UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MICROELETRÔNICA

LUCIANA MENDES DA SILVA

Minimização Lógica por Fusão de Portas

Dissertação apresentada como requisito parcial para a obtenção do grau de Mestre em Microeletrônica

Orientador: Prof. Dr. Ricardo Reis
Co-orientador: Prof Dr. Guilherme Bontorin

Porto Alegre
2018

CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Silva, Luciana Mendes da

Minimização Lógica por Fusão de Portas / Luciana Mendes da Silva. – Porto Alegre: PGMICRO da UFRGS, 2018.

100 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Microeletrônica, Porto Alegre, BR–RS, 2018. Orientador: Ricardo Reis; Co-orientador: Guilherme Bontorin.

1. Minimização Lógica. 2. Algoritmo. 3. EDA. 4. Microeletrônica. I. Reis, Ricardo. II. Bontorin, Guilherme. III. Título.

"Bem-aventurados os humildes, porque herdarão a terra.
Bem-aventurados os que têm fome e sede de justiça, porque serão fartos."
— BÍBLIA, MATHEUS 5:6

AGRADECIMENTOS

Agradeço primeiramente a Deus e aos espíritos de luz que guiaram os meus passos por esta caminhada. Agradeço também aos meus pais pelo apoio incondicional, ao meu namorado pelo incentivo e compreensão e a todos os professores que fizeram parte da minha vida de estudante deste a primeira série do Ensino Fundamental até o Mestrado em Microeletrônica.

RESUMO

Neste trabalho é apresentado um método para redução do número de transistores em circuitos integrados. Foram desenvolvidos um algoritmo e uma ferramenta de EDA baseada no mesmo, denominada de LOMGAM (Logic Minimization by Gate Merging - minimização Lógica por Fusão de Portas), com o objetivo de investigar a redução do número de transistores em um circuito integrado por meio da fusão de portas lógicas. Essa técnica é aplicada sobre uma netlist já mapeado para uma biblioteca de células, e trabalha substituindo conjuntos de portas lógicas interconectadas de fanout unitário por novas portas complexas de função lógica equivalente, independente delas estarem disponíveis em uma biblioteca. A ferramenta desenvolvida, LOMGAM é usada para explorar a aplicação de um conjunto de parâmetros na realização da fusão de portas, e observar o comportamento da aplicação destes sobre o número final de transistores e de conexões do circuito.

O LOMGAM é composto por 7 etapas: Identificação das Interconexões e das Portas Combinacionais; Identificação do Fanout; Geração das Equações Booleanas; Processo de Fusão; Conversão para CMOS; Geração da Netlist minimizada no Formato eqn e Geração da netlist no formato Spice. O suporte a três tipos de parâmetros de controle foi implementado: Índice de Fusão de Portas (Gate Merging Index - GMI), Quantidade Máxima de Transistores em Série (QTMS), e Número Máximo de Inversores (NMI) nas entradas das portas aglutinadas. Sendo que o usuário pode adotar separadamente pelo uso do GMI ou do QMTS. Através da aplicação do LOMGAM verificou-se uma redução média de 11% no número de transistores e de 26,17%, em média, da quantidade de interconexões para o parâmetro Quantidade Máxima de Transistores em Série (QMTS). Para o parâmetro Índice de Fusão de Portas (GMI), ocorreu uma redução média de 6,89% na quantidade de transistores e uma redução média de 13,31% na quantidade de interconexões. Além disso, observa-se que o tempo de execução da ferramenta não aumenta exponencialmente em relação ao aumento do Índice de Fusão de Portas (GMI).

Palavras-chave: Minimização Lógica. Algoritmo. EDA. Microeletrônica.

Logic Minimization by Gate Merging

ABSTRACT

In this work, we introduce a method to reduce the number of transistors in integrated circuits. We have developed an algorithm and an EDA tool based on it, called LOMGAM (Logic Minimization by Gate Merging), they have goal to investigate the reduction of the number of transistors in an integrated circuit by merging logic gate. This technique is applied over a netlist already mapped to a cell library, and it works by replacing interconnected sets of logic gates with unitary fanout for new complex gates with equivalent logic function, regardless of whether they are available in a library. The tool developed, LOMGAM, is used to explore the application of a set of parameters in the realization of the gate merging process, and it observes the behavior of the application of these on the final number of transistors and circuit connections.
The LOMGAM is composed of 7 steps: Identification of interconnections and Combinational Gates; Identification of Fanout; Generation of Boolean Equations; Gate Merging Process; Conversion to CMOS Technology; Generation of Optimized Netlist in eqn Format and Generation of Optimized Netlist in Spice Format. The support to three types of control parameters were implemented: Gate Merging Index (GMI), Maximum Number of Transistors in Series (QTMS), and Maximum Number of Inverters (NMI) in the inputs of the merged gates. Being that the user can adopt separately by the use of GMI or QMTS. Through the application of LOMGAM, we observe an average reduction of 11 % in the number of transistors and 26.17 %, on average, of the number of interconnections for the parameter Maximum Quantity of Transistors in Series (QMTS). For the parameter Gate Merging Index (GMI), there was an average reduction of 6.89 % in the number of transistors and an average reduction of 13.31 % in the amount of interconnections. In addition, we observed that the execution time of tool does not increase exponentially in relation to the increase of the Gate Merging Index (GMI).

Keywords: Logic minimization, algorithm, EDA, Microelectronics.

LISTA DE ABREVIATURAS E SIGLAS

ABC	*A System for Sequential Synthesis and Verification*
AIG	Grafo de Inversores e portas E
ASTG	Grafo de Transição de Sinal
BDD	Diagrama de Decisão Binária
BBDD	Diagrama de Decisão Binária Bicondicional
BLIF	*Berkeley Logic Interchange Format*
CAD	Computer-Aided Design
CMOS	Metal-Óxido-Semicondutor Complementar
DAG	Grafo Acíclico Direto
EDA	Design Eletrônico Automático
ELIS	Ambiente para Síntese Lógica
FRAIG	Grafo de Inversores e portas E Funcionalmente Reduzido
GMI	Índice de Fusão de Porta
LFLS	Síntese Lógica livre de biblioteca
LOMGAM	Minimização Lógica por Fusão de Portas
MB	Baseado em Multiplexadores
MIG	Grafo de Porta Majoritária e Inversores
MIS	Sistema Multi Nível de minimização Lógica
MOTO-X	*Multiple Output Transistor Optimization with bridging*
MVSIS	Síntese Lógica com Multi Valor
NMI	Número Máximo de Inversores nas Entradas das Portas Aglutinadas
OBDD	Diagrama de Decisão Binária Ordenado
PGA	Vetores de Portas Programáveis
QCA	*Quantum dot Cellular Automata*

QMTS	Quantidade Máxima de Transistores em Série
ROBDD	Diagrama de Decisão Binária Ordenado e Reduzido
ROBBDD	Diagrama de Decisão Binária Bicondicional Ordenado e Reduzido
SIS	Sistema Sequencial Interativo
STG	Gráfico de Transição de Estados
TABA	Atribuição de Transistor com BDDs representando Portas Complexas
TSBDD	Diagrama de Decisão Binária com Terminal Suprimido
TNG	Grafo de Rede de Transistores Genéricos
TLU	*Table Look up*
VIRMA	Ferramenta de Mapeamento Tecnológico Virtual
VLSI	Integração em Escala Muito Grande

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

O aumento da densidade de integração de transistores superou um milhão no final dos anos 1980 e nos dias atuais já atingiu bilhões (MANES; PELOSI, 2015). Devido a isso, foram desenvolvidos métodos de automatização do projeto de circuitos integrados, estes são conhecidos como ferramentas de EDA. Estas contribuíram para a redução do tempo de projeto além de proporcionar a criação de circuitos otimizados de alta qualidade e praticamente livres de erros (DE MICHELI, 1994). Porém os desafios da indústria de EDA não param de crescer por causa da crescente complexidade dos circuitos projetados, dos novos processos tecnológicos e das novas metodologias de projeto, como projetos de circuitos independentes de bibliotecas de células.

Em termos econômicos a indústria de circuitos VLSI envolve altos custos tanto na fase projeto, com mão-de-obra altamente qualificada, quanto no processo de fabricação. No caso, da manufatura há o custo com a produção das máscaras, processamento do silício, testes e encapsulamento. Esses custos são proporcionais ao volume de chips produzidos e ao tamanho dos mesmos, ou seja, quanto menor a área do chip mais circuitos podem ser produzidos no mesmo wafer de silício (RABEY; CHANDRAKASAN; NIKOLIC, 2003).

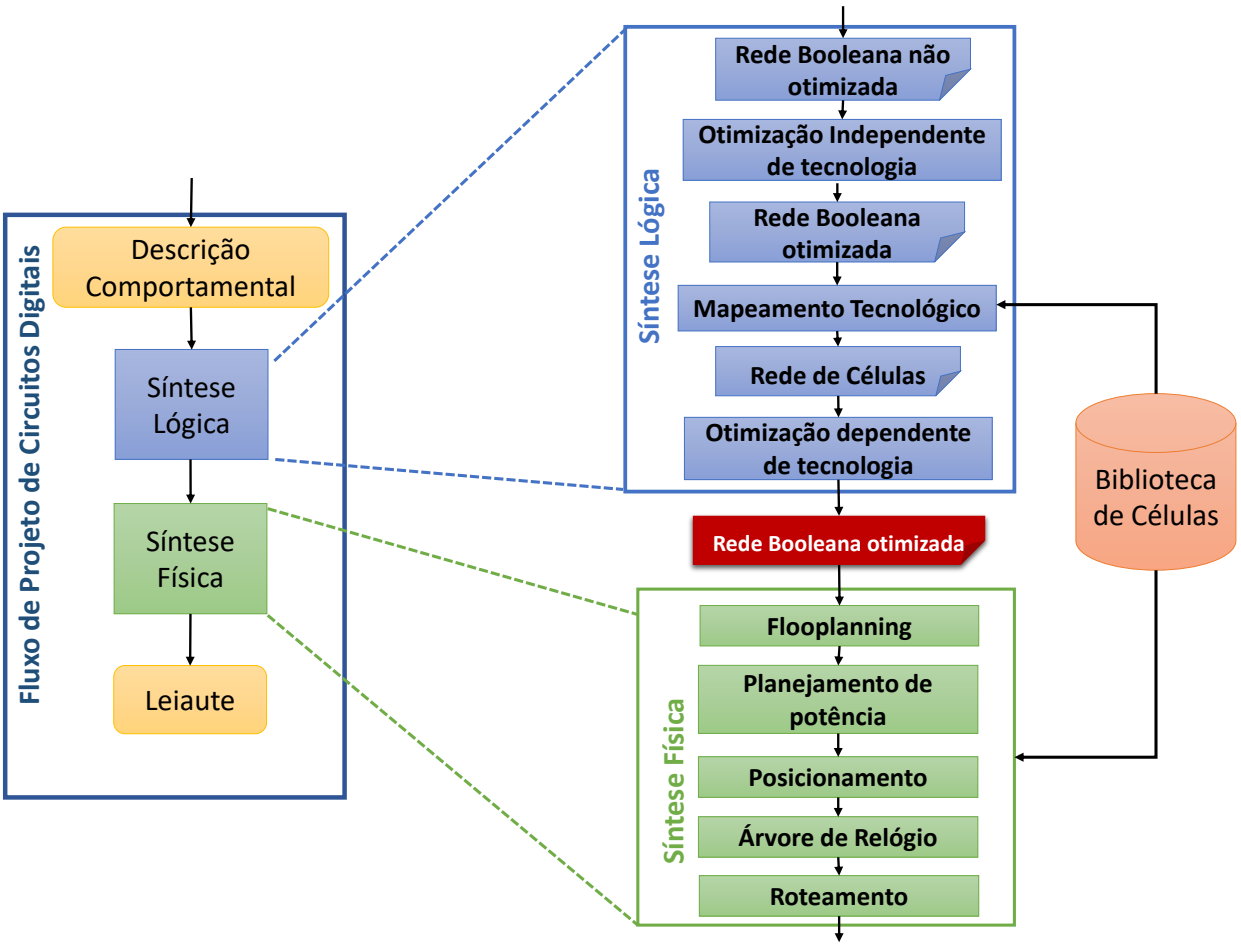
Uma biblioteca de células é desenvolvida e caracterizada uma única vez para um determinado processo tecnológico e pode ser utilizada em um número incontável de vezes e em projetos diferentes. Porém elas possuem um número limitado de funções lógicas, que geralmente não ultrapassa a 150 funções (WESTE; HARRIS, 2011), (REIS, 2011). Pois uma biblioteca de células é composta por funções básicas: inversores, AND, OR; e funções complexas como AOI, OAI, multiplexadores, decodificadores e somadores. Isso faz com que durante o mapeamento tecnológico uma função que poderia ser representada por uma única célula, se esta não existir na biblioteca, tem que ser representada por células equivalentes, fato que contribui para um aumento na quantidade de transistores e interconexões (WESTE; HARRIS, 2011), (CONCEIÇÃO et al., 2017).

Além disso, as células que compõe uma biblioteca possuem altura fixa e posições determinadas para as linhas de alimentação, isso facilita a distribuição da alimentação bastando colocar uma célula encostada na outra, lado a lado (RABEY; CHANDRAKASAN; NIKOLIC, 2003). Outro fator importante é o uso da largura mínima do transistor, para, deste modo, reduzir a capacitância da célula (RABEY; CHANDRAKASAN; NIKOLIC, 2003) (WESTE; HARRIS, 2011) (SCARTEZZINI,).

O fluxo de projeto de um circuito digital VLSI inicia com uma descrição RTL do cir-

cuito, esta etapa consiste em descrever o circuito utilizando uma linguagem de descrição de hardware como o VHDL ou o Verilog, que serve de entrada para a Síntese Lógica. Na figura 1.1 mostra o fluxo completo do projeto de circuitos digitais VLSI. Nesta figura, é mostrado que o fluxo pode ser divido em duas grandes fases: Síntese Lógica e Síntese Física (SCARTEZZINI,).

Figura 1.1 – Fluxo de projeto de circuitos digitais com uso de biblioteca de células.



Fonte:(ALEGRETTI,) (SCARTEZZINI,)

1.1 Síntese Lógica

A Síntese Lógica, a partir da descrição comportamental do circuito, realiza uma minimização do circuito e o mapeamento do mesmo para as células de uma biblioteca; em seguida, levando em consideração as funções lógicas disponíveis na biblioteca, realiza um novo processo de minimização (SCARTEZZINI,). O quadro azul, chamado de Síntese Lógica, da figura 1.1 apresenta as etapas que compõe essa fase:

- *Minimização independente de tecnologia*: consiste na minimização lógica, compartilhamento de recursos e subexpressões booleanas e eliminação de funções desnecessárias (SCARTEZZINI,);
- *Mapeamento Tecnológico*: é o mapeamento das funções booleanas para as células presentes na biblioteca respeitando as restrições de tempo e de área (SCARTEZZINI,);
- *Minimização dependente de tecnologia*: é um processo de minimização que respeita as restrições impostas pelo usuário em termos de atraso e de área, bem como as causadas pela biblioteca devido ao número limitado de células.

O resultado da Síntese Lógica é uma rede booleana minimizada descrita em Verilog Estruturado ou VHDL. Este arquivo serve de entrada na fase de Síntese Física.

O projeto de circuitos digitais sem o uso de uma biblioteca de células ou utilizando uma biblioteca virtual descreve as funções lógicas como uma rede de transistores, deste modo, qualquer função booleana pode ser representada (SCARTEZZINI,).

Na Síntese Lógica independente de biblioteca de células há uma maior capacidade de minimização porque não há um número limitado de funções lógicas como ocorre quando é utilizada uma biblioteca. XUE; AL-KHALILI; ROZON apresentam uma metodologia de mapeamento tecnológico que tem por base uma árvore de transistores. Esse algoritmo é formado por quatro etapas: transformação lógica orientada a tempo, particionamento lógico e mapeamento de porta. Em comparação com o Synopsys Design Analizer o algoritmo proporcionou uma melhora de 42% na área e de 43% na potência para uma tecnologia de 180nm (XUE; AL-KHALILI; ROZON, 2005).

