPR-MP	$ \delta $
C17v1	0.94219	0.94219	0.000%	3.71	0.83	5
C17v2	0.95252	0.95252	0.000%	1.38	0.73	2
C17v3	0.95193	0.95193	0.000%	1.32	0.66	2
C17v4	0.94675	0.94675	0.000%	1.04	0.35	3
Multiplex	0.95900	0.95900	0.000%	36.11	1.53	24
SCv1	0.95260	0.95260	0.000%	1.21	0.23	6
SCv2	0.92510	0.92510	0.000%	2.52	2.50	0
SCv3	0.92526	0.92526	0.000%	1197.56	9.63	125
C8	0.94303	0.94303	0.000%	5.16	5.12	0
C9	0.97575	0.97575	0.000%	14.67	1.21	13
C10	0.97716	0.97716	0.000%	23.87	0.19	126
C11	0.97521	0.97521	0.000%	530.89	1.27	419
C20	0.98823	0.98823	0.000%	89546.13	5.43	16491

Tabela 7: Comparação entre PTM Serial e SPR-MP

Circuito	PTM Serial			SPR-MP		
	tempo(ms)	somas	produtos	tempo(ms)	somas	produtos
C17v1	47.04	3.27e+4	2.95e+5	0.83	1.65e+3	2.85e+3
C17v2	9.44	4.09e+3	2.50e+4	0.73	2.73e+3	4.68e+3
C17v3	7.94	4.09e+3	2.50e+4	0.66	5.47e+3	9.37e+3
C17v4	13.38	4.09e+3	2.50e+4	0.35	2.79e+3	4.82e+3
Multiplex	1297.10	2.62e+5	2.88e+6	1.53	6.48e+4	1.08e+5
SCv1	0.53	2.55e+2	1.53e+3	0.23	2.13e+3	3.62e+3
SCv2	3868.34	2.09e+6	1.51e+7	2.50	1.30e+5	2.25e+5
SCv3	4959.20	2.62e+5	3.67e+6	9.63	6.28e+5	1.05e+6
C8	12.82	8.19e+3	6.63e+4	5.12	2.33e+5	4.03e+5
C9	2886.07	5.24e+5	5.79e+6	1.21	2.54e+4	4.33e+4
C10	1194.97	1.04e+6	9.47e+6	0.19	4.67e+2	8.04e+2
C11	1307323.76*	5.36e+8	6.45e+9	1.27	7.64e+3	1.29e+4
C20	107726635008**	1.23e+12	9.85e+14	5.43	4.82e+4	8.21e+4

* 21 minutos e 47 segundos

** 3 anos 5 meses e 15 dias, baseado em tempo de execução de 13 índices da ITM do C20, os quais levaram, em média, 1643778 ms cada.

Serial, vale ressaltar novamente, que não existe uma relação direta entre o tempo exigido e o número de operações desta abordagem, visto que são necessárias várias buscas nos valores de confiabilidade das portas, tornando o processo de análise mais demorado. Assim sendo, é possível inferir que, em questão de paralelismo, o método SPR-MP seria uma melhor opção que o PTM-Serial.

5.2 Expandindo as análises com o SPR-MP

Os resultados anteriores mostraram que o SPR-MP é um método exato como o método PTM, desde que sejam considerados todos os *fanouts*. Além disso, o tempo de processamento demandado para obter os resultados demonstra que é possível analisar circuitos maiores com o método. Assim sendo, nesta seção serão apresentadas análises baseadas em uma nova amostragem de circuitos, com o objetivo de verificar se é possível obter resultados de confiabilidades próximos ao exato não considerando a totalidade dos *fanouts*.

5.2.1 Nova amostragem de circuitos

Os métodos implementados neste trabalho analisam a confiabilidade de circuitos combinacionais. O ISCAS85 é um conjunto de circuitos *benchmark* combinacionais. O próximo conjunto de circuitos, dentre os *benchmarks* mais disponíveis, é o ISCAS89 (BRGLEZ; BRYAN; KOZMINSKI, 1989). Porém os circuitos do ISCAS89 são sequenciais, impossibilitando a análise de confiabilidade na ferramenta desenvolvida e desfocando do escopo deste trabalho. Assim, foram gerados circuitos combinacionais baseados nos circuitos sequenciais do ISCAS89, visando criar *benchmarks* para atender os diversos trabalhos em desenvolvimento nessa linha de pesquisa. Além disso, devido ao tamanho dos circuitos do ISCAS89, foram criados novos circuitos a partir das saídas dos circuitos combinacionais convertidos. Para conversão dos circuitos, foi as técnicas de conversão contidas no trabalho de (CZUTRO, 2013). O conceito proposto é de que, para converter um circuito sequencial em combinacional, as entradas de *flip-flops* se tornam saídas nos circuitos convertidos, e as saídas de *flip-flops* se tornam entradas dos circuitos convertidos. O processo de conversão pode ser observado na Figura 22.

Executando os procedimentos de conversão nos circuitos ISCAS89, foram obtidos os circuitos que estão descritos na Tabela 8. É possível observar que as características dos circuitos não são um problema para o método SPR. Já para o método SPR-MP, a quantidade de *fanouts* reconvergentes inviabiliza a análise a partir do circuito “s298”, devido ao processamento necessário. Pois, se tratando do SPR-MP, se não se levar em conta as possíveis simplificações, o número de iterações será 4^f , onde f é a quantidade de *fanouts*. Assim, aproveitando que foram gerados circuitos menores baseados nas saídas do ISCAS89 “combinacional”, foram selecionados circuitos de tamanhos e características que pudessem ser analisados pelo método SPR-MP. O objetivo da seleção foi compor uma lista de circuitos o mais distinta possível, em termos de tipos de portas, quantidade de portas, número de níveis e, principalmente, número de *fanouts*. Sendo assim, a Tabela 9 traz esta nova amostragem de circuitos e suas respectivas características.

Além da nova amostragem apresentada, notou-se que ao se lidar com valores de confiabilidades próximas a 1, utilizar a métrica MTBF se mostra visualmente melhor, ou seja, fica mais fácil perceber as diferenças entre os valores. Para exemplificar, foram selecio-

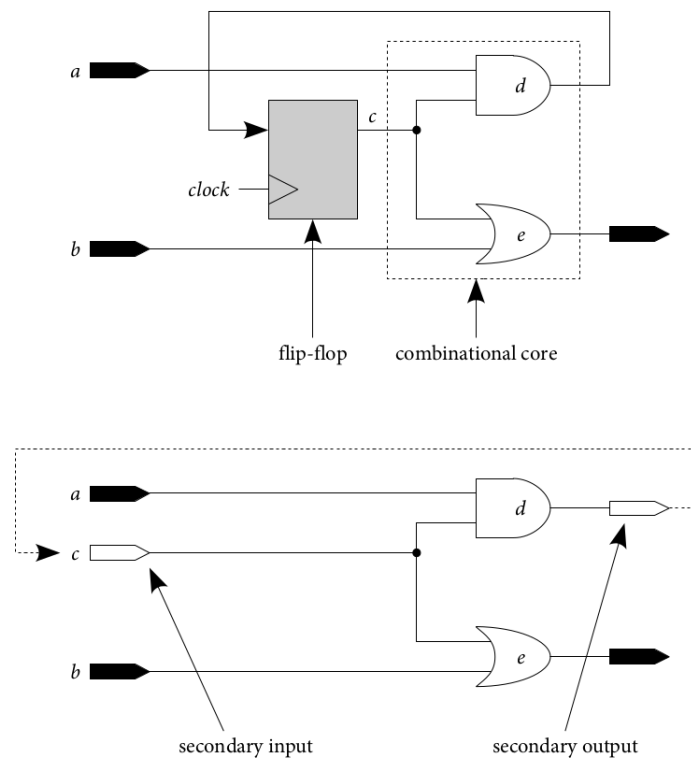


Figura 22: Convertendo um circuito sequencial em combinacional conforme a proposta de (CZUTRO, 2013)

nados alguns circuitos da nova amostragem e foi executado o método SPR-MP com uma confiabilidade de 0.999999 em todas as portas dos circuitos. Na Tabela 10 pode ser observado a diferença em analisar um valor de confiabilidade e um valor de MTBF. Portanto, a partir deste ponto do trabalho, os resultados serão convertidos em valores de MTBF.

No método SPR-MP pode ser selecionada qualquer quantidade de *fanouts*, afetando a confiabilidade final do circuito. Para se garantir um valor exato, todos os caminhos reconvergentes devem ser considerados. Mesmo assim, simulações com diferentes quantidades de *fanouts* foram realizadas, a fim de verificar o quão próximo do resultado exato é possível chegar. Foram realizadas simulações com diferentes quantidades de *fanouts*, totalizando oito amostragens, as quais levaram em consideração:

- *fanouts* que são entradas;
- *fanouts* localizados no meio dos circuitos;
- 50% dos *fanouts* mais próximos à entrada;
- 50% dos *fanouts* mais próximos à saída;
- 25% dos *fanouts* mais próximos à entrada;
- 25% dos *fanouts* mais próximos à saída;

Tabela 8: ISCAS89 versão combinacional

Circuito	Portas	E / S	Fanouts	Níveis
s27	10	7 / 4	8	4
s208	52	19 / 10	24	5
s298	78	17 / 20	30	5
s344	75	24 / 26	45	8
s349	76	24 / 26	45	8
s382	93	24 / 27	43	5
s386	88	13 / 13	32	5
s400	95	24 / 27	48	8
s420	100	35 / 18	47	8
s444	93	24 / 27	42	5
s510	145	25 / 12	46	5
s641	117	54 / 43	57	10
s713	117	54 / 42	58	10
s820	196	23 / 24	66	6
s832	198	23 / 24	68	6
s838	193	67 / 34	91	13
s953	241	22 / 29	117	6
s1196	321	31 / 31	123	10
s1238	344	31 / 31	122	11
s1423	344	91 / 79	193	26
s1488	373	14 / 25	95	8
s1494	376	14 / 25	95	8
s5378	910	214 / 227	521	9
s9234	1216	247 / 250	580	14
s13207	1770	695 / 785	763	15
s15850	2090	610 / 683	1026	19
s38417	5275	1664 / 1742	2778	13
s38584	6654	1457 / 1730	2936	13

- 10% dos *fanouts* mais próximos à entrada;
- 10% dos *fanouts* mais próximos à saída.

Devido ao grande número de resultados, foi escolhido apresentar nesta seção os valores de maneira sintetizada, onde para cada amostragem foram apresentados o maior e menor valores absolutos, a média de todos os resultados e o desvio padrão destes. Na Tabela 11 estão contidos os resultados obtidos com as simulações de maneira sintetizada e as colunas estão diretamente relacionadas as considerações elencadas anteriormente. No Apêndice C estão contidos os resultados em sua totalidade.

Analisando a média e o desvio padrão dos resultados é possível destacar que os resultados são distintos entre sim, visto que a os valores de média estão abaixo do desvio. Contudo, os resultados demonstram que fazendo determinada seleção de *fanouts* é possível

Tabela 9: Nova amostragem de circuitos

Circuito	Portas	E / S	Fanouts	Níveis
s27*	10	7 / 4	8	4
s208*	52	19 / 10	24	5
s298	22	8 / 1	11	5
s344	26	13 / 1	14	8
s349	29	11 / 1	16	8
s382	20	14 / 1	6	5
s386	19	12 / 1	8	5
s400	20	14 / 1	9	8
s420	47	34 / 1	23	8
s444	18	8 / 1	11	5
s510	48	20 / 1	17	5
s641	41	22 / 1	16	10
s713	41	22 / 1	16	10
s820	53	20 / 1	18	6
s832	56	20 / 1	16	6
s838	10	9 / 1	7	13
s953	45	17 / 1	22	6
s1196	48	18 / 1	22	10
s1238	50	18 / 1	22	11
s1423	52	30 / 1	20	26
s1488	52	14 / 1	18	8
s1494	68	14 / 1	19	8
s5378	32	9 / 1	24	9
s9234	58	56 / 1	4	14
s13207	34	18 / 1	16	15
s15850	44	20 / 1	24	19
s38417	83	65 / 1	17	13
s38584	45	23 / 1	17	13

* circuitos completos

Tabela 10: Apresentação de valores de confiabilidade em MTBF (confiabilidade das portas em 0.999999)

Circuito	Confiabilidade	MTBF
s27	0.9999933125	192193
s208	0.9999726735	36594
s444	0.9999956289	228776
s510	0.9999867560	75505
s1488	0.9999945397	183139
s1494	0.9999937019	158776
s38417	0.9999934924	153665
s38584	0.9999957962	237876

Tabela 11: Diferença Percentual entre diferentes números de fanouts

	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
MAX(abs)	s27 24.85%	s1494 13.72%	s27 18.85%	s400 13.51%	s27 19.76%	s1494 14.95%	s5378 32.86%	s5378 50.10%
MIN(abs)	s838 0.00%	s9234 0.02%	s838 0.00%	s386 0.01%	s9234 0.01%	s9234 0.01%	s838 0.07%	s838 0.17%
MED	2.98%	3.22%	3.61%	2.34%	4.53%	3.20%	5.89%	5.28%
DESVIO	5.35%	3.92%	4.67%	3.40%	5.40%	4.35%	7.74%	9.90%

chegar a resultados próximos ao exato. Nestas simulações, os valores mais aproximados foram obtidos considerando 50% dos *fanouts* mais próximos às saídas dos circuitos.

O principal objetivo em considerar números inferiores de *fanouts* é o tempo de processamento. Assim, a Tabela 12 trata sobre a redução dos tempos de processamento em cada amostragem para cada circuito. O formato de tabela acompanha o mesmo da Tabela 11, trazendo as diferenças percentuais entre os resultados. O primeiro ponto a se destacar é que realmente há ganho de tempo de processamento ao considerar uma menor quantidade de *fanouts*, apesar do custo de precisão de confiabilidade. Porém, o que chamou bastante atenção foi o resultado do circuito "s5378". Considerando apenas os *fanouts* intermediários, não houve uma redução, mas sim o acréscimo de processamento de 767%. Isso se deve a características introduzidas quando o método foi discutido na Seção 3:

- Em fanouts de entrada, os valores de 0 e 1 incorretos são, geralmente, considerados nulos (0% de probabilidade);
- Se em uma combinação corrente de estados de *fanouts*, um destes estados for 0, o resultado desta confiabilidade parcial (*pass*) será 0.

Assim, neste caso em específico do s5378, o número de simplificações considerando todos os seus *fanouts* foi maior do que considerando apenas os fanouts intermediários.

Tabela 12: Diferenças Percentuais entre os tempos de processamento

	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
MAX	s15850 -99.99%	s5378 767.19%	s15850 -99.99%	s1423 -99.95%	s15850 -99.99%	s15850 -99.99%	s15850 -99.99%	s15850 -99.99%
MIN	s386 -81.40%	s832 -1.62%	s838 -90.70%	s510 -30.48%	s386 -95.35%	s9234 -92.31%	s386 -95.35%	s382 -96.43%
MED	98.20%	104.80%	98.69%	82.44%	99.46%	98.65%	99.61%	99.52%
DESVIO	3.89%	131.77%	2.39%	18.97%	1.21%	2.39%	1.02%	0.96%

Vale lembrar que nestas simulações com o método SPR-MP não foram considerados simplificações de iterações por *threshold*. Mesmo assim, os demais resultados demonstram que reduzir o número de *fanouts* pode ser uma alternativa para expandir as análises de confiabilidade, utilizando o método SPR-MP.

Vale destacar que simulações com quantidades menores de *fanouts* foram analisadas em (PONTES et al., 2018). Nesse trabalho a amostragem de circuitos para comparar os três métodos (Seção 5.1) foi utilizada. Devido ao tamanho dos circuitos e, principalmente devido a quantidade de *fanouts*, não foi possível perceber que a redução do tempo de processamento era tão acentuada. Além disso, com a nova amostragem foi possível observar que selecionar os 50% dos *fanouts* mais próximos à(s) saída(s) do circuito pode ser a melhor opção.

5.3 SPR como métrica de confiabilidade

Já foi demonstrado que o método SPR, se comparado ao método PTM, é extremamente rápido em termos de processamento, pois realiza apenas uma propagação no circuito. Contudo, os valores gerados pelo SPR não são exatos e, os resultados da Seção 5.1 demonstram que não é possível traçar um padrão das diferenças de confiabilidades em diferentes circuitos. Nesta seção serão mostrados resultados que objetivam verificar se, mesmo gerando valores não exatos, o SPR possa ser uma métrica de estimativa de confiabilidade. Para isso, o primeiro passo foi verificar os valores de MTBF e tempos de processamento obtidos pelo SPR em comparação ao SPR-MP. Esta comparação se encontra na Tabela 13. Os resultados da Tabela 13 reforçam a questão de desempenho do SPR, mas mais uma vez, demonstram que as diferenças para os resultados exatos do SPR-MP não seguem qualquer padrão. No circuito s5378, por exemplo, a diferença para o resultado exato chega a quase 42%.

Por outro lado, pode ser considerada a hipótese de que, mesmo não gerando valores exatos, o SPR possa ser uma métrica de confiabilidade. Para explorar esta hipótese foi feita um outro experimento, o qual envolveu classificar os circuitos que possuem o menor valor de MTBF, ou seja, uma classificação dos circuitos mais confiáveis. O objetivo desta simulação foi verificar se ambos os métodos classificam a listagem de circuitos da mesma maneira. O resultado desta simulação pode ser observado na Tabela 14. Como é possível

Tabela 13: Diferenças entre SPR-MP e SPR

Circuito	SPR-MP MTBF	SPR-MP Tempo(ms)	SPR δ %	SPR Tempo(ms)
s27	149532	708	-24.28%	0.22
s208	36594	21756004	-5.40%	2.66
s298	192193	714	-1.17%	0.60
s344	157679	48243	6.65%	0.42
s349	166451	52684	5.01%	1.27
s382	287091	56	-1.89%	0.21
s386	592764	43	1.47%	0.21
s400	270819	390	8.35%	0.22
s420	60350	8866581	1.16%	1.63
s444	228776	755	-13.27%	0.33
s510	75505	119002	-0.51%	1.34
s641	213167	1086663	0.88%	0.90
s713	181863	1115342	6.00%	0.82
s820	198691	254040	1.66%	0.89
s832	174021	56300	-1.55%	0.88
s838	268344	43	0.09%	0.33
s953	101135	2472164	1.27%	0.91
s1196	190906	6924286	-1.86%	3.29
s1238	219147	3185367	-0.38%	1.00
s1423	352520	32704473	5.92%	0.63
s1488	183139	182881	9.45%	1.23
s1494	158776	804524	14.62%	1.02
s5378	50000	10839774	41.99%	0.47
s9234	249429	39	-0.01%	1.10
s13207	163381	89384	-5.59%	1.11
s15850	143920	308344145	-7.82%	0.83
s38417	153665	4868606	3.41%	1.98
s38584	237876	111084	2.10%	0.52

notar, as classificações não são idênticas. Contudo, a maior distância de erro foram 4 posições (s444). Tendo em vista que os circuitos da amostragem são distintos entre si, em termos de função lógica, número de *fanouts* e etc, ainda não se pode descartar o uso do SPR como métrica de confiabilidade.

Nesse sentido, foi realizado outro experimento, o qual teve como objetivo verificar se o método SPR pode identificar as portas que geram maior impacto no valor de confiabilidade de todo o circuito. Caso o método consiga fazer isso, dado o seu tempo de execução, o SPR poderia auxiliar o projetista sobre quais pontos do circuito merecem atenção em termos de confiabilidade.

Para verificar essa possibilidade, a metodologia utilizada foi fixar a confiabilidade em 0 em uma única porta e o restante das portas em 0.9999. Após isso, executar os métodos SPR e SPR-MP e repetir o processo para as outras portas dos circuito. Quanto menor o número de MTBF relacionado a porta, mais relevância esta terá na confiabilidade final do circuito. Então, para cada circuito analisado, foi feita uma classificação utilizando os dois métodos. Vale lembrar que muitos dos circuitos levaram um tempo considerável de processamento quando foi utilizado o SPR-MP. Além disso, cada circuito foi simulado

Tabela 14: Classificação dos circuitos mais confiáveis segundo os dois métodos

SPR-MP		SPR	
1	s208	1	s208
2	s5378	2	s420
3	s420	3	s5378
4	s510	4	s510
5	s953	5	s953
6	s15850	6	s27
7	s27	7	s15850
8	s38417	8	s13207
9	s344	9	s38417
10	s1494	10	s344
11	s13207	11	s832
12	s349	12	s349
13	s832	13	s1494
14	s713	14	s1196
15	s1488	15	s298
16	s1196	16	s713
17	s298	17	s444
18	s820	18	s1488
19	s641	19	s820
20	s1238	20	s641
21	s444	21	s1238
22	s38584	22	s38584
23	s9234	23	s9234
24	s838	24	s838
25	s400	25	s382
26	s382	26	s400
27	s1423	27	s1423
28	s386	28	s386

o número de vezes igual às suas respectivas quantidades de portas. Sendo assim, foram selecionados os 16 circuitos que tiverem o menor custo temporal para se obter o MTBF com o SPR-MP.

Da mesma forma que o experimento da Tabela 14, o SPR não classificou da mesma forma que o SPR-MP. Contudo, as portas mais importantes dos circuitos ficaram nas primeiras posições nos dois métodos. Baseado nisso, foi elaborada uma outra simulação, onde ao invés de colocar a porta com confiabilidade 0, colocou-se a porta com confiabilidade 1, o objetivo foi verificar o comportamento da classificação do SPR diante desta mudança de um extremo ao outro de confiabilidade. Como esperado, o SPR-MP manteve a mesma classificação, já o SPR alterou algumas posições das portas comparado-se com a classificação com confiabilidade em 0. A Tabela 15 contém os resultados de um dos circuitos analisados. Na classificação do SPR em confiabilidade 1, como o valor de confiabilidade individual é maior que a da fixada nas outras portas (0.9999), quanto maior o valor de MTBF, a porta contribuirá mais para a confiabilidade final do circuito.

Tabela 15: Portas que mais afetam a confiabilidade segundo o SPR e SPR-MP no circuito s444

s444					
SPR-MP em 0.9999 = 2288.17 // SPR em 0.9999 = 1984.73					
SPR-MP - Conf = 0		SPR - Conf = 0		SPR - Conf = 1	
classificação	MTBF	classificação	MTBF	classificação	MTBF
g84	0.13	g84	0.13	g84	2475.97
g80	0.55	g80	0.57	g62	2434.80
g83	0.72	g62	0.93	g80	2374.91
g62	1.55	g83	0.93	g83	2282.26
g21	3.47	g61	1.88	g61	2186.83
g61	3.74	g60	4.16	g60	2084.12
g82	5.16	g82	5.97	g82	2047.28
g81	5.57	g81	6.43	g81	2042.99
g63	7.00	g63	6.71	g63	2040.60
g60	7.73	g36	7.33	g36	2037.76
g36	12.23	g57	9.02	g57	2026.76
g19	14.47	g21	13.03	g21	2014.07
g47	27.63	g19	17.90	g19	2007.20
g59	61.84	g47	21.74	g47	2002.62
g57	82.50	g59	32.57	g59	1996.63
g58	120.86	g70	53.71	g70	1991.82
g70	120.89	g58	64.20	g58	1990.66
g56	230.04	g56	141.64	g56	1987.31

No exemplo da Tabela 15, a porta mais importante é a que gera o sinal de saída do circuito. Em todos os circuitos analisados, o SPR em 0, conseguiu destacar as portas mais importantes tal qual o SPR-MP, já o SPR em 1 trocou algumas posições. Assim, as análises realizadas mostraram que o SPR pode ser utilizado para identificar pontos que merecem maior atenção no circuito. Mas se existe um ponto que precisa ser melhorado e, existem técnicas para deixar esses pontos mais robustos, seria interessante poder acompanhar esse incremento de confiabilidade. Nesse sentido, talvez o SPR possa conseguir acompanhar esta evolução na confiabilidade do circuito.

A fim de verificar esta nova hipótese, foram selecionadas 10 portas de cada circuito das classificações anteriores. O ponto central das comparações foi o valor de MTBF de todos os circuitos obtidos através da confiabilidade 0.9999 nos dois métodos. Individualmente, cada porta que ficava em evidência teve sua confiabilidade variada para baixo e para cima, ou seja, a intenção era acompanhar o comportamento da confiabilidade geral do circuito, diminuindo e aumentando a confiabilidade de cada porta. A Tabela 16 demonstra os valores de MTBF obtidos por meio da diferença da confiabilidade 0.9999 para cada valor de confiabilidade das colunas. Os valores obtidos demonstram que o SPR conseguiu acompanhar os valores de maneira bem próxima ao SPR-MP, principalmente nas variações das portas mais importantes.