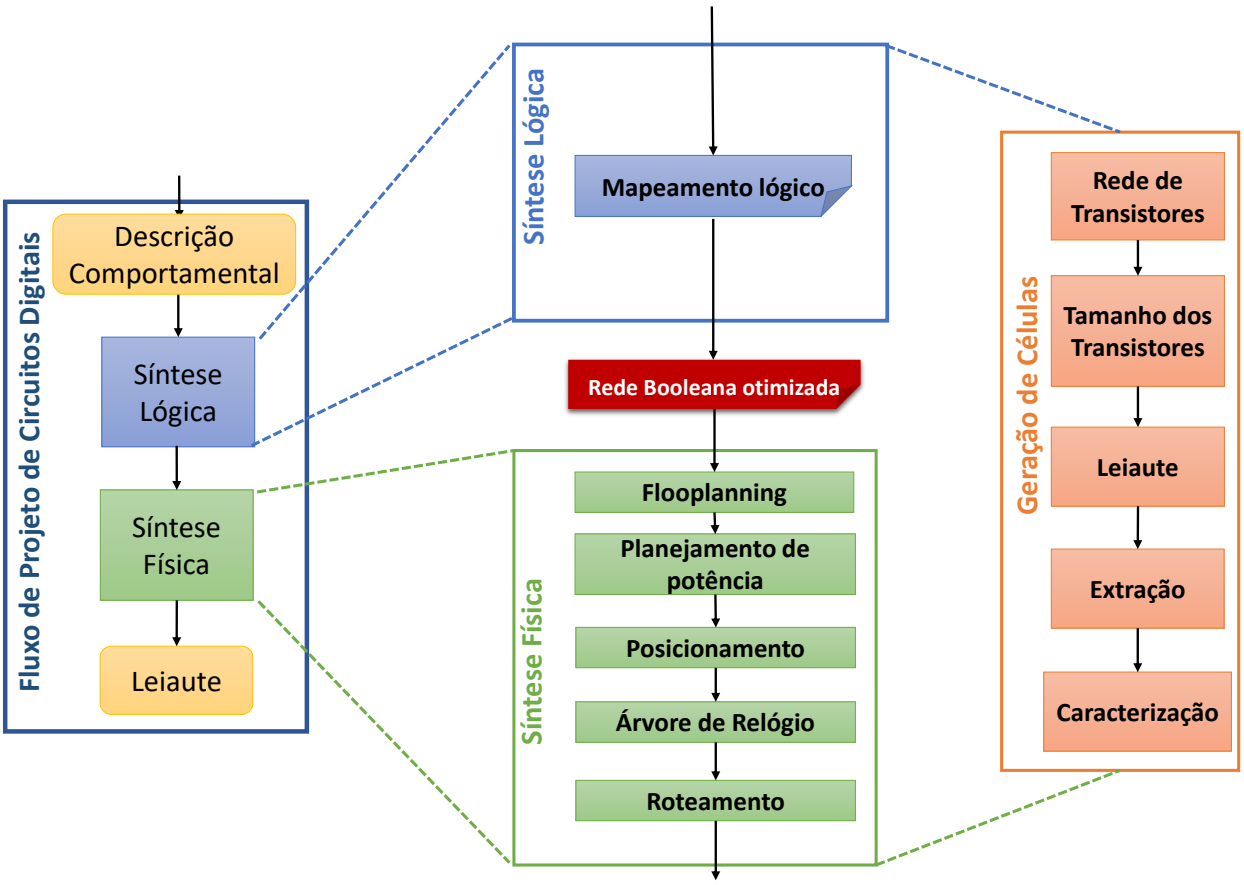
O fluxo de projeto independente de biblioteca é ilustrado na figura 1.2. A fase de Síntese Lógica consiste no mapeamento lógico, ou seja, mapear a descrição RTL do circuito em portas lógicas. De acordo com essas equações as redes de transistores são definidas e o processo de geração de células é iniciado (SCARTEZZINI,). Este processo de confecção das células é apresentado no quadro chamado de Geração de Células da figura 1.2, sendo composto por 4 etapas:

- *Dimensionamento do transistor*: etapa que define o tamanho dos transistores que compõem a rede. Pequenos tamanhos de transistores reduzem a área e a capacitância interna das células. Porém quando utilizados com valores altos de fanout tem sua capacidade de carga reduzida o que aumenta o atraso. Quando é utilizado transistores maiores, o atraso é reduzido, em contrapartida, ocorre um aumento na capacitância e na área (SCARTEZZINI,);

- *Leiaute*: é a etapa de desenho da célula utilizando retângulos, este desenho é posteriormente utilizado no processo de fabricação. Para esse processo são usados geradores de células como ASTRAN (ZIESEMER,), por exemplo (SCARTEZZINI,);
- *Extração*: consiste em extrair e avaliar os componentes parasitas que podem alterar o funcionamento da célula (SCARTEZZINI,) (STEMMER,);
- *Caracterização*: etapa que determina o comportamento da célula através da aplicação de parâmetros como inclinação de entrada, carga de saída, temperatura e tensão. Com os dados obtidos são criados modelos que são utilizados nas Sínteses Lógica e Física (SCARTEZZINI,).

O quadro chamado de Síntese Física da figura 1.2, mostra que a fase de Síntese Física desta metodologia é igual à descrita anteriormente para o método que usa biblioteca de células, portanto, tem as mesmas etapas.

Figura 1.2 – Fluxo de projeto de circuitos digitais sem o uso de biblioteca de células.



Fonte:(ALEGRETTI,) (SCARTEZZINI,)

1.2 Síntese Física

A Síntese Física compreende a elaboração do leiaute do circuito, o qual consiste em um desenho que possui informações sobre o posicionamento das células e macro blocos, distribuição da grade de alimentação e de relógio e também dos elementos parasitas que compõem o circuito (LIENIG et al., 2011). A Síntese Física é formada pelas etapas apresentadas a seguir:

- *Floorplanning*: nesta etapa são determinadas as posições dos macroblocos, pinos externos e dos módulos de subcircuito (LIENIG et al., 2011);
- *Planejamento de Potência*: distribui as redes de alimentação (Vdd) e terra (Gnd) pelo chip (LIENIG et al., 2011);
- *Posicionamento*: etapa responsável por encontrar espaços para a colocação das células dentro de cada bloco (LIENIG et al., 2011);
- *Árvore de Relógio*: realiza a distribuição do sinal de relógio por todo o chip respeitando as restrições de atraso e de *skew* (LIENIG et al., 2011);
- *Roteamento*: este é dividido em duas subetapas: Roteamento Global (faz a alocação de recursos para as conexões como controles de roteamento para os canais e as caixas de distribuição (switch boxes)) e o Roteamento Detalhado (faz a atribuição de rotas para as camadas de metais) (LIENIG et al., 2011).

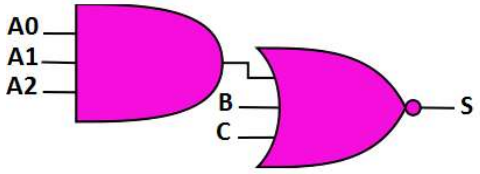
1.3 Motivação

A principal motivação deste trabalho é reduzir o número de transistores presentes no circuito, pois uma quantidade menor de transistores contribui para uma redução no número de conexões e de vias utilizados no leiaute e também para a redução do consumo estático (REIS, 2011), (CONCEIÇÃO et al., 2017). A redução no número de interconexões pode contribuir para uma redução na capacitância e na resistência em alguns circuitos. Devido ao processo de fusão de portas (criação de portas complexas), proposto neste trabalho, ocorre uma diminuição na quantidade de células no circuito investigado e consequentemente de interconexões (SILVA; BONTORIN; REIS, 2015). Esse processo de produção de novas portas consiste em uma minimização lógica independente de biblioteca de células e de tecnologia. A figura 1.3 apresenta um exemplo que compara duas implementações de uma mesma função booleana em CMOS: uma utilizando uma biblioteca de células e a outra não. Observando, na figura 1.3, a porta AND de três entradas é possível observar, que em CMOS, esta porta lógica é composta por uma

NAND de três entradas seguida de um inversor na saída, portanto, a AND3 contém 8 transistores. Somando esses 8 transistores com os 6 transistores da NOR3 é obtida uma quantidade de 14 transistores, como mostra a tabela do item a) da referida figura. Porém, quando a função booleana em questão é representada por uma porta complexa ocorre uma redução de 4 transistores, como mostra o item b) da figura 1.3. Esse exemplo mostra que o uso de uma porta complexa reduz a quantidade de transistores e não causa grande impacto na média do tempo de transição, no atraso e da potência dinâmica, no caso de uma tecnologia de 65nm(SCARTEZZINI,).

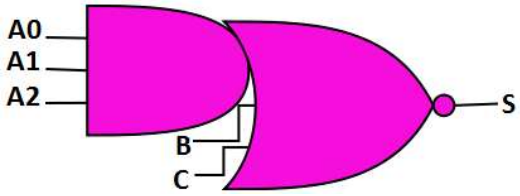
Figura 1.3 – Implementação da função booleana S=!((((A0*A1)*A2)+B)+C) em 65nm

a) Usando uma biblioteca de células



Quantidade de Transistores (Unid.)	Média de Tempo de Transição (ns)	Média do Atraso (ns)	Média da Potência Dinâmica (nW)
14	0,0507449	0,052095	0,00946828

b) Sem o uso de uma biblioteca de células



Quantidade de Transistores (Unid.)	Média de Tempo de Transição (ns)	Média do Atraso (ns)	Média da Potência Dinâmica (nW)
10	0,0626034	0,0558041	0,0092566

Fonte:(SCARTEZZINI,)

Este trabalho é organizado do seguinte modo: capítulo 2 apresenta as metodologias de minimização (em dois níveis e em multiníveis) durante a Síntese Lógica; o capítulo 3 descreve algumas técnicas baseadas em grafos que são utilizadas para representar e minimizar redes booleanas durante a Síntese Lógica como AIG, BDD e MIG, por exemplo; o capítulo 4 aborda

ferramentas acadêmicas para Síntese Lógica que abrangem todo fluxo desta ou apenas alguma etapa da mesma como o mapeamento tecnológico; o capítulo 5 detalha metodologia proposta, ou seja, a minimização lógica por fusão de portas; o capítulo 6 apresenta os resultados obtidos e, por último, o capítulo 7 contém as conclusões e os trabalhos futuros.

2 MINIMIZAÇÃO DURANTE A SÍNTESE LÓGICA

2.1 Minimização em Dois Níveis

A minimização em dois níveis inicialmente era utilizada para minimização lógica de circuitos baseados em PLA. Atualmente essa técnica é usada na etapa de minimização lógica independente de tecnologia em multiníveis (SCHEFFER; LAVAGNO; MARTIN, 2006).

Esse tipo de minimização é uma instância do *Unate Covering Problem* (UCP). Nesse caso, uma função de n entradas e uma saída possui uma matriz de cobertura com O(2^n) linhas (mintermos) e O($(3n)/n$) colunas (primos da função). Uma solução típica realiza várias iterações para estabelecer a dominância de linha e coluna e também para extrair os primos essenciais até que a matriz não possa mais ser reduzida. A partir disso é utilizado o algoritmo de Quine-McCluskey (SCHEFFER; LAVAGNO; MARTIN, 2006). Uma outra técnica que também realiza esse tipo de simplificação utiliza uma representação do circuito em ROBDD, o qual é baseado no BDD, ambos serão abordados no capítulo 3. Para solucionar UCPs, o ROBDD é combinado com a Relaxação Lagrangiana (SCHEFFER; LAVAGNO; MARTIN, 2006). A ferramenta acadêmica EXPRESSO é um exemplo de exploração da minimização em dois níveis. Basicamente, ela apresenta três tipos de operações: *Redução*, que reduz os cubos de modo ordenado; *Expansão*, que expande os cubos em primos e remove os que são cobertos por outros; *Irredundante*, que remove os cubos redundantes durante a etapa de cobertura (SCHEFFER; LAVAGNO; MARTIN, 2006) e (KATZ, 1994).

Esses algoritmos têm por base o cofatoramento, respeitando a maioria das variáveis *binates* da cobertura até que as folhas *unates* sejam obtidas. Quando as operações descritas anteriormente encontram um mínimo local, então, é chamado um algoritmo que adiciona primos de modo seletivo até que ocorra o escape do mínimo local. Depois disso as operações são chamadas novamente pela ferramenta (SCHEFFER; LAVAGNO; MARTIN, 2006).

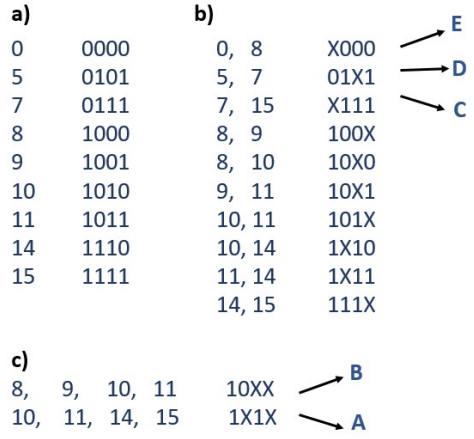
O método Quine-McCluskey, usado na minimização lógica em dois níveis, é composto por 4 etapas: geração dos implicantes primos, criação da tabela de implicantes primos, identificação dos implicantes primos essenciais, identificação de colunas e linhas dominantes (DEVADAS; GHOSH; KEUZER, 1994). A primeira etapa começa associando um decimal a cada mintermo da lista. O número é obtido através da conversão do mintermo (número binário) para decimal. Em seguida, é realizada a fusão dos cubos, sendo que dois cubos só podem ser aglutinados se diferirem em apenas uma posição como mostra a parte b) da figura 2.1. Os implicantes primos terminados em 1-cubo, porque possuem exatamente um X, são analisados

para serem aglutinados em 2-cubos como ilustra a parte c) da figura 2.1. Quando um k-cubo é gerado a partir de dois k-1-cubos, eles não são primos, por isso serão descartados mais tarde. Esse processo é encerrado quando no conjunto de L-cubos não há mais como realizar fusões (DEVADAS; GHOSH; KEUZER, 1994) e (KATZ, 1994).

A segunda etapa consiste em gerar uma tabela com implicantes primos obtidos anteriormente como mostra a figura 2.2. A primeira coluna contém os mintermos da lista e a primeira linha tem os implicantes primos encontrados. O X determina qual mintermo produziu determinado implicante primo (DEVADAS; GHOSH; KEUZER, 1994) (KATZ, 1994). Além disso, através da tabela da figura 2.2, é percebido que A, B, D e E são implicantes primos essenciais (terceira etapa) (DEVADAS; GHOSH; KEUZER, 1994).

A quarta etapa ocorre quando a função não possui implicantes primos essenciais. Por exemplo, considerando a tabela da parte a) da figura 2.2 que contém implicantes primos hipotéticos, o processo, que escolhe de modo arbitrário uma coluna, é iniciado. Selecionando a coluna e as linhas do implicante A, a tabela é reduzida como na parte b) da figura 2.2. Como uma coluna P é dominante sobre outra somente se a outra conter cada linha de P. Deste modo, B é dominante sobre C e H sobre G. Assim a tabela adquire a constituição apresentada na parte c) da figura 2.2. Deste modo, é possível diminuir a quantidade de literais presentes na função. Pois, as colunas excluídas correspondem aos primos com número igual ou maior de literais que o dominante primo. Pela tabela do exemplo da figura 2.2, C e G podem ser considerados implicantes primos. Assim escolhendo-se C e G resulta na seleção de E, o que leva a obtenção da cobertura f = A, C, E, G (DEVADAS; GHOSH; KEUZER, 1994). Porém não há garantia que essa seja a cobertura mínima, uma vez que a escolha da primeira coluna a ser excluída foi de modo arbitrário (DEVADAS; GHOSH; KEUZER, 1994).

Figura 2.1 – Identificação dos cubos a partir de uma lista de mintermos



Fonte: (DEVADAS; GHOSH; KEUZER, 1994)

Figura 2.2 – Tabela de Implicantes Primos

	A	B	C	D	E
0000					X
0101				X	
0111			X	X	
1000		X			
1001		X			X
1010	X	X			
1011	X	X			
1110	X				
1111	X		X		

Fonte: (DEVADAS; GHOSH; KEUZER, 1994)

Figura 2.3 – Redução da Tabela de Implicantes Primos através da exclusão das colunas dominantes

a)

	A	B	C	D	E	F	G	H
0000	X							X
0101	X	X						
0111		X	X					
1000			X	X				
1001								
1010							X	X
1011						X	X	
1110					X	X		
1111				X	X			

b)

	B	C	D	E	F	G	H
0111	X	X					
1000		X	X				
1001							
1010						X	X
1011					X	X	
1110				X	X		
1111			X	X			

c)

	C	D	E	F	G
0111	X				
1000	X	X			
1001					
1010					X
1011				X	X
1110			X	X	
1111		X	X		

Fonte: (DEVADAS; GHOSH; KEUZER, 1994)

2.2 Minimização em Multiníveis

A minimização em dois níveis funciona muito bem quando os produtos de uma soma de produtos possuem poucos literais. Porém em circuitos com múltiplos níveis esse processo deixa de ser trivial. Devido a isso, foi desenvolvida a minimização em multiníveis a qual é composta por duas etapas: minimização independente de tecnologia e minimização dependente de tecnologia. Como a área ocupada pelos nodos internos do grafo costuma ser pequena, mas as conexões de um grafo costumam exigir um aumento significativo dessa área, por isso há o costume de buscar uma redução da quantidade de literais presentes (KATZ, 1994). Normalmente a rede lógica multinível pode ser representada por um grafo acíclico direto, onde as arestas representam as interconexões e os nodos os elementos de memória ou as portas lógicas primitivas (SCHEFFER; LAVAGNO; MARTIN, 2006).

Na minimização independente de tecnologia é levado em consideração a quantidade de literais de modo fatorado. Porque essa quantidade de literais está relacionada com a área do circuito. Entre as técnicas de minimização em multiníveis, pode-se citar: a Divisão, o *Kerneling* e o uso de *don't cares*. A primeira consiste no processo de fatoração. Pode ser de dois tipos: Booleana, que explora as possíveis funções que podem dividir a equação que vai ser fatorada. Porém esse método é computacionalmente expansivo. Por isso, geralmente, é utilizada a divisão algébrica a qual é menos flexível mas é mais rápida do que a anterior. A segunda, é usada para escolher o divisor da função que passará pelo processo de fatoração. Eles são computacionalmente eficientes quando são usados em operações algébricas e também produzem divisores primários livres de cubos (SCHEFFER; LAVAGNO; MARTIN, 2006). A terceira, utiliza uma rede lógica representada em ROBDD, na qual os *don't cares* são localizados através do fanin de cada nodo (SCHEFFER; LAVAGNO; MARTIN, 2006).

Na etapa de minimização dependente de tecnologia, o circuito otimizado na etapa anterior é mapeado para as portas lógicas contidas em uma biblioteca de células. Assim as estimativas utilizadas na etapa anterior são trocadas por outras mais realistas obtidas da biblioteca. Deste modo, surgem novas possibilidades de otimizações locais na rede lógica. Por isso, depois do mapeamento, um novo processo de minimização é realizado (SCHEFFER; LAVAGNO; MARTIN, 2006).

Existe ainda a minimização baseada no esforço lógico (SUTHERLAND; SPROULL; HARRIS, 1998) na qual o atraso de um circuito pode ser obtido a partir dos esforços lógico e elétrico através da equação $d = \tau * (p + g * h)$. Onde τ é um parâmetro dependente do processo de fabricação, p é o atraso parasita, g corresponde ao esforço lógico e h ao esforço elétrico

(SCHEFFER; LAVAGNO; MARTIN, 2006).

O esforço lógico é a capacidade que a porta lógica possui de transmitir um sinal lógico para a saída. Já o esforço elétrico é caracterizado pelo ganho da porta, sendo a razão entre as capacitâncias de entrada e saída do pino de *gate*. Logo, o esforço lógico não é dependente do tamanho do *gate*, mas o esforço elétrico sim (SCHEFFER; LAVAGNO; MARTIN, 2006). O objetivo dessa técnica não é reduzir o número de literais mais sim o atraso do circuito (SCHEFFER; LAVAGNO; MARTIN, 2006).

3 MINIMIZAÇÃO LÓGICA USANDO GRAFOS

Um grafo é um modelo matemático que pode ser definido como uma tríplice G =(V(G), E(G), Ig) (RANGANATHAN; BALAKRISHNAN, 2012) no qual V(G) é um conjunto não-vazio, E(G) corresponde ao conjunto disjunto de V(G) (CORMEN et al., 2009) (REIS et al., 1997) e (DE MICHELI, 1994) e Ig é uma relação de incidência que relaciona cada elemento de E(G) a dois elementos de V(G). Além disso, os elementos de V(G) são chamados de vértices e os de E(G) são conhecidos como arestas (RANGANATHAN; BALAKRISHNAN, 2012) (CORMEN et al., 2009) (REIS et al., 1997) e (DE MICHELI, 1994).

Um grafo pode ser representado de duas formas: vetor, por exemplo: V(G) = v1, v2, v3, v4, v5, E(G) = e1, e2, e3, e4, e5, e6, Ig (e1) = v1, v5, Ig(e2) = v2, v3, Ig(e3) = v2, v4, Ig(e4) = v2, v5, Ig(e5) = v2, v5, Ig(e6) = v3, v3; ou de forma gráfica, como mostra a figura 3.1.

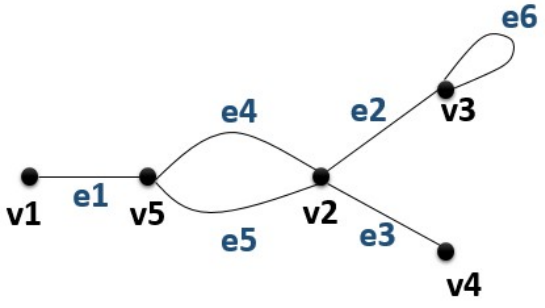
Existe na literatura diferentes tipos de grafos, porém, neste trabalho, é apresentado uma breve introdução dos grafos utilizados para minimização lógica, como AIG, MIG e DAG e BDDs, os quais são descritos nas seções a seguir.

3.1 Grafo Acíclico Direto - DAG

DAG (Grafo Acíclico Direto) (KEUTZER, 1987) é um tipo de grafo. Na minimização lógica, os vértices de um DAG representam as portas AND e OR e suas arestas o valor 1 ou 0.

Tendo por base o DAG, foi desenvolvido o DAGON (KEUTZER, 1987). Este é um algoritmo canônico independente de tecnologia que descreve um circuito combinacional. Basicamente, o DAGON particiona o circuito em árvores e depois as compara com os padrões

Figura 3.1 – Exemplo de representação gráfica de um Grafo



Fonte: (RANGANATHAN; BALAKRISHNAN, 2012)

tecnológicos (células da biblioteca) escolhendo o padrão que combina com a árvore, levando em consideração os custos de tempo, obtidos a partir do fanout.

Em (MARQUES et al., 2007) foi apresentado uma ferramenta, chamada de VIRMA-WF, que visa reduzir o atraso em circuitos combinacionais através da redução do número de transistores em série. A VIRMA-WF recebe o circuito mapeado em um DAG e o número máximo de transistores na rede PMOS e para a rede NMOS. Comparado com o SIS, a ferramenta apresentou uma redução média de 27% e 33% no atraso, considerando as bibliotecas virtuais produzidas para os pares (3,3) e (3,4). Esses representam o número de transistores em série para as redes *Pull Up e Pull Down* respectivamente.

3.2 Grafo de ANDs e Inversores - AIG

O grafo de ANDs e Inversores (AIG) usa somente portas ANDs de duas entradas. Este grafo consiste em uma forma não canônica de representar uma função booleana, ou seja, uma mesma equação pode ser representada por diferentes AIGs. Além disso, um AIG também é um tipo de DAG (MISHCHENKO et al., 2005), (FIGUEIRÓ; RIBAS; REIS, 2011), (FIGUEIRÓ; RIBAS; REIS, 2010) e (MISHCHENKO; BRAYTON, 2013). O uso de ANDs com duas entradas garante a representação do grafo em multiníveis e também a sua uniformidade (MISHCHENKO et al., 2005). Na figura 3.2 há um exemplo de AIG da função Booleana $F = A*!B*C$.

A associação do paradigma da Composição Funcional (MARTINS; RIBAS; REIS, 2012) com o AIG permite minimizar a quantidade de nodos e a profundidade do grafo. Nessa técnica a profundidade lógica é utilizada como um parâmetro secundário, pois o método usa o peso das entradas como um critério também. O peso de uma entrada está associado ao atraso estimado da mesma. Porém, na prática, esse peso corresponde ao número de nodos que formam o maior caminho da parte do circuito que está sendo analisada. O uso desse segundo parâmetro proporcionou uma redução de 6,65% na profundidade lógica sem aumentar a quantidade de nodos do grafo (FIGUEIRÓ; RIBAS; REIS, 2011).

A Composição Funcional, associada ao AIG, é baseada em quatro princípios: a função Booleana é representada por um par composto pelas representações estrutural e funcional; começa com um grupo inicial de funções; cria associações de equações simples para produzir funções mais complexas; as ordens parciais respeitam o conceito de programação dinâmica (MARTINS; RIBAS; REIS, 2012). Além disso, mantém somente um grupo de funções permitidas, para, deste modo, reduzir o tempo de execução e quantidade de memória utilizada

(FIGUEIRÓ; RIBAS; REIS, 2011) e (MARTINS; RIBAS; REIS, 2012).

O AIG é utilizado na ferramenta ABC (Berkeley Logic Synthesis and Verification Group.,) no qual a quantidade de nodos é reduzida através da seleção adequada da polaridade dos sinais de entrada e saída de cada nodo. As portas ANDs e ORs podem ser substituídas por NANDs ou NORs através da aplicação do Teorema de De Morgan. Pois, assim é possível reduzir a quantidade de transistores no circuito (MATOS,).

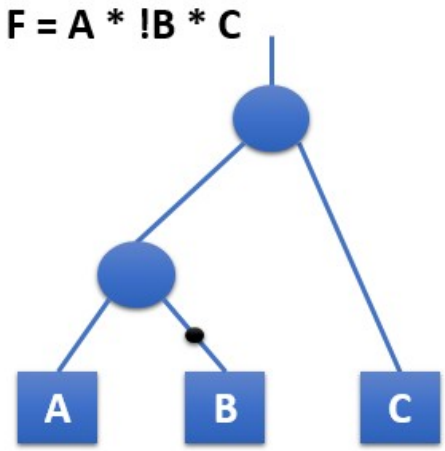
Outro dado importante é que a quantidade de nodos presentes em um AIG serve para estimar a área do circuito. Assim como a redução da profundidade do grafo pode contribuir para a diminuição do atraso depois da etapa de mapeamento tecnológico (MATOS,).

Em (MATOS,) e (MATOS et al., 2014) é apresentado um conjunto de algoritmos para a redução da quantidade de transistores. Nesta metodologia o circuito contido em um AIG passa por um processo de minimização dos nodos. Em seguida, este circuito é submetido a um processo de minimização de inversores, o qual tem por base o algoritmo de Coloração de Grafos. Finalmente, o circuito é mapeado para um conjunto de portas simples como NANDs e NORs. O fato dos algoritmos de redução de nodos em AIGs tenderem a aumentar a quantidade de nodos compartilhados, pode causar um aumento no fanout das células maior do que o valor desejado. Isso pode acarretar um prejuízo na performance dos circuitos em tecnologias mais novas. Por causa disso, (MATOS,) e (MATOS et al., 2014) apresentam o desenvolvimento de um método que limita o fanout e insere árvores de inversores quando uma célula possui o fanout muito grande. Os algoritmos de (MATOS,) proporcionaram, em média, uma redução de 32,07% na quantidade de transistores.

Uma variação dessa técnica é a FRAIG (MISHCHENKO et al., 2005) que é a união entre o AIG e o processo de detecção e junção de nodos equivalentes presentes no grafo. Durante o processo de construção do grafo é realizada uma análise para identificar se existe um nodo equivalente ao nodo filho, se houver, o nodo não é criado (MISHCHENKO et al., 2005) e (FIGUEIRÓ; RIBAS; REIS, 2010). A figura 3.3 mostra o FRAIG da função $F = !A*!B*C*!D+!A*!B*!C*D$. Segundo (MISHCHENKO et al., 2005), essa técnica pode ser aplicada em diferentes etapas da Síntese Lógica como no Mapeamento Tecnológico, por exemplo.

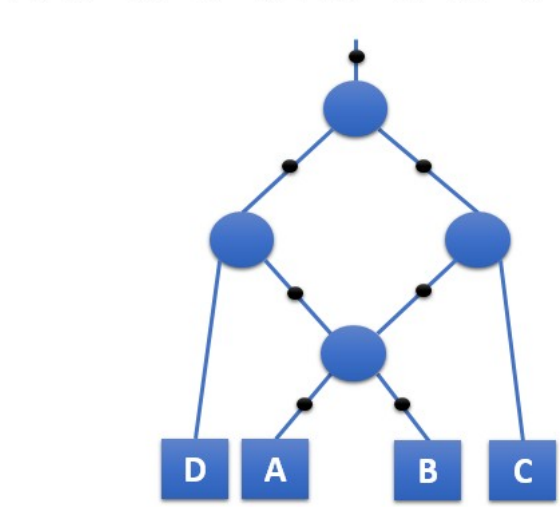
Em (GRANDHI et al., 2014) o AIG foi utilizado em um algoritmo para analisar e posteriormente aplicar transformações locais em regiões do grafo que apresentam pouca confiabilidade, para, deste modo, reduzir os impactos que erros nas portas podem causar nas saídas primárias. As transformações locais são baseadas em 5 regras: redução do fanout através do uso da lei da distributividade; redução do tamanho do caminho mais longo, tendo por base a lei da associatividade; distribuição igualitária dos inversores entre os caminhos do grafo; uso

Figura 3.2 – AIG da função $F = A*!B*C$



Fonte: (FIGUEIRÓ; RIBAS; REIS, 2010)

Figura 3.3 – FRAIG da função $F = !A*!B*C*!D+!A*!B*!C*D$



Fonte: (FIGUEIRÓ; RIBAS; REIS, 2010)

de portas majoritárias; os sinais com pouca probabilidade de ser igual a 1 devem ficar mais próximos do nodo raiz do subgrafo, deste modo, a capacidade de mascaramento da porta AND é aumentada e a influência de erros provenientes de outra partes do grafo é reduzida. Segundo (GRANDHI et al., 2014) para circuitos com uma quantidade de nodos que varie entre 30 e 1500 há uma melhora na confiabilidade do circuito em torno de 7,5%.

3.3 Grafo com Porta Majoritária e Inversores - MIG

Em (COHN, M.; LINDAMAN, 1961) foi apresentada a Lógica de Decisão com porta Majoritária. Este tipo de lógica é baseada em 10 axiomas e 14 teoremas, os quais foram usados para elaborar um somador utilizando apenas três portas majoritárias. Já em (SHELDON B. AKERS, 1961) foi apresentado um estudo que mostra a compatibilidade entre as identidades da lógica booleana com as da lógica de portas majoritárias. Neste estilo, foi utilizado somente portas majoritárias de 3 entradas para representar qualquer função booleana. A porta AND é definida através da fixação de uma das entradas da majoritária em 0; para a porta OR uma das entradas deve ser colocada em 1 (COHN, M.; LINDAMAN, 1961) (AMARÚ; GAILLARDON; DE MICHELI, 2014). Na figura 3.4 há na parte a) a representação da função $f = x3 * (x2+!(!x1 + x0))$ em um AOIG e na parte b) a mesma em um MIG e por último, na parte c), uma versão minimizada da mesma.

Tendo por base os teoremas de (COHN, M.; LINDAMAN, 1961), em (AMARÚ; GAILLARDON; DE MICHELI, 2014) é apresentado um novo tipo de grafo chamado de MIG. Este utiliza somente inversores e portas majoritárias. Embora os autores AMARÚ; GAILLARDON; DE MICHELI não enformem a tecnologia utilizada, em (AMARÚ; GAILLARDON; DE MICHELI, 2014) explicam que essa nova técnica causa uma redução, em média, de 18% nos níveis lógicos, de 22% no atraso estimado, de 14% na área do circuito e de 11% na potência consumida.

Aproveitando a capacidade de mascarar erros lógicos que a porta majoritária possui, é possível usar tal característica para ampliar a capacidade de minimização lógica usando MIGs. Porém os erros inseridos no grafo devem ser cuidadosamente escolhidos para evitar a perda da funcionalidade lógica do circuito. Devido a isso, esses devem satisfazer a equação $(fA \oplus f) * (fB \oplus f) = 0$. Onde f é a função original e fA e fB são as funções que contêm erros. Estes são chamados de ortogonais se satisfazem a equação citada anteriormente. Antes da inserção dos erros no grafo, é necessário identificar os que são considerados críticos, ou seja, aqueles que possuem um grande impacto no processo de simplificação (AMARÚ; GAILLARDON; De Micheli, 2015).

Quando a inserção de erros no grafo causa um aumento considerável no tamanho deste, durante as etapas intermediárias de minimização, o MIG é particionado em sub-grafos que devem conter os nodos críticos. Assim os sub-grafos serão simplificados separadamente (AMARÚ; GAILLARDON; De Micheli, 2015) e (AMARÚ; GAILLARDON; De Micheli, 2016). É importante também que os erros inseridos não ocorram ao mesmo tempo, porque isso causa um

erro no resultado da função. Pois uma porta majoritária de três entradas só mascara um erro por vez (AMARÚ; GAILLARDON; De Micheli, 2016).

Além disso, após inseridos os erros, são aplicados métodos Booleanos de simplificação. Finalmente os sub-grafos são reconectados ao nodo raiz da função. Posteriormente, os métodos algébricos de minimização de MIGs são aplicados no grafo resultante (AMARÚ; GAILLARDON; De Micheli, 2015) e (AMARÚ; GAILLARDON; De Micheli, 2016).