Ao fazer uma análise exclusiva das variações percentuais de todos os circuitos da amostragem, é possível observar que o SPR consegue demonstrar, de forma aproximada, tanto os pontos mais sensíveis dos circuitos, quanto o impacto na confiabilidade de todo

Tabela 16: Diferença percentual entre os valores e o valor de MTBF com confiabilidade 0.9999

s838							
Gate	Method	0.9	0.99	0.999	0.99999	0.999999	1
g186	SPR-MP	-99.65%	-96.39%	-70.73%	31.83%	36.17%	36.67%
	SPR	-99.65%	-96.39%	-70.74%	31.87%	36.21%	36.71%
g185	SPR-MP	-99.65%	-96.38%	-70.69%	31.76%	36.08%	36.58%
	SPR	-99.65%	-96.39%	-70.71%	31.79%	36.12%	36.62%
g183	SPR-MP	-99.64%	-96.34%	-70.44%	31.27%	35.51%	35.99%
	SPR	-99.64%	-96.34%	-70.46%	31.31%	35.55%	36.04%
g184	SPR-MP	-99.17%	-92.09%	-51.38%	11.81%	13.15%	13.30%
	SPR	-99.17%	-92.08%	-51.35%	11.80%	13.13%	13.28%
g182	SPR-MP	-97.12%	-76.87%	-23.19%	3.11%	3.43%	3.47%
	SPR	-97.08%	-76.63%	-22.96%	3.07%	3.39%	3.42%
g173	SPR-MP	-97.12%	-76.86%	-23.18%	3.11%	3.43%	3.47%
	SPR	-97.11%	-76.81%	-23.14%	3.10%	3.42%	3.46%
g003	SPR-MP	-89.35%	-45.35%	-7.01%	0.76%	0.84%	0.85%
	SPR	-89.32%	-45.29%	-7.00%	0.76%	0.83%	0.84%
g001	SPR-MP	-80.69%	-29.27%	-3.62%	0.38%	0.42%	0.42%
	SPR	-80.67%	-29.24%	-3.62%	0.38%	0.41%	0.42%
g000	SPR-MP	-51.14%	-9.40%	-0.93%	0.09%	0.10%	0.10%
	SPR	-50.82%	-9.29%	-0.92%	0.09%	0.10%	0.10%
g004	SPR-MP	-51.14%	-9.40%	-0.93%	0.09%	0.10%	0.10%
	SPR	-50.82%	-9.29%	-0.92%	0.09%	0.10%	0.10%

o circuito que estes pontos geram, dos 16 circuitos analisados foram retirados os dois melhores e os dois piores casos com relação ao percentual de incremento na confiabilidade de cada porta, considerando o incremento de 0.9999 para 0.99999. Os gráficos podem ser observados nas Figuras 23 e 24. Um dos circuitos relacionados como “pior caso” é o s27, o qual é um dos dois únicos circuitos que possuem mais do que uma saída. O s208 é o outro circuito que também está completo. Contudo, como esta última análise exigiu um número de execuções do SPR-MP, no mínimo, igual ao número de portas do circuito, não foi possível realizar a análise sobre o s208. Assim sendo, um outro possível caminho a seguir nas pesquisas seria identificar quais características dos circuitos mais impactam na imprecisão do SPR.

Por outro lado, mesmo nos piores casos das simulações, é possível notar uma tendência do SPR em relação ao SPR-MP. O que confirma, pelo menos em circuitos compatíveis com as características dos analisados neste trabalho que, o SPR pode ser utilizado como uma métrica de confiabilidade, pois ele consegue identificar tanto os pontos mais sensíveis do circuito, quanto o incremento ou decremento da confiabilidade geral do circuito quando há alterações nestes pontos, na mesma proporção que os resultados identificados com o método SPR-MP

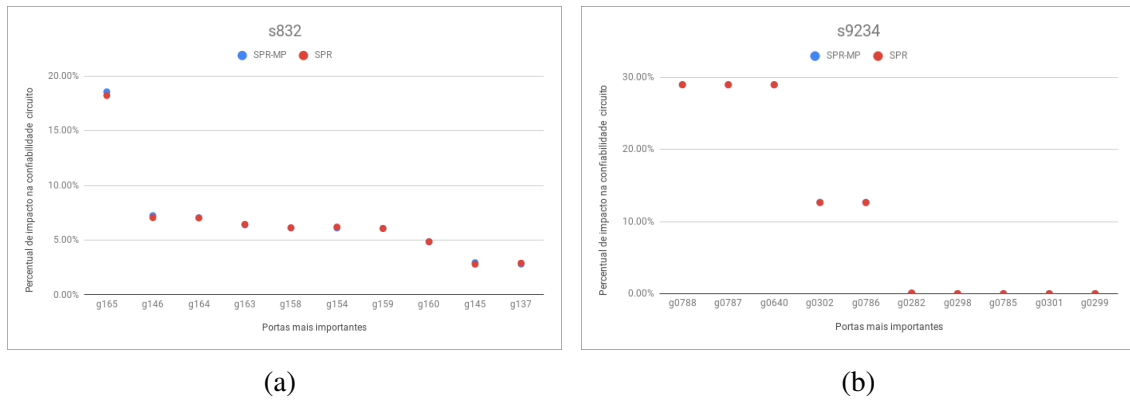


Figura 23: Percentual de incremento no MTBF do circuito, melhores casos: (a) Circuito s832 e (b) Circuito s9234

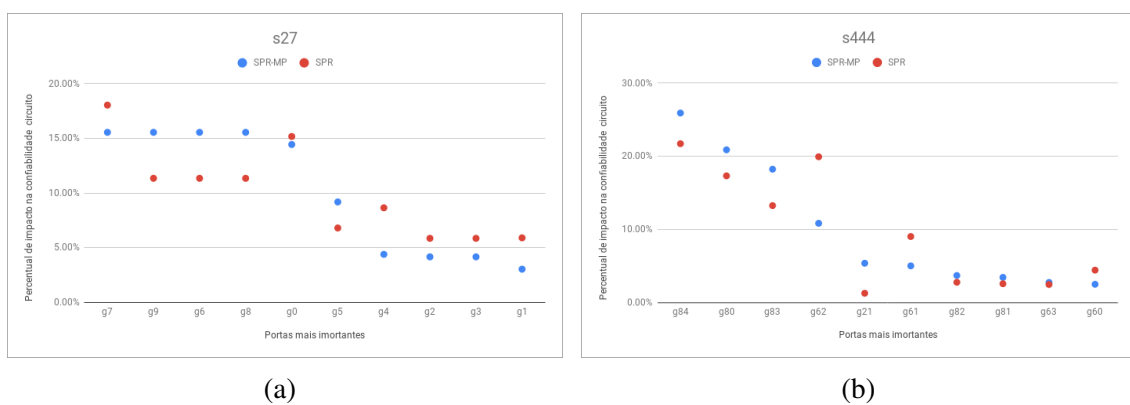


Figura 24: Percentual de incremento no MTBF do circuito, piores casos: (a) Circuito s27 e (b) Circuito s444

6 CONSIDERAÇÕES FINAIS

Esse trabalho teve como objetivo trazer uma revisão dos métodos analíticos que estimam a confiabilidade de um circuito digital. Para tanto, foram realizadas implementações que possibilitaram verificar as vantagens e desvantagens dos métodos PTM, SPR e SPR-MP. O método PTM é um método para estimar a confiabilidade de um circuito, o qual é, por definição, exato. Contudo, foi demonstrado neste trabalho que a abordagem PTM, em sua forma original, é impraticável, pois para a representação das matrizes em sua totalidade, existe um problema exponencial que pode ser observado mesmo em circuitos pequenos, como é o caso dos presentes no *benchmark* ISCAS85. Por outro lado, foi idealizada uma forma de lidar com o problema de memória, tornando o cálculo das probabilidades da matriz final um processo sequencial de acumulação. Notou-se que o problema mudou de forma, mas não foi resolvido, ou seja, o problema exponencial não afetou a memória mas sim o tempo de processamento.

O método SPR-MP é uma variação do SPR que lida com *fanouts*. É uma abordagem flexível que pode gerar resultados exatos, a um custo temporal compatível. Mas este trabalho demonstrou que, considerando determinados *fanouts*, o tempo de processamento pode ser reduzido em grandes quantidades, afetando a confiabilidade final de maneira sucinta. Além disso, o método se mostrou superior, em termos de desempenho, ao PTM sequencial. Assim, o SPR-MP pode ser utilizado para encontrar valores exatos de circuitos um pouco maiores dos que o método PTM. Além disso, tendo em vista que os ciclos do SPR-MP são independentes entre si, é possível utilizar técnicas de paralelismo para tornar a abordagem mais eficiente e ampliar ainda mais o tamanho dos circuitos analisados.

A confiabilidade gerada pelo método SPR não é exata, visto que os caminhos reconvergentes não são considerados pelo método. Porém, o desempenho demonstrado nas análises o torna uma possibilidade viável a ser explorada. Foram destacados neste trabalho resultados que apontam uma tendência do SPR, principalmente na identificação de portas que mais impactam na confiabilidade final do circuito. Além disso, pôde ser notado que o SPR consegue observar o impacto que o aumento de confiabilidade do circuito, quando ocorre um aumento de confiabilidade nas portas que mais geram impacto na confiabilidade final. Talvez a amostragem de circuitos não tenha sido suficiente para validar o

comportamento do SPR. Visto que, um dos casos que mais se distinguiu do SPR-MP foi o circuito s27, o qual era um dos únicos que possuíam mais saídas. Contudo, o problema de escalabilidade para circuitos maiores ainda persiste. Assim, pesquisar formas de ampliar a estimativa de confiabilidade exata em circuitos maiores se faz necessário.

Os estudos desenvolvidos neste trabalho geraram uma publicação e uma contribuição em outro trabalho, as quais podem ser observadas logo abaixo:

- ***The Suitability of the SPR-MP Method to Evaluate the Reliability of Logic Circuits*** (PONTES et al., 2018)
- ***Reliability evaluation of circuits designed in multi-and single-stage versions*** (SCHVITZ et al., 2018)

REFERÊNCIAS

ALBRECHT, C. IWLS 2005 benchmarks. In: INTERNATIONAL WORKSHOP FOR LOGIC SYNTHESIS (IWLS): [HTTP://WWW. IWLS. ORG](http://www.iwls.org), 2005. ... [S.l.: s.n.], 2005.

AVIZIENIS, A.; LAPRIE, J.-C.; RANDELL, B. et al. **Fundamental concepts of dependability**. [S.l.]: University of Newcastle upon Tyne, Computing Science, 2001.

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.

BHADURI, D.; SHUKLA, S. K.; GRAHAM, P. S.; GOKHALE, M. B. Reliability analysis of large circuits using scalable techniques and tools. **IEEE Transactions on Circuits and Systems I: Regular Papers**, [S.l.], v.54, n.11, p.2447–2460, 2007.

BIROLINI, A. **Quality and reliability of technical systems: theory, practice, management**. [S.l.]: Springer Science & Business Media, 2012.

BLOCH, J. **Effective java**. [S.l.]: Pearson Education India, 2016.

BRGLEZ, F.; BRYAN, D.; KOZMINSKI, K. Notes on the ISCAS'89 Benchmark Circuits. **North-Carolina State University**, [S.l.], 1989.

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.

BRUEGGE, B.; DUTOIT, A. H. **Object-Oriented Software Engineering Using UML, Patterns and Java-(Required)**. [S.l.]: Prentice Hall, 2004. v.2004.

BUTZEN, P. F. Aging aware design techniques and CMOS gate degradation estimative. , [S.l.], 2012.

CHOUDHURY, M. R.; MOHANRAM, K. Accurate and scalable reliability analysis of logic circuits. In: DESIGN, AUTOMATION AND TEST IN EUROPE, 2007. **Proceedings...** [S.l.: s.n.], 2007. p.1454–1459.

CZUTRO, A. **Efficiency and Applications of SAT-Based Test Pattern Generation**. 2013. Tese (Doutorado em Ciência da Computação) — Dissertation, Technischen Fakultät der Albert-Ludwigs-Universität in . . .

EJML. **EJML Efficient Java Matrix Library**.

FINKELSTEIN, M. **Failure rate modelling for reliability and risk**. [S.l.]: Springer Science & Business Media, 2008.

FRANCO, D. T. **Fiabilité du Signal des Circuits Logiques Combinatoires sous Fautes Simultanées Multiples**. 2008. Tese (Doutorado em Ciência da Computação) — .

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.

FRITZ, R. I. **Desenvolvimento de Ferramenta para Cálculo da Confiabilidade de Circuitos Combinacionais Utilizando o Método PTM**. 2017. dissertation — Universidade Federal do Rio Grande.

GEORGE, N.; LACH, J. Characterization of logical masking and error propagation in combinational circuits and effects on system vulnerability. , [S.l.], 2011.

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.], v.63, n.6, p.1336–1350, 2014.

HARRISON, R. L. Introduction to monte carlo simulation. In: **AIP CONFERENCE PROCEEDINGS, 2010. . . .** [S.l.: s.n.], 2010. v.1204, n.1, p.17–21.

HASAN, O.; PATEL, J.; TAHAR, S. Formal reliability analysis of combinational circuits using theorem proving. **Journal of Applied Logic**, [S.l.], v.9, n.1, p.41–60, 2011.

HSUEH, M.-C.; TSAI, T. K.; IYER, R. K. Fault injection techniques and tools. **Computer**, [S.l.], v.30, n.4, p.75–82, 1997.

ICHINOMIYA, Y.; TANOUE, S.; AMAGASAKI, M.; IIDA, M.; KUGA, M.; SU-EYOSHI, T. Improving the robustness of a softcore processor against SEUs by using

TMR and partial reconfiguration. In: FIELD-PROGRAMMABLE CUSTOM COMPUTING MACHINES (FCCM), 2010 18TH IEEE ANNUAL INTERNATIONAL SYMPOSIUM ON, 2010. ... [S.l.: s.n.], 2010. p.47–54.

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

KRAVETS, V. N. **Constructive multi-level synthesis by way of functional properties**. [S.l.]: University of Michigan, 2001.

KRISHNASWAMY, S.; MARKOV, I. L.; HAYES, J. P. On the role of timing masking in reliable logic circuit design. In: DESIGN AUTOMATION CONFERENCE, 45., 2008. **Proceedings...** [S.l.: s.n.], 2008. p.924–929.

KRISHNASWAMY, S.; VIAMONTES, G. F.; MARKOV, I. L.; HAYES, J. P. Accurate reliability evaluation and enhancement via probabilistic transfer matrices. **Proceedings -Design, Automation and Test in Europe, DATE '05**, [S.l.], v.I, p.282–287, 2005.

KWIATKOWSKA, M.; NORMAN, G.; PARKER, D. PRISM 2.0: A tool for probabilistic model checking. In: QUANTITATIVE EVALUATION OF SYSTEMS, 2004. QEST 2004. PROCEEDINGS. FIRST INTERNATIONAL CONFERENCE ON THE, 2004. ... [S.l.: s.n.], 2004. p.322–323.

LALA, P. K. **Self-checking and fault-tolerant digital design**. [S.l.]: Morgan Kaufmann, 2001.

LIENIG, J.; BRUEMMER, H. Design Process and Its Fundamentals. In: **Fundamentals of Electronic Systems Design**. [S.l.]: Springer, 2017. p.5–30.

LIU, B.; CAI, L. Monte Carlo Reliability Model for Single-Event Transient on Combinational Circuits. **IEEE Transactions on Nuclear Science**, [S.l.], v.64, n.12, p.2933–2937, 2017.

MISKOV-ZIVANOV, N.; MARCULESCU, D. Circuit reliability analysis using symbolic techniques. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, [S.l.], v.25, n.12, p.2638–2649, 2006.

MOORE, G. Moore's law. **Electronics Magazine**, [S.l.], v.38, n.8, p.114, 1965.

PAGLIARINI, S. N. **Reliability analysis methods and improvement techniques applicable to digital circuits**. 2013. Tese (Doutorado em Ciência da Computação) — Télécom ParisTech.

PARTRIDGE, J.; HALL, E. C.; HANLEY, L. D. The application of failure analysis in procuring and screening of integrated circuits. **Physics of Failure in Electronics**, [S.l.], v.4, p.95–139, 1965.

PATEL, K.; HAYES, J.; MARKOV, I. Evaluating circuit reliability under probabilistic gate-level fault models. **Proceedings of the International Workshop on Logic and Synthesis**, [S.l.], p.59–64, 2003.

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.

RAMANARAYANAN, R.; DEGALAHAL, V. S.; KRISHNAN, R.; KIM, J.; NARAYANAN, V.; XIE, Y.; IRWIN, M. J.; UNLU, K. Modeling soft errors at the device and logic levels for combinational circuits. **IEEE Transactions on Dependable and Secure Computing**, [S.l.], v.6, n.3, p.202–216, 2009.

SCHVITZ, R.; PONTES, M.; MEINHARDT, C.; FRANCO, D. T.; NAVINER, L.; ROSA, L. da; BUTZEN, P. F. Reliability evaluation of circuits designed in multi-and single-stage versions. In: IEEE 9TH LATIN AMERICAN SYMPOSIUM ON CIRCUITS & SYSTEMS (LASCAS), 2018., 2018. ... [S.l.: s.n.], 2018. p.1–4.

SIEWIOREK, D.; SWARZ, R. **Reliable Computer Systems: Design and Evaluation**. [S.l.]: Digital Press, 2017.

XIAO, J.; LEE, W.; YANG, X.; HU, H.; HUANG, Y. A Method of Gate-level Circuit Yield Calculation Based on PTM. **Procedia Computer Science**, [S.l.], v.107, n.Icict, p.674–684, 2017.