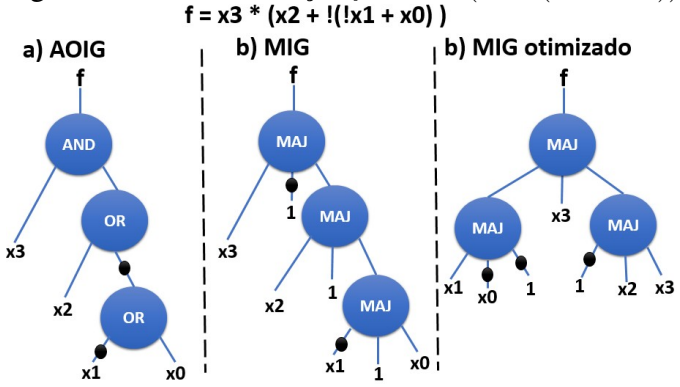
Assim como o BDD clássico, o ordenamento do MIG pode influenciar o tamanho do grafo e a quantidade de oportunidades de simplificação que ele pode oferecer. Isso significa que o algoritmo caiu em um mínimo local. para resolver esse problema é necessário reordenar o grafo (AMARÚ; GAILLARDON; De Micheli, 2016).

Os axiomas propostos em (AMARÚ; GAILLARDON; De Micheli, 2015) e (AMARÚ; GAILLARDON; De Micheli, 2016) para lógica com porta majoritária de três entradas foram expandidos para majoritárias com n entradas, sendo que n deve ser um número ímpar. Observando a figura 3.5 que contem um somador completo, o qual pode ser representado com três portas majoritárias de três entradas. Porém, com a ampliação dos axiomas, esse circuito é representado por apenas duas portas majoritárias, sendo uma com cinco entradas e a outra com três entradas (AMARÚ; GAILLARDON; DE MICHELI, 2015).

Esse tipo de grafo tem sido utilizado em tecnologias não CMOS, como Quantum dot Cellular Automata (QCA), nanofios e portas lógicas baseadas em spin (AMARÚ; GAILLARDON; DE MICHELI, 2015).

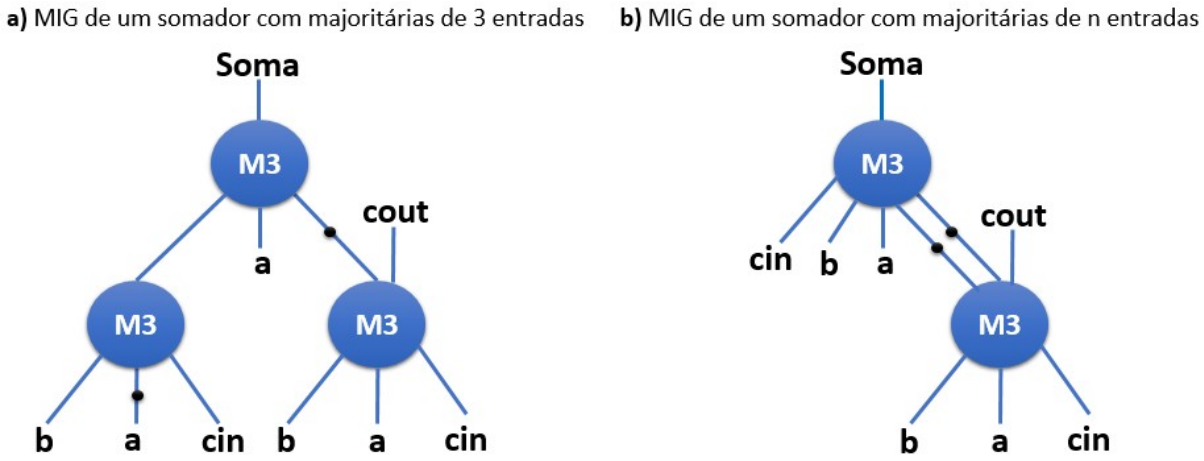
Tendo por base as portas majoritárias de três entradas, foi criada uma nova versão de MIG para minimização, onde o grafo do circuito é cortado em sub-grafos de 4 variáveis. Estes são mapeados para MIGs de quatro entradas que foram pré-computados. Deste modo é possível mapear o circuito para um conjunto de funções Booleanas que serão sintetizadas (SOEKEM et al., 2016).

Figura 3.4 – MIG da função $f = x3 * (x2 + !(!x1 + x0))$



Fonte: (AMARÚ; GAILLARDON; De Micheli, 2016)

Figura 3.5 – MIG do Somador



Fonte: (AMARÚ; GAILLARDON; DE MICHELI, 2015)

3.4 Diagrama de Decisão Binária - BDD

O aumento da complexidade das funções lógicas e dos sistemas digitais tornou necessário o desenvolvimento de novas técnicas para representar e minimizar essas funções. Pois a representação destas utilizando Soma de produtos, Tabelas Verdades e Mapas de Karnaugh ficaram impraticáveis. Por causa do fato da complexidade desses métodos crescer exponencialmente de acordo com o número de variáveis presentes em cada equação Booleana (AKERS, 1978), (DEVADAS; GHOSH; KEUZER, 1994), (AMARU; GAILLARDON; MICHELI, 2013) e (EBENDT; FEY; DRECHSLER, 2005). Devido a isso foi criado o Diagrama de Decisão Binária (BDD) (AMARU; GAILLARDON; MICHELI, 2013) que consiste em um grafo acíclico. Para cada vértice deste, que não é terminal, é atribuído um índice e uma variável de entrada (AKERS, 1978), (DEVADAS; GHOSH; KEUZER, 1994), (ARORA; BANEJIE; JIDGE, 2013) e (AMARU; GAILLARDON; MICHELI, 2013). Cada variável da função lógica corresponde a um nodo no diagrama, o qual pode assumir um dos valores 0 ou 1. Deste modo, para a função

$f = A+!B * C$, se A for igual a 1, f será também igual a 1. Porém se A for 0 é necessário analisar o valor do nodo B. Se este for igual a 1 f terá o número 0. Mas para $B = 0$ é necessário analisar o nodo C que, neste caso, determinará o valor de f (AKERS, 1978). O BDD desta equação pode ser observado na figura 3.6.

É possível também simplificar o BDD, reduzindo assim a quantidade de nodos. Por exemplo, o diagrama da equação $f =!A * B*!C + A * C$ é mostrado na parte a) da figura 3.7. Analisando o grafo da parte a) da figura mencionada anteriormente, é observado que quando A e B são iguais a 0, f é igual a 0, independentemente do valor de C. Logo, nesse caso, o C pode ser suprimido do diagrama (parte b da figura 3.7). Quando A é igual a 1, independentemente do valor de B, o diagrama mostra que o próximo nodo é C (parte b) da figura 3.7). Desse modo, o nodo B pode ser retirado do BDD, como mostra a parte c) da figura 3.7 (AKERS, 1978).

No pior caso, a complexidade de um diagrama de decisão binária é $O(2^n/n)$, onde n é o número de variáveis presentes na função.(AKERS, 1978).

O tamanho de um BDD é influenciado pelo ordenamento das variáveis que compõe a equação Booleana que este diagrama representa (ARORA; BANEJIE; JIDGE, 2013) e (AMARU; GAILLARDON; MICHELI, 2013). Como é visto no exemplo da figura 3.8, onde a representação de uma mesma função com um ordenamento diferente das variáveis produz dois grafos com uma quantidade diferente de nodos (ARORA; BANEJIE; JIDGE, 2013). Por este motivo, foi desenvolvido o Diagrama de Decisão Binária Ordenado (OBDD) que pode ser construído através do uso da Expansão de Shannon (SHANNON, 1938). Nesse tipo de BDD o índice do vértice pai deve ser menor que o índice do vértice filho (REIS et al., 1997). Porém esse tipo de grafo, apesar de ordenado, não elimina os nodos redundantes (AKERS, 1978), (ARORA; BANEJIE; JIDGE, 2013) e (EBENDT; FEY; DRECHSLER, 2005).

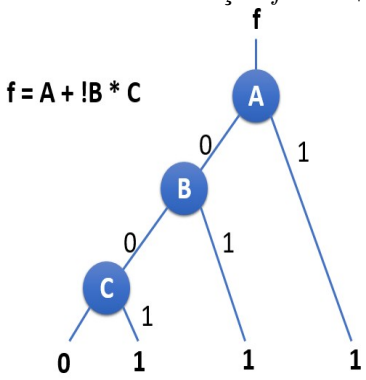
O Diagrama de Decisão Binária Ordenado e Reduzido (ROBDD) (AMARU; GAILLARDON; MICHELI, 2013) foi criado para solucionar o problema dos vértices redundantes e também ser mais uma forma canônica de representação de equações Booleanas (REIS et al., 1997) e (ARORA; BANEJIE; JIDGE, 2013). Pois o ROBDD evita que nodos distintos, que representam uma mesma variável, com os mesmos sucessores, faça parte do grafo. Além disso, também não permite que um vértice tenha um sucessor idêntico $(alto(x) = baixo(x))$ (ARORA; BANEJIE; JIDGE, 2013) e (AKERS, 1978).

O Diagrama de Decisão Binária com Terminal Suprimido (TSBDD) (REIS et al., 1995) também deriva do BDD clássico proposto em 1978. Esse diagrama permite a associação direta de seus arcos com transistores. Assim a etapa de mapeamento tecnológico pode ser realizada sem o uso de uma biblioteca de células (REIS et al., 1995). O TSBDD possui quatro propri-

edades básicas: somente os arcos S0 são conectados ao terminal 1; somente os arcos S1 são conectados ao terminal 0; Todos os arcos que chegam ao mesmo vértice são S0 ou S1; existe sempre um caminho que passa por todos os vértices. (REIS et al., 1995) e (REIS et al., 1997). É importante destacar que, se não houver perda de informação, esses arcos podem ser suprimidos (REIS et al., 1997). A figura 3.9 mostra o exemplo das redes de transistores PMOS (parte c) e de transistores NMOS (parte b) extraídas do TSBDD de uma função complexa (parte a) (REIS et al., 1997).

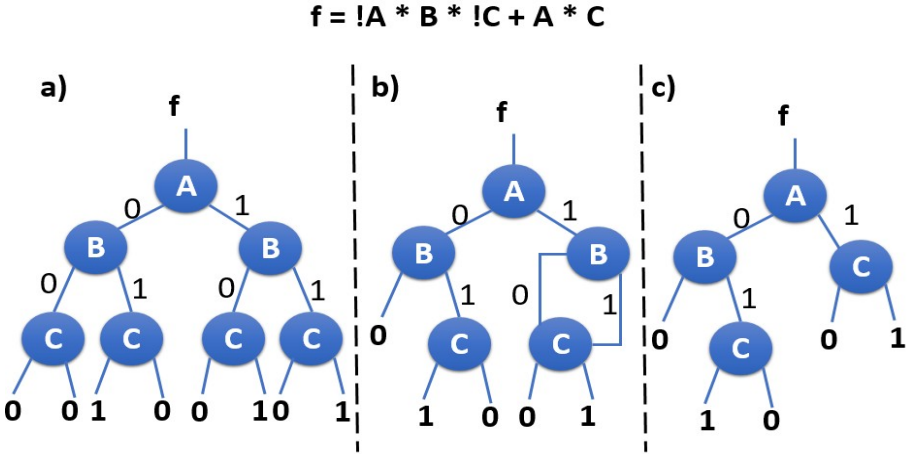
O BBDD é uma variação do BDD onde a expansão de Shannon é substituída por uma expansão bicondicional baseada em operações com NXOR e XOR. Este diagrama assim como o BDD tradicional, também possui uma versão ordenada e reduzida: o ROBBDD. Este diagrama quando comparado com o ROBDD produz uma rede booleana, em média, com 37% menos nodos. Sendo que em CMOS, quando comparado com uma ferramenta comercial, reduz, em média, em 31,5% o número de transistores (AMARÚ; GAILLARDON; De Micheli, 2013).

Figura 3.6 – BDD da função $f = A + !B * C$



Fonte: (AKERS, 1978)

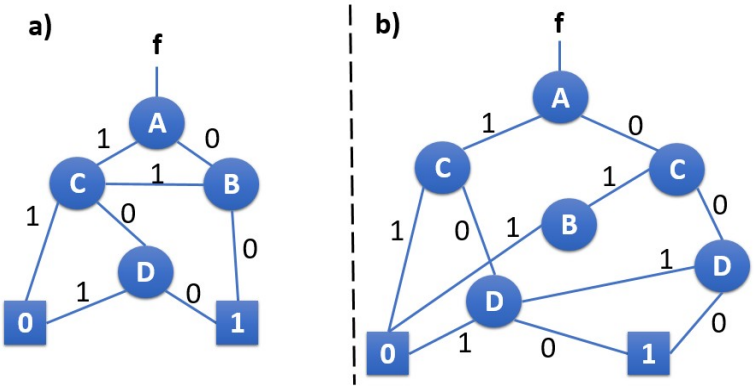
Figura 3.7 – Simplificação de BDD



Fonte: (AKERS, 1978)

Figura 3.8 – Ordenamento das variáveis em um BDD

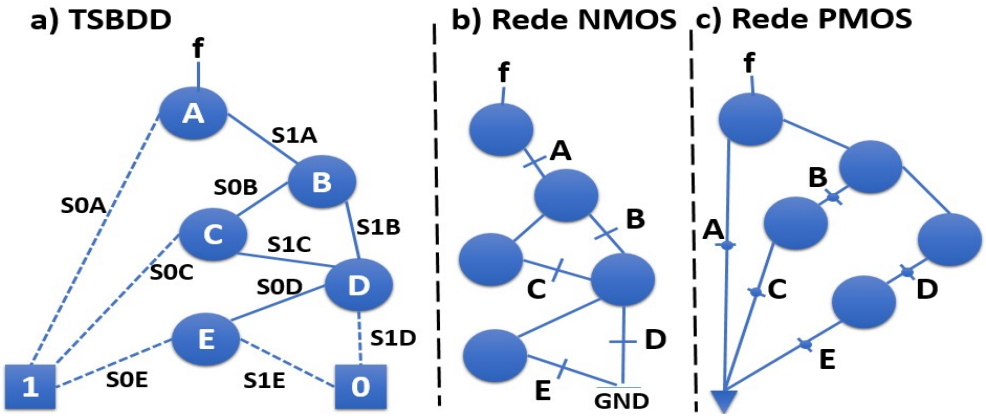
f = A * B + A * !B * !C * D + !A * C * D



Fonte: (ARORA; BANEJIE; JIDGE, 2013)

Figura 3.9 – Diagrama de Decisão Binária com Terminal Suprimido

f = A * (B + C) * (D + E)



Fonte: (REIS et al., 1997)

4 FERRAMENTAS DE SÍNTESE LÓGICA

A literatura apresenta várias ferramentas acadêmicas que abordam todas as etapas da Síntese Lógica como a SIS (SENTOVICH et al., 1992) e a ABC (Berkeley Logic Synthesis and Verification Group.,) (BRAYTON; MISHCHENKO, 2010) ou apenas determinadas etapas desta como o TABA (REIS et al., 1997) por exemplo. Neste capítulo são apresentadas as principais ferramentas que tratam desta fase do fluxo VLSI: seção 4.1 descreve a MIS (BRAYTON et al., 1987); a seção 4.2 apresenta a SIS ; a 4.3 mostra a ABC ; a seção 4.4 aborda a TABA (REIS et al., 1997); a seção 4.5 descreve a ELIS (MARQUES et al., 2004); a seção 4.6 apresenta a VIRMA (MARQUES,), a seção 4.7 descreve a MIXSyn (AMARÚ; GAILLARDON; MICHELI, 2013) e a última mostra a MOTO-X (KAGARIS, 2016).

4.1 MIS

A MIS (BRAYTON et al., 1987) é uma ferramenta de Síntese Lógica desenvolvida na década de 1980. A MIS recebe como entrada as equações booleanas, as quais representam uma macrocélula, extraídas do circuito pela ferramenta BDSYN. Em seguida, otimiza essas equações para produzir uma rede booleana ótima que respeita o comportamento dos pinos de entrada e de saída da macrocélula.

A minimização global nessa ferramenta engloba algoritmos de decomposição, fatoração, minimização e minimização de tempo para funções com multinível. As equações da rede booleana podem estar representadas na forma de soma de produtos ou na forma fatorada. Nesta última é possível a obtenção de uma estimativa da área por nodo, além de permitir, através da contagem do número de literais, uma estimativa da complexidade da rede booleana (BRAYTON et al., 1987).

A MIS também utiliza a minimização local para decompor portas grandes em pequenas, o que pode causar um aumento na área do circuito. Assim como as demais ferramentas de Síntese Lógica, a MIS pode reduzir o atraso sem causar um grande aumento na área, através do uso de uma representação em DAG do circuito. Para fazer uma estimativa do atraso é utilizado o modelo RC, a largura do transistor, a quantidade de transistores em série, a capacitância de carga e a média entre o número de transistores em paralelo e os transistores do caminho a ter o atraso minimizado (BRAYTON et al., 1987).