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

APÊNDICE A AMOSTRA DE CIRCUITOS DA PRIMEIRA ANÁLISE

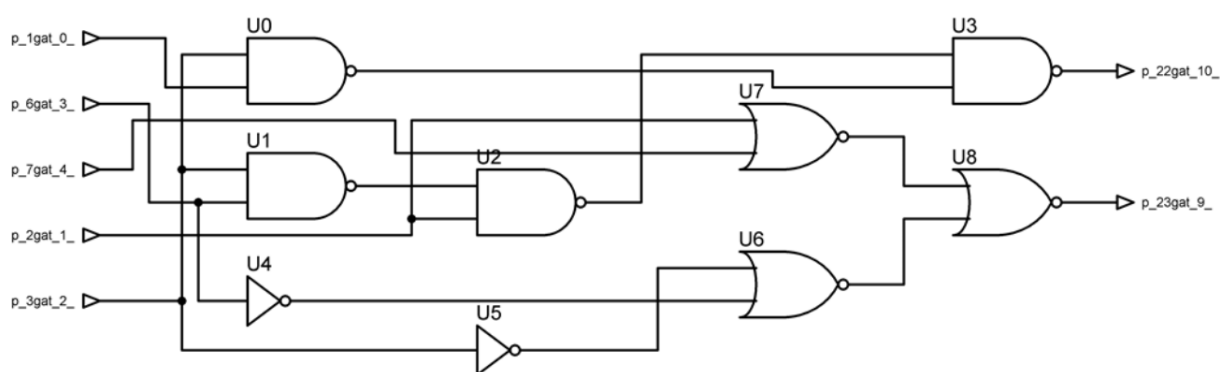


Figura 25: C17 - Versão 1

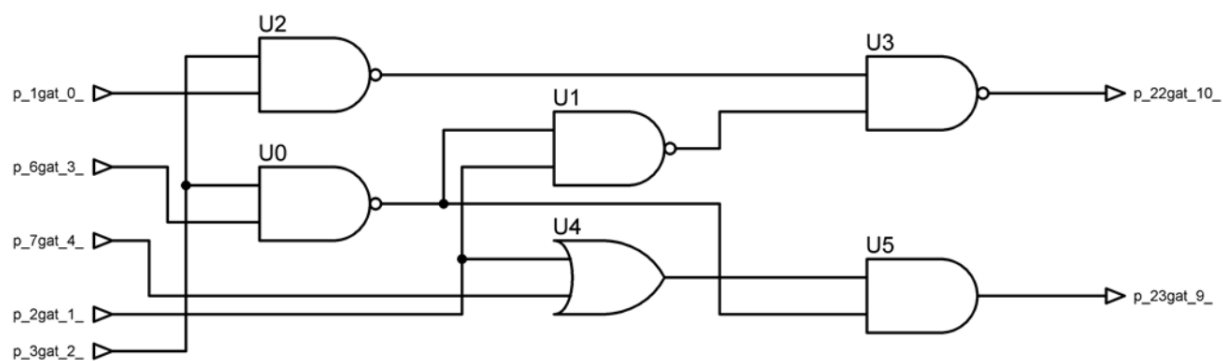


Figura 26: C17 - Versão 2

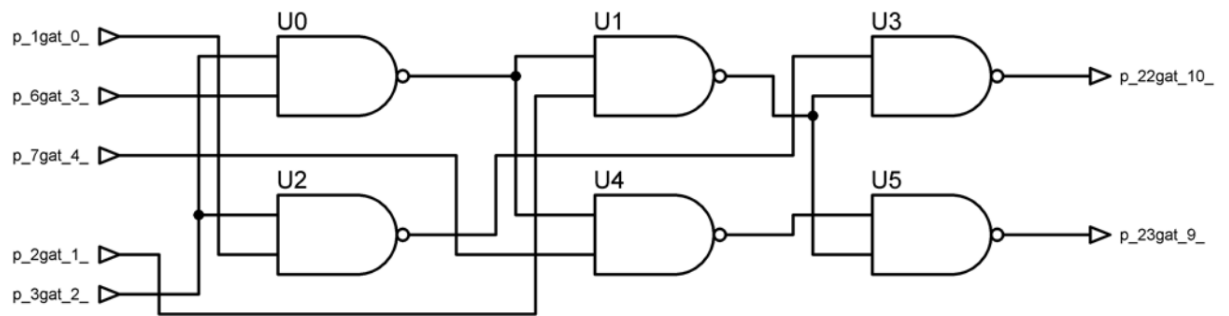


Figura 27: C17 - Versão 3

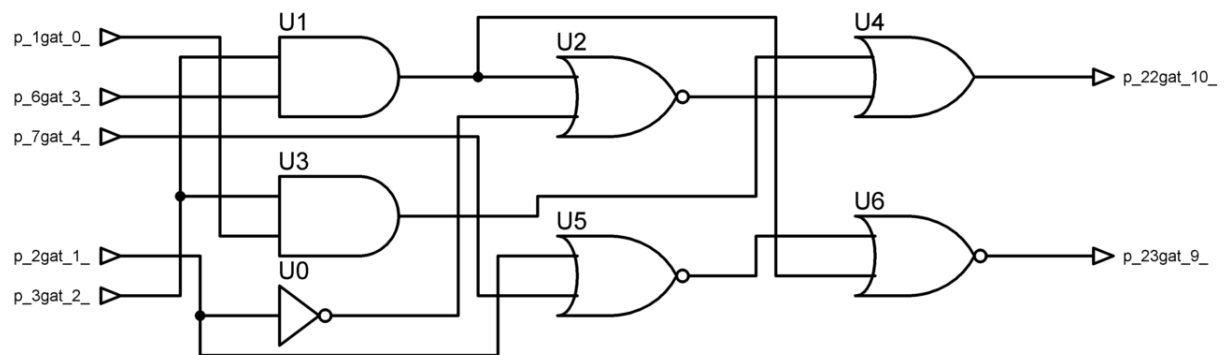


Figura 28: C17 - Versão 4

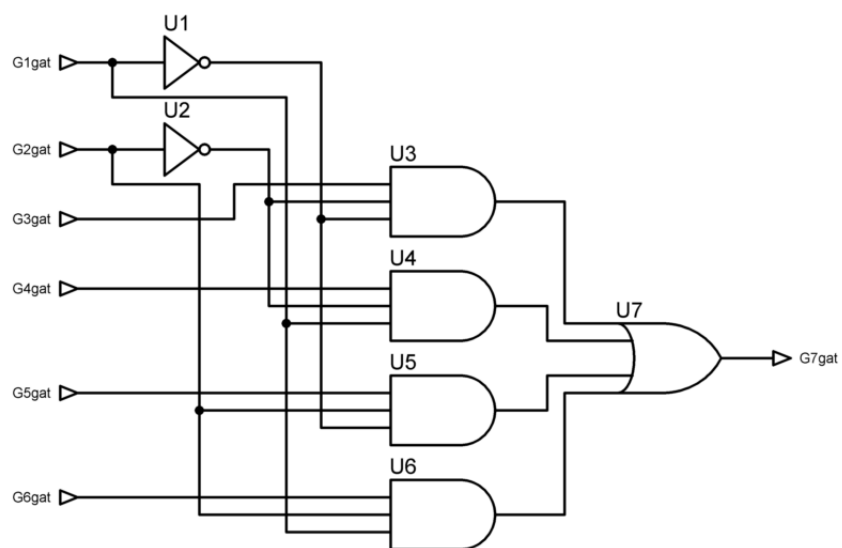


Figura 29: Multiplexador 4-bits

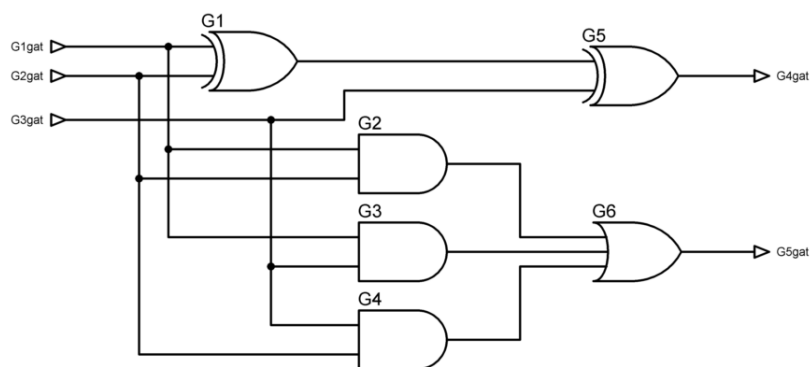


Figura 30: Somador Completo - Versão 1

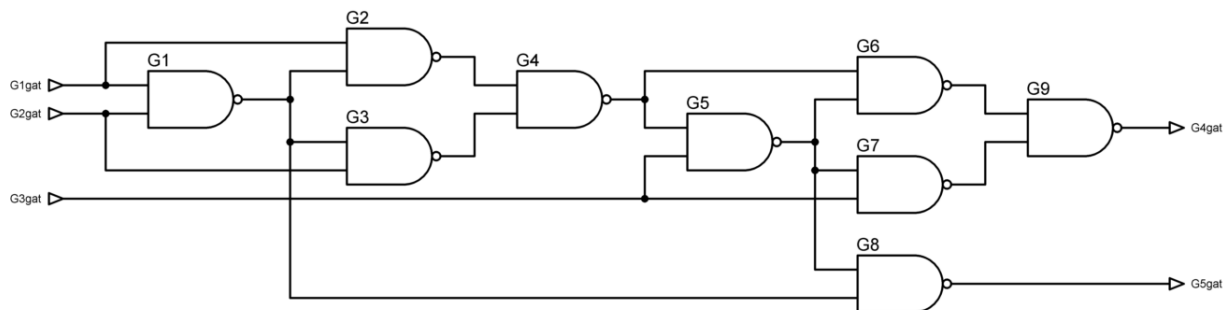


Figura 31: Somador Completo - Versão 2

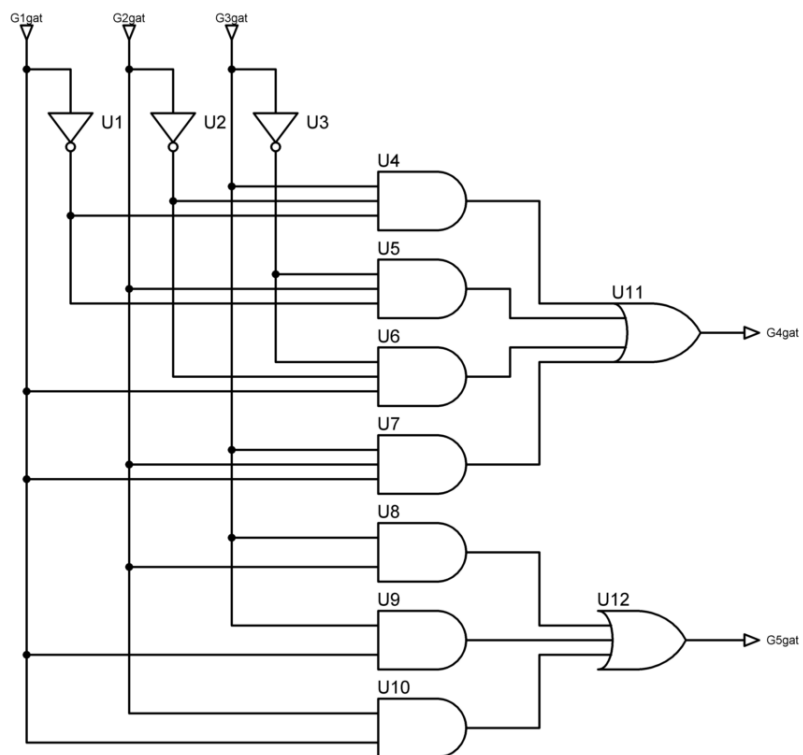


Figura 32: Somador Completo - Versão 3

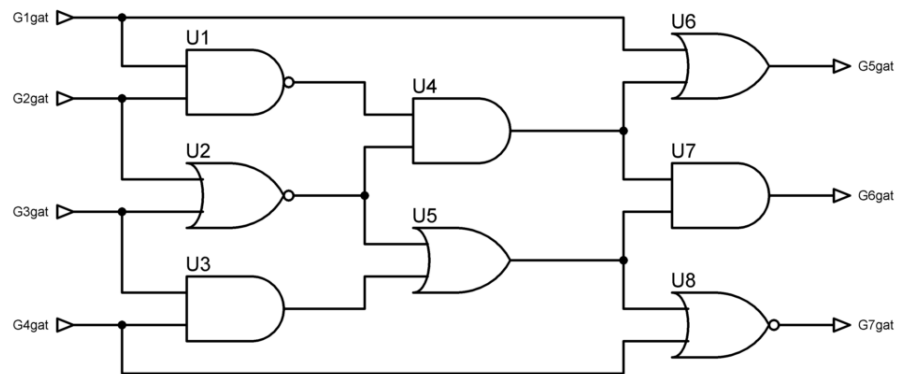


Figura 33: C8

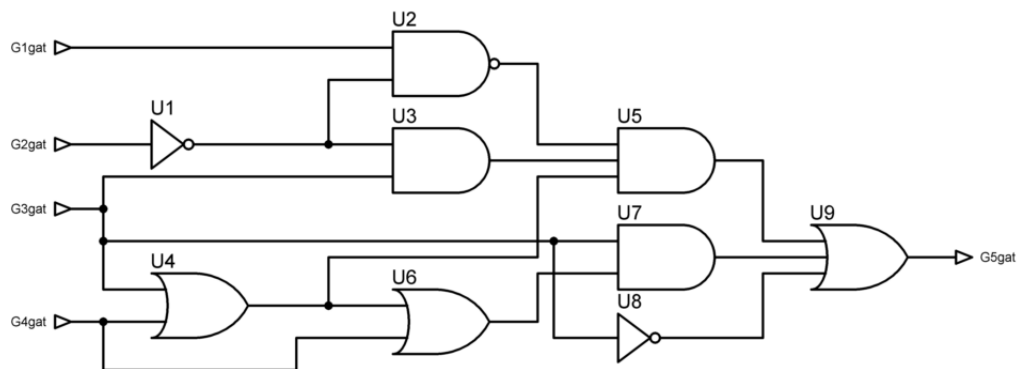


Figura 34: C9

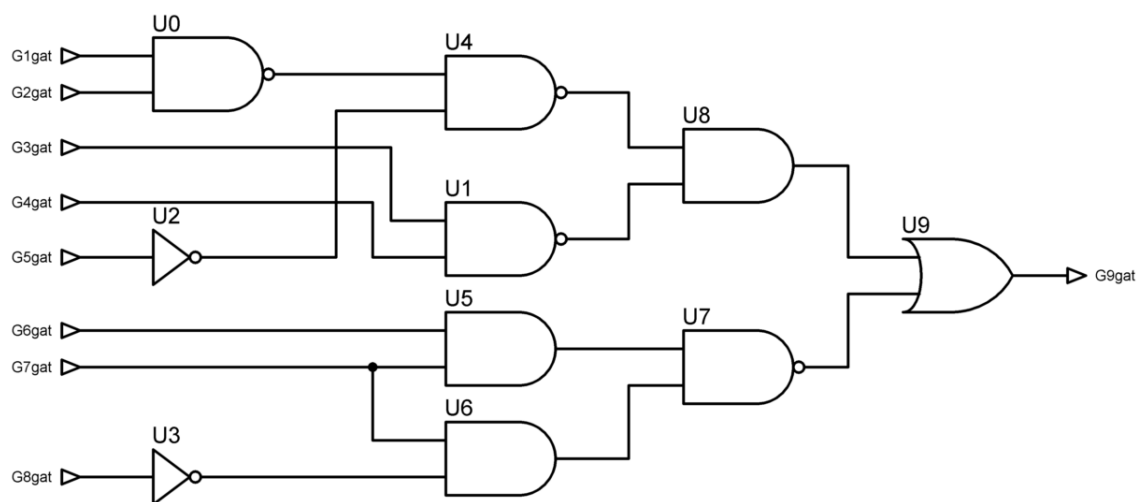


Figura 35: C10

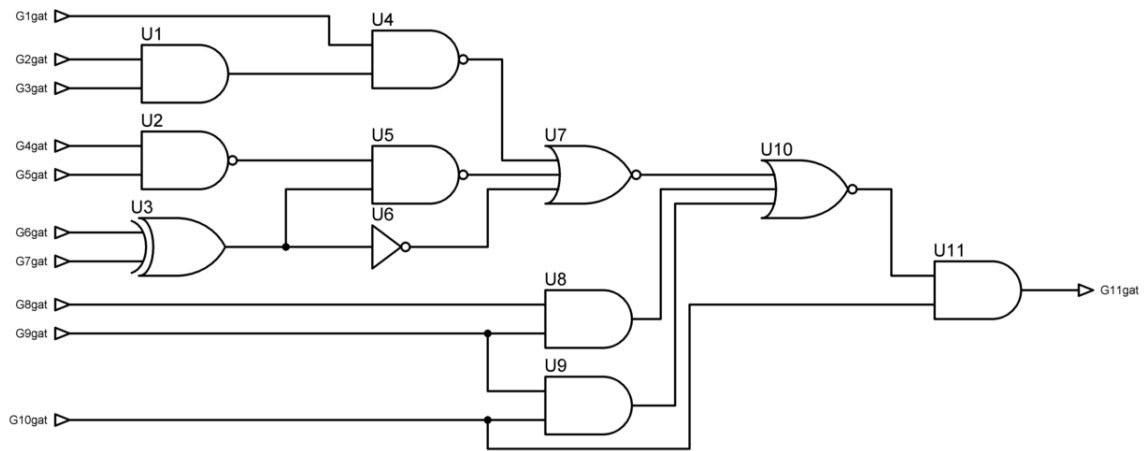


Figura 36: C11

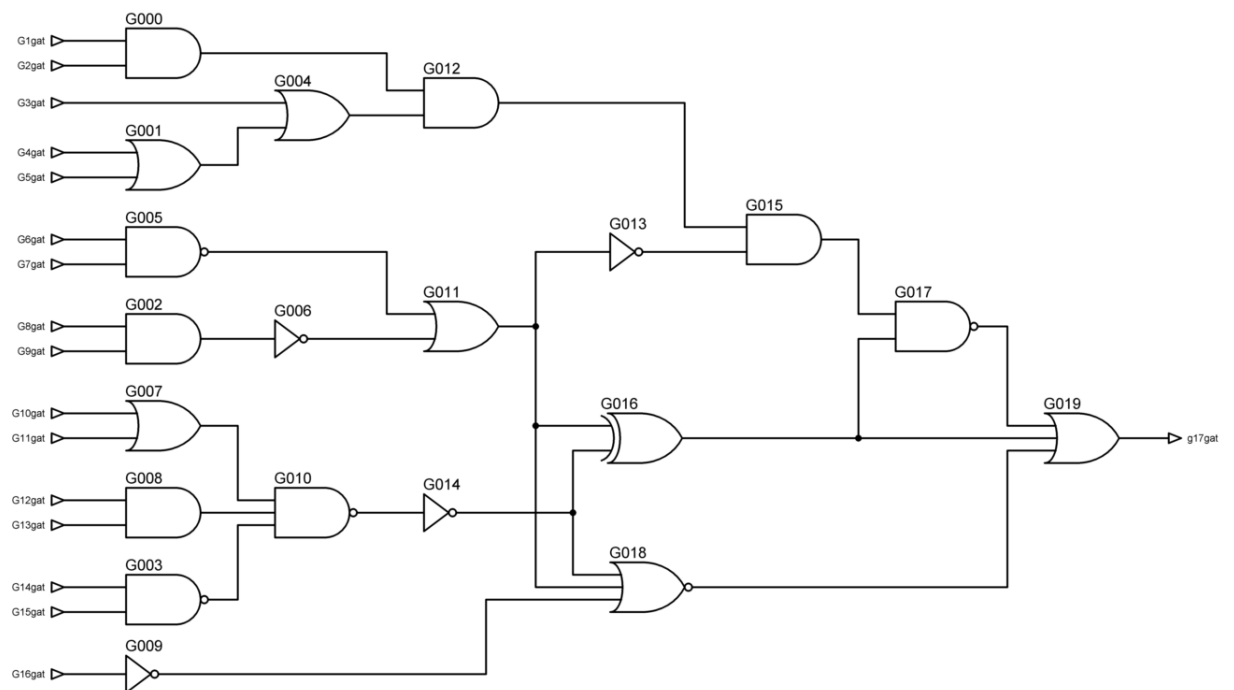


Figura 37: C20

APÊNDICE B DESCRIÇÃO VERILOG DOS CIRCUITOS DA SEGUNDA AMOSTRAGEM

```
//Converted to Combinational , Module name: s27
module s27 ( clock, G0, G1, G2, G3, G5, G6, G7, G17, n11, n16, n21 );
input clock, G0, G1, G2, G3, G5, G6, G7;
output G17, n11, n16, n21;
wire n14, n15, n16g1, n17, n18, n19;
INVX1 g0(.A(G0), .Y(n14));
AOI21X1 g1(.A0(G6), .A1(n14), .B0(G3), .Y(n15));
INVX1 g2(.A(G1), .Y(n16g1));
INVX1 g3(.A(G7), .Y(n17));
AOI22X1 g4(.A0(G6), .A1(n14), .B0(n16g1), .B1(n17), .Y(n18));
OR2X1 g5(.A(n18), .B(G5), .Y(n19));
OR2X1 g6(.A(n19), .B(n15), .Y(G17));
NOR3X1 g7(.A(n18), .B(n15), .C(G5), .Y(n16));
NOR2X1 g8(.A(n16), .B(n14), .Y(n11));
AOI21X1 g9(.A0(n17), .A1(n16g1), .B0(G2), .Y(n21));
endmodule
```

Figura 38: Descrição do s27

```

//Converted to Combinational , Module name: s208
module s208 ( X, Clear, C_8, C_7, C_6, C_5, C_4, C_3, C_2, C_1, C_0, Y_4, Y_3,
Y_2, Y_1, Y_8, Y_7, Y_6, Y_5, W, Z, n27, n32, n37, n42, n47, n52, n57, n62 );
input X, Clear, C_8, C_7, C_6, C_5, C_4, C_3, C_2, C_1, C_0, Y_4, Y_3, Y_2, Y_1,
Y_8, Y_7, Y_6, Y_5;
output W, Z, n27, n32, n37, n42, n47, n52, n57, n62;
wire n37_1, n38, n40, n41, n42_1, n43, n44, n45, n46, n47_1, n48, n49, n50, n51,
n52_1, n53, n54, n55, n56, n57_1, n58, n59, n60, n61, n62_1, n64, n65, n66, n67,
n68, n70, n71, n72, n74, n75, n78, n79, n80, n81, n83, n84, n85;
INVX1 g00(.A(Y_8), .Y(n37_1));
NAND3X1 g01(.A(Y_5), .B(Y_6), .C(Y_7), .Y(n38));
NOR2X1 g02(.A(n38), .B(n37_1), .Y(W));
INVX1 g03(.A(Y_4), .Y(n40));
INVX1 g04(.A(Y_5), .Y(n41));
INVX1 g05(.A(X), .Y(n42_1));
NOR4X1 g06(.A(Y_2), .B(Y_3), .C(n42_1), .D(Y_1), .Y(n43));
AND2X1 g07(.A(Y_6), .B(C_6), .Y(n44));
NAND4X1 g08(.A(n43), .B(n41), .C(n40), .D(n44), .Y(n45));
NAND4X1 g09(.A(Y_5), .B(n40), .C(C_5), .D(n43), .Y(n46));
NAND3X1 g10(.A(n43), .B(Y_4), .C(C_4), .Y(n47_1));
NAND2X1 g11(.A(Y_3), .B(C_3), .Y(n48));
NOR4X1 g12(.A(Y_1), .B(Y_2), .C(n42_1), .D(n48), .Y(n49));
NAND2X1 g13(.A(Y_2), .B(C_2), .Y(n50));
NOR3X1 g14(.A(n50), .B(Y_1), .C(n42_1), .Y(n51));
NAND2X1 g15(.A(C_0), .B(X), .Y(n52_1));
NAND3X1 g16(.A(Y_1), .B(C_1), .C(X), .Y(n53));
NAND2X1 g17(.A(n53), .B(n52_1), .Y(n54));
NOR3X1 g18(.A(n54), .B(n51), .C(n49), .Y(n55));
AND2X1 g19(.A(n55), .B(n47_1), .Y(n56));
OR4X1 g20(.A(Y_2), .B(Y_3), .C(n42_1), .D(Y_1), .Y(n57_1));
NOR4X1 g21(.A(Y_5), .B(Y_6), .C(Y_4), .D(n57_1), .Y(n58));
NAND2X1 g22(.A(Y_8), .B(C_8), .Y(n59));
NOR2X1 g23(.A(n59), .B(Y_7), .Y(n60));
AND2X1 g24(.A(Y_7), .B(C_7), .Y(n61));
OAI21X1 g25(.A0(n61), .A1(n60), .B0(n58), .Y(n62_1));
NAND4X1 g26(.A(n56), .B(n46), .C(n45), .D(n62_1), .Y(Z));
INVX1 g27(.A(Clear), .Y(n64));
NAND4X1 g28(.A(Y_2), .B(n64), .C(X), .D(Y_1), .Y(n65));
NAND2X1 g29(.A(Y_3), .B(n40), .Y(n66));
NAND3X1 g30(.A(Y_1), .B(Y_2), .C(Y_3), .Y(n67));
NAND4X1 g31(.A(Y_4), .B(n64), .C(X), .D(n67), .Y(n68));
OAI21X1 g32(.A0(n66), .A1(n65), .B0(n68), .Y(n27));
INVX1 g33(.A(Y_3), .Y(n70));
AND2X1 g34(.A(Y_1), .B(Y_2), .Y(n71));
OR4X1 g35(.A(n70), .B(Clear), .C(n42_1), .D(n71), .Y(n72));
OAI21X1 g36(.A0(n65), .A1(Y_3), .B0(n72), .Y(n32));
INVX1 g37(.A(Y_1), .Y(n74));
NOR3X1 g38(.A(n74), .B(Clear), .C(n42_1), .Y(n75));
NOR3X1 g39(.A(Y_1), .B(Clear), .C(n42_1), .Y(n42));
MX2X1 g40(.A(n75), .B(n42), .S0(Y_2), .Y(n37));
NAND4X1 g41(.A(Y_3), .B(Y_4), .C(n64), .D(n71), .Y(n78));
NOR4X1 g42(.A(n41), .B(n40), .C(Clear), .D(n67), .Y(n79));
NAND4X1 g43(.A(Y_6), .B(Y_7), .C(n37_1), .D(n79), .Y(n80));
NAND2X1 g44(.A(n38), .B(Y_8), .Y(n81));
OAI21X1 g45(.A0(n81), .A1(n78), .B0(n80), .Y(n47));
NAND2X1 g46(.A(n79), .B(Y_6), .Y(n83));
NAND2X1 g47(.A(Y_5), .B(Y_6), .Y(n84));
NAND2X1 g48(.A(n84), .B(Y_7), .Y(n85));
OAI22X1 g49(.A0(n83), .A1(Y_7), .B0(n78), .B1(n85), .Y(n52));
NOR4X1 g50(.A(Y_5), .B(n40), .C(Clear), .D(n67), .Y(n62));
MX2X1 g51(.A(n79), .B(n62), .S0(Y_6), .Y(n57));
endmodule

```

Figura 39: Descrição do s208

```

//Converted to Combinational (Partial output: n64) , Module name: s298_n64
module s298_n64 ( G19, G10, G13, G14, G15, G22, G12, G11, n64 );
input G19, G10, G13, G14, G15, G22, G12, G11;
output n64;
wire n102, n106, n88, n53, n71, n82, n105, n87, n84_1, n85, n86, n81, n78, n80,
n104, n61, n58, n68, n79_1, n59_1, n103;
NOR2X1 g56(.A(n106), .B(n102), .Y(n64));
NOR4X1 g51(.A(G19), .B(n71), .C(n53), .D(n88), .Y(n102));
MX2X1 g55(.A(G10), .B(n105), .S0(n82), .Y(n106));
AOI22X1 g37(.A0(n86), .A1(n85), .B0(n84_1), .B1(n87), .Y(n88));
INVX1 g02(.A(G13), .Y(n53));
INVX1 g20(.A(G14), .Y(n71));
OAI22X1 g31(.A0(n80), .A1(n78), .B0(G15), .B1(n81), .Y(n82));
AOI21X1 g54(.A0(n61), .A1(G14), .B0(n104), .Y(n105));
NAND4X1 g36(.A(G22), .B(n71), .C(G13), .D(n61), .Y(n87));
INVX1 g33(.A(G15), .Y(n84_1));
NOR2X1 g34(.A(G12), .B(n58), .Y(n85));
NOR3X1 g35(.A(G22), .B(G14), .C(n53), .Y(n86));
NOR4X1 g30(.A(n79_1), .B(G14), .C(n53), .D(n68), .Y(n81));
NAND2X1 g27(.A(n59_1), .B(G11), .Y(n78));
NAND3X1 g29(.A(n79_1), .B(n71), .C(G13), .Y(n80));
NAND2X1 g53(.A(n103), .B(n53), .Y(n104));
NOR2X1 g10(.A(G12), .B(G11), .Y(n61));
INVX1 g07(.A(G11), .Y(n58));
OR2X1 g17(.A(G12), .B(G11), .Y(n68));
INVX1 g28(.A(G22), .Y(n79_1));
INVX1 g08(.A(G12), .Y(n59_1));
NAND3X1 g52(.A(G19), .B(G14), .C(G12), .Y(n103));
endmodule

```

Figura 40: Descrição do s298 (saída n64)

```

//Converted to Combinational (Partial output: n61) , Module name: s344 n61
module s344 n61 ( START, ACVQN2, CT1, CT2, CT0, AX3, MRVQN0, ACVQN3, AX2, AX1,
AX0, ACVQN1, ACVQN0, n61 );
input START, ACVQN2, CT1, CT2, CT0, AX3, MRVQN0, ACVQN3, AX2, AX1, AX0, ACVQN1,
ACVQN0;
output n61;
wire n82, n103, n87, n102, P6, READY, n86_1, n95, n97, n101_1, n98, n73, n75, n94,
n91_1, n93, P7, P0, n96_1, n92, n90, n88, n89, P5, P4;
NAND2X1 g39(.A(n103), .B(n82), .Y(n61));
INVX1 g17(.A(START), .Y(n82));
MX2X1 g38(.A(P6), .B(n102), .S0(n87), .Y(n103));
NOR2X1 g22(.A(n86_1), .B(READY), .Y(n87));
OAI22X1 g37(.A0(n98), .A1(n101_1), .B0(n97), .B1(n95), .Y(n102));
INVX1 g02(.A(ACVQN2), .Y(P6));
NOR3X1 g13(.A(n75), .B(CT1), .C(n73), .Y(READY));
NOR3X1 g21(.A(CT0), .B(CT1), .C(CT2), .Y(n86_1));
AOI21X1 g30(.A0(n93), .A1(n91_1), .B0(n94), .Y(n95));
NAND3X1 g32(.A(AX3), .B(P0), .C(P7), .Y(n97));
AND2X1 g36(.A(n96_1), .B(n95), .Y(n101_1));
OAI21X1 g33(.A0(n96_1), .A1(n95), .B0(n97), .Y(n98));
INVX1 g08(.A(CT2), .Y(n73));
INVX1 g10(.A(CT0), .Y(n75));
NOR3X1 g29(.A(n92), .B(MRVQN0), .C(ACVQN2), .Y(n94));
OAI21X1 g26(.A0(n89), .A1(n88), .B0(n90), .Y(n91_1));
OAI21X1 g28(.A0(n92), .A1(MRVQN0), .B0(ACVQN2), .Y(n93));
INVX1 g03(.A(ACVQN3), .Y(P7));
INVX1 g04(.A(MRVQN0), .Y(P0));
AOI21X1 g31(.A0(AX3), .A1(P0), .B0(P7), .Y(n96_1));
INVX1 g27(.A(AX2), .Y(n92));
NAND3X1 g25(.A(AX1), .B(P0), .C(P5), .Y(n90));
NAND3X1 g23(.A(AX0), .B(P0), .C(P4), .Y(n88));
AOI21X1 g24(.A0(AX1), .A1(P0), .B0(P5), .Y(n89));
INVX1 g01(.A(ACVQN1), .Y(P5));
INVX1 g00(.A(ACVQN0), .Y(P4));
endmodule

```

Figura 41: Descrição do s344 (saída n61)

```

//Converted to Combinational (Partial output: n66) , Module name: s349 n66
module s349 n66 ( START, ACVQN1, CT1, CT2, CT0, MRVQN0, ACVQN0, ACVQN2, AX1, AX0,
AX2, n66 );
input START, ACVQN1, CT1, CT2, CT0, MRVQN0, ACVQN0, ACVQN2, AX1, AX0, AX2;
output n66;
wire n83, n118, n89, n117, P5, READY, n88, n112, n113, n116, n115, n65, n67,
n111_1, n108, n110, n96_1, n93, n95, n114, n109, n107, n94, n92, n90, n91_1, P0,
P4;
NAND2X1 g53(.A(n118), .B(n83), .Y(n66));
INVX1 g18(.A(START), .Y(n83));
MX2X1 g52(.A(P5), .B(n117), .S0(n89), .Y(n118));
NOR2X1 g23(.A(n88), .B(READY), .Y(n89));
OAI22X1 g51(.A0(n115), .A1(n116), .B0(n113), .B1(n112), .Y(n117));
INVX1 g11(.A(ACVQN1), .Y(P5));
NOR3X1 g05(.A(n67), .B(CT1), .C(n65), .Y(READY));
NOR3X1 g22(.A(CT0), .B(CT1), .C(CT2), .Y(n88));
AOI21X1 g46(.A0(n110), .A1(n108), .B0(n111_1), .Y(n112));
INVX1 g47(.A(n96_1), .Y(n113));
NOR2X1 g50(.A(n95), .B(n93), .Y(n116));
OAI21X1 g49(.A0(n114), .A1(n112), .B0(n113), .Y(n115));
INVX1 g00(.A(CT2), .Y(n65));
INVX1 g02(.A(CT0), .Y(n67));
NOR3X1 g45(.A(n109), .B(MRVQN0), .C(ACVQN1), .Y(n111_1));
NOR3X1 g42(.A(n107), .B(MRVQN0), .C(ACVQN0), .Y(n108));
OAI21X1 g44(.A0(n109), .A1(MRVQN0), .B0(ACVQN1), .Y(n110));
NOR3X1 g30(.A(n94), .B(MRVQN0), .C(ACVQN2), .Y(n96_1));
OAI21X1 g27(.A0(n91_1), .A1(n90), .B0(n92), .Y(n93));
OAI21X1 g29(.A0(n94), .A1(MRVQN0), .B0(ACVQN2), .Y(n95));
INVX1 g48(.A(n95), .Y(n114));
INVX1 g43(.A(AX1), .Y(n109));
INVX1 g41(.A(AX0), .Y(n107));
INVX1 g28(.A(AX2), .Y(n94));
NAND3X1 g26(.A(AX1), .B(P0), .C(P5), .Y(n92));
NAND3X1 g24(.A(AX0), .B(P0), .C(P4), .Y(n90));
AOI21X1 g25(.A0(AX1), .A1(P0), .B0(P5), .Y(n91_1));
INVX1 g06(.A(MRVQN0), .Y(P0));
INVX1 g10(.A(ACVQN0), .Y(P4));
endmodule

```

Figura 42: Descrição do s349 (saída n66)

```

//Converted to Combinational (Partial output: n69) , Module name: s382_n69
module s382_n69 ( C3_Q2, CLR, C3_Q3, C3_Q1, C3_Q0, TESTL, UC_8, UC_9, UC_10,
UC_11, UC_16, UC_17, UC_18, UC_19, n69 );
input C3_Q2, CLR, C3_Q3, C3_Q1, C3_Q0, TESTL, UC_8, UC_9, UC_10, UC_11, UC_16,
UC_17, UC_18, UC_19;
output n69;
wire n126, n128, n130, n74_1, n124, n125, n86, n79_1, n80, n129, n122, n118, n121,
n96, n115, n116, n117, n119_1, n120;
NOR3X1 g59(.A(n130), .B(n128), .C(n126), .Y(n69));
OAI21X1 g54(.A0(n125), .A1(n124), .B0(n74_1), .Y(n126));
NOR4X1 g56(.A(n80), .B(n79_1), .C(n86), .D(n124), .Y(n128));
AOI21X1 g58(.A0(n122), .A1(n129), .B0(C3_Q2), .Y(n130));
INVX1 g02(.A(CLR), .Y(n74_1));
NAND2X1 g52(.A(n121), .B(n118), .Y(n124));
OAI21X1 g53(.A0(n96), .A1(C3_Q2), .B0(C3_Q3), .Y(n125));
INVX1 g14(.A(C3_Q2), .Y(n86));
INVX1 g07(.A(C3_Q1), .Y(n79_1));
INVX1 g08(.A(C3_Q0), .Y(n80));
AND2X1 g57(.A(n121), .B(n118), .Y(n129));
AND2X1 g50(.A(C3_Q0), .B(C3_Q1), .Y(n122));
OAI21X1 g46(.A0(n117), .A1(n116), .B0(n115), .Y(n118));
NOR2X1 g49(.A(n120), .B(n119_1), .Y(n121));
OR2X1 g24(.A(C3_Q0), .B(C3_Q1), .Y(n96));
INVX1 g43(.A(TESTL), .Y(n115));
INVX1 g44(.A(UC_8), .Y(n116));
NOR3X1 g45(.A(UC_11), .B(UC_10), .C(UC_9), .Y(n117));
INVX1 g47(.A(UC_16), .Y(n119_1));
NOR3X1 g48(.A(UC_19), .B(UC_18), .C(UC_17), .Y(n120));

endmodule

```

Figura 43: Descrição do s382 (saída n69)

```

//Converted to Combinational (Partial output: n49) , Module name: s386_n49
module s386_n49 ( v0, v1, v10, v9, v12, v7, v11, v8, v3, v2, v5, v4, n49 );
input v0, v1, v10, v9, v12, v7, v11, v8, v3, v2, v5, v4;
output n49;
wire n81, n113, n116, n70, n71, n33, n108, n112, n34_1, n114, n115, n58, n37,
n107, n111, n109, n110, n36;
AOI21X1 g85(.A0(n116), .A1(n113), .B0(n81), .Y(n49));
NAND3X1 g49(.A(n71), .B(v0), .C(n70), .Y(n81));
OAI21X1 g81(.A0(n112), .A1(n108), .B0(n33), .Y(n113));
OAI21X1 g84(.A0(n115), .A1(n114), .B0(n34_1), .Y(n116));
INVX1 g38(.A(v1), .Y(n70));
NOR2X1 g39(.A(v9), .B(v10), .Y(n71));
INVX1 g01(.A(v12), .Y(n33));
NOR3X1 g76(.A(n107), .B(n37), .C(n58), .Y(n108));
OAI21X1 g80(.A0(n110), .A1(n109), .B0(n111), .Y(n112));
INVX1 g02(.A(v7), .Y(n34_1));
NOR3X1 g82(.A(v8), .B(v11), .C(n33), .Y(n114));
NOR3X1 g83(.A(n37), .B(n36), .C(v12), .Y(n115));
INVX1 g26(.A(v3), .Y(n58));
INVX1 g05(.A(v8), .Y(n37));
MX2X1 g75(.A(v2), .B(v11), .S0(v7), .Y(n107));
NAND4X1 g79(.A(n37), .B(v11), .C(v5), .D(v7), .Y(n111));
AOI21X1 g77(.A0(n37), .A1(v2), .B0(v7), .Y(n109));
NAND2X1 g78(.A(n36), .B(v4), .Y(n110));
INVX1 g04(.A(v11), .Y(n36));

endmodule

```

Figura 44: Descrição do s386 (saída n49)

```

//Converted to Combinational (Partial output: n64) , Module name: s400 n64
module s400_n64 ( C3_Q3, C3_Q1, C3_Q0, C3_Q2, CLR, TESTL, UC_8, UC_9, UC_10,
UC_11, UC_16, UC_17, UC_18, UC_19, n64 );
input C3_Q3, C3_Q1, C3_Q0, C3_Q2, CLR, TESTL, UC_8, UC_9, UC_10, UC_11, UC_16,
UC_17, UC_18, UC_19;
output n64;
wire n123, n126, n128, n122, n93, n86, n121, n124, n125, n74_1, n127, n117, n120,
n96, n114_1, n115, n116, n118, n119_1;
NOR3X1 g57(.A(n128), .B(n126), .C(n123), .Y(n64));
NOR4X1 g51(.A(n121), .B(n86), .C(n93), .D(n122), .Y(n123));
AOI21X1 g54(.A0(n125), .A1(n124), .B0(C3_Q3), .Y(n126));
OAI21X1 g56(.A0(n127), .A1(n121), .B0(n74_1), .Y(n128));
NAND2X1 g50(.A(C3_Q0), .B(C3_Q1), .Y(n122));
INVX1 g21(.A(C3_Q3), .Y(n93));
INVX1 g14(.A(C3_Q2), .Y(n86));
NAND2X1 g49(.A(n120), .B(n117), .Y(n121));
AND2X1 g52(.A(n120), .B(n117), .Y(n124));
NOR2X1 g53(.A(n122), .B(n86), .Y(n125));
INVX1 g02(.A(CLR), .Y(n74_1));
OAI21X1 g55(.A0(n96), .A1(C3_Q2), .B0(C3_Q3), .Y(n127));
OAI21X1 g45(.A0(n116), .A1(n115), .B0(n114_1), .Y(n117));
NOR2X1 g48(.A(n119_1), .B(n118), .Y(n120));
OR2X1 g24(.A(C3_Q0), .B(C3_Q1), .Y(n96));
INVX1 g42(.A(TESTL), .Y(n114_1));
INVX1 g43(.A(UC_8), .Y(n115));
NOR3X1 g44(.A(UC_11), .B(UC_10), .C(UC_9), .Y(n116));
INVX1 g46(.A(UC_16), .Y(n118));
NOR3X1 g47(.A(UC_19), .B(UC_18), .C(UC_17), .Y(n119_1));

endmodule

```

Figura 45: Descrição do s400 (saída n64)

```

//Converted to Combinational (Partial output: Z) , Module name: s420_Z
module s420_Z ( Y_1, Y_13, Y_12, Y_11, Y_10, C_1, C_14, Y_14, C_15, Y_15, C_8,
Y_8, C_13, C_12, C_11, C_16, Y_16, Y_4, Y_6, Y_5, Y_7, Y_9, C_10, C_9, Y_3, Y_2,
X, C_0, C_2, C_7, C_6, C_5, C_4, C_3, Z );
input Y_1, Y_13, Y_12, Y_11, Y_10, C_1, C_14, Y_14, C_15, Y_15, C_8, Y_8, C_13,
C_12, C_11, C_16, Y_16, Y_4, Y_6, Y_5, Y_7, Y_9, C_10, C_9, Y_3, Y_2, X, C_0, C_2,
C_7, C_6, C_5, C_4, C_3;
output Z;
wire n82, n112, n117, n81, n72, n73_1, n111, n84, n86, n88_1, n113_1, n114, n116,
n80, n110, n90, n92, n93_1, n83_1, n85, n87, n115, n79, n74, n75, n76, n109, n95,
n97, n99, n89, n91, n78_1, n108_1, n101, n105, n106, n94, n96, n98_1, n77, n107,
n100, n104, n102, n103_1;
NAND3X1 g49(.A(n117), .B(n112), .C(n82), .Y(Z));
OR4X1 g13(.A(Y_13), .B(n73_1), .C(n72), .D(n81), .Y(n82));
NOR4X1 g43(.A(n88_1), .B(n86), .C(n84), .D(n111), .Y(n112));
OAI21X1 g48(.A0(n116), .A1(n114), .B0(n113_1), .Y(n117));
OR4X1 g12(.A(Y_10), .B(Y_11), .C(Y_12), .D(n80), .Y(n81));
INVX1 g03(.A(C_14), .Y(n72));
INVX1 g04(.A(Y_14), .Y(n73_1));
NAND4X1 g42(.A(n93_1), .B(n92), .C(n90), .D(n110), .Y(n111));
NOR2X1 g15(.A(n83_1), .B(n81), .Y(n84));
NOR4X1 g17(.A(n80), .B(Y_10), .C(Y_11), .D(n85), .Y(n86));
NOR3X1 g19(.A(n87), .B(n80), .C(Y_10), .Y(n88_1));
NOR3X1 g44(.A(n81), .B(Y_13), .C(Y_14), .Y(n113_1));
AND2X1 g45(.A(Y_15), .B(C_15), .Y(n114));
NOR2X1 g47(.A(n115), .B(Y_15), .Y(n116));
NAND4X1 g11(.A(n76), .B(n75), .C(n74), .D(n79), .Y(n80));
NOR4X1 g41(.A(n99), .B(n97), .C(n95), .D(n109), .Y(n110));
OR2X1 g21(.A(n89), .B(n80), .Y(n90));
NAND4X1 g23(.A(n79), .B(n75), .C(n74), .D(n91), .Y(n92));
NAND4X1 g24(.A(n75), .B(Y_8), .C(C_8), .D(n79), .Y(n93_1));
NAND2X1 g14(.A(Y_13), .B(C_13), .Y(n83_1));
NAND2X1 g16(.A(Y_12), .B(C_12), .Y(n85));
NAND2X1 g18(.A(Y_11), .B(C_11), .Y(n87));
NAND2X1 g46(.A(Y_16), .B(C_16), .Y(n115));
NOR4X1 g10(.A(Y_5), .B(Y_6), .C(Y_4), .D(n78_1), .Y(n79));
INVX1 g05(.A(Y_8), .Y(n74));
INVX1 g06(.A(Y_7), .Y(n75));
INVX1 g07(.A(Y_9), .Y(n76));
NAND4X1 g40(.A(n106), .B(n105), .C(n101), .D(n108_1), .Y(n109));
AND2X1 g26(.A(n94), .B(n79), .Y(n95));
NOR4X1 g28(.A(n78_1), .B(Y_5), .C(Y_4), .D(n96), .Y(n97));
NOR3X1 g30(.A(n98_1), .B(n78_1), .C(Y_4), .Y(n99));
NAND2X1 g20(.A(Y_10), .B(C_10), .Y(n89));
AND2X1 g22(.A(Y_9), .B(C_9), .Y(n91));
OR4X1 g09(.A(Y_2), .B(Y_3), .C(n77), .D(Y_1), .Y(n78_1));
OAI21X1 g39(.A0(n107), .A1(C_0), .B0(X), .Y(n108_1));
OR2X1 g32(.A(n100), .B(n78_1), .Y(n101));
NAND4X1 g36(.A(n103_1), .B(n102), .C(X), .D(n104), .Y(n105));
NAND4X1 g37(.A(Y_2), .B(C_2), .C(X), .D(n103_1), .Y(n106));
AND2X1 g25(.A(Y_7), .B(C_7), .Y(n94));
NAND2X1 g27(.A(Y_6), .B(C_6), .Y(n96));
NAND2X1 g29(.A(Y_5), .B(C_5), .Y(n98_1));
INVX1 g08(.A(X), .Y(n77));
AND2X1 g38(.A(Y_1), .B(C_1), .Y(n107));
NAND2X1 g31(.A(Y_4), .B(C_4), .Y(n100));
AND2X1 g35(.A(Y_3), .B(C_3), .Y(n104));
INVX1 g33(.A(Y_2), .Y(n102));
INVX1 g34(.A(Y_1), .Y(n103_1));

endmodule

```

Figura 46: Descrição do s420 (saída Z)

```

//Converted to Combinational (Partial output: n109) , Module name: s444_n109
module s444_n109 ( G17, G19, G20, G23, G21, G22, G0, G24, n109 );
input G17, G19, G20, G23, G21, G22, G0, G24;
output n109;
wire n155, n152, n93, n135, n108, n153, n154, n132, n134, n129, n91, n142, n119_1,
n131, n130, n133, n128;
OAI22X1 g84(.A0(n135), .A1(n93), .B0(n152), .B1(n155), .Y(n109));
AOI21X1 g83(.A0(n154), .A1(n153), .B0(n108), .Y(n155));
AND2X1 g80(.A(n134), .B(n132), .Y(n152));
INVX1 g21(.A(G17), .Y(n93));
AOI21X1 g63(.A0(n134), .A1(n132), .B0(n129), .Y(n135));
OR2X1 g36(.A(G20), .B(G19), .Y(n108));
NAND4X1 g81(.A(n142), .B(G21), .C(n91), .D(G23), .Y(n153));
NAND3X1 g82(.A(G22), .B(n119_1), .C(n91), .Y(n154));
NAND4X1 g60(.A(G21), .B(n130), .C(G19), .D(n131), .Y(n132));
NOR2X1 g62(.A(n133), .B(G0), .Y(n134));
NOR4X1 g57(.A(n128), .B(n119_1), .C(G0), .D(n108), .Y(n129));
INVX1 g19(.A(G0), .Y(n91));
INVX1 g70(.A(G22), .Y(n142));
INVX1 g47(.A(G21), .Y(n119_1));
NOR2X1 g59(.A(G23), .B(G22), .Y(n131));
INVX1 g58(.A(G20), .Y(n130));
INVX1 g61(.A(G24), .Y(n133));
INVX1 g56(.A(G23), .Y(n128));

endmodule

```

Figura 47: Descrição do s444 (saída n109)

```

//Converted to Combinational (Partial output: n78) , Module name: s510_n78
module s510_n78 ( cnt44, st_4, st_0, st_5, st_3, cnt261, st_1, cnt45, st_2,
cnt509, cnt10, cnt283, cnt21, cnt511, cnt567, john, cnt272, cnt591, cnt284, pcnt6,
n78 );
input cnt44, st_4, st_0, st_5, st_3, cnt261, st_1, cnt45, st_2, cnt509, cnt10,
cnt283, cnt21, cnt511, cnt567, john, cnt272, cnt591, cnt284, pcnt6;
output n78;
wire n187, n168, n173, n181, n186, n79, n183, n161, n167, n88, n169, n172, n180,
n175, n176, n179, n185, n80, n121, n184, n182, n151, n143, n160, n53_1, n166,
n163, n164, n165, n141, n171, n155, n157, n84, n64, n174, n148, n177, n178, n48,
n66, n51, n98, n162, n44, n170, n69;
NAND4X1 g144(.A(n181), .B(n173), .C(n168), .D(n187), .Y(n78));
AOI22X1 g143(.A0(n183), .A1(cnt44), .B0(n79), .B1(n186), .Y(n187));
OAI21X1 g124(.A0(n167), .A1(n161), .B0(st_4), .Y(n168));
OAI21X1 g129(.A0(n172), .A1(n169), .B0(n88), .Y(n173));
NOR4X1 g137(.A(n179), .B(n176), .C(n175), .D(n180), .Y(n181));
OAI22X1 g142(.A0(n184), .A1(n121), .B0(n80), .B1(n185), .Y(n186));
INVX1 g035(.A(st_0), .Y(n79));
OAI21X1 g139(.A0(n151), .A1(st_4), .B0(n182), .Y(n183));
NOR4X1 g117(.A(st_0), .B(n53_1), .C(n160), .D(n143), .Y(n161));
NAND4X1 g123(.A(n165), .B(n164), .C(n163), .D(n166), .Y(n167));
NOR2X1 g044(.A(st_4), .B(st_5), .Y(n88));
NOR4X1 g125(.A(st_0), .B(n53_1), .C(n141), .D(n143), .Y(n169));
OAI21X1 g128(.A0(n157), .A1(n155), .B0(n171), .Y(n172));
NOR4X1 g136(.A(n64), .B(n79), .C(st_4), .D(n84), .Y(n180));
NOR4X1 g131(.A(n143), .B(st_0), .C(st_3), .D(n174), .Y(n175));
NOR4X1 g132(.A(n79), .B(st_1), .C(cnt261), .D(n148), .Y(n176));
AND2X1 g135(.A(n178), .B(n177), .Y(n179));
NAND3X1 g141(.A(n66), .B(n48), .C(cnt45), .Y(n185));
AOI21X1 g036(.A0(st_3), .A1(st_4), .B0(st_5), .Y(n80));
NAND3X1 g077(.A(n66), .B(st_2), .C(st_3), .Y(n121));
NAND2X1 g140(.A(st_5), .B(cnt509), .Y(n184));
NAND4X1 g138(.A(st_1), .B(n48), .C(st_5), .D(n79), .Y(n182));
NAND4X1 g107(.A(st_1), .B(n48), .C(n53_1), .D(n79), .Y(n151));
NAND2X1 g099(.A(st_1), .B(st_2), .Y(n143));
INVX1 g116(.A(cnt10), .Y(n160));
INVX1 g009(.A(st_3), .Y(n53_1));
NAND3X1 g122(.A(n98), .B(n51), .C(cnt283), .Y(n166));
NAND4X1 g119(.A(n66), .B(st_2), .C(st_3), .D(n162), .Y(n163));
NAND3X1 g120(.A(n98), .B(n53_1), .C(cnt21), .Y(n164));
NAND4X1 g121(.A(st_2), .B(n53_1), .C(cnt45), .D(n44), .Y(n165));
INVX1 g097(.A(cnt511), .Y(n141));
OR4X1 g127(.A(n69), .B(n48), .C(st_3), .D(n170), .Y(n171));
INVX1 g111(.A(cnt567), .Y(n155));
NAND2X1 g113(.A(n98), .B(n51), .Y(n157));
XOR2X1 g040(.A(st_1), .B(st_2), .Y(n84));
OR2X1 g020(.A(st_3), .B(st_5), .Y(n64));
AOI22X1 g130(.A0(st_5), .A1(cnt10), .B0(john), .B1(st_4), .Y(n174));
NAND3X1 g104(.A(st_2), .B(n53_1), .C(st_5), .Y(n148));
MX2X1 g133(.A(cnt591), .B(cnt272), .S0(st_2), .Y(n177));
NOR4X1 g134(.A(n53_1), .B(st_4), .C(st_5), .D(n69), .Y(n178));
INVX1 g004(.A(st_2), .Y(n48));
INVX1 g022(.A(st_1), .Y(n66));
AND2X1 g007(.A(st_1), .B(st_3), .Y(n51));
NOR2X1 g054(.A(st_0), .B(st_2), .Y(n98));
OR2X1 g118(.A(st_0), .B(cnt21), .Y(n162));
NOR2X1 g000(.A(st_0), .B(st_1), .Y(n44));
NAND2X1 g126(.A(pcnt6), .B(cnt284), .Y(n170));
OR2X1 g025(.A(st_0), .B(st_1), .Y(n69));

endmodule