4.2 SIS

A SIS (SENTOVICH et al., 1992) é uma ferramenta interativa para Síntese Lógica e minimização de circuitos sequenciais. Assim como a MIS, a SIS visa minimizar a área, a performance e aumentar a testabilidade. A Síntese Lógica dessa ferramenta possui várias etapas: atribuição e minimização de estados; minimização global da área e da performance; minimização local, e mapeamento tecnológico.

Essa ferramenta pode receber o circuito descrito em diferentes formatos como: BLIF; que descreve máquinas de estados finitos, e ASTG que corresponde a um grafo de transição de sinal baseado em eventos, normalmente utilizado para representar circuitos assíncronos (SENTOVICH et al., 1992).

A SIS engloba também várias técnicas implementadas no MIS as quais proporcionam uma melhoria na reestruturação do circuito, no mapeamento tecnológico, no uso de *don't cares* e na minimização de nodos. Para que a síntese de circuitos sequenciais fosse possível, novas técnicas foram incluídas na ferramenta como minimização de estados em máquinas de estados finitos, algoritmos de minimização de circuitos sequenciais, verificação de máquinas de estados finitos e algoritmos para a síntese de circuitos assíncronos (SENTOVICH et al., 1992).

No processo de simplificação e de tratamento dos *don't cares*, a SIS utiliza BDDs. Como a ordem das variáveis influencia no tamanho do BDD, a SIS possui algoritmos para determinar a ordem das variáveis para reduzir o tamanho desse diagrama. Por isso a ferramenta pode sintetizar circuitos grandes. Além de realizar a Síntese Lógica de circuitos sequenciais e de circuitos assíncronos, a SIS também permite a síntese de Vetores de Portas Programáveis (PGAs). Esse tipo de arquitetura possui duas principais categorias *Table Look up* (TLU) e Baseado em Multiplexadores (MB) (SENTOVICH et al., 1992).

4.3 ABC

A ABC é uma ferramenta EDA para síntese e verificação de circuitos síncronos sequenciais. Nesta ferramenta a minimização lógica utiliza AIGs, já a minimização do atraso é baseada em DAGs e o mapeamento tecnológico pode ser realizado para TLUs ou para uma biblioteca padrão de células. A ABC engloba outra ferramenta, desenvolvida pelo mesmo grupo de pesquisa da Universidade de Berkeley, a SIS.

A ABC é o resultado de uma nova implementação do MVSIS, que foi desenvolvida pela Berkeley Logic Synthesis and Verification Group (BRAYTON; MISHCHENKO, 2010), o qual

realiza a Síntese Lógica em multiníveis utilizando AIGs. Esse sistema está apto a trabalhar com diferentes representações booleanas como BDDs e soma de produtos para solucionar tarefas especializadas. Porém a estrutura principal de representação da rede booleana continua sendo os AIGs (Berkeley Logic Synthesis and Verification Group.).

Além disso, é um sistema de domínio público utilizado para o desenvolvimento de novos algoritmos de síntese e também é amplamente comparado a outras ferramentas que estão no estado da arte, como ocorre em (AMARÚ; GAILLARDON; MICHELI, 2013), onde desempenho da MYXSyn é comparado ao da ABC.

4.4 TABA

A TABA (REIS et al., 1997) é uma ferramenta para mapeamento tecnológico. Foi desenvolvida usando variações de BDD como Diagrama de Decisão Binária com Terminal Suprimido (TSBDD), por exemplo. Como entrada recebe uma rede booleana formada por portas lógicas simples. Primeiramente ela decompõe essa rede em cone lógicos, os quais possuem as portas de fanout unitário que estão interconectadas entre si. Cada um destes cones é representado por um TSBDD. Esse diagrama, então, é associado a um algoritmo de limitação do número de transistores em série. Finalmente a porta complexa é gerada respeitando a restrição do número máximo de transistores em série definido pelo usuário (REIS et al., 1997). Além disso, essa ferramenta utiliza uma biblioteca virtual com um número bem limitado de células (REIS; ROBERT; REIS, 1998). O TABA reduz, em média, 17% o número de transistores (REIS et al., 1997).

Um estudo de seis parâmetros (número de transistores em série, número de portas complexas, número de transistores em série em cada porta complexa, número máximo de portas complexas, número máximo de fanout de cada porta complexa e o número de transistores por onde passa o sinal partindo das entradas primárias para a saída primária) do TABA foi apresentado em (REIS; ROBERT; REIS, 1998). Após uma análise de cada parâmetro, foi concluído que o TABA permitiu, além da redução da quantidade de transistores, uma diminuição no número de células (REIS; ROBERT; REIS, 1998). Fato que pode favorecer uma redução no atraso, na área do circuito e na profundidade lógica (REIS; ROBERT; REIS, 1998).

4.5 ELIS

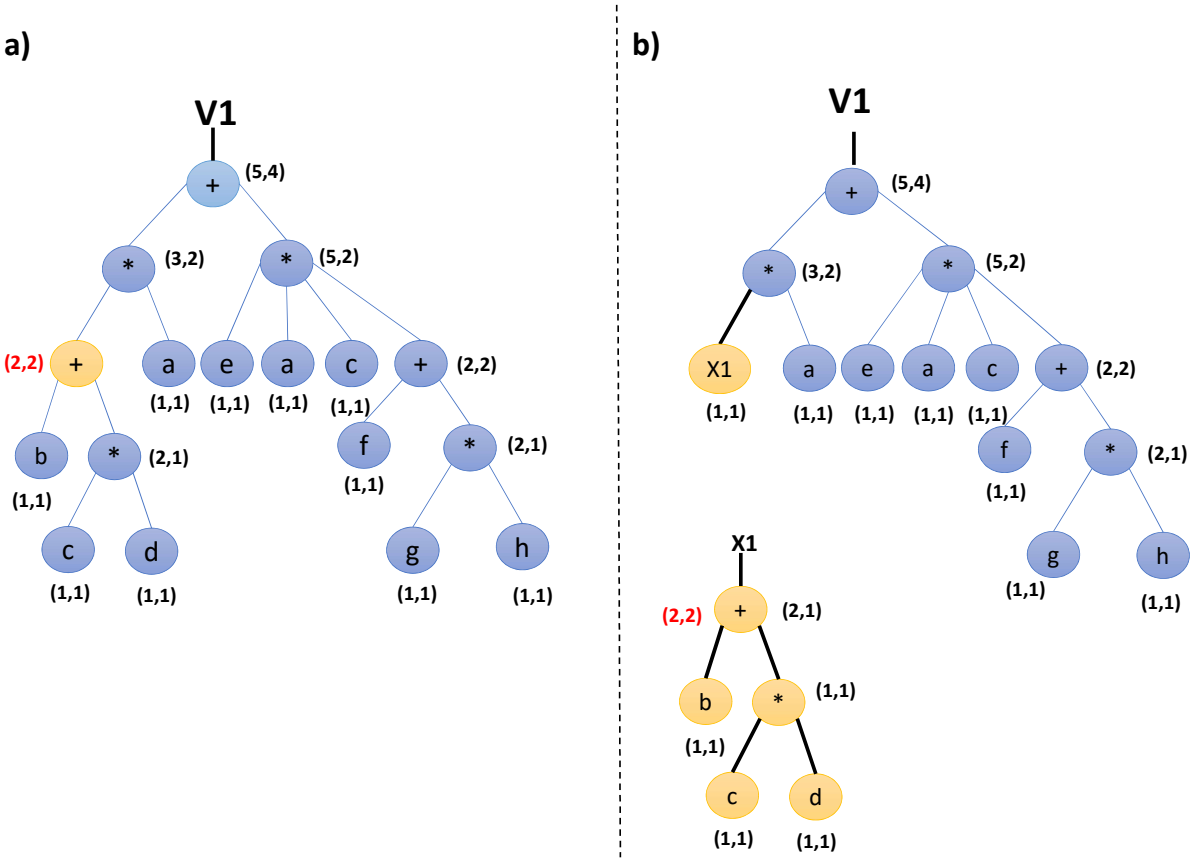
A ELIS (LOGICS) (MARQUES et al., 2004) é uma ferramenta de Síntese Lógica que possui uma estrutura de dados que serve de base para a implementação de diversos algoritmos de Síntese Lógica. Para garantir a compatibilidade da ELIS com outros softwares foram implementados leitores de arquivos para os formatos EQN e NETBLIF os quais são compatíveis com o SIS. Além disso, os arquivos de saída são em VHDL ou em SPICE também visando a compatibilidade como outras ferramentas, como o HSPICE por exemplo.

O mapeamento tecnológico dessa ferramenta, tanto para bibliotecas virtuais simétricas quanto para assimétricas, utiliza árvores cujos nodos podem ter múltiplos nodos filhos. Sendo que a estratégia de cobertura explora várias decomposições dessas árvores buscando uma minimização do índice lógico (MARQUES,). No algoritmo proposto (CORREIA; REIS, 2004) o DAG é decomposto em árvores. O nodo raiz de cada árvore, além de receber um *label* de AND ou OR, é associado a um custo (s, p) e a um índice lógico 1. O custo do nodo raiz é a soma dos custos dos seus nodos filhos, sendo que cada nodo filho tem o seu custo definido (1,1). Os cortes são feitos quando um nodo atinge o custo máximo, em seguida essa sub-árvore é associada a uma porta complexa CMOS. Já o nodo de corte é realocado como um novo nodo folha com custo (1,1) e seu índice lógico é o valor do índice da raiz da árvore recentemente cortada mais 1. A figura 4.1 exemplifica esse método, o nodo raiz V está associado ao índice 1 e ao custo (5,4), como as partes a) e b) da referida figura ilustram. Se o custo máximo for (2,2), por exemplo, é realizado o corte mostrado na parte b) da figura 4.1, como dito anteriormente, o nodo raiz X, que pertence a nova árvore, tem índice 1, ou seja, 0 (valor do índice imediatamente após o corte) mais 1. Na árvore de nodo raiz V1, o nodo X1 é considerado um novo nodo folha com custo (1,1), assim como os demais nodos folhas dessa árvore. Já a árvore com nodo raiz X1 é mapeada para uma porta lógica complexa CMOS. Usando a programação dinâmica várias decomposições são avaliadas e aquelas que apresentam o menor número de portas em série no caminho crítico são escolhidas.

Outro fator importante é a verificação da equivalência de sinais, a menos que o objetivo seja reduzir o atraso, não há necessidade de portas lógicas iguais que produzam o mesmo sinal, logo, basta que este sinal seja compartilhado. A ELIS também consegue remover nodos inalcançáveis dos grafos, ou seja, situações ou portas lógicas que nunca serão utilizadas. Devido ao uso de algoritmos de mapeamento tecnológico, a ferramenta consegue minimizar a quantidade de inversores utilizados (CORREIA; REIS, 2004).

A ELIS também produz uma descrição em formato spice do circuito(CORREIA; REIS,

Figura 4.1 – Exemplo de corte em um DAG



Fonte: (CORREIA; REIS, 2004)

2004). Porém para um mapeamento independente de biblioteca de células, os transistores devem possuir o tamanho mínimo, pois a ferramenta define a mesma largura para todos os transistores. Logo, não há uma preocupação da ELIS em definir valores coerentes para tal parâmetro (POSSER,).

4.6 VIRMA

A VIRMA (MARQUES,) é uma ferramenta de mapeamento tecnológico que utiliza uma biblioteca virtual de células e o conceito de menor célula conectada. As bibliotecas virtuais não dispõem de informações como atraso, área e potência consumida. Por isso são utilizados algoritmos que fazem uma estimativa desses dados (MARQUES,).

Os custos de cada caminho tanto para a rede PMOS quanto para a NMOS são obtidos a partir da soma da quantidade de transistores em série de cada célula que está no caminho. Assim como a ELIS, a VIRMA representa a rede booleana do circuito em um DAG, que para aumentar

a granularidade do circuito e garantir maior liberdade para a geração de células complexas, é decomposto em portas AND/OR de duas entradas (MARQUES,).

Uma outra versão da ferramenta, a VIRMA-WF (VIRMA wavefront), utiliza o algoritmo wavefront (MARQUES,). Esta ferramenta realiza o mapeamento de um DAG para o mínimo atraso possível através do uso de uma biblioteca estática de células. Mas devido ao uso de uma biblioteca virtual de células, a fase de combinação desse algoritmo foi alterada para trabalhar puramente com funções booleanas, as quais são representados por BDDs. A partir desses BDDs são calculados o número de transistores em série. Essa fase deve sempre fazer o custo da rede PMOS maior ou igual a custo da rede NMOS e também respeitar o valor máximo de transistores em série definido. Para permitir uma avaliação do número excessivo de combinações, são usados como limite o número de AND/OR e o número de literais (MARQUES,).

4.7 MIXSyn

A MIXSyn (AMARÚ; GAILLARDON; MICHELI, 2013) é uma ferramenta de Síntese Lógica independente de biblioteca que atua sobre as portas AND/OR e XOR presentes no circuito. Além disso, suporta dois tipos de tecnologia: CMOS e Ambipolar. Essa ferramenta é dividida em duas partes:

- *Minimização Lógica*: primeiro otimiza usando a porta lógica XOR e, posteriormente, utilizando AND/OR e *don't cares* (AMARÚ; GAILLARDON; MICHELI, 2013);
- *Mapeamento tecnológico independente de biblioteca*: a rede booleana é organizada em um grafo, o qual é decomposto em sub-árvores. Essas sub-árvores contêm um número m de entradas, que correspondem a um fanout máximo, e devem conter o mínimo possível de folhas (AMARÚ; GAILLARDON; MICHELI, 2013).

Comparando o desempenho da MIXSyn com as ferramentas acadêmicas ABC e BDC e a ferramenta comercial da Synopsys Design Compiler, foi verificado que, para a tecnologia CMOS, houve uma redução na quantidade de transistores, em média, de 18% e 9,2% respectivamente (AMARÚ; GAILLARDON; MICHELI, 2013).

4.8 MOTO-X

A MOTO-X (*Multiple Output Transistor Optimization with bridging*) (KAGARIS, 2016) é uma ferramenta de Síntese Lógica que visa reduzir o número de transistores. Ela recebe como entrada uma função de múltiplas saídas em forma de Soma de Produtos. Essa função é mapeada para um grafo chamado de Grafo de Rede de Transistores Genéricos (TNG) no qual os nodos representam os pontos de ramificação e as arestas, os transistores; de forma semelhante ao TSBDD (REIS et al., 1995) que foi abordado anteriormente. O número máximo de transistores em série, assim como ocorre com a ferramenta TABA (REIS et al., 1997) e (REIS; ROBERT; REIS, 1998), também é definido pelo usuário.

Essa ferramenta, ao tratar da Soma de Produtos, não está preocupada com o ordenamento das variáveis durante a inserção destas no grafo. Porém, quando uma determinada ordem tem um custo, que o algoritmo considera infinito, um novo ordenamento das variáveis é realizado. Além disso, esse método procura tratar dos *don't cares* adicionando o mínimo de transistores possível no grafo que representa a função que está sendo simplificada. Uma outra característica é que o MOTO-X não tenta tratar das variáveis negadas que compõem o TNG resultante (KAGARIS, 2016).

5 MINIMIZAÇÃO LÓGICA POR FUSÃO DE PORTAS

A Minimização Lógica é um problema de Otimização que pode ser classificado como *NP Hard*, ou seja, um problema cuja solução ótima não pode ser obtida em tempo polinomial. Devido a isso, algoritmos que visam solucionar problemas dessa natureza, apresentam resultados satisfatórios, porém, em algumas situações podem cair em um mínimo local e ter o seu desempenho prejudicado (KLEINBERG; TARDOS, 2006) e (CORMEN et al., 2009).

A metodologia utilizada neste trabalho é um algoritmo guloso que possui o objetivo de reduzir a quantidade de transistores e interconexões através da aglutinação de portas combinacionais de fanout unitário para gerar portas complexas. No Fluxo de Projeto de Circuitos Digitais, o método apresentado corresponde a uma etapa de minimização lógica que está localizada entre as etapas de Síntese Lógica e Síntese Física, como mostra a figura 5.1. Este método foi implementado por meio de um algoritmo guloso denominado de Minimização Lógica por Fusão de Portas (LOMGAM). Este algoritmo realiza a minimização lógica de modo independente de uma biblioteca de células, logo, produz portas complexas exclusivas para o circuito que está sendo minimizado. O novo Netlist gerado pelo LOMGAM consiste no arquivo de entrada da fase de Síntese Física, sendo esta realizada por uma ferramenta capaz de gerar o leiaute de qualquer rede de transistores, como por exemplo o ASTRAN (ZIESEMER,).