```

Figura 48: Descrição do s510 (saída n78)


```

//Converted to Combinational (Partial output: n178) , Module name: s641_n178
module s641_n178 ( G5, G2, G75, G3, G71, G76, G64, G4, G72, G77, G65, G13, G9,
G10, G22, G11, G78, G23, G67, G66, G24, G25, n178 );
input G5, G2, G75, G3, G71, G76, G64, G4, G72, G77, G65, G13, G9, G10, G22, G11,
G78, G23, G67, G66, G24, G25;
output n178;
wire n216, n119, n215, n152, n154, n179, n137, G86BF, n214, n148_1, n204, n129,
n151, n175, n171, n172, n124_1, n147, n188_1, n163_1, n164, n135, n173_1, n174,
n145, n146, n156, n160, n134_1, n131, n133, n139_1, n141, n155, n158_1, n159,
n132, n138, n140, n157;
OAI21X1 g101(.A0(n215), .A1(n119), .B0(n216), .Y(n178));
OAI21X1 g100(.A0(G5), .A1(n154), .B0(n152), .Y(n216));
AOI21X1 g064(.A0(G86BF), .A1(n137), .B0(n179), .Y(n119));
OR4X1 g099(.A(n129), .B(n204), .C(n148_1), .D(n214), .Y(n215));
INVX1 g036(.A(n151), .Y(n152));
INVX1 g038(.A(G2), .Y(n154));
INVX1 g063(.A(G75), .Y(n179));
INVX1 g021(.A(G3), .Y(n137));
OAI21X1 g060(.A0(n172), .A1(n171), .B0(n175), .Y(G86BF));
NAND3X1 g098(.A(G71), .B(G5), .C(n124_1), .Y(n214));
INVX1 g032(.A(n147), .Y(n148_1));
INVX1 g088(.A(n188_1), .Y(n204));
NOR2X1 g049(.A(n164), .B(n163_1), .Y(n129));
OAI21X1 g035(.A0(n135), .A1(G3), .B0(G76), .Y(n151));
NOR2X1 g059(.A(n174), .B(n173_1), .Y(n175));
OAI21X1 g055(.A0(n164), .A1(n163_1), .B0(n147), .Y(n171));
NAND3X1 g056(.A(n151), .B(G64), .C(n154), .Y(n172));
INVX1 g008(.A(G4), .Y(n124_1));
OR2X1 g031(.A(n146), .B(n145), .Y(n147));
AND2X1 g072(.A(n135), .B(G72), .Y(n188_1));
INVX1 g047(.A(G77), .Y(n163_1));
AOI21X1 g048(.A0(n160), .A1(n156), .B0(G3), .Y(n164));
NOR4X1 g019(.A(n133), .B(G65), .C(n131), .D(n134_1), .Y(n135));
NOR4X1 g057(.A(G10), .B(G9), .C(G3), .D(G13), .Y(n173_1));
OAI21X1 g058(.A0(G11), .A1(G3), .B0(G22), .Y(n174));
INVX1 g029(.A(G78), .Y(n145));
AOI21X1 g030(.A0(n141), .A1(n139_1), .B0(G3), .Y(n146));
OAI21X1 g040(.A0(n146), .A1(n145), .B0(n155), .Y(n156));
NOR2X1 g044(.A(n159), .B(n158_1), .Y(n160));
NOR2X1 g018(.A(G11), .B(G3), .Y(n134_1));
INVX1 g015(.A(G23), .Y(n131));
NOR4X1 g017(.A(n132), .B(G9), .C(G3), .D(G13), .Y(n133));
NAND4X1 g023(.A(G10), .B(G9), .C(n137), .D(n138), .Y(n139_1));
NOR3X1 g025(.A(n134_1), .B(G67), .C(n140), .Y(n141));
AND2X1 g039(.A(G66), .B(n154), .Y(n155));
NOR4X1 g042(.A(G10), .B(n157), .C(G3), .D(G13), .Y(n158_1));
OAI21X1 g043(.A0(G11), .A1(G3), .B0(G24), .Y(n159));
INVX1 g016(.A(G10), .Y(n132));
INVX1 g022(.A(G13), .Y(n138));
INVX1 g024(.A(G25), .Y(n140));
INVX1 g041(.A(G9), .Y(n157));

endmodule

```

Figura 49: Descrição do s641 (saída n178)

```

//Converted to Combinational (Partial output: n177) , Module name: s713_n177
module s713_n177 ( G5, G2, G75, G3, G77, G76, G64, G72, G65, G13, G9, G10, G22,
G11, G66, G78, G24, G23, G71, G67, G4, G25, n177 );
input G5, G2, G75, G3, G77, G76, G64, G72, G65, G13, G9, G10, G22, G11, G66, G78,
G24, G23, G71, G67, G4, G25;
output n177;
wire n216, n117, n215, n147_1, n126, n116, n115, G86BF, n214, n117_1, n133, n139,
n143, n134, n140, n151, n213, n128, n132_1, n138, n141, n142_1, n125, n203,
n127_1, n118, n124, n130, n131, n137_1, n135, n122_1, n172_1, n120, n123, n129,
n136, n165, n119, n121;
OAI21X1 g102(.A0(n215), .A1(n117), .B0(n216), .Y(n177));
OAI21X1 g101(.A0(G5), .A1(n126), .B0(n147_1), .Y(n216));
AOI21X1 g030(.A0(G86BF), .A1(n115), .B0(n116), .Y(n117));
OAI21X1 g100(.A0(n133), .A1(n117_1), .B0(n214), .Y(n215));
INVX1 g032(.A(n139), .Y(n147_1));
INVX1 g011(.A(G2), .Y(n126));
INVX1 g001(.A(G75), .Y(n116));
INVX1 g000(.A(G3), .Y(n115));
OAI21X1 g029(.A0(n140), .A1(n134), .B0(n143), .Y(G86BF));
NOR2X1 g099(.A(n213), .B(n151), .Y(n214));
INVX1 g002(.A(G77), .Y(n117_1));
AOI21X1 g018(.A0(n132_1), .A1(n128), .B0(G3), .Y(n133));
OAI21X1 g024(.A0(n138), .A1(G3), .B0(G76), .Y(n139));
NOR2X1 g028(.A(n142_1), .B(n141), .Y(n143));
OAI21X1 g019(.A0(n133), .A1(n117_1), .B0(n125), .Y(n134));
NAND3X1 g025(.A(n139), .B(G64), .C(n126), .Y(n140));
INVX1 g036(.A(n125), .Y(n151));
NAND4X1 g098(.A(n138), .B(G72), .C(G5), .D(n203), .Y(n213));
OAI21X1 g013(.A0(n124), .A1(n118), .B0(n127_1), .Y(n128));
NOR2X1 g017(.A(n131), .B(n130), .Y(n132_1));
NOR4X1 g023(.A(n122_1), .B(G65), .C(n135), .D(n137_1), .Y(n138));
NOR4X1 g026(.A(G10), .B(G9), .C(G3), .D(G13), .Y(n141));
OAI21X1 g027(.A0(G11), .A1(G3), .B0(G22), .Y(n142_1));
OR2X1 g010(.A(n124), .B(n118), .Y(n125));
INVX1 g088(.A(n172_1), .Y(n203));
AND2X1 g012(.A(G66), .B(n126), .Y(n127_1));
INVX1 g003(.A(G78), .Y(n118));
AOI21X1 g009(.A0(n123), .A1(n120), .B0(G3), .Y(n124));
NOR4X1 g015(.A(G10), .B(n129), .C(G3), .D(G13), .Y(n130));
OAI21X1 g016(.A0(G11), .A1(G3), .B0(G24), .Y(n131));
NOR4X1 g022(.A(n136), .B(G9), .C(G3), .D(G13), .Y(n137_1));
INVX1 g020(.A(G23), .Y(n135));
NOR2X1 g007(.A(G11), .B(G3), .Y(n122_1));
NAND2X1 g057(.A(G71), .B(n165), .Y(n172_1));
NAND4X1 g005(.A(G10), .B(G9), .C(n115), .D(n119), .Y(n120));
NOR3X1 g008(.A(n122_1), .B(G67), .C(n121), .Y(n123));
INVX1 g014(.A(G9), .Y(n129));
INVX1 g021(.A(G10), .Y(n136));
INVX1 g050(.A(G4), .Y(n165));
INVX1 g004(.A(G13), .Y(n119));
INVX1 g006(.A(G25), .Y(n121));

endmodule

```

Figura 50: Descrição do s713 (saída n177)

```

//Converted to Combinational (Partial output: n95) , Module name: s820_n95
module s820_n95 ( G4, G1, G18, G16, G38, G39, G40, G0, G41, G42, G15, G13, G7, G8,
G2, G5, G9, G6, G14, n95 );
input G3, G4, G1, G18, G16, G38, G39, G40, G0, G41, G42, G15, G13, G7, G8, G2, G5,
G9, G6, G14;
output n95;
wire n234, n245, n223, n233, n119, n242, n244, n222, n53, n219, n226, n73, n232,
n117, n118, n238, n241, n243, n77, n201, n212, n220, n221, n218, n156, n217, n225,
n113, n224, n231, n227, n230, n116, n58, n236, n237, n240, n82, n239, n95_1, n107,
n56, n54, n52, n229, n228, n235, n76, n175, n104, n136, n83;
AOI21X1 g194(.A0(n245), .A1(n234), .B0(G18), .Y(n95));
OAI21X1 g182(.A0(n233), .A1(n223), .B0(G16), .Y(n234));
NOR3X1 g193(.A(n244), .B(n242), .C(n119), .Y(n245));
AOI21X1 g171(.A0(n219), .A1(n53), .B0(n222), .Y(n223));
OAI21X1 g181(.A0(n232), .A1(n73), .B0(n226), .Y(n233));
NOR2X1 g067(.A(n118), .B(n117), .Y(n119));
AOI21X1 g190(.A0(n241), .A1(n238), .B0(G38), .Y(n242));
NOR4X1 g192(.A(n212), .B(n201), .C(n77), .D(n243), .Y(n244));
OR2X1 g170(.A(n221), .B(n220), .Y(n222));
INVX1 g001(.A(G39), .Y(n53));
NAND4X1 g167(.A(n217), .B(n156), .C(G40), .D(n218), .Y(n219));
NAND4X1 g174(.A(n224), .B(n218), .C(n113), .D(n225), .Y(n226));
OR2X1 g021(.A(G39), .B(G38), .Y(n73));
AOI21X1 g180(.A0(n230), .A1(n227), .B0(n231), .Y(n232));
AOI21X1 g065(.A0(n116), .A1(G38), .B0(n113), .Y(n117));
NAND2X1 g066(.A(G39), .B(G4), .Y(n118));
OAI21X1 g186(.A0(n237), .A1(n236), .B0(n58), .Y(n238));
AOI21X1 g189(.A0(n239), .A1(n82), .B0(n240), .Y(n241));
AND2X1 g191(.A(G39), .B(G0), .Y(n243));
INVX1 g025(.A(G38), .Y(n77));
NAND3X1 g149(.A(G42), .B(G41), .C(G40), .Y(n201));
NOR2X1 g160(.A(G39), .B(G4), .Y(n212));
AOI21X1 g168(.A0(n107), .A1(n113), .B0(n95_1), .Y(n220));
OAI21X1 g169(.A0(G40), .A1(G15), .B0(G41), .Y(n221));
OR2X1 g166(.A(G42), .B(G15), .Y(n218));
NAND2X1 g104(.A(n56), .B(G38), .Y(n156));
NAND3X1 g165(.A(G42), .B(G15), .C(G13), .Y(n217));
OAI21X1 g173(.A0(n56), .A1(G41), .B0(n53), .Y(n225));
INVX1 g061(.A(G40), .Y(n113));
NAND3X1 g172(.A(G42), .B(G39), .C(G15), .Y(n224));
OAI22X1 g179(.A0(n56), .A1(G15), .B0(G1), .B1(n54), .Y(n231));
NOR3X1 g175(.A(G42), .B(n113), .C(n52), .Y(n227));
NOR4X1 g178(.A(G8), .B(G7), .C(n228), .D(n229), .Y(n230));
NOR2X1 g064(.A(G42), .B(G41), .Y(n116));
AND2X1 g006(.A(G40), .B(G39), .Y(n58));
NOR4X1 g184(.A(n56), .B(G41), .C(G2), .D(n235), .Y(n236));
AOI21X1 g185(.A0(n76), .A1(G5), .B0(G42), .Y(n237));
NOR4X1 g188(.A(n136), .B(G39), .C(n104), .D(n175), .Y(n240));
NOR3X1 g030(.A(G42), .B(n76), .C(G40), .Y(n82));
NOR3X1 g187(.A(G39), .B(n52), .C(n83), .Y(n239));
NAND2X1 g043(.A(G39), .B(G38), .Y(n95_1));
NAND4X1 g055(.A(G8), .B(G7), .C(G6), .D(G9), .Y(n107));
INVX1 g004(.A(G42), .Y(n56));
OR2X1 g002(.A(G41), .B(G40), .Y(n54));
INVX1 g000(.A(G15), .Y(n52));
INVX1 g177(.A(G9), .Y(n229));
INVX1 g176(.A(G6), .Y(n228));
OR2X1 g183(.A(G3), .B(G1), .Y(n235));
INVX1 g024(.A(G41), .Y(n76));
NOR2X1 g123(.A(G42), .B(G40), .Y(n175));
INVX1 g052(.A(G4), .Y(n104));
NOR2X1 g084(.A(G41), .B(G40), .Y(n136));
INVX1 g031(.A(G14), .Y(n83));

endmodule

```

Figura 51: Descrição do s820 (saída n95)

```

//Converted to Combinational (Partial output: n90) , Module name: s832_n90
module s832_n90 ( G1, G3, G4, G18, G38, G0, G40, G41, G42, G39, G2, G15, G16, G9,
G6, G7, G8, G13, G14, G5, n90 );
input G1, G3, G4, G18, G38, G0, G40, G41, G42, G39, G2, G15, G16, G9, G6, G7, G8,
G13, G14, G5;
output n90;
wire n200, n218, n76, n191, n199, n217, n208, n212, n213, n190, n186, n187, n188,
n193, n196, n198, n197, n214, n215, n216, n207, n202, n204, n205, n209, n211, n86,
n73, n78, n189, n172, n62, n174, n185, n53, n114, n192, n194, n195, n80_1, n61,
n147, n66, n57, n60, n206, n95_1, n96, n201, n203, n210, n102, n54, n52, n109;
AOI21X1 g165(.A0(n218), .A1(n200), .B0(G18), .Y(n90));
OAI21X1 g146(.A0(n199), .A1(n191), .B0(n76), .Y(n200));
NOR4X1 g164(.A(n213), .B(n212), .C(n208), .D(n217), .Y(n218));
INVX1 g024(.A(G38), .Y(n76));
NOR4X1 g137(.A(n188), .B(n187), .C(n186), .D(n190), .Y(n191));
OAI22X1 g145(.A0(n197), .A1(n198), .B0(n196), .B1(n193), .Y(n199));
NOR3X1 g163(.A(n216), .B(n215), .C(n214), .Y(n217));
NOR4X1 g154(.A(n205), .B(n204), .C(n202), .D(n207), .Y(n208));
NOR2X1 g158(.A(n211), .B(n209), .Y(n212));
NOR3X1 g159(.A(n209), .B(n86), .C(G0), .Y(n213));
OAI22X1 g136(.A0(n172), .A1(n189), .B0(n78), .B1(n73), .Y(n190));
NOR3X1 g132(.A(n185), .B(n174), .C(n62), .Y(n186));
NOR3X1 g133(.A(G42), .B(G41), .C(G40), .Y(n187));
OAI21X1 g134(.A0(n114), .A1(n73), .B0(n53), .Y(n188));
NOR2X1 g139(.A(n192), .B(G39), .Y(n193));
NAND2X1 g142(.A(n195), .B(n194), .Y(n196));
NAND4X1 g144(.A(n147), .B(G2), .C(n61), .D(n80_1), .Y(n198));
AOI22X1 g143(.A0(n57), .A1(G42), .B0(n62), .B1(n66), .Y(n197));
NAND4X1 g160(.A(n114), .B(G39), .C(n73), .D(n78), .Y(n214));
NOR2X1 g161(.A(n60), .B(n62), .Y(n215));
NOR3X1 g162(.A(G41), .B(n62), .C(G15), .Y(n216));
OAI21X1 g153(.A0(n95_1), .A1(G42), .B0(n206), .Y(n207));
NOR2X1 g148(.A(n201), .B(n96), .Y(n202));
OAI21X1 g150(.A0(G39), .A1(n76), .B0(n203), .Y(n204));
AOI21X1 g151(.A0(G42), .A1(G15), .B0(G41), .Y(n205));
NAND3X1 g155(.A(G42), .B(G41), .C(G40), .Y(n209));
NAND3X1 g157(.A(n210), .B(n53), .C(n73), .Y(n211));
NAND2X1 g034(.A(G39), .B(G38), .Y(n86));
INVX1 g021(.A(G4), .Y(n73));
NAND2X1 g026(.A(G42), .B(G41), .Y(n78));
OR2X1 g135(.A(G42), .B(G40), .Y(n189));
NAND2X1 g118(.A(G15), .B(n102), .Y(n172));
INVX1 g010(.A(G16), .Y(n62));
NOR2X1 g120(.A(G42), .B(G40), .Y(n174));
NOR4X1 g131(.A(G41), .B(n114), .C(n52), .D(n54), .Y(n185));
INVX1 g001(.A(G39), .Y(n53));
INVX1 g060(.A(G40), .Y(n114));
NOR3X1 g138(.A(G42), .B(n80_1), .C(G4), .Y(n192));
AOI21X1 g140(.A0(n80_1), .A1(n109), .B0(n114), .Y(n194));
OAI21X1 g141(.A0(n80_1), .A1(G16), .B0(G42), .Y(n195));
INVX1 g028(.A(G41), .Y(n80_1));
INVX1 g009(.A(G1), .Y(n61));
INVX1 g093(.A(G3), .Y(n147));
NOR2X1 g014(.A(G40), .B(G39), .Y(n66));
AND2X1 g005(.A(G40), .B(G39), .Y(n57));
OR2X1 g006(.A(G42), .B(G41), .Y(n60));
AOI21X1 g152(.A0(n53), .A1(G15), .B0(G4), .Y(n206));
NAND2X1 g043(.A(G38), .B(G15), .Y(n95_1));
NAND4X1 g044(.A(G8), .B(G7), .C(G6), .D(G9), .Y(n96));
AOI21X1 g147(.A0(G38), .A1(G15), .B0(n80_1), .Y(n201));
NOR2X1 g149(.A(G40), .B(n62), .Y(n203));
NAND3X1 g156(.A(G16), .B(G15), .C(G13), .Y(n210));
INVX1 g048(.A(G14), .Y(n102));
INVX1 g002(.A(G42), .Y(n54));
INVX1 g000(.A(G15), .Y(n52));
INVX1 g055(.A(G5), .Y(n109));

endmodule

```

Figura 52: Descrição do s832 (saída n90)

```

//Converted to Combinational (Partial output: n215) , Module name: s838_n215
module s838_n215 ( Y_31, Y_30, Y_32, Clear, Y_29, Y_27, Y_26, Y_25, Y_28, n215 );
input Y_31, Y_30, Y_32, Clear, Y_29, Y_27, Y_26, Y_25, Y_28;
output n215;
wire n316, n318, n315, n133, n134, n317, n306, n136, n137;
NAND2X1 g186(.A(n318), .B(n316), .Y(n215));
NAND4X1 g183(.A(Y_30), .B(Y_31), .C(n133), .D(n315), .Y(n316));
NAND3X1 g185(.A(n317), .B(n134), .C(Y_32), .Y(n318));
NOR4X1 g182(.A(n137), .B(n136), .C(Clear), .D(n306), .Y(n315));
INVX1 g000(.A(Y_32), .Y(n133));
NAND3X1 g001(.A(Y_29), .B(Y_30), .C(Y_31), .Y(n134));
NOR3X1 g184(.A(n306), .B(n136), .C(Clear), .Y(n317));
NAND3X1 g173(.A(Y_25), .B(Y_26), .C(Y_27), .Y(n306));
INVX1 g003(.A(Y_28), .Y(n136));
INVX1 g004(.A(Y_29), .Y(n137));

endmodule

```

Figura 53: Descrição do s838 (saída n215)

```

//Converted to Combinational (Partial output: n104) , Module name: s953_n104
module s953_n104 ( Rdy2RtHS1, WantRtHS1, Rdy1RtHS1, WantBmHS1, State_0, Prog_0,
Full0HS1, State_3, State_1, State_4, Rdy1BmHS1, Rdy2BmHS1, State_5, Prog_2,
State_2, InDoneHS1, FullIHS1, n104 );
input Rdy2RtHS1, WantRtHS1, Rdy1RtHS1, WantBmHS1, State_0, Prog_0, Full0HS1,
State_3, State_1, State_4, Rdy1BmHS1, Rdy2BmHS1, State_5, Prog_2, State_2,
InDoneHS1, FullIHS1;
output n104;
wire n251, n242, n245, n248, n250, n209, n249, n241, n168, n235, n244, n246, n230,
n247, n190, n132, n223, n206, n208, n178, n147, n148, n167, n131, n133_1, n205,
n234, n243, n135, n139, n128, n229, n231, n189_1, n188, n136, n129_1, n127, n207,
n162, n126, n153_1, n236, n187;
NAND4X1 g126(.A(n248), .B(n245), .C(n242), .D(n251), .Y(n104));
AOI21X1 g125(.A0(n249), .A1(n209), .B0(n250), .Y(n251));
NAND4X1 g116(.A(n168), .B(WantRtHS1), .C(Rdy2RtHS1), .D(n241), .Y(n242));
NAND2X1 g119(.A(n244), .B(n235), .Y(n245));
AOI21X1 g122(.A0(n247), .A1(n230), .B0(n246), .Y(n248));
AOI21X1 g124(.A0(n223), .A1(n132), .B0(n190), .Y(n250));
NOR3X1 g083(.A(n208), .B(n206), .C(n132), .Y(n209));
NOR2X1 g123(.A(n178), .B(Rdy1RtHS1), .Y(n249));
NAND3X1 g115(.A(WantBmHS1), .B(n148), .C(n147), .Y(n241));
NOR4X1 g042(.A(n133_1), .B(State_0), .C(n131), .D(n167), .Y(n168));
MX2X1 g109(.A(n234), .B(n205), .S0(Prog_0), .Y(n235));
NOR2X1 g118(.A(n243), .B(Full0HS1), .Y(n244));
NOR4X1 g120(.A(State_1), .B(State_3), .C(n139), .D(n135), .Y(n246));
AOI21X1 g104(.A0(n229), .A1(State_4), .B0(n128), .Y(n230));
AOI21X1 g121(.A0(Rdy2BmHS1), .A1(Rdy1BmHS1), .B0(n231), .Y(n247));
AOI21X1 g064(.A0(n188), .A1(State_3), .B0(n189_1), .Y(n190));
INVX1 g006(.A(State_1), .Y(n132));
OR4X1 g097(.A(n129_1), .B(n135), .C(State_4), .D(n136), .Y(n223));
NAND4X1 g080(.A(n128), .B(n127), .C(State_5), .D(n135), .Y(n206));
NOR2X1 g082(.A(n207), .B(State_4), .Y(n208));
INVX1 g052(.A(Rdy2RtHS1), .Y(n178));
INVX1 g021(.A(Rdy1BmHS1), .Y(n147));
INVX1 g022(.A(Rdy2BmHS1), .Y(n148));
NAND2X1 g041(.A(n136), .B(n162), .Y(n167));
INVX1 g005(.A(Prog_2), .Y(n131));
NAND4X1 g007(.A(n128), .B(n127), .C(State_5), .D(n132), .Y(n133_1));
NOR2X1 g079(.A(Rdy2RtHS1), .B(n126), .Y(n205));
NOR2X1 g108(.A(Rdy2BmHS1), .B(n147), .Y(n234));
NAND3X1 g117(.A(n236), .B(n153_1), .C(n131), .Y(n243));
INVX1 g009(.A(State_0), .Y(n135));
INVX1 g013(.A(State_5), .Y(n139));
INVX1 g002(.A(State_2), .Y(n128));
MX2X1 g103(.A(InDoneHS1), .B(n136), .S0(Prog_2), .Y(n229));
NOR2X1 g105(.A(State_4), .B(Prog_2), .Y(n231));
NOR4X1 g063(.A(State_2), .B(State_4), .C(State_5), .D(State_1), .Y(n189_1));
NOR4X1 g062(.A(State_0), .B(n148), .C(n147), .D(n187), .Y(n188));
NOR2X1 g010(.A(FullIHS1), .B(Full0HS1), .Y(n136));
NAND3X1 g003(.A(n128), .B(n127), .C(State_5), .Y(n129_1));
INVX1 g001(.A(State_3), .Y(n127));
NOR3X1 g081(.A(n131), .B(FullIHS1), .C(Full0HS1), .Y(n207));
INVX1 g036(.A(State_4), .Y(n162));
INVX1 g000(.A(Rdy1RtHS1), .Y(n126));
NOR4X1 g027(.A(State_2), .B(State_3), .C(n139), .D(State_0), .Y(n153_1));
NOR2X1 g110(.A(State_1), .B(State_4), .Y(n236));
OR2X1 g061(.A(FullIHS1), .B(Full0HS1), .Y(n187));
endmodule

```

Figura 54: Descrição do s953 (saída n104)

```

//Converted to Combinational (Partial output: G542) , Module name: s1196_G542
module s1196_G542 ( G4, G1, G3, G10, G7, G8, G9, G13, G6, G34, G46, G12, G11, G5,
G31, G30, G2, G0, G542 );
input G4, G3, G1, G10, G7, G8, G9, G13, G6, G34, G46, G12, G11, G5, G31, G30, G2,
G0;
output G542;
wire n285, n293, n283, n284, n292, n286, n157, n287, n86, n115, n209, n87_1,
n112_1, n256, n288, n290, n291, n130, n205, n111, n137_1, n289, n116, n129, n121,
n125, n106, n110, n126, n88, n128, n118, n120, n122_1, n123, n124, n105, n109,
n85, n94, n127_1, n117_1, n114, n119, n108, n113, n107_1;
NAND2X1 g212(.A(n293), .B(n285), .Y(G542));
NAND2X1 g203(.A(n284), .B(n283), .Y(n285));
AOI22X1 g211(.A0(n287), .A1(n157), .B0(n286), .B1(n292), .Y(n293));
NOR3X1 g201(.A(G10), .B(n115), .C(n86), .Y(n283));
OAI22X1 g202(.A0(n256), .A1(n112_1), .B0(n87_1), .B1(n209), .Y(n284));
NAND3X1 g210(.A(n291), .B(n290), .C(n288), .Y(n292));
NOR2X1 g204(.A(n209), .B(n87_1), .Y(n286));
AND2X1 g075(.A(G10), .B(G7), .Y(n157));
AOI21X1 g205(.A0(G9), .A1(G8), .B0(n256), .Y(n287));
INVX1 g004(.A(G7), .Y(n86));
INVX1 g033(.A(G9), .Y(n115));
OR4X1 g127(.A(n111), .B(G13), .C(n205), .D(n130), .Y(n209));
INVX1 g005(.A(G6), .Y(n87_1));
INVX1 g030(.A(G8), .Y(n112_1));
INVX1 g174(.A(G34), .Y(n256));
INVX1 g206(.A(n137_1), .Y(n288));
NAND2X1 g208(.A(n289), .B(n115), .Y(n290));
OAI21X1 g209(.A0(n115), .A1(n86), .B0(n116), .Y(n291));
NAND4X1 g048(.A(n125), .B(n121), .C(G46), .D(n129), .Y(n130));
INVX1 g123(.A(G12), .Y(n205));
NOR2X1 g029(.A(n110), .B(n106), .Y(n111));
NOR4X1 g055(.A(n88), .B(n115), .C(G8), .D(n126), .Y(n137_1));
OAI21X1 g207(.A0(G8), .A1(n86), .B0(n88), .Y(n289));
AND2X1 g034(.A(G10), .B(G8), .Y(n116));
NAND4X1 g047(.A(n126), .B(G9), .C(n86), .D(n128), .Y(n129));
OAI21X1 g039(.A0(n120), .A1(n118), .B0(G11), .Y(n121));
NAND3X1 g043(.A(n124), .B(n123), .C(n122_1), .Y(n125));
NOR2X1 g024(.A(G5), .B(n105), .Y(n106));
AOI22X1 g028(.A0(n94), .A1(G3), .B0(n85), .B1(n109), .Y(n110));
INVX1 g044(.A(G11), .Y(n126));
INVX1 g006(.A(G10), .Y(n88));
MX2X1 g046(.A(G10), .B(n127_1), .S0(G8), .Y(n128));
NOR4X1 g036(.A(n114), .B(G31), .C(n112_1), .D(n117_1), .Y(n118));
OAI22X1 g038(.A0(G30), .A1(G6), .B0(G9), .B1(n119), .Y(n120));
NAND3X1 g040(.A(G30), .B(G7), .C(n87_1), .Y(n122_1));
NOR3X1 g041(.A(G11), .B(G10), .C(G9), .Y(n123));
NAND2X1 g042(.A(G31), .B(G8), .Y(n124));
INVX1 g023(.A(G4), .Y(n105));
AOI21X1 g027(.A0(n108), .A1(G2), .B0(G3), .Y(n109));
INVX1 g003(.A(G5), .Y(n85));
NOR2X1 g012(.A(G4), .B(G0), .Y(n94));
INVX1 g045(.A(G31), .Y(n127_1));
NOR2X1 g035(.A(n116), .B(n115), .Y(n117_1));
NOR3X1 g032(.A(n113), .B(n86), .C(G6), .Y(n114));
OR2X1 g037(.A(G8), .B(G7), .Y(n119));
OAI22X1 g026(.A0(n85), .A1(G1), .B0(n105), .B1(n107_1), .Y(n108));
INVX1 g031(.A(G30), .Y(n113));
AND2X1 g025(.A(G5), .B(G3), .Y(n107_1));

endmodule

```

Figura 55: Descrição do s1196 (saída G542)

```

//Converted to Combinational (Partial output: n117) , Module name: s1238_n117
module s1238_n117 ( G1, G3, G4, G10, G34, G13, G7, G9, G6, G8, G12, G11, G46, G5,
G31, G0, G2, G30, n117 );
input G1, G3, G4, G10, G34, G13, G7, G9, G6, G8, G12, G11, G46, G5, G31, G0, G2,
G30;
output n117;
wire n394, n214, n216, n393, n120, n392, n101, n104, n130, n100, n215, n105, n223,
n119, n82_1, n97_1, n125, n387, n118, n98, n111, n115, n84, n86, n96, n117_1, n99,
n107_1, n110, n103, n113, n114, n83, n85, n87_1, n88, n95, n116, n106, n108, n109,
n102_1, n112_1, n94, n90, n89, n93, n92_1, n91;
A0I21X1 g313(.A0(n216), .A1(n214), .B0(n394), .Y(n117));
OAI21X1 g312(.A0(n392), .A1(n120), .B0(n393), .Y(n394));
NOR3X1 g132(.A(G10), .B(n104), .C(n101), .Y(n214));
OAI22X1 g134(.A0(n215), .A1(n100), .B0(n130), .B1(n120), .Y(n216));
NAND3X1 g311(.A(n223), .B(n105), .C(G34), .Y(n393));
OR4X1 g038(.A(n97_1), .B(G13), .C(n82_1), .D(n119), .Y(n120));
OR2X1 g310(.A(n387), .B(n125), .Y(n392));
INVX1 g019(.A(G7), .Y(n101));
INVX1 g022(.A(G9), .Y(n104));
INVX1 g048(.A(G6), .Y(n130));
INVX1 g018(.A(G8), .Y(n100));
INVX1 g133(.A(G34), .Y(n215));
AND2X1 g023(.A(G10), .B(G8), .Y(n105));
NAND2X1 g141(.A(G9), .B(G7), .Y(n223));
OR4X1 g037(.A(n115), .B(n111), .C(n98), .D(n118), .Y(n119));
INVX1 g000(.A(G12), .Y(n82_1));
A0I21X1 g015(.A0(n96), .A1(n86), .B0(n84), .Y(n97_1));
NAND2X1 g043(.A(G10), .B(G7), .Y(n125));
AND2X1 g305(.A(G9), .B(G6), .Y(n387));
NOR4X1 g036(.A(G11), .B(n104), .C(G7), .D(n117_1), .Y(n118));
INVX1 g016(.A(G46), .Y(n98));
A0I21X1 g029(.A0(n110), .A1(n107_1), .B0(n99), .Y(n111));
NOR3X1 g033(.A(n114), .B(n113), .C(n103), .Y(n115));
NOR2X1 g002(.A(G5), .B(n83), .Y(n84));
NAND3X1 g004(.A(n83), .B(G3), .C(n85), .Y(n86));
OAI21X1 g014(.A0(n95), .A1(n88), .B0(n87_1), .Y(n96));
MX2X1 g035(.A(n116), .B(G31), .S0(G8), .Y(n117_1));
INVX1 g017(.A(G11), .Y(n99));
OR4X1 g025(.A(n103), .B(G31), .C(n100), .D(n106), .Y(n107_1));
NOR2X1 g028(.A(n109), .B(n108), .Y(n110));
NOR3X1 g021(.A(n102_1), .B(n101), .C(G6), .Y(n103));
NAND2X1 g031(.A(n112_1), .B(n104), .Y(n113));
AND2X1 g032(.A(G31), .B(G8), .Y(n114));
INVX1 g001(.A(G4), .Y(n83));
INVX1 g003(.A(G0), .Y(n85));
INVX1 g005(.A(G5), .Y(n87_1));
NOR3X1 g006(.A(G3), .B(G2), .C(n85), .Y(n88));
A0I21X1 g013(.A0(n94), .A1(G0), .B0(G4), .Y(n95));
INVX1 g034(.A(G10), .Y(n116));
NOR2X1 g024(.A(n105), .B(n104), .Y(n106));
NOR3X1 g026(.A(G9), .B(G8), .C(G7), .Y(n108));
NOR2X1 g027(.A(G30), .B(G6), .Y(n109));
INVX1 g020(.A(G30), .Y(n102_1));
NOR2X1 g030(.A(G11), .B(G10), .Y(n112_1));
OAI21X1 g012(.A0(n93), .A1(n89), .B0(n90), .Y(n94));
INVX1 g008(.A(G3), .Y(n90));
INVX1 g007(.A(G2), .Y(n89));
A0I22X1 g011(.A0(G5), .A1(n91), .B0(G4), .B1(n92_1), .Y(n93));
NAND2X1 g010(.A(G5), .B(G3), .Y(n92_1));
INVX1 g009(.A(G1), .Y(n91));
endmodule

```

Figura 56: Descrição do s1238 (saída n117)

```

//Converted to Combinational (Partial output: n90) , Module name: s1423_n90
module s1423_n90 ( G1, G2, G3, G4, G8, G14, G31, G30, G29, G92, G28, G27, G24,
G25, G26, G46, G90, G45, G84, G78, G85, G44, G64, G77, G76, G43, G42, G0, G75,
G74, n90 );
input G1, G2, G3, G4, G8, G14, G31, G30, G29, G92, G28, G27, G24, G25, G26, G46,
G90, G45, G84, G78, G85, G44, G64, G77, G76, G43, G42, G0, G75, G74;
output n90;
wire n331, n376, n377, n373, n293, n292, n247, n291, n290_1, n248, n249, n260_1,
n262, n289, n258, n259, n261, n267, n269, n288, n257, n250_1, n251, n254, n266,
n268, n273, n275_1, n287, n255_1, n256, n252, n253, n264, n265_1, n271, n272,
n274, n280_1, n281, n286, n263, n270_1, n276, n278, n279, n282, n284, n285_1,
n277, n283;
NOR3X1 g134(.A(n377), .B(n376), .C(n331), .Y(n90));
INVX1 g087(.A(G14), .Y(n331));
AND2X1 g132(.A(n373), .B(G31), .Y(n376));
AOI21X1 g133(.A0(n293), .A1(G30), .B0(G31), .Y(n377));
AND2X1 g129(.A(n293), .B(G30), .Y(n373));
AOI21X1 g049(.A0(n292), .A1(G92), .B0(G29), .Y(n293));
NAND3X1 g048(.A(n291), .B(G28), .C(n247), .Y(n292));
INVX1 g003(.A(G27), .Y(n247));
NOR4X1 g047(.A(n249), .B(n248), .C(G24), .D(n290_1), .Y(n291));
AOI21X1 g046(.A0(n289), .A1(n262), .B0(n260_1), .Y(n290_1));
INVX1 g004(.A(G25), .Y(n248));
INVX1 g005(.A(G26), .Y(n249));
AOI21X1 g016(.A0(n259), .A1(n258), .B0(G46), .Y(n260_1));
XOR2X1 g018(.A(n261), .B(G46), .Y(n262));
AOI21X1 g045(.A0(n288), .A1(n269), .B0(n267), .Y(n289));
OR4X1 g014(.A(n254), .B(n251), .C(n250_1), .D(n257), .Y(n258));
OR2X1 g015(.A(G90), .B(G4), .Y(n259));
AND2X1 g017(.A(n259), .B(n258), .Y(n261));
OR2X1 g023(.A(n266), .B(G45), .Y(n267));
XOR2X1 g025(.A(n266), .B(n268), .Y(n269));
AOI21X1 g044(.A0(n287), .A1(n275_1), .B0(n273), .Y(n288));
NOR3X1 g013(.A(n256), .B(n255_1), .C(G84), .Y(n257));
INVX1 g006(.A(G78), .Y(n250_1));
INVX1 g007(.A(G90), .Y(n251));
AOI21X1 g010(.A0(n253), .A1(n252), .B0(G85), .Y(n254));
AND2X1 g022(.A(n265_1), .B(n264), .Y(n266));
INVX1 g024(.A(G45), .Y(n268));
AOI21X1 g029(.A0(n272), .A1(n271), .B0(G44), .Y(n273));
XOR2X1 g031(.A(n274), .B(G44), .Y(n275_1));
AOI21X1 g043(.A0(n286), .A1(n281), .B0(n280_1), .Y(n287));
AND2X1 g011(.A(G90), .B(G64), .Y(n255_1));
NOR2X1 g012(.A(G90), .B(G8), .Y(n256));
NAND2X1 g008(.A(G90), .B(G64), .Y(n252));
OR2X1 g009(.A(G90), .B(G8), .Y(n253));
OR4X1 g020(.A(n254), .B(n251), .C(n263), .D(n257), .Y(n264));
OR2X1 g021(.A(G90), .B(G3), .Y(n265_1));
OR4X1 g027(.A(n254), .B(n251), .C(n270_1), .D(n257), .Y(n271));
OR2X1 g028(.A(G90), .B(G2), .Y(n272));
AND2X1 g030(.A(n272), .B(n271), .Y(n274));
AOI21X1 g036(.A0(n279), .A1(n278), .B0(n276), .Y(n280_1));
NOR3X1 g037(.A(n279), .B(n278), .C(n276), .Y(n281));
NOR3X1 g042(.A(n285_1), .B(n284), .C(n282), .Y(n286));
INVX1 g019(.A(G77), .Y(n263));
INVX1 g026(.A(G76), .Y(n270_1));
INVX1 g032(.A(G43), .Y(n276));
NOR3X1 g034(.A(n277), .B(n257), .C(n254), .Y(n278));
NOR2X1 g035(.A(G90), .B(G1), .Y(n279));
INVX1 g038(.A(G42), .Y(n282));
NOR3X1 g040(.A(n283), .B(n257), .C(n254), .Y(n284));
NOR2X1 g041(.A(G90), .B(G0), .Y(n285_1));
NAND2X1 g033(.A(G90), .B(G75), .Y(n277));
NAND2X1 g039(.A(G90), .B(G74), .Y(n283));

endmodule

```

Figura 57: Descrição do s1423 (saída n90)


```

//Converted to Combinational (Partial output: n75) , Module name: s1488_n75
module s1488_n75 ( v1, CLR, v7, v8, v2, v12, v9, v3, v11, v10, v0, v6, v5, v4,
n75 );
input v1, CLR, v7, v8, v2, v12, v9, v3, v11, v10, v0, v6, v5, v4;
output n75;
wire n267, n382, n395, n45, n376, n381, n394, n386, n374, n375, n380, n103, n379,
n393, n218, n392, n385, n72, n184, n383, n371, n373, n70_1, n324, n143, n64, n363,
n83, n377, n378, n57, n63, n217, n387, n391, n390, n68, n384, n82, n104, n372,
n69, n54, n111, n142, n47, n135, n154, n388, n389, n90;
AOI21X1 g351(.A0(n395), .A1(n382), .B0(n267), .Y(n75));
INVX1 g222(.A(CLR), .Y(n267));
OAI21X1 g337(.A0(n381), .A1(n376), .B0(n45), .Y(n382));
AOI21X1 g350(.A0(n386), .A1(v7), .B0(n394), .Y(n395));
INVX1 g000(.A(v7), .Y(n45));
AOI21X1 g331(.A0(n375), .A1(n374), .B0(v8), .Y(n376));
OAI21X1 g336(.A0(n379), .A1(n103), .B0(n380), .Y(n381));
AOI22X1 g349(.A0(n392), .A1(v2), .B0(n218), .B1(n393), .Y(n394));
OAI22X1 g341(.A0(n383), .A1(n184), .B0(n72), .B1(n385), .Y(n386));
AOI21X1 g329(.A0(n373), .A1(n371), .B0(v12), .Y(n374));
AOI22X1 g330(.A0(n143), .A1(n324), .B0(n70_1), .B1(v9), .Y(n375));
AOI22X1 g335(.A0(n83), .A1(n363), .B0(n64), .B1(v9), .Y(n380));
INVX1 g058(.A(v2), .Y(n103));
AOI21X1 g334(.A0(n378), .A1(n72), .B0(n377), .Y(n379));
AOI22X1 g348(.A0(n63), .A1(v12), .B0(v3), .B1(n57), .Y(n393));
INVX1 g173(.A(n217), .Y(n218));
AOI22X1 g347(.A0(n390), .A1(n391), .B0(n387), .B1(v8), .Y(n392));
MX2X1 g340(.A(n184), .B(n384), .S0(n68), .Y(n385));
INVX1 g027(.A(v8), .Y(n72));
NAND2X1 g139(.A(v11), .B(n82), .Y(n184));
MX2X1 g338(.A(n63), .B(v8), .S0(n68), .Y(n383));
NOR2X1 g326(.A(n68), .B(v11), .Y(n371));
AOI21X1 g328(.A0(n372), .A1(n104), .B0(v10), .Y(n373));
INVX1 g025(.A(n69), .Y(n70_1));
NOR2X1 g279(.A(n54), .B(v12), .Y(n324));
OAI21X1 g098(.A0(n142), .A1(v0), .B0(n111), .Y(n143));
AND2X1 g019(.A(v11), .B(v12), .Y(n64));
NOR2X1 g318(.A(n72), .B(v12), .Y(n363));
NOR2X1 g038(.A(v9), .B(v10), .Y(n83));
NOR3X1 g332(.A(n47), .B(v10), .C(v1), .Y(n377));
OAI21X1 g333(.A0(n135), .A1(v3), .B0(n63), .Y(n378));
NOR2X1 g012(.A(v10), .B(v12), .Y(n57));
INVX1 g018(.A(v9), .Y(n63));
NOR2X1 g172(.A(n72), .B(v11), .Y(n217));
OAI22X1 g342(.A0(n184), .A1(n68), .B0(n45), .B1(n154), .Y(n387));
NOR2X1 g346(.A(v7), .B(n104), .Y(n391));
NAND2X1 g345(.A(n389), .B(n388), .Y(n390));
INVX1 g023(.A(v10), .Y(n68));
AOI21X1 g339(.A0(n63), .A1(v12), .B0(n104), .Y(n384));
INVX1 g037(.A(v12), .Y(n82));
INVX1 g059(.A(v11), .Y(n104));
OR2X1 g327(.A(v3), .B(v6), .Y(n372));
NOR3X1 g024(.A(n68), .B(v11), .C(v12), .Y(n69));
NAND2X1 g009(.A(v4), .B(v5), .Y(n54));
OR2X1 g066(.A(v10), .B(v11), .Y(n111));
NAND2X1 g097(.A(v10), .B(v11), .Y(n142));
NAND2X1 g002(.A(v11), .B(v12), .Y(n47));
NAND2X1 g090(.A(v10), .B(v12), .Y(n135));
NAND2X1 g109(.A(n63), .B(v12), .Y(n154));
NAND3X1 g343(.A(n90), .B(v8), .C(n82), .Y(n388));
NAND3X1 g344(.A(n68), .B(v12), .C(v1), .Y(n389));
AND2X1 g045(.A(v4), .B(v5), .Y(n90));
endmodule

```

Figura 58: Descrição do s1488 (saída n75)

```

//Converted to Combinational (Partial output: n70) , Module name: s1494_n70
module s1494_n70 ( v1, CLR, v7, v10, v9, v12, v2, v8, v11, v5, v4, v3, v6, v0, n70 );
input v1, CLR, v7, v10, v9, v12, v2, v8, v11, v5, v4, v3, v6, v0;
output n70;
wire n267, n365, n374, n68, n346, n364, n373, n116, n369, n62, n341, n345, n363, n349,
n372, n93, n139, n160, n368, n244, n70_1, n67, n84, n45, n343, n344, n362, n354, n348,
n174, n124, n272, n371, n87, n119, n366, n367, n94, n137, n342, n357, n360, n361, n350,
n353, n304, n347, n107, n74, n370, n111, n136, n100, n355, n356, n358, n359, n123, n86,
n352, n351, n53, n54, n64, n140, n254, n83;
AOI21X1 g329(.A0(n374), .A1(n365), .B0(n267), .Y(n70));
INVX1 g221(.A(CLR), .Y(n267));
OAI21X1 g319(.A0(n364), .A1(n346), .B0(n68), .Y(n365));
AOI22X1 g328(.A0(n369), .A1(n116), .B0(v7), .B1(n373), .Y(n374));
INVX1 g023(.A(v7), .Y(n68));
AOI21X1 g300(.A0(n345), .A1(n341), .B0(n62), .Y(n346));
OAI21X1 g318(.A0(n349), .A1(v10), .B0(n363), .Y(n364));
OAI22X1 g327(.A0(n160), .A1(n139), .B0(n93), .B1(n372), .Y(n373));
NOR2X1 g071(.A(v9), .B(v10), .Y(n116));
OAI21X1 g323(.A0(n244), .A1(v12), .B0(n368), .Y(n369));
INVX1 g017(.A(v2), .Y(n62));
NAND4X1 g295(.A(v10), .B(n84), .C(n67), .D(n70_1), .Y(n341));
OAI21X1 g299(.A0(n344), .A1(n343), .B0(n45), .Y(n345));
AOI21X1 g317(.A0(n354), .A1(n93), .B0(n362), .Y(n363));
AOI22X1 g303(.A0(n272), .A1(n124), .B0(n174), .B1(n348), .Y(n349));
AOI22X1 g326(.A0(n87), .A1(n67), .B0(n70_1), .B1(n371), .Y(n372));
INVX1 g048(.A(v8), .Y(n93));
NAND2X1 g093(.A(v11), .B(n67), .Y(n139));
NOR2X1 g114(.A(v9), .B(n45), .Y(n160));
OAI21X1 g322(.A0(n367), .A1(n366), .B0(n119), .Y(n368));
INVX1 g198(.A(n94), .Y(n244));
INVX1 g025(.A(v9), .Y(n70_1));
INVX1 g022(.A(v12), .Y(n67));
INVX1 g039(.A(v11), .Y(n84));
INVX1 g000(.A(v10), .Y(n45));
AOI21X1 g297(.A0(n342), .A1(n70_1), .B0(n137), .Y(n343));
NOR4X1 g298(.A(n70_1), .B(v11), .C(v12), .D(v8), .Y(n344));
NAND3X1 g316(.A(n361), .B(n360), .C(n357), .Y(n362));
OAI21X1 g308(.A0(n353), .A1(v9), .B0(n350), .Y(n354));
AOI22X1 g302(.A0(n74), .A1(n107), .B0(n347), .B1(n304), .Y(n348));
INVX1 g128(.A(v1), .Y(n174));
AND2X1 g078(.A(v4), .B(v5), .Y(n124));
NOR3X1 g226(.A(v9), .B(v11), .C(v12), .Y(n272));
OAI21X1 g325(.A0(n111), .A1(n84), .B0(n370), .Y(n371));
OR2X1 g042(.A(v10), .B(v11), .Y(n87));
INVX1 g073(.A(v3), .Y(n119));
NOR4X1 g320(.A(v8), .B(n67), .C(v6), .D(v7), .Y(n366));
NOR2X1 g321(.A(n93), .B(v12), .Y(n367));
NOR2X1 g049(.A(n93), .B(v11), .Y(n94));
INVX1 g091(.A(n136), .Y(n137));
NAND2X1 g296(.A(v12), .B(v1), .Y(n342));
OAI21X1 g311(.A0(n356), .A1(n355), .B0(n100), .Y(n357));
OAI21X1 g314(.A0(n359), .A1(n358), .B0(v8), .Y(n360));
NAND4X1 g315(.A(v10), .B(n84), .C(v12), .D(v9), .Y(n361));
NAND4X1 g304(.A(v9), .B(n67), .C(n86), .D(n123), .Y(n350));
AOI22X1 g307(.A0(n53), .A1(n45), .B0(n351), .B1(n352), .Y(n353));
NAND2X1 g258(.A(n93), .B(v9), .Y(n304));
INVX1 g301(.A(n54), .Y(n347));
NAND2X1 g062(.A(v11), .B(v12), .Y(n107));
NAND2X1 g029(.A(v8), .B(n62), .Y(n74));
AOI21X1 g324(.A0(v10), .A1(n84), .B0(n67), .Y(n370));
AND2X1 g066(.A(v10), .B(v2), .Y(n111));
AND2X1 g090(.A(v8), .B(v11), .Y(n136));
NAND2X1 g055(.A(v4), .B(v5), .Y(n100));
NOR3X1 g309(.A(n64), .B(v8), .C(n45), .Y(n355));
NOR4X1 g310(.A(n70_1), .B(v10), .C(n84), .D(n93), .Y(n356));
NOR3X1 g312(.A(n87), .B(v12), .C(n119), .Y(n358));
AND2X1 g313(.A(n254), .B(n140), .Y(n359));
NOR2X1 g077(.A(v10), .B(v11), .Y(n123));
INVX1 g041(.A(v6), .Y(n86));
NAND2X1 g306(.A(v11), .B(n83), .Y(n352));
NOR2X1 g305(.A(n45), .B(v12), .Y(n351));
AND2X1 g008(.A(v11), .B(v12), .Y(n53));
NOR2X1 g009(.A(v11), .B(v12), .Y(n54));
OR2X1 g019(.A(v9), .B(v12), .Y(n64));
NAND2X1 g094(.A(v10), .B(v11), .Y(n140));
AND2X1 g208(.A(v9), .B(v12), .Y(n254));
INVX1 g038(.A(v0), .Y(n83));
endmodule