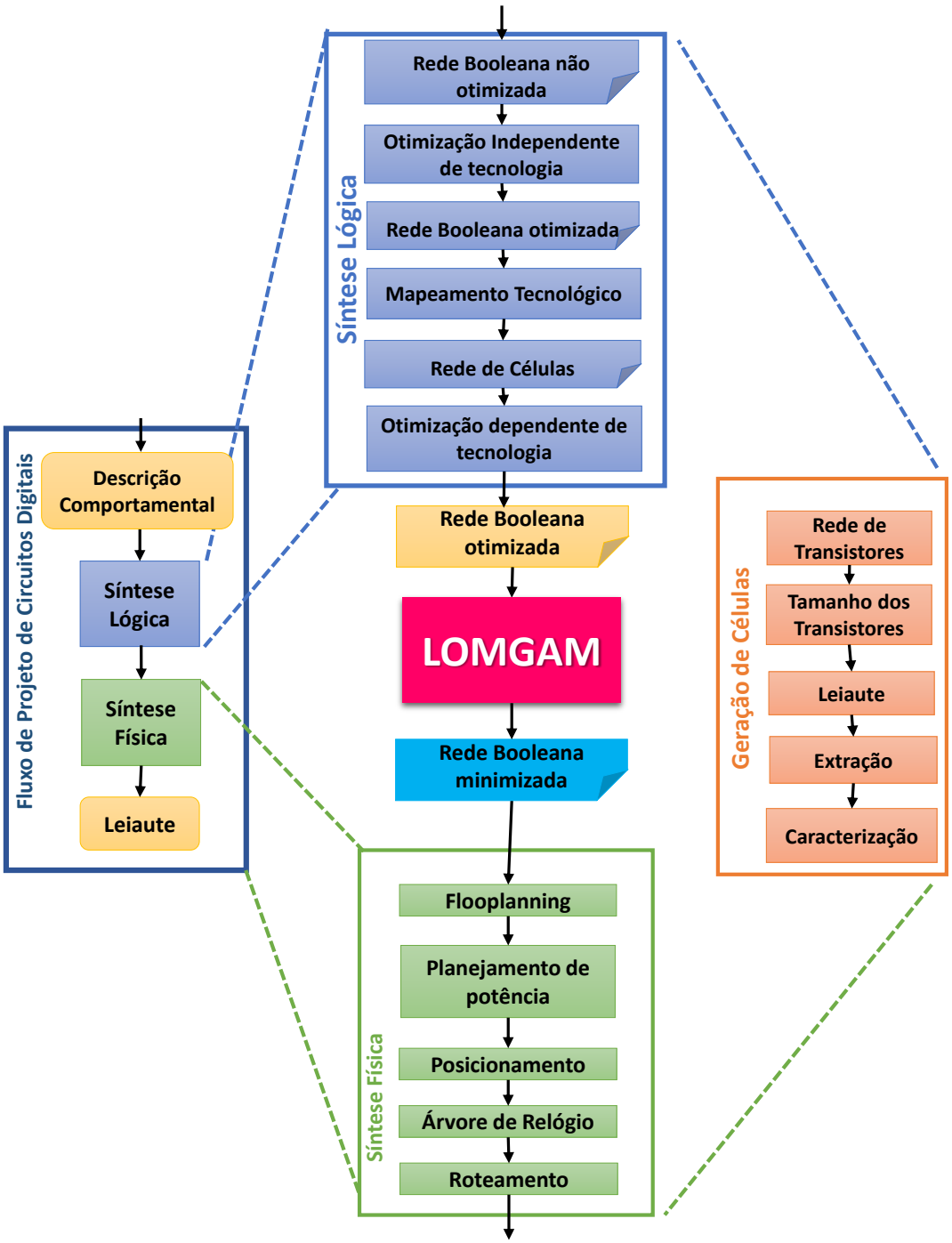
O algoritmo proposto, representado na figura 5.2, é composto por sete etapas as quais são abordadas a seguir:

- *Identificação das Interconexões e das Portas Combinacionais*: O algoritmo inicia lendo o arquivo de Netlist (em formato Verilog Estruturado) para identificar e armazenar as interconexões e portas combinacionais. Em seguida, realiza a análise das entradas e das saídas de cada porta. Pois, elas serão as variáveis nas equações booleanas de suas respectivas portas;
- *Identificação do Fanout*: As interconexões de entrada e de saída de cada uma das portas lógicas são analisadas para determinar o fanout de cada uma. No circuito da figura 5.3 é observado que a porta lógica A tem fanout igual a 2, porque sua saída corresponde a uma das entradas das portas B e C. O fanout unitário ocorre quando a saída de uma determinada porta é interconectada apenas à outra porta, como acontece com a porta C da figura 5.3;
- *Geração das Equações Booleanas*: Nesta etapa as equações são mapeadas do formato Verilog estruturado para a forma de equação Booleana. Tendo por base as interconexões identificadas na primeira etapa;

- *Processo de Fusão*: Esta etapa só é aplicada nas equações booleanas de fanout unitário. A fusão consiste em aglutinar um porta de fanout unitário em outra porta lógica, desde que, estas portas estejam interconectadas. Além disso, o usuário pode escolher entre dois tipos de parâmetros de controle: GMI (Índice de Fusão de Portas - *Gate Merging Index*) e Quantidade Máxima de transistores em série (QMTS) tanto para a rede transistores PMOS, quanto para a NMOS. Tanto o processo de fusão quanto esses parâmetros de controle serão abordados nas próximas seções.
- *Conversão para CMOS*: na tecnologia CMOS, uma porta AND é formada por uma porta NAND com um inversor na saída, o mesmo ocorre com uma porta OR que é composta por uma porta NOR com um inversor na saída como mostra a parte a) da figura 5.4. Analisando as portas complexas D e E do primeiro circuito da parte b) da figura 5.4 é notado que essas não estão em CMOS. Para transformar em CMOS, é usado o mesmo princípio apresentado na parte a) da referida figura, deste modo, o segundo circuito da parte b) desta figura é obtido.
- *Geração da Netlist Minimizada no Formato eqn*: Produção de um arquivo de netlist minimizada do circuito. A netlist em formato eqn foi adotada por ser compatível com a ferramenta ELIS a qual é utilizada na próxima etapa e também devido à sua compatibilidade com outras ferramentas acadêmicas como o SIS e o ABC, por exemplo. Isso possibilita futuras comparações, em termos de redução no número de transistores, da LOMGAM com estas;
- *Geração da Netlist Minimizada no formato Spice*: A última etapa do algoritmo consiste em converter a Netlist minimizada do formato eqn para o formato Spice. Esta etapa foi acrescentada ao método para permitir simulações elétricas através do uso de ferramentas como o Hspice, por exemplo. Além disso, a descrição Spice do circuito garante a compatibilidade com o ASTRAN o qual é uma ferramenta de geração de leiaute de rede de transistores independente de biblioteca de células (ZIESEMER,). Deste modo, é possível gerar o leiaute dos circuitos minimizados pela LOMGAM e assim analisar as características elétricas, atraso e área desses. A conversão para a descrição Spice é realizada através da ferramenta ELIS (MARQUES et al., 2004), (CORREIA; REIS, 2004) através dos comandos: re *arquivo.eqn* para ler a Netlist minimizada do circuito e ws *arquivo.sp* para gerar a descrição Spice e salvá-la em arquivo. A referida ferramenta não permite o dimensionamento dos transistores de cada equação, apenas um dimensionamento padrão, ou seja, o mesmo tamanho para todos os transistores(POSSER,). Como o dimensionamento de transistores não faz parte do escopo deste trabalho, por isso foi decidido utilizar

o dimensionamento padrão do ELIS para definir o tamanho dos transistores.

Figura 5.1 – Fluxo de Projeto de Circuitos Digitais com a inserção da LOMGAM.



Adaptado de(ALEGRETTI,)

Figura 5.2 – Minimização Lógica por Fusão de Portas.

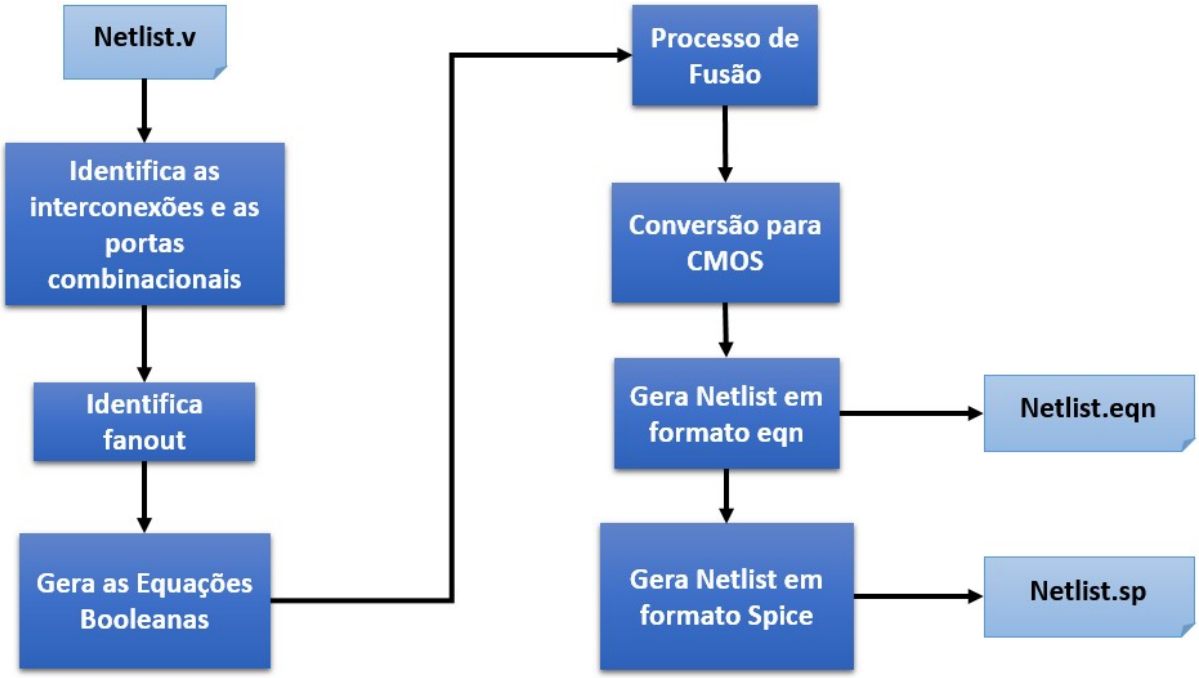
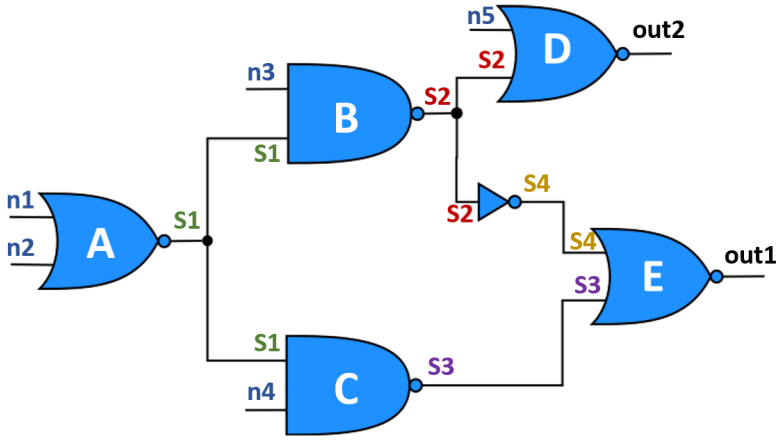


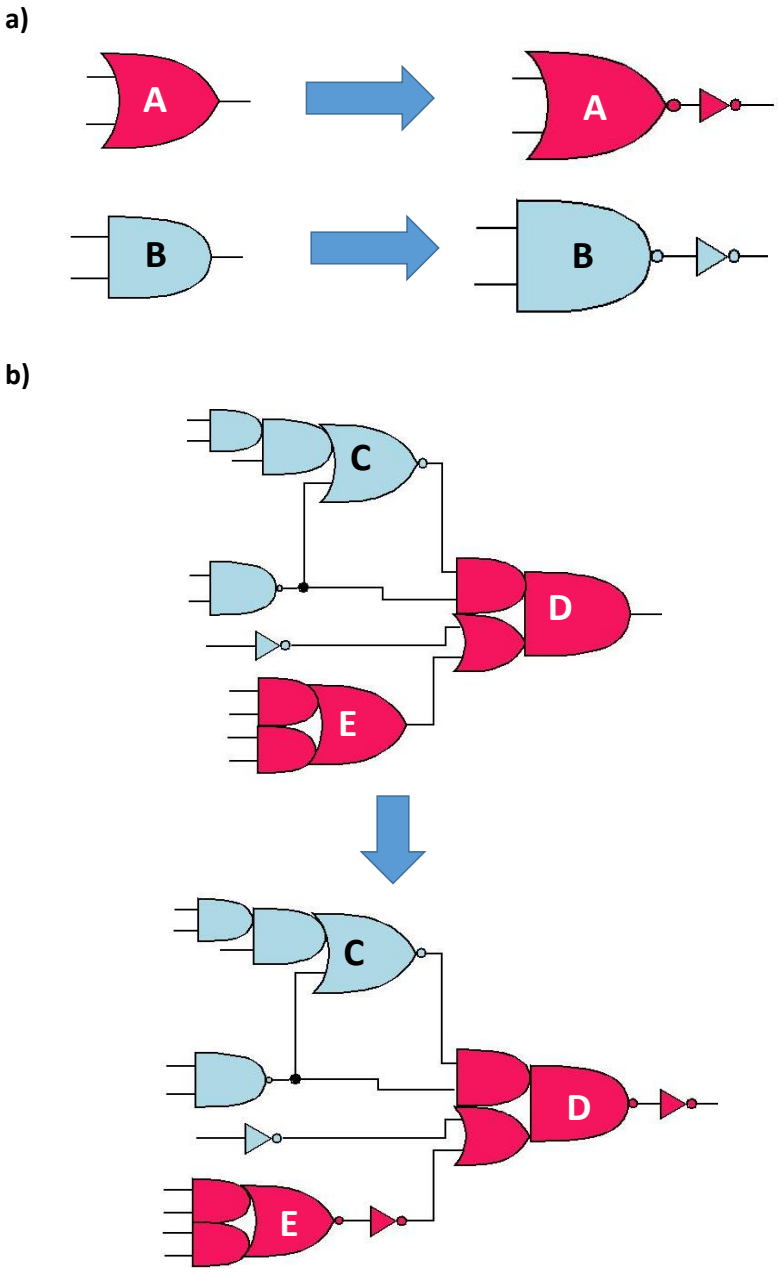
Figura 5.3 – Determinação do fanout.



As informações obtidas de cada porta, durante a execução de cada etapa do algoritmo são armazenadas na estrutura de dados apresentada na figura 5.5. Essa estrutura, chamada de Porta, possui sete campos, como é observado na figura citada anteriormente: o *Nome* recebe a identificação da porta (por exemplo: NAND2, NOR2, AOI22); O *Saída* armazena a interconexão de saída da porta; o *Fanout* contem o valor do fanout; o *Equação* Booleana, recebe a equação Booleana que representa a porta e os demais são utilizados de acordo com o parâmetro de controle selecionado. Se o Índice de Fusão de Portas (GMI) foi selecionado o campo Status é usado e os campos PMOS e NMOS não são utilizados. Porém se o parâmetro de controle escolhido for a Quantidade Máxima de Transistores em Série (QMTS) o campo PMOS é preenchido com a quantidade de transistores PMOS em série que a porta contem e o campo NMOS é

preenchido como o número de transistores NMOS em série que a porta possui. Nesta situação, o campo Status não é utilizado. É importante destacar que se a porta está somente conectada à uma saída primária o campo Fanout é preenchido com 0. O campo Status, quando o parâmetro GMI é usado, também é preenchido com 0.

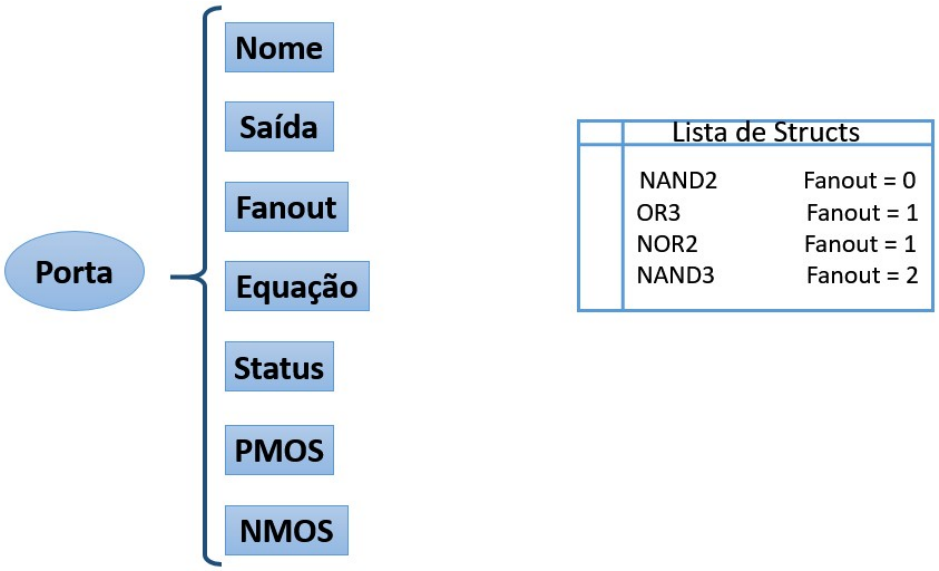
Figura 5.4 – Conversão para CMOS.