```

Figura 59: Descrição do s1494 (saída n70)

```

//Converted to Combinational (Partial output: n240) , Module name: s5378 n240
module s5378_n240 ( n722gat, n726gat, n842gat, n271gat, n160gat, n337gat, n394gat,
n703gat, n341gat, n240 );
input n722gat, n726gat, n842gat, n271gat, n160gat, n337gat, n394gat, n703gat,
n341gat;
output n240;
wire n776, n772, n774_1, n775, n773, n762, n770, n753, n771, n761, n769_1, n765,
n767, n768, n752, n749_1, n750, n751, n760, n756, n758, n759_1, n763, n764_1,
n766, n711, n748, n744_1, n754_1, n755, n757;
NAND4X1 g156(.A(n775), .B(n774_1), .C(n772), .D(n776), .Y(n240));
NAND3X1 g155(.A(n770), .B(n762), .C(n773), .Y(n776));
NAND3X1 g151(.A(n771), .B(n762), .C(n753), .Y(n772));
NAND3X1 g153(.A(n771), .B(n761), .C(n773), .Y(n774_1));
NAND3X1 g154(.A(n770), .B(n761), .C(n753), .Y(n775));
INVX1 g152(.A(n753), .Y(n773));
INVX1 g141(.A(n761), .Y(n762));
NOR4X1 g149(.A(n768), .B(n767), .C(n765), .D(n769_1), .Y(n770));
NOR4X1 g132(.A(n751), .B(n750), .C(n749_1), .D(n752), .Y(n753));
INVX1 g150(.A(n770), .Y(n771));
NOR4X1 g140(.A(n759_1), .B(n758), .C(n756), .D(n760), .Y(n761));
NOR3X1 g148(.A(n842gat), .B(n726gat), .C(n722gat), .Y(n769_1));
NOR3X1 g144(.A(n764_1), .B(n763), .C(n722gat), .Y(n765));
NOR3X1 g146(.A(n764_1), .B(n726gat), .C(n766), .Y(n767));
NOR3X1 g147(.A(n842gat), .B(n763), .C(n766), .Y(n768));
NOR3X1 g131(.A(n337gat), .B(n160gat), .C(n271gat), .Y(n752));
NOR3X1 g128(.A(n337gat), .B(n748), .C(n711), .Y(n749_1));
NOR3X1 g129(.A(n744_1), .B(n748), .C(n271gat), .Y(n750));
NOR3X1 g130(.A(n744_1), .B(n160gat), .C(n711), .Y(n751));
NOR3X1 g139(.A(n341gat), .B(n703gat), .C(n394gat), .Y(n760));
NOR3X1 g135(.A(n755), .B(n754_1), .C(n394gat), .Y(n756));
NOR3X1 g137(.A(n341gat), .B(n754_1), .C(n757), .Y(n758));
NOR3X1 g138(.A(n755), .B(n703gat), .C(n757), .Y(n759_1));
INVX1 g142(.A(n726gat), .Y(n763));
INVX1 g143(.A(n842gat), .Y(n764_1));
INVX1 g145(.A(n722gat), .Y(n766));
INVX1 g090(.A(n271gat), .Y(n711));
INVX1 g127(.A(n160gat), .Y(n748));
INVX1 g123(.A(n337gat), .Y(n744_1));
INVX1 g133(.A(n703gat), .Y(n754_1));
INVX1 g134(.A(n341gat), .Y(n755));
INVX1 g136(.A(n394gat), .Y(n757));
endmodule

```

Figura 60: Descrição do s5378 (saída n240)

```

//Converted to Combinational (Partial output: n676) , Module name: s9234 n676
module s9234 n676 ( g1, g2, g6, g24, g28, g693, g677, g683, g684, g41, g541, g682,
g681, g676, g702, g699, g688, g536, g689, g685, g698, g516, g236, g465, g48, g230,
g512, g520, g242, g524, g532, g278, g276, g277, g281, g279, g280, g212, g500, g224,
g508, g528, g254, g504, g218, g248, g260, g206, g204, g205, g207, g208, g209, g1,
g10, g18, g14, n676 );
input g1, g2, g6, g24, g28, g693, g677, g683, g684, g41, g541, g682, g681, g676,
g702, g699, g688, g536, g689, g685, g698, g516, g236, g465, g48, g230, g512, g520,
g242, g524, g532, g278, g276, g277, g281, g279, g280, g212, g500, g224, g508, g528,
g254, g504, g218, g248, g260, g206, g204, g205, g207, g208, g209, g1, g10, g18, g14;
output n676;
wire n1011_1, n1497, n1010, n1007, n1009, n1349, n1496, n798, n826_1, n727, n1005,
n1006_1, n1008, n1213, n991_1, n1495, n724, n726_1, n758, n789, n939, n990, n978,
n981_1, n984, n1494, n1122, n870, n716_1, n723, n725, n989, n979, n980, n982, n983,
n964, n869, n866_1, n867, n868, n719, n722, n988, n985, n986_1, n987, n793, n955,
n963, n960, n961_1, n962, n717, n718, n720, n721_1;
MX2X1 g0788(.A(n1497), .B(g693), .S0(n1011_1), .Y(n676));
NOR4X1 g0302(.A(n1009), .B(n1007), .C(g677), .D(n1010), .Y(n1011_1));
MAND2X1 g0787(.A(n1496), .B(n1349), .Y(n1497));
MAND4X1 g0301(.A(n826_1), .B(g684), .C(n798), .D(g683), .Y(n1010));
OR4X1 g0298(.A(n1006_1), .B(g41), .C(n1005), .D(n727), .Y(n1007));
INVX1 g0300(.A(n1008), .Y(n1009));
INVX1 g0640(.A(g541), .Y(n1349));
MAND3X1 g0786(.A(n1495), .B(n991_1), .C(n1213), .Y(n1496));
INVX1 g0090(.A(g682), .Y(n798));
INVX1 g0118(.A(g681), .Y(n826_1));
OAI21X1 g0019(.A0(n726_1), .A1(n724), .B0(g676), .Y(n727));
INVX1 g0296(.A(g702), .Y(n1005));
INVX1 g0297(.A(g699), .Y(n1006_1));
NOR4X1 g0299(.A(n939), .B(n789), .C(n758), .D(g688), .Y(n1008));
INVX1 g0504(.A(g536), .Y(n1213));
NOR4X1 g0282(.A(n984), .B(n981_1), .C(n978), .D(n990), .Y(n991_1));
MX2X1 g0785(.A(n870), .B(n1122), .S0(n1494), .Y(n1495));
AND2X1 g0016(.A(n723), .B(n716_1), .Y(n724));
OAI21X1 g0018(.A0(n723), .A1(n716_1), .B0(n725), .Y(n726_1));
INVX1 g0050(.A(g689), .Y(n758));
INVX1 g0081(.A(g685), .Y(n789));
INVX1 g0230(.A(g698), .Y(n939));
INVX1 g0281(.A(n989), .Y(n990));
XOR2X1 g0269(.A(g236), .B(g516), .Y(n978));
OR2X1 g0272(.A(n980), .B(n979), .Y(n981_1));
MAND2X1 g0275(.A(n983), .B(n982), .Y(n984));
INVX1 g0784(.A(g465), .Y(n1494));
INVX1 g0413(.A(n964), .Y(n1122));
NOR4X1 g0162(.A(n868), .B(n867), .C(n866_1), .D(n869), .Y(n870));
INVX1 g0008(.A(g48), .Y(n716_1));
XOR2X1 g0015(.A(n722), .B(n719), .Y(n723));
INVX1 g0017(.A(g41), .Y(n725));
NOR4X1 g0280(.A(n987), .B(n986_1), .C(n985), .D(n988), .Y(n989));
XOR2X1 g0270(.A(g512), .B(g230), .Y(n979));
XOR2X1 g0271(.A(g242), .B(g520), .Y(n980));
XOR2X1 g0273(.A(g524), .B(n793), .Y(n982));
XOR2X1 g0274(.A(g532), .B(n955), .Y(n983));
OR4X1 g0255(.A(n962), .B(n961_1), .C(n960), .D(n963), .Y(n964));
MAND3X1 g0161(.A(g277), .B(g276), .C(g278), .Y(n869));
INVX1 g0158(.A(g281), .Y(n866_1));
INVX1 g0159(.A(g279), .Y(n867));
INVX1 g0160(.A(g280), .Y(n868));
XOR2X1 g0011(.A(n718), .B(n717), .Y(n719));
XOR2X1 g0014(.A(n721_1), .B(n720), .Y(n722));
XOR2X1 g0279(.A(g500), .B(g212), .Y(n988));
XOR2X1 g0276(.A(g508), .B(g224), .Y(n985));
XOR2X1 g0277(.A(g254), .B(g528), .Y(n986_1));
XOR2X1 g0278(.A(g218), .B(g504), .Y(n987));
INVX1 g0085(.A(g248), .Y(n793));
INVX1 g0246(.A(g260), .Y(n955));
MAND3X1 g0254(.A(g205), .B(g204), .C(g206), .Y(n963));
INVX1 g0251(.A(g207), .Y(n960));
INVX1 g0252(.A(g208), .Y(n961_1));
INVX1 g0253(.A(g209), .Y(n962));
XOR2X1 g0009(.A(g1), .B(g2), .Y(n717));
XOR2X1 g0010(.A(g10), .B(g6), .Y(n718));
XOR2X1 g0012(.A(g14), .B(g18), .Y(n720));
XOR2X1 g0013(.A(g28), .B(g24), .Y(n721_1));
endmodule

```

Figura 61: Descrição do s9234 (saída n676)

```

//Converted to Combinational (Partial output: n594) , Module name: s13207_n594
module s13207_n594 ( g1251, g1477, g1513, g1467, g1472, g1462, g150, g1524, g1519,
g1034, g1494, g1499, g1489, g1481, g1504, g1509, g1514, g174, n594 );
input g1251, g1477, g1513, g1467, g1472, g1462, g150, g1524, g1519, g1034, g1494,
g1499, g1489, g1481, g1504, g1509, g1514, g174;
output n594;
wire n2725, n2746_1, n2649, n2724, n2716, n2743, n2727, n2745, n2723, n2742,
n2726_1, n2744, n2722, n2740, n2741_1, n2307, n2721_1, n2733, n2739, n2720,
n2717_1, n2718, n2719, n2729, n2732, n2738, n2734, n2735, n2728, n2730, n2731_1,
n2736_1, n2737;
NOR2X1 g0616(.A(n2746_1), .B(n2725), .Y(n594));
MX2X1 g0594(.A(n2716), .B(n2724), .S0(n2649), .Y(n2725));
OAI21X1 g0615(.A0(n2745), .A1(n2727), .B0(n2743), .Y(n2746_1));
INVX1 g0519(.A(g1251), .Y(n2649));
XOR2X1 g0593(.A(n2723), .B(g1477), .Y(n2724));
INVX1 g0585(.A(g1477), .Y(n2716));
INVX1 g0612(.A(n2742), .Y(n2743));
INVX1 g0596(.A(n2726_1), .Y(n2727));
XOR2X1 g0614(.A(n2744), .B(g1513), .Y(n2745));
NAND4X1 g0592(.A(g1462), .B(g1472), .C(g1467), .D(n2722), .Y(n2723));
AND2X1 g0611(.A(n2741_1), .B(n2740), .Y(n2742));
AND2X1 g0595(.A(n2307), .B(g150), .Y(n2726_1));
INVX1 g0613(.A(g1524), .Y(n2744));
AND2X1 g0591(.A(n2721_1), .B(g1519), .Y(n2722));
OR2X1 g0609(.A(n2739), .B(n2733), .Y(n2740));
AOI21X1 g0610(.A0(n2307), .A1(g150), .B0(g1251), .Y(n2741_1));
INVX1 g0013(.A(g1034), .Y(n2307));
NOR4X1 g0590(.A(n2719), .B(n2718), .C(n2717_1), .D(n2720), .Y(n2721_1));
AND2X1 g0602(.A(n2732), .B(n2729), .Y(n2733));
NOR4X1 g0608(.A(n2720), .B(n2735), .C(n2734), .D(n2738), .Y(n2739));
NAND4X1 g0589(.A(g1481), .B(g1489), .C(g1499), .D(g1494), .Y(n2720));
INVX1 g0586(.A(g1504), .Y(n2717_1));
INVX1 g0587(.A(g1509), .Y(n2718));
INVX1 g0588(.A(g1514), .Y(n2719));
AND2X1 g0598(.A(n2728), .B(g1494), .Y(n2729));
NOR2X1 g0601(.A(n2731_1), .B(n2730), .Y(n2732));
OR4X1 g0607(.A(n2737), .B(n2736_1), .C(g174), .D(n2719), .Y(n2738));
INVX1 g0603(.A(g1462), .Y(n2734));
INVX1 g0604(.A(g1519), .Y(n2735));
AND2X1 g0597(.A(g1481), .B(g1489), .Y(n2728));
NAND4X1 g0599(.A(g174), .B(g1504), .C(g1499), .D(g1467), .Y(n2730));
NAND4X1 g0600(.A(g1462), .B(g1472), .C(g1477), .D(g1519), .Y(n2731_1));
INVX1 g0605(.A(g1467), .Y(n2736_1));
INVX1 g0606(.A(g1472), .Y(n2737));

endmodule

```

Figura 62: Descrição do s13207 (saída n594)

```

//Converted to Combinational (Partial output: n460) , Module name: s15850_n460
module s15850_n460 ( g1822, g1882, g1834, g1814, g1828, g1840, g1936, g1945,
g1864, g1868, g1861, g1872, g1887, g1845, g1857, g1900, g1891, g1927, g1918,
g1909, n460 );
input g1822, g1882, g1834, g1814, g1828, g1840, g1936, g1945, g1864, g1868, g1861,
g1872, g1887, g1845, g1857, g1900, g1891, g1927, g1918, g1909;
output n460;
wire n2348, n2330, n2342, n2347, n2345, n2346_1, n2341_1, n2320, n2329, n2306_1,
n2331_1, n2313, n2343, n2344, n2314, n2337, n2340, n2319, n2328, n2312, n2308,
n2311_1, n2336_1, n2326_1, n2339, n2318, n2327, n2321_1, n2307, n2309, n2310,
n2335, n2332, n2333, n2334, n2325, n2324, n2338, n2317, n2322, n2323, n2315,
n2316_1;
OAI21X1 g0512(.A0(n2342), .A1(n2330), .B0(n2348), .Y(n460));
NAND4X1 g0511(.A(n2341_1), .B(n2346_1), .C(n2345), .D(n2347), .Y(n2348));
MX2X1 g0493(.A(n2306_1), .B(n2329), .S0(n2320), .Y(n2330));
OR2X1 g0505(.A(n2341_1), .B(n2331_1), .Y(n2342));
AOI21X1 g0510(.A0(n2343), .A1(g1822), .B0(n2313), .Y(n2347));
INVX1 g0508(.A(n2344), .Y(n2345));
INVX1 g0509(.A(n2331_1), .Y(n2346_1));
AOI21X1 g0504(.A0(n2340), .A1(n2337), .B0(n2314), .Y(n2341_1));
NAND2X1 g0483(.A(n2319), .B(n2314), .Y(n2320));
XOR2X1 g0492(.A(n2328), .B(g1882), .Y(n2329));
INVX1 g0469(.A(g1882), .Y(n2306_1));
NOR4X1 g0494(.A(g1828), .B(g1822), .C(g1814), .D(g1834), .Y(n2331_1));
NOR2X1 g0476(.A(n2312), .B(g1840), .Y(n2313));
INVX1 g0506(.A(g1828), .Y(n2343));
NOR3X1 g0507(.A(n2343), .B(g1822), .C(g1814), .Y(n2344));
AOI21X1 g0477(.A0(n2313), .A1(n2311_1), .B0(n2308), .Y(n2314));
OR4X1 g0500(.A(n2326_1), .B(g1945), .C(g1936), .D(n2336_1), .Y(n2337));
NAND4X1 g0503(.A(n2326_1), .B(g1945), .C(g1936), .D(n2339), .Y(n2340));
OAI21X1 g0482(.A0(n2318), .A1(g1840), .B0(n2308), .Y(n2319));
MX2X1 g0491(.A(n2321_1), .B(n2327), .S0(n2319), .Y(n2328));
NAND2X1 g0475(.A(g1834), .B(g1814), .Y(n2312));
NOR4X1 g0471(.A(g1861), .B(g1868), .C(g1864), .D(n2307), .Y(n2308));
NOR2X1 g0474(.A(n2310), .B(n2309), .Y(n2311_1));
NAND4X1 g0499(.A(n2334), .B(n2333), .C(n2332), .D(n2335), .Y(n2336_1));
OAI21X1 g0489(.A0(n2324), .A1(g1828), .B0(n2325), .Y(n2326_1));
NOR4X1 g0502(.A(n2334), .B(n2333), .C(n2332), .D(n2338), .Y(n2339));
INVX1 g0481(.A(n2317), .Y(n2318));
XOR2X1 g0490(.A(n2326_1), .B(g1872), .Y(n2327));
INVX1 g0484(.A(g1887), .Y(n2321_1));
INVX1 g0470(.A(g1845), .Y(n2307));
INVX1 g0472(.A(g1857), .Y(n2309));
NOR2X1 g0473(.A(g1828), .B(g1822), .Y(n2310));
NOR4X1 g0498(.A(g1891), .B(g1900), .C(g1882), .D(g1872), .Y(n2335));
INVX1 g0495(.A(g1927), .Y(n2332));
INVX1 g0496(.A(g1918), .Y(n2333));
INVX1 g0497(.A(g1909), .Y(n2334));
OAI21X1 g0488(.A0(g1828), .A1(g1814), .B0(n2322), .Y(n2325));
INVX1 g0487(.A(n2323), .Y(n2324));
NAND4X1 g0501(.A(g1891), .B(g1900), .C(g1882), .D(g1872), .Y(n2338));
NAND4X1 g0480(.A(n2316_1), .B(g1814), .C(n2315), .D(n2310), .Y(n2317));
INVX1 g0485(.A(g1822), .Y(n2322));
NOR2X1 g0486(.A(n2322), .B(g1814), .Y(n2323));
INVX1 g0478(.A(g1840), .Y(n2315));
INVX1 g0479(.A(g1834), .Y(n2316_1));
endmodule

```

Figura 63: Descrição do s15850 (saída n460)

```

//Converted to Combinational (Partial output: n7962) , Module name: s38417_n7962
module s38417_n7962 ( g1880, g1890, g1909, g1943, g1939, g1924, g1937, g1905, g1925,
g1985, g1991, g1930, g1931, g1988, g2100, g2099, g2101, g1231, g1230, g1928, g2116,
g2114, g2115, g2097, g2096, g2098, g2093, g2095, g2094, g2112, g2106, g2105, g2107,
g2085, g2084, g2086, g2082, g2081, g2083, g2003, g2009, g2006, g544, g1186, g2103,
g2102, g2104, g2118, g2117, g2119, g2088, g2087, g2089, g2091, g2090, g2092, g2079,
g2078, g2080, g1234, g499, g548, g2111, g2113, g545, n7962 );
input g1880, g1890, g1909, g1943, g1939, g1924, g1937, g1905, g1925, g1985, g1991,
g1930, g1931, g1988, g2100, g2099, g2101, g1231, g1230, g1928, g2116, g2114, g2115,
g2097, g2096, g2098, g2093, g2095, g2094, g2112, g2106, g2105, g2107, g2085, g2084,
g2086, g2082, g2081, g2083, g2003, g2009, g2006, g544, g1186, g2103, g2102, g2104,
g2118, g2117, g2119, g2088, g2087, g2089, g2091, g2090, g2092, g2079, g2078, g2080,
g1234, g499, g548, g2111, g2113, g545;
output n7962;
wire n9586, n9589, n9597, n9593, n5315, n9571, n9585, n9588, n9595, n9596, n9592, n9591,
n9570, n5317, n5318, n9584, n9574, n9580, n9587, n9594, n9590, n8048, n8306, n5313,
n5316_1, n9583, n9572, n9573, n9579, n8333, n8337, n9575, n8172, n8046, n8047_1, n8304,
n8305, n5312, n5306, n5307_1, n5314, n9581, n9582, n8295, n8297, n8302, n8309, n8308,
n9578, n8315, n8319, n8344, n8331, n8332, n8335, n8336, n8324, n8326, n8327, n8294,
n8213, n8296, n5305, n2204, n5304, n8300, n8301, n9576, n9577, n8313, n8314, n8317,
n8318, n8343, n8322, n8323, n5303, n5296, n5298_1, n8342, n8341, n5297;
OAI22X1 g4548 (.A0(n9593), .A1(n9597), .B0(n9589), .B1(n9586), .Y(n7962));
NOR3X1 g4536 (.A(n9585), .B(n9571), .C(n5315), .Y(n9586));
INVX1 g4539 (.A(n9588), .Y(n9589));
OAI21X1 g4547 (.A0(n9596), .A1(n9571), .B0(n9595), .Y(n9597));
NOR4X1 g4543 (.A(n9591), .B(n9571), .C(n5315), .D(n9592), .Y(n9593));
INVX1 g0273 (.A(g1880), .Y(n5315));
OAI21X1 g4521 (.A0(n5318), .A1(n5317), .B0(n9570), .Y(n9571));
AOI21X1 g4535 (.A0(n9580), .A1(n9574), .B0(n9584), .Y(n9585));
NOR2X1 g4538 (.A(g1890), .B(n9587), .Y(n9588));
INVX1 g4545 (.A(n9594), .Y(n9595));
OR2X1 g4546 (.A(n9590), .B(g1909), .Y(n9596));
XOR2X1 g4542 (.A(n8306), .B(n8048), .Y(n9592));
INVX1 g4541 (.A(n9590), .Y(n9591));
NOR2X1 g4520 (.A(g1939), .B(g1943), .Y(n9570));
NOR2X1 g0275 (.A(n5316_1), .B(n5313), .Y(n5317));
NOR2X1 g0276 (.A(n5315), .B(g1924), .Y(n5318));
INVX1 g4534 (.A(n9583), .Y(n9584));
AND2X1 g4524 (.A(n9573), .B(n9572), .Y(n9574));
NOR4X1 g4530 (.A(n9575), .B(n8337), .C(n8333), .D(n9579), .Y(n9580));
INVX1 g4537 (.A(g1937), .Y(n9587));
OAI21X1 g4544 (.A0(g1905), .A1(g1890), .B0(n9587), .Y(n9594));
OAI21X1 g4540 (.A0(g1905), .A1(g1937), .B0(n8172), .Y(n9590));
AND2X1 g3003 (.A(n8047_1), .B(n8046), .Y(n8048));
NOR2X1 g3261 (.A(n8305), .B(n8304), .Y(n8306));
NOR3X1 g0271 (.A(n5307_1), .B(n5306), .C(n5312), .Y(n5313));
OAI21X1 g0274 (.A0(n5314), .A1(g1925), .B0(n5315), .Y(n5316_1));
NOR2X1 g4533 (.A(n9582), .B(n9581), .Y(n9583));
NOR3X1 g4522 (.A(n8302), .B(n8297), .C(n8295), .Y(n9572));
NOR4X1 g4523 (.A(n8308), .B(n8305), .C(n8304), .D(n8309), .Y(n9573));
NAND4X1 g4529 (.A(n8344), .B(n8319), .C(n8315), .D(n9578), .Y(n9579));
NOR2X1 g3288 (.A(n8332), .B(n8331), .Y(n8333));
NOR2X1 g3292 (.A(n8336), .B(n8335), .Y(n8337));
OAI21X1 g4525 (.A0(n8327), .A1(n8326), .B0(n8324), .Y(n9575));
INVX1 g3127 (.A(g1890), .Y(n8172));
NAND2X1 g3001 (.A(g1985), .B(g1925), .Y(n8046));
AOI22X1 g3002 (.A0(g1988), .A1(g1931), .B0(g1930), .B1(g1991), .Y(n8047_1));
NOR2X1 g3259 (.A(g2100), .B(n8294), .Y(n8304));
OAI22X1 g3260 (.A0(g2101), .A1(n8296), .B0(n8213), .B1(g2099), .Y(n8305));
INVX1 g0270 (.A(g1925), .Y(n5312));
AOI21X1 g0264 (.A0(n2204), .A1(g1231), .B0(n5305), .Y(n5306));
NOR2X1 g0265 (.A(n5304), .B(g1230), .Y(n5307_1));
INVX1 g0272 (.A(g1928), .Y(n5314));
NOR2X1 g4531 (.A(g2116), .B(n8296), .Y(n9581));
OAI22X1 g4532 (.A0(g2115), .A1(n8294), .B0(n8213), .B1(g2114), .Y(n9582));
NOR2X1 g3250 (.A(g2097), .B(n8294), .Y(n8295));
OAI22X1 g3252 (.A0(g2098), .A1(n8296), .B0(n8213), .B1(g2096), .Y(n8297));
NOR2X1 g3257 (.A(n8301), .B(n8300), .Y(n8302));
OAI22X1 g3264 (.A0(g2095), .A1(n8296), .B0(n8213), .B1(g2093), .Y(n8309));
NOR2X1 g3263 (.A(g2094), .B(n8294), .Y(n8308));
OR2X1 g4528 (.A(n9577), .B(n9576), .Y(n9578));
OR2X1 g3270 (.A(n8314), .B(n8313), .Y(n8315));
NOR2X1 g3274 (.A(n8318), .B(n8317), .Y(n8319));
OAI21X1 g3299 (.A0(g2112), .A1(n8294), .B0(n8343), .Y(n8344));
NOR2X1 g3286 (.A(g2106), .B(n8294), .Y(n8331));
OAI22X1 g3287 (.A0(g2107), .A1(n8296), .B0(n8213), .B1(g2105), .Y(n8332));
NOR2X1 g3290 (.A(g2085), .B(n8294), .Y(n8335));
OAI22X1 g3291 (.A0(g2086), .A1(n8296), .B0(n8213), .B1(g2084), .Y(n8336));
NOR2X1 g3279 (.A(n8323), .B(n8322), .Y(n8324));
NOR2X1 g3281 (.A(g2082), .B(n8294), .Y(n8326));
OAI22X1 g3282 (.A0(g2083), .A1(n8296), .B0(n8213), .B1(g2081), .Y(n8327));
INVX1 g3249 (.A(g2003), .Y(n8294));
INVX1 g3168 (.A(g2009), .Y(n8213));
INVX1 g3251 (.A(g2006), .Y(n8296));
OAI21X1 g0263 (.A0(n5303), .A1(g1231), .B0(n5304), .Y(n5305));
MX2X1 g0257 (.A(g544), .B(n5298_1), .S0(n5296), .Y(n2204));
INVX1 g0262 (.A(g1186), .Y(n5304));
NOR2X1 g3255 (.A(g2103), .B(n8294), .Y(n8300));
OAI22X1 g3256 (.A0(g2104), .A1(n8296), .B0(n8213), .B1(g2102), .Y(n8301));
NOR2X1 g4526 (.A(g2118), .B(n8294), .Y(n9576));
OAI22X1 g4527 (.A0(g2119), .A1(n8296), .B0(n8213), .B1(g2117), .Y(n9577));
NOR2X1 g3268 (.A(g2088), .B(n8294), .Y(n8313));
OAI22X1 g3269 (.A0(g2089), .A1(n8296), .B0(n8213), .B1(g2087), .Y(n8314));
NOR2X1 g3272 (.A(g2091), .B(n8294), .Y(n8317));
OAI22X1 g3273 (.A0(g2092), .A1(n8296), .B0(n8213), .B1(g2090), .Y(n8318));
AOI22X1 g3298 (.A0(n8341), .A1(g2006), .B0(g2009), .B1(n8342), .Y(n8343));
NOR2X1 g3277 (.A(g2079), .B(n8294), .Y(n8322));
OAI22X1 g3278 (.A0(g2080), .A1(n8296), .B0(n8213), .B1(g2078), .Y(n8323));
INVX1 g0261 (.A(g1234), .Y(n5303));
INVX1 g0254 (.A(g499), .Y(n5296));
AND2X1 g0256 (.A(g548), .B(n5297), .Y(n5298_1));
INVX1 g3297 (.A(g2111), .Y(n8342));
INVX1 g3296 (.A(g2113), .Y(n8341));
INVX1 g0255 (.A(g545), .Y(n5297));
endmodule