As estruturas Porta são armazenadas em Listas. Essas listas são um mecanismo de armazenamento de dados disponível na linguagem de programação C++. Depois que todas as portas combinacionais estão armazenadas na Lista, esta é organizada em ordem crescente de fanout. O objetivo deste ordenamento é facilitar a leitura do circuito combinacional, uma vez que a leitura do mesmo é sempre realizada partindo-se das saídas primárias em direção às

entradas primárias.

Figura 5.5 – Estrutura de dados Porta.



5.1 Processo de Fusão

Como mencionado anteriormente neste capítulo, o processo de fusão é uma etapa do LOMGAM que faz a união de duas portas lógicas, que sejam interconectadas, para formar uma porta complexa. Sendo que a porta que será aglutinada deve obrigatoriamente ter fanout unitário já a porta que recebe a aglutinação não necessita ter fanout igual a 1. O circuito da figura 5.6 exemplifica como esse processo ocorre. A parte a) dessa figura mostra um circuito composto por três portas NAND de duas entradas e um inversor. Além disso, é observado que todas as portas do circuito mencionado possuem fanout igual a um, logo, todas podem ser aglutinadas para formar uma porta complexa. Na parte b) da figura 5.6 é possível perceber que a porta A tem sua saída conectada à entrada do inversor através da interconexão O3. Como uma NAND que tem sua saída conectada a um inversor é logicamente equivalente a uma AND, portanto, é produzida a porta AC como o segundo circuito da parte b) ilustra. A parte c) da figura 5.6 apresenta um outro modo do processo de fusão ser realizado. Neste caso, o teorema de De Morgan é aplicado antes da fusão. Observando a porta B da parte a) da figura mencionada, é percebido que esta é uma NAND2, logo, há necessidade da aplicação do teorema de De Morgan. Assim é obtida uma porta OR2 com um inversor em cada uma de suas entradas. A porta OR2 é aglutinada com a porta complexa AC produzida na etapa anterior, deste modo, a porta complexa ABC, mostrada na parte c) da figura 5.6, é obtida.

A partir do exemplo mostrado na figura 5.6 é possível perceber que toda vez que uma

porta negada (NAND e NOR por exemplo) vai ser aglutinada a outra porta é necessário aplicar o teorema de De Morgan. A parte a) da figura 5.7 mostra como esse teorema funciona: a NOR de duas entradas, depois da aplicação do referido teorema, torna-se uma AND2 com um inversor ligado a cada uma de suas entradas. No primeiro circuito da parte b) da figura 5.7 é observado que as portas B e C podem ser aglutinadas com a porta A. Mas para isso as portas B e C, devido ao fato de serem negadas, devem ser submetidas ao teorema de De Morgan. Consequentemente, existe a necessidade de ser acrescentado um inversor em cada uma das entradas das portas B e C. Como essas portas foram aglutinadas na porta A, a porta ABC é gerada e substitui as portas A, B e C no circuito, como é visto no segundo circuito da parte b) da figura 5.7. Observando o circuito que contém a porta ABC (segundo circuito da parte b) da figura 5.7), nota-se que somente três dos quatro inversores produzidos pelo teorema de De Morgan foram adicionados no circuito. Isso acontece porque a entrada A1 já está conectada a um inversor, o qual originalmente estava conectado a uma das entradas da porta E, portanto, basta conectá-lo à entrada da porta complexa ABC.

Durante o processo de fusão, toda vez que o teorema De Morgan é utilizado, é feita uma varredura no circuito a procura de inversores que sejam iguais (inversores que invertam o mesmo sinal) aos produzidos pelo teorema mencionado, deste modo, evita-se a duplicação de inversores no circuito como mostra o exemplo da parte b) da figura 5.7.

Analisando a parte a) da figura 5.7 em termos de quantidade de transistores, é observado que a NAND2 possui 4 transistores e que a AND2 com entradas negadas tem 6 transistores mais os 4 transistores dos 2 inversores, logo, são necessários 10 transistores para representar uma função booleana que pode ser representada por apenas 4 transistores de uma NAND2. Essa mesma situação é vista nos circuitos da parte b) da figura 5.7, onde o primeiro é composto por 24 transistores e o segundo por 26 transistores. Isso ocorre, porque para produzir a porta complexa ABC, foi necessário usar o teorema de De Morgan. Como cada uma das portas (portas B e C) têm duas entradas haveria a necessidade de uma adição de 4 inversores (8 transistores). Porém, como um desses inversores já existe no circuito, somente 3 inversores (6 transistores são adicionados). Essa situação faz com que, em alguns casos, o processo de fusão, ao invés de reduzir o número de transistores no circuito, acaba aumentando a quantidade de transistores e de interconexões presentes no circuito. Por este motivo foi criado o parâmetro de controle chamado de Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI). Por exemplo, se o valor de NMI for igual a 3, significa que o teorema de De Morgan somente pode ser aplicado em portas com até três entradas. Apesar, deste parâmetro limitar a quantidade de fusões e consequentemente a quantidade de portas complexas produzidas, o NMI busca limitar a quanti-

dade de inversores acrescentados no circuito. Uma vez que nem todos os inversores produzidos pelo teorema já existem no circuito e muitos dos inversores gerados não são "absorvidos"pelo processo de fusão na próxima iteração do mesmo.

Figura 5.6 – Exemplo do processo de fusão.

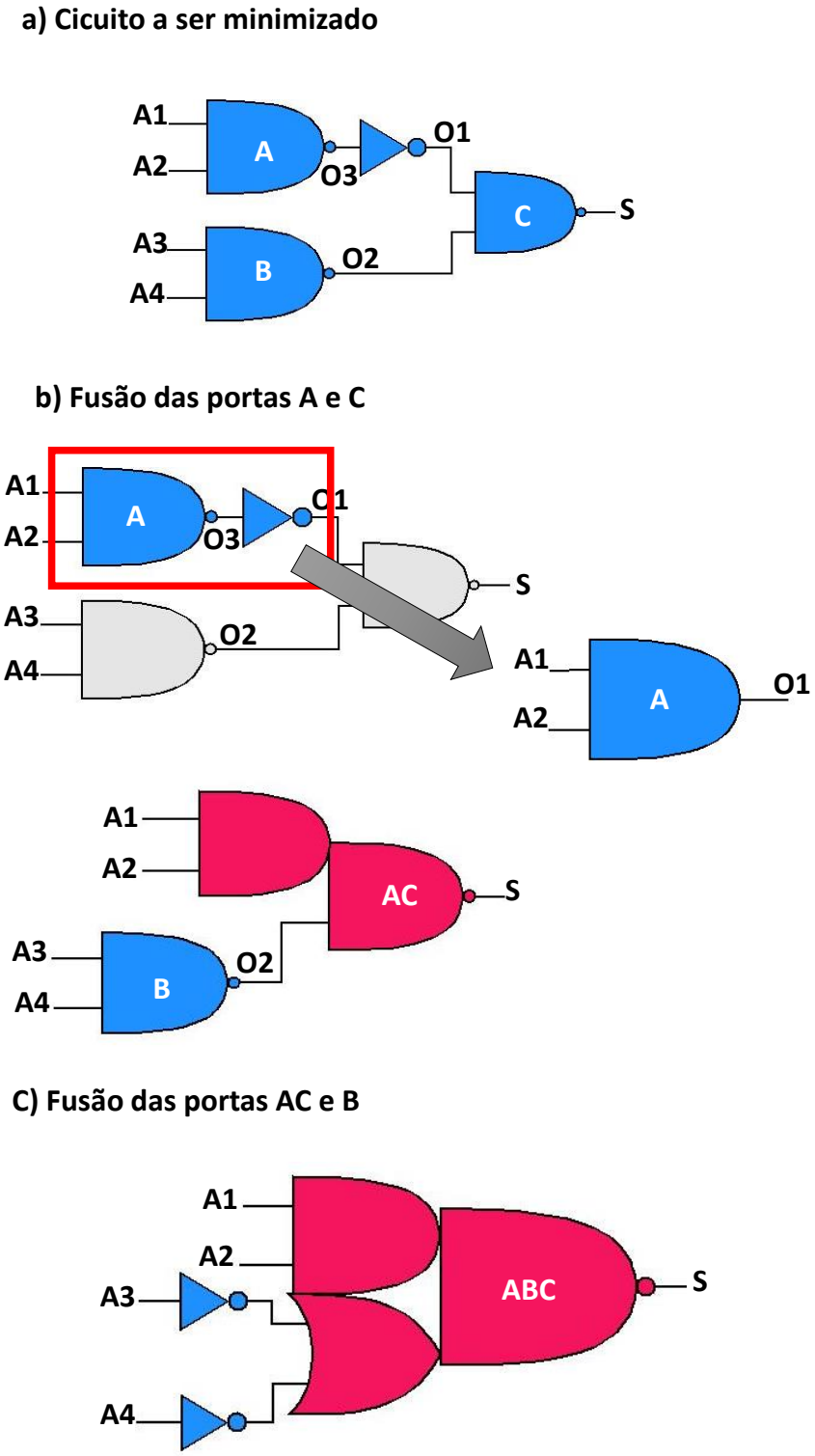
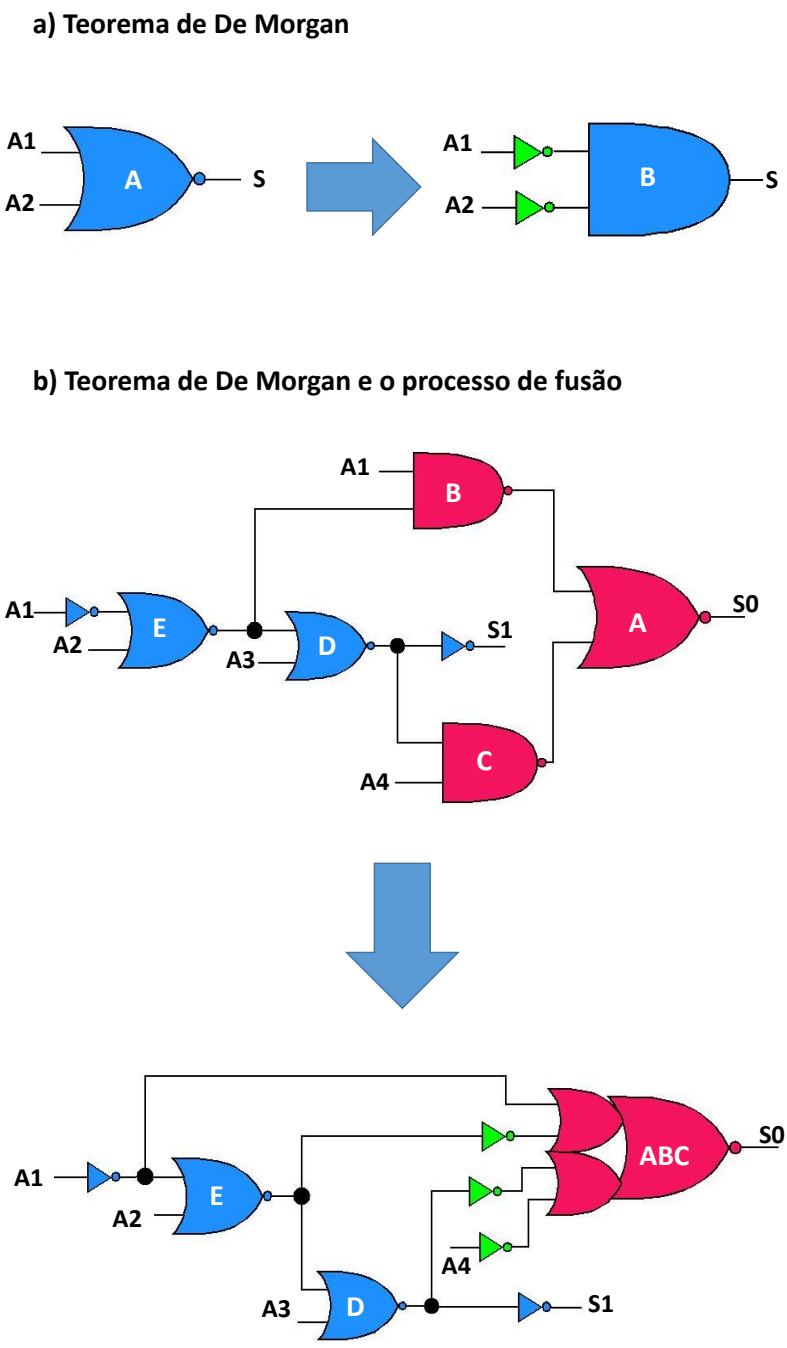


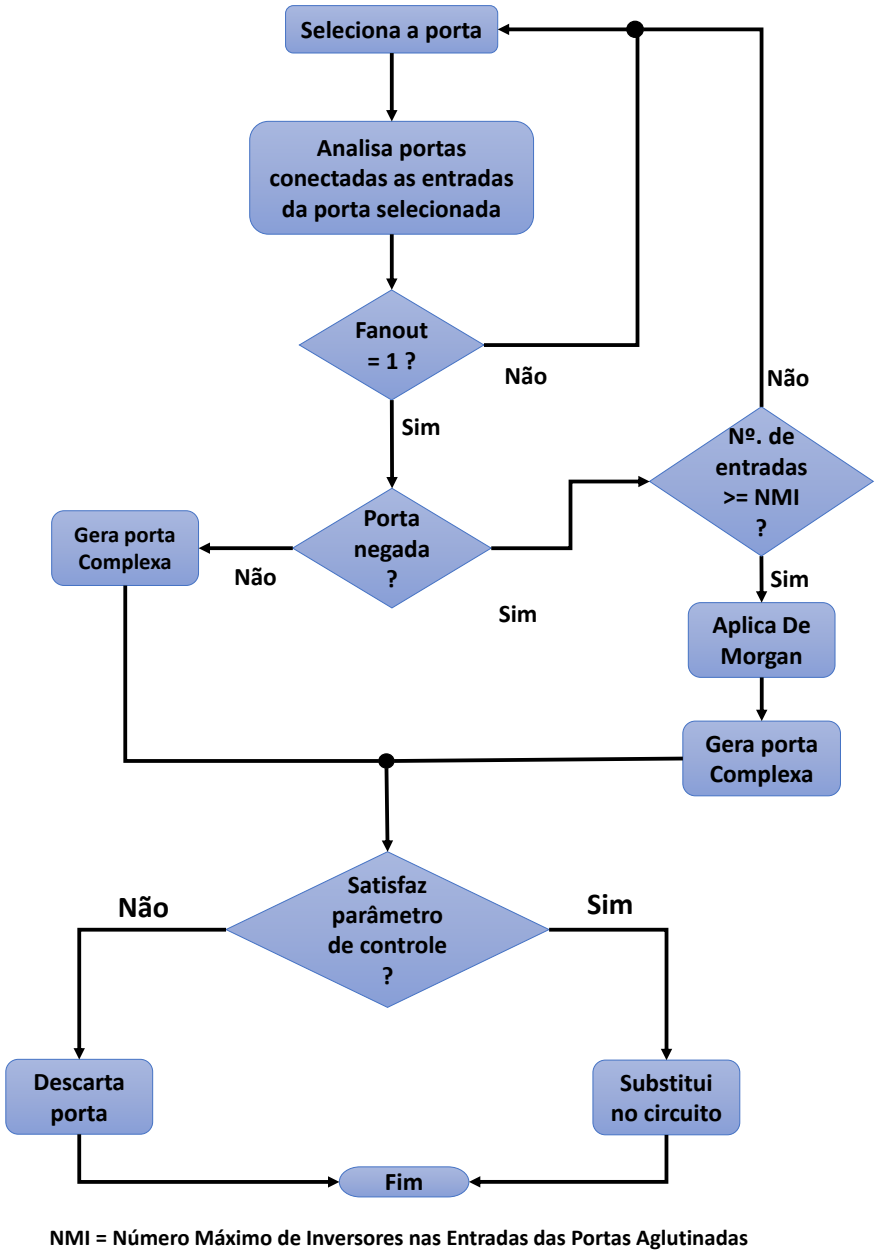
Figura 5.7 – Teorema de De Morgan.