```

Figura 64: Descrição do s38417 (saída n7962)

```

//Converted to Combinational (Partial output: n7656) , Module name: s38584_n7656
module s38584_n7656 ( g35, g4821, g4826, g4646, g4674, g4688, g4681, g128, g4831,
g4776, g4801, g4793, g4732, g4722, g4727, g4717, g4765, g4698, g4785, g4709,
g4737, g4754, g4743, n7656 );
input g35, g4821, g4826, g4646, g4674, g4688, g4681, g128, g4831, g4776, g4801,
g4793, g4732, g4722, g4727, g4717, g4765, g4698, g4785, g4709, g4737, g4754, g4743;
output n7656;
wire n4620, n7425, n7443, n7424, n7421, n7422_1, n7423, n7442_1, n7420, n7014,
n4967, n7441, n7426, n7432_1, n7440, n7427_1, n7435, n4721, n7431, n7030, n7439,
n4822_1, n7434, n4969_1, n7433, n4720, n7429, n7430, n4970, n7438, n4821, n4813,
n4816, n4819, n7428, n4812_1, n4820, n6720, n7437_1, n7436, n4814, n4815, n4817_1,
n4818;
AOI21X1 g2799(.A0(n7443), .A1(n7425), .B0(n4620), .Y(n7656));
INVX1 g0000(.A(g35), .Y(n4620));
AOI21X1 g2780(.A0(n7422_1), .A1(n7421), .B0(n7424), .Y(n7425));
NAND2X1 g2798(.A(n7442_1), .B(n7423), .Y(n7443));
INVX1 g2779(.A(n7423), .Y(n7424));
INVX1 g2776(.A(n7420), .Y(n7421));
AOI22X1 g2777(.A0(n4967), .A1(g4826), .B0(n7014), .B1(g4821), .Y(n7422_1));
NOR4X1 g2778(.A(g4681), .B(g4688), .C(g4674), .D(g4646), .Y(n7423));
AOI21X1 g2797(.A0(n7432_1), .A1(n7426), .B0(n7441), .Y(n7442_1));
AOI22X1 g2775(.A0(g4681), .A1(g4831), .B0(g128), .B1(g4646), .Y(n7420));
INVX1 g2370(.A(g4674), .Y(n7014));
INVX1 g0347(.A(g4688), .Y(n4967));
AOI21X1 g2796(.A0(n7435), .A1(n7427_1), .B0(n7440), .Y(n7441));
INVX1 g2781(.A(n4721), .Y(n7426));
XOR2X1 g2787(.A(n7431), .B(n7427_1), .Y(n7432_1));
NAND2X1 g2795(.A(n7439), .B(n7030), .Y(n7440));
INVX1 g2782(.A(n4822_1), .Y(n7427_1));
AOI22X1 g2790(.A0(n7433), .A1(g4776), .B0(n4969_1), .B1(n7434), .Y(n7435));
NOR3X1 g0101(.A(g4793), .B(g4801), .C(n4720), .Y(n4721));
AND2X1 g2786(.A(n7430), .B(n7429), .Y(n7431));
INVX1 g2386(.A(n4970), .Y(n7030));
OR4X1 g2794(.A(g4793), .B(g4801), .C(g4776), .D(n7438), .Y(n7439));
NOR4X1 g0202(.A(n4819), .B(n4816), .C(n4813), .D(n4821), .Y(n4822_1));
OR2X1 g2789(.A(g4801), .B(g4776), .Y(n7434));
INVX1 g0349(.A(g4793), .Y(n4969_1));
NAND2X1 g2788(.A(g4793), .B(g4801), .Y(n7433));
INVX1 g0100(.A(g4776), .Y(n4720));
AOI22X1 g2784(.A0(n4812_1), .A1(g4722), .B0(g4732), .B1(n7428), .Y(n7429));
AOI22X1 g2785(.A0(n6720), .A1(g4717), .B0(g4727), .B1(n4820), .Y(n7430));
NOR3X1 g0350(.A(n4969_1), .B(g4801), .C(n4720), .Y(n4970));
AOI21X1 g2793(.A0(n4820), .A1(n7436), .B0(n7437_1), .Y(n7438));
AND2X1 g0201(.A(n4820), .B(g4765), .Y(n4821));
AND2X1 g0193(.A(n4812_1), .B(g4698), .Y(n4813));
NOR3X1 g0196(.A(g4785), .B(n4815), .C(n4814), .Y(n4816));
NOR3X1 g0199(.A(n4818), .B(g4709), .C(n4817_1), .Y(n4819));
NOR2X1 g2783(.A(g4785), .B(n4815), .Y(n7428));
NOR2X1 g0192(.A(g4785), .B(g4709), .Y(n4812_1));
AND2X1 g0200(.A(g4785), .B(g4709), .Y(n4820));
NOR2X1 g2079(.A(n4818), .B(g4709), .Y(n6720));
XOR2X1 g2792(.A(g4785), .B(g4709), .Y(n7437_1));
INVX1 g2791(.A(g4737), .Y(n7436));
INVX1 g0194(.A(g4754), .Y(n4814));
INVX1 g0195(.A(g4709), .Y(n4815));
INVX1 g0197(.A(g4743), .Y(n4817_1));
INVX1 g0198(.A(g4785), .Y(n4818));
endmodule

```

Figura 65: Descrição do s38584 (saída n7656)

APÊNDICE C ANÁLISE FANOUTS SPR-MP

Tabela 17: Análise SPR-MP: Quantidade de *fanouts* analisados

Circuito	100%	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
s27	8	2	6	4	4	2	2	1	1
s208	24	10	14	12	12	6	6	3	3
s298_n64	11	7	4	6	6	3	3	2	2
s344_n61	14	7	7	7	7	4	4	2	2
s349_n66	16	9	7	8	8	4	4	2	2
s382_n69	6	3	3	3	3	2	2	1	1
s386_n49	8	5	3	4	4	2	2	1	1
s400_n64	9	4	5	5	5	3	3	1	1
s420_Z	23	16	7	12	12	6	6	3	3
s444_n109	11	6	5	6	6	3	3	2	2
s510_n78	17	9	8	9	9	5	5	2	2
s641_n178	16	6	10	8	8	4	4	2	2
s713_n177	16	6	10	8	8	4	4	2	2
s820_n95	18	12	6	9	9	5	5	2	2
s832_n90	16	8	8	8	8	4	4	2	2
s838_n215	7	5	2	4	4	2	2	1	1
s953_n104	22	13	9	11	11	6	6	3	3
s1196_G542	22	11	11	11	11	6	6	3	3
s1238_n117	22	14	8	11	11	6	6	3	3
s1423_n90	20	8	12	10	10	5	5	2	2
s1488_n75	18	11	7	9	9	5	5	2	2
s1494_n70	19	11	8	10	10	5	5	2	2
s5378_n240	24	9	15	12	12	6	6	3	3
s9234_n676	4	1	3	2	2	1	1	***	***
s13207_n594	16	13	3	8	8	4	4	2	2
s15850_n460	24	11	13	12	12	6	6	3	3
s38417_n7962	17	7	10	9	9	5	5	2	2
s38584_n7656	17	9	8	9	9	5	5	2	2

Tabela 18: Análise SPR-MP: Confiabilidade dos circuitos com diferentes quantidades de fanouts

Circuito	100%	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
s27	149532.70	112379.48	142380.393	121341.46	136043.62	119992.5054	128148.3232	112379.4785	127173.5273
s208	36594.02	34165.32	37233.87273	34699.36	36993.49	34200.39033	36875.39443	34269.9669	35873.39842
s298_n64	192193.00	187959.70	188412.1591	186589.58	191877.59	185439.5347	188327.166	188974.1305	190019.7902
s344_n61	157679.78	168829.45	157202.8771	168829.45	157202.88	167731.5897	154926.4865	169158.167	171314.7694
s349_n66	166451.15	170526.50	168882.2746	170981.66	170256.82	170788.0691	165903.6876	176947.0368	170684.6964
s382_n69	287091.84	273773.22	293892.034	273773.22	293892.03	282732.9352	285804.9331	282283.3115	284258.5338
s386_n49	592764.42	592078.96	592957.214	592249.76	592806.98	592518.1814	594350.6323	595744.5459	595513.8714
s400_n64	270819.70	266275.35	307402.3005	266275.33	307402.30	294336.1508	302634.4701	293788.2592	295071.0352
s420_Z	60350.17	60386.02	61636.18082	59517.25	60674.33	60957.19117	61604.21561	60957.87675	61566.94977
s444_n109	228776.30	198066.37	226311.9373	198066.37	225182.73	198225.622	225859.3048	198123.5819	224680.4912
s510_n78	75505.54	75538.99	74673.16934	75538.99	74669.04	76411.11811	74757.50181	75293.33419	74933.50169
s641_n178	213167.75	211899.39	215679.9613	211499.49	216509.65	211463.4022	215721.3936	211455.6888	214930.5401
s713_n177	181863.19	181102.29	192950.9414	180843.63	193528.65	180846.1637	192893.088	180826.3071	192596.332
s820_n95	198691.77	198432.08	201297.9191	198425.34	201294.31	197968.7747	201286.1444	197974.7164	201910.2501
s832_n90	174021.76	173446.79	173777.9332	173446.79	173777.93	170743.0386	172795.4731	169221.6295	172130.3426
s838_n215	268344.19	268344.11	268996.1511	268344.11	269349.84	268478.3992	268996.1511	268544.4632	268793.2499
s953_n104	101135.46	101131.16	102527.6346	101129.11	102509.40	101154.505	102526.2828	101717.3125	102515.4537
s1196_G542	190906.93	188785.26	190481.0748	188785.26	190481.07	189368.5596	190783.3758	187911.5641	190136.0342
s1238_n117	219147.64	218828.36	220203.9885	217290.12	219683.63	221046.4304	219615.4326	217480.6997	219532.9401
s1423_n90	352520.65	351378.88	381359.7548	373440.86	351478.05	373384.8882	351415.7902	373331.569	351328.354
s1488_n75	183139.04	181911.02	200716.9323	181908.87	200715.49	186148.0869	200696.9437	199107.9362	199852.3603
s1494_n70	158776.52	158376.75	180557.7399	177411.36	160662.75	177574.9743	182514.2943	179444.135	182107.7936
s5378_n240	50000.25	50000.54	52991.44341	50000.45	52862.60	58767.88807	52627.97885	66428.52392	75048.11593
s9234_n676	249429.18	249450.24	249391.756	249450.25	249391.69	249402.0189	249398.3615	***	***
s13207_n594	163381.79	164721.32	153657.5315	153669.25	164661.40	153661.085	153665.9894	153656.4619	154245.7401
s15850_n460	143920.50	132620.92	145161.5845	132618.81	145118.50	132709.5949	139903.4464	132709.5685	139866.6919
s38417_n7962	153665.38	146763.96	159730.4186	147316.26	158474.29	146764.5495	158711.3666	158994.4636	159361.4555
s38584_n7656	237876.73	230410.83	236988.0186	230410.83	237325.47	242109.8735	236881.7599	242798.7009	236617.7725

Tabela 19: Análise SPR-MP: Diferenças percentuais

Circuito	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
s27	-24.85%	-4.78%	-18.85%	-9.02%	-19.76%	-14.30%	-24.85%	-14.95%
s208	-6.64%	1.75%	-5.18%	1.09%	-6.54%	0.77%	-6.35%	-1.97%
s298_n64	-2.20%	-1.97%	-2.92%	-0.16%	-3.51%	-2.01%	-1.67%	-1.13%
s344_n61	7.07%	-0.30%	7.07%	-0.30%	6.37%	-1.75%	7.28%	8.65%
s349_n66	2.45%	1.46%	2.72%	2.29%	2.61%	-0.33%	6.31%	2.54%
s382_n69	-4.64%	2.37%	-4.64%	2.37%	-1.52%	-0.45%	-1.67%	-0.99%
s386_n49	-0.12%	0.03%	-0.09%	0.01%	-0.04%	0.27%	0.50%	0.46%
s400_n64	-1.68%	13.51%	-1.68%	13.51%	8.68%	11.75%	8.48%	8.95%
s420_Z	0.06%	2.13%	-1.38%	0.54%	1.01%	2.08%	1.01%	2.02%
s444_n109	-13.42%	-1.08%	-13.42%	-1.57%	-13.35%	-1.28%	-13.40%	-1.79%
s510_n78	0.04%	-1.10%	0.04%	-1.11%	1.20%	-0.99%	-0.28%	-0.76%
s641_n178	-0.60%	1.18%	-0.78%	1.57%	-0.80%	1.20%	-0.80%	0.83%
s713_n177	-0.42%	6.10%	-0.56%	6.41%	-0.56%	6.06%	-0.57%	5.90%
s820_n95	-0.13%	1.31%	-0.13%	1.31%	-0.36%	1.31%	-0.36%	1.62%
s832_n90	-0.33%	-0.14%	-0.33%	-0.14%	-1.88%	-0.70%	-2.76%	-1.09%
s838_n215	0.00%	0.24%	0.00%	0.37%	0.05%	0.24%	0.07%	0.17%
s953_n104	0.00%	1.38%	-0.01%	1.36%	0.02%	1.38%	0.58%	1.36%
s1196_G542	-1.11%	-0.22%	-1.11%	-0.22%	-0.81%	-0.06%	-1.57%	-0.40%
s1238_n117	-0.15%	0.48%	-0.85%	0.24%	0.87%	0.21%	-0.76%	0.18%
s1423_n90	-0.32%	8.18%	5.93%	-0.30%	5.92%	-0.31%	5.90%	-0.34%
s1488_n75	-0.67%	9.60%	-0.67%	9.60%	1.64%	9.59%	8.72%	9.13%
s1494_n70	-0.25%	13.72%	11.74%	1.19%	11.84%	14.95%	13.02%	14.69%
s5378_n240	0.00%	5.98%	0.00%	5.72%	17.54%	5.26%	32.86%	50.10%
s9234_n676	0.01%	-0.02%	0.01%	-0.02%	-0.01%	-0.01%	***	***
s13207_n594	0.82%	-5.95%	-5.94%	0.78%	-5.95%	-5.95%	-5.95%	-5.59%
s15850_n460	-7.85%	0.86%	-7.85%	0.83%	-7.79%	-2.79%	-7.79%	-2.82%
s38417_n7962	-4.49%	3.95%	-4.13%	3.13%	-4.49%	3.28%	3.47%	3.71%
s38584_n7656	-3.14%	-0.37%	-3.14%	-0.23%	1.78%	-0.42%	2.07%	-0.53%

Tabela 20: Análise SPR-MP: Tempos de processamento(ms)

Circuito	100%	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
s27	714	2	195	15	42	3	5	2	2
s208	21756004	878	29378248	3214	3759562	74	3826	20	99
s298_n64	708	29	56	15	106	3	17	4	9
s344_n61	48243	26	1322	24	1241	16	37	2	9
s349_n66	52684	88	2448	51	4271	4	61	1	8
s382_n69	56	4	7	2	7	2	2	1	2
s386_n49	43	8	15	3	18	2	3	2	1
s400_n64	390	5	159	15	137	3	20	2	4
s420_Z	8866581	31752	4152	2243	20591	76	1168	12	53
s444_n109	755	14	163	16	270	2	13	2	15
s510_n78	119002	299	46117	305	82732	39	843	20	24
s641_n178	1086663	40	147129	380	9251	18	122	3	7
s713_n177	1115342	24	153020	358	9262	7	55	2	7
s820_n95	254040	1700	2501	240	17548	18	599	5	18
s832_n90	56300	208	55388	149	38059	13	220	6	22
s838_n215	43	3	4	4	6	1	3	1	1
s953_n104	2472164	3207	147407	875	324652	62	2091	13	38
s1196_G542	6924286	881	2058748	970	1970854	58	2474	6	87
s1238_n117	3185367	5627	22814	826	123892	54	1526	5	44
s1423_n90	32704473	134	1649051	5857	15541	28	152	3	8
s1488_n75	182881	755	6653	199	24761	14	485	4	14
s1494_n70	804524	1100	45199	581	163688	32	873	6	24
s5378_n240	10839774	122	94001231	1270	1661798	33	566	3	49
s9234_n676	39	2	19	2	9	1	3	***	***
s13207_n594	89384	1994	25	88	429	8	42	2	13
s15850_n460	308344145	884	12325551	6638	777726	32	539	5	36
s38417_n7962	4868606	133	641161	1962	160921	41	923	8	32
s38584_n7656	111084	183	26863	188	39711	18	480	2	13

Tabela 21: Análise SPR-MP: Tempos de processamento - Diferença Percentual

Circuito	= Entrada	= Meio	50% E	50% S	25% E	25% S	10% E	10% S
s27	-99.719888%	-72.689076%	-97.899160%	-94.117647%	-99.579832%	-99.299720%	-99.719888%	-99.719888%
s208	-99.995964%	35.035129%	-99.985227%	-82.719428%	-99.999660%	-99.982414%	-99.999908%	-99.999545%
s298_n64	-95.903955%	-92.090395%	-97.881356%	-85.028249%	-99.576271%	-97.598870%	-99.435028%	-98.728814%
s344_n61	-99.946106%	-97.259706%	-99.950252%	-97.427606%	-99.966835%	-99.923305%	-99.995854%	-99.981344%
s349_n66	-99.832966%	-95.353428%	-99.903196%	-91.893174%	-99.992408%	-99.884215%	-99.998102%	-99.984815%
s382_n69	-92.857143%	-87.500000%	-96.428571%	-87.500000%	-96.428571%	-96.428571%	-98.214286%	-96.428571%
s386_n49	-81.395349%	-65.116279%	-93.023256%	-58.139535%	-95.348837%	-93.023256%	-95.348837%	-97.674419%
s400_n64	-98.717949%	-59.230769%	-96.153846%	-64.871795%	-99.230769%	-94.871795%	-99.487179%	-98.974359%
s420_Z	-99.641891%	-99.953172%	-99.974703%	-99.767768%	-99.999143%	-99.986827%	-99.999865%	-99.999402%
s444_n109	-98.145695%	-78.410596%	-97.880795%	-64.238411%	-99.735099%	-98.278146%	-99.735099%	-98.013245%
s510_n78	-99.748744%	-61.246870%	-99.743702%	-30.478479%	-99.967227%	-99.291609%	-99.983194%	-99.979832%
s641_n178	-99.996319%	-86.460476%	-99.965031%	-99.148678%	-99.998344%	-99.988773%	-99.999724%	-99.999356%
s713_n177	-99.997848%	-86.280441%	-99.967902%	-99.169582%	-99.999372%	-99.995069%	-99.999821%	-99.999372%
s820_n95	-99.330814%	-99.015509%	-99.905527%	-93.092426%	-99.992915%	-99.764210%	-99.998032%	-99.992915%
s832_n90	-99.630551%	-1.619893%	-99.735346%	-32.399645%	-99.976909%	-99.609236%	-99.989343%	-99.960924%
s838_n215	-93.023256%	-90.697674%	-90.697674%	-86.046512%	-97.674419%	-93.023256%	-97.674419%	-97.674419%
s953_n104	-99.870276%	-94.037329%	-99.964606%	-86.867700%	-99.997492%	-99.915418%	-99.999474%	-99.998463%
s1196_G542	-99.987277%	-70.267721%	-99.985991%	-71.537080%	-99.999162%	-99.964271%	-99.999913%	-99.998744%
s1238_n117	-99.823348%	-99.283787%	-99.974069%	-96.110589%	-99.998305%	-99.952093%	-99.999843%	-99.998619%
s1423_n90	-99.999590%	-94.957720%	-99.982091%	-99.952481%	-99.999914%	-99.999535%	-99.999991%	-99.999976%
s1488_n75	-99.587163%	-96.362115%	-99.891186%	-86.460595%	-99.992345%	-99.734800%	-99.997813%	-99.992345%
s1494_n70	-99.863273%	-94.381895%	-99.927783%	-79.654056%	-99.996022%	-99.891489%	-99.999254%	-99.997017%
s5378_n240	-99.998875%	767.188107%	-99.988284%	-84.669441%	-99.999696%	-99.994778%	-99.999972%	-99.999548%
s9234_n676	-94.871795%	-51.282051%	-94.871795%	-76.923077%	-97.435897%	-92.307692%	***	***
s13207_n594	-97.769176%	-99.972031%	-99.901548%	-99.520048%	-99.991050%	-99.953012%	-99.997762%	-99.985456%
s15850_n460	-99.999713%	-96.002664%	-99.997847%	-99.747773%	-99.999990%	-99.999825%	-99.999998%	-99.999988%
s38417_n7962	-99.997268%	-86.830707%	-99.959701%	-96.694721%	-99.999158%	-99.981042%	-99.999836%	-99.999343%
s38584_n7656	-99.835260%	-75.817399%	-99.830759%	-64.251377%	-99.983796%	-99.567895%	-99.998200%	-99.988297%