O fluxograma da figura 5.8 descreve detalhadamente o processo de fusão. Este começa selecionando uma porta lógica no circuito e analisa as portas que estão conectadas às entradas da mesma. Em seguida, o método seleciona e analisa uma dessas portas, se ela possuir fanout unitário, é verificado se possui uma negação, ou seja, se é uma NAND ou uma NOR por exemplo. Se a porta não for negada (OR ou AND, por exemplo) a porta complexa é gerada. Então é analisado se a porta complexa produzida está dentro dos parâmetros de controle, que pode ser o Índice de Fusão de Portas ou a Quantidade Máxima de Transistores em Série, se sim a

porta substitui no circuito as portas que a originaram. Retornando ao pondo onde o algoritmo verifica se a porta é negada: se a porta for negada, antes de aplicar o teorema de De Morgan, o método compara o NMI com a quantidade de entradas da porta. Se o NMI for menor ou igual ao número de entradas da porta em questão, o teorema é aplicado, a fusão é realizada e se a porta complexa produzida não for descartada, é realizada uma varredura no circuito para verificar se os inversores produzidos pelo teorema já existem. Se existirem são conectados às entradas da porta complexa produzida. Porém se não existirem são adicionados no circuito como o exemplo da figura 5.7 mostrou.

Figura 5.8 – Fluxograma do processo de fusão.



5.2 Parâmetros de Controle

Conforme dito anteriormente, o usuário pode escolher entre dois tipos de parâmetros de controle: GMI (Índice de Fusão de Portas) que controla somente a quantidade de fusões que uma porta lógica pode sofrer ou QMTS (Quantidade Máxima de Transistores em Série), neste caso, a quantidade máxima de transistores em série, tanto na rede PMOS quanto na rede NMOS, é controlada. Sendo assim, o usuário define os números máximo de PMOS em série e NMOS em série também.

O parâmetro GMI (Índice de Fusão de Portas) foi desenvolvido para controlar a quantidade de portas com fanout unitário que são aglutinadas para formar uma porta complexa. O GMI visa evitar que um grande número de portas de fanout igual a um que estejam em sequência, ao serem submetidas ao processo de fusão, gerem uma porta complexa com uma quantidade elevada de entradas e de transistores em série.

Como o GMI não controla o número de transistores em série nas redes de transistores PMOS e NMOS durante o processo de fusão. Além disso, como a quantidade transistores em série pode influenciar a performance do circuito, foi decidido criar um parâmetro mais eficiente no controle do número de transistores em série, o qual foi denominado de Quantidade Máxima de Transistores em Série (QMTS).

Esta seção está dividida em duas subseções, a primeira aborda o parâmetro GMI e a segunda o parâmetro QMTS.

5.2.1 Índice de Fusão de Portas - GMI

O Índice de Fusão de Portas representa a quantidade máxima de portas lógicas de fanout unitário que podem ser aglutinadas para produzir uma nova porta complexa. A cada iteração do processo de fusão, o GMI é comparado com a quantidade de fusões que já foram realizadas com a porta em questão. Essa quantidade foi denominada de Status. A fusão só é feita se o Status for menor ou igual ao GMI. O Índice de Fusão de Portas foi adotado como parâmetro para evitar que o algoritmo intercale todas as portas de fanout unitário que estiverem em sequência. Isso causaria a produção de portas complexas com um número elevado de entradas e possivelmente de transistores em série. A figura 5.9 detalha o processo de fusão, a parte a) mostra o circuito que passará pelo processo de fusão para um GMI (Índice de Fusão de Portas) = 2 e um NMI (Número Máximo de Inversores nas Entradas das Portas Aglutinadas) = 2. Observe que todas as portas de fanout unitário possuem o Status igual a 0, logo, ainda não passaram pelo processo de

fusão. A parte b) da figura 5.9 mostra o processo de fusão para o circuito da parte a). Primeiro é realizada as fusões das portas G com H e C com F. É possível notar que a porta E também poderia ser aglutinada com as portas G e H, porém isso não é feito porque a porta E tem três entradas que, ao passar pelo teorema de De Morgan, resultaria na criação de três inversores, ou seja, uma quantidade de inversores maior do que a definida (NMI = 2) no exemplo da figura 5.9. Para cada uma dessas fusões os Status das primeiras portas são atualizados. Por exemplo, o Status da porta H é igual ao Status de H mais o Status de G mais 1, como os valores dos Status de H e G são 0, o novo valor do Status de H é 1. Esse valor é comparado com o valor de GMI, que neste exemplo é 2, como 1 é menor que 2, a fusão de G com H é realizada e porta complexa, que foi produzida, substitui as portas H e G no circuito.

A parte b) da figura 5.9 mostra uma segunda interação no processo de fusão, analisando o segundo circuito, vê-se que a porta D foi aglutinada com a porta complexa GH. Primeiramente isso foi possível porque o número de inversores que a porta D produz, ao ser submetida ao teorema de De Morgan, é igual a 2 inversores, ou seja, uma quantidade de inversores igual à quantidade máxima permitida (NMI) que no presente exemplo foi definida em 2. Além disso, o Status da porta complexa DGH é igual ao Status de GH mais o Status de D mais 1, logo, o novo valor do Status da porta DGH é 2, o qual é igual a valor de GMI (no exemplo GMI foi definido em 2). Por tanto, a porta DGH é gerada e substituída no circuito, como o segundo circuito da parte b) da figura 5.9 mostra.

5.2.2 Quantidade Máxima de Transistores em Série - QMTS

No parâmetro QMTS (PMOS, NMOS), o usuário especifica um valor para quantidade máxima de transistores em série para cada uma das redes (*Pull Up e Pull Down*). Neste caso, o processo de fusão começa verificando quantos transistores em série cada uma das redes possui, tanto na equação que vai receber a aglutinação quanto na que será intercalada. Se as duas tiverem um número de transistores menor ou igual aos valores especificados pelo projetista, a fusão é iniciada.

Usando o parâmetro Quantidade Máxima de Transistores em Série, após a quantidade de transistores em série em cada uma das portas ser verificada, a porta complexa é gerada. Então uma nova verificação do número de transistores em série é feita. A parte a) da figura 5.10 mostra o fluxograma que detalha como este processo é realizado. Inicialmente, é gerado um arquivo no formato eqn que contém a porta complexa produzida. Além deste, um arquivo de script também é produzido. Ambos os arquivos são carregados na ferramenta ELIS a qual produz a

descrição Spice da porta complexa que foi produzida. Em seguida, o processo de análise de nodos identifica a quantidade de transistores em série na porta complexa. Essa quantidade é comparada com a Quantidade Máxima de Transistores em Série determinada pelo usuário. Se o par QTS (PMOS, NMOS) (Quantidade de Transistores em Série) for menor ou igual ao par QMTS (PMOS, NMOS), a porta complexa substitui no circuito as portas que a originaram, caso contrário, a porta complexa é descartada.

Figura 5.9 – Processo de Fusão usando GMI.

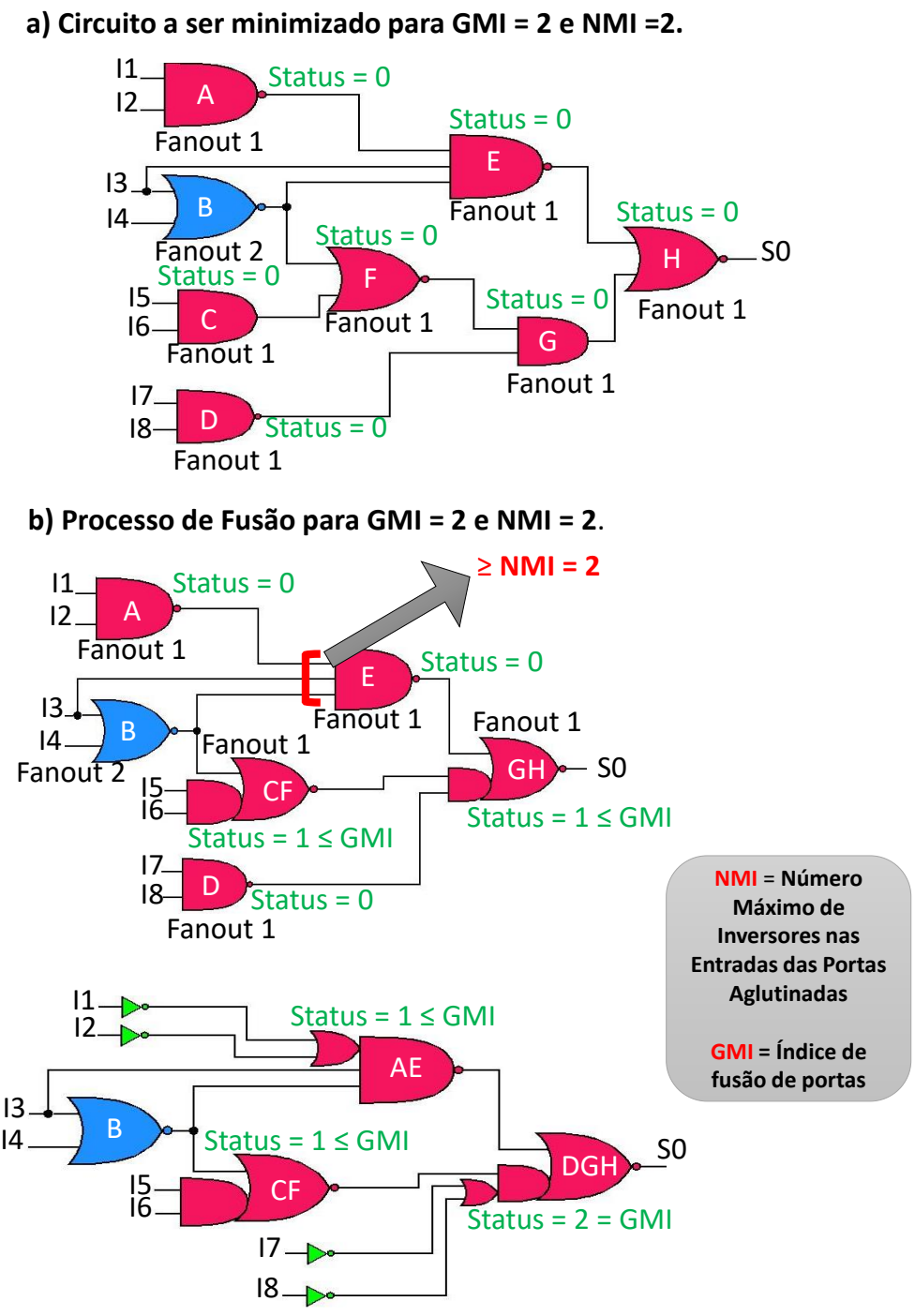
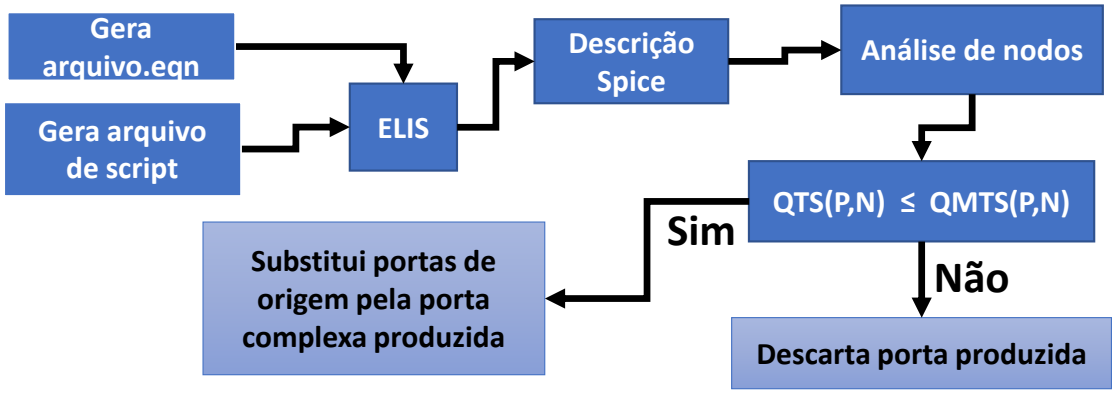


Figura 5.10 – Processo de análise de nodos da equação descrita no formato Spice.

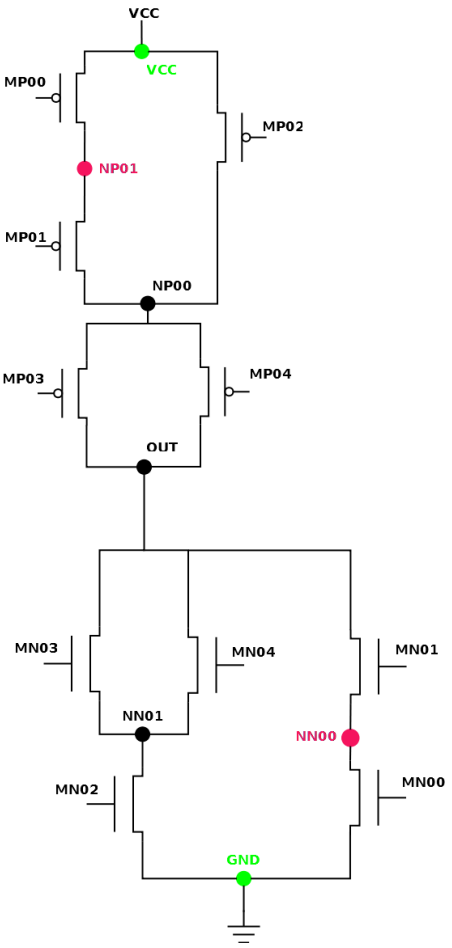
a) Identificação da quantidade de transistores em série.



b) Processo de análise de nodos da equação

S2 = !((A4*A5)+((A1+A2)+A3))

```
.subckt S2 a b c d e out vcc gnd
MP00 NP01 d vcc vcc MODP W=0.630u L=0.050u
MP01 NP00 c NP01 vcc MODP W=0.630u L=0.050u
MP02 NP00 e vcc vcc MODP W=0.630u L=0.050u
MP03 out a NP00 vcc MODP W=0.630u L=0.050u
MP04 out b NP00 vcc MODP W=0.630u L=0.050u
MN00 NN00 b gnd gnd MODN W=0.415u L=0.050u
MN01 out a NN00 gnd MODN W=0.415u L=0.050u
MN02 NN01 e gnd gnd MODN W=0.415u L=0.050u
MN03 out c NN01 gnd MODN W=0.415u L=0.050u
MN04 out d NN01 gnd MODN W=0.415u L=0.050u
.ends S2
```

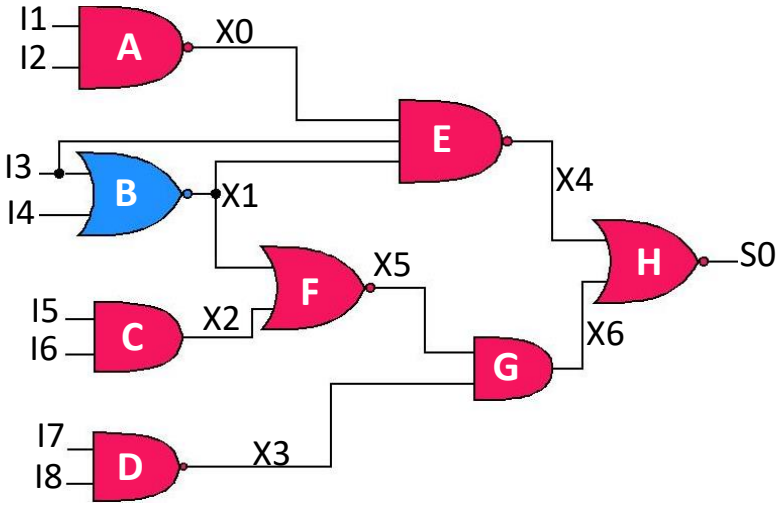


QMTS (PMOS,NMOS) = Quantidade Máxima de Transistores em Série

QTS (PMOS,NMOS) = Quantidade de Transistores em Série

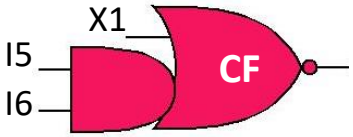
Figura 5.11 – Processo de Fusão usando QMTS(PMOS, NMOS).

a) Circuito a ser minimizado para QMTS (2,3) e NMI =2



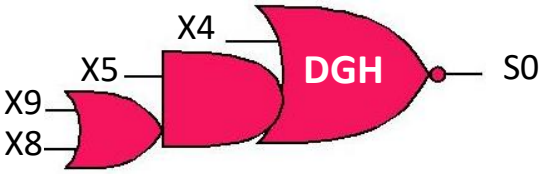
b) Processo de fusão

Fusão das portas C e F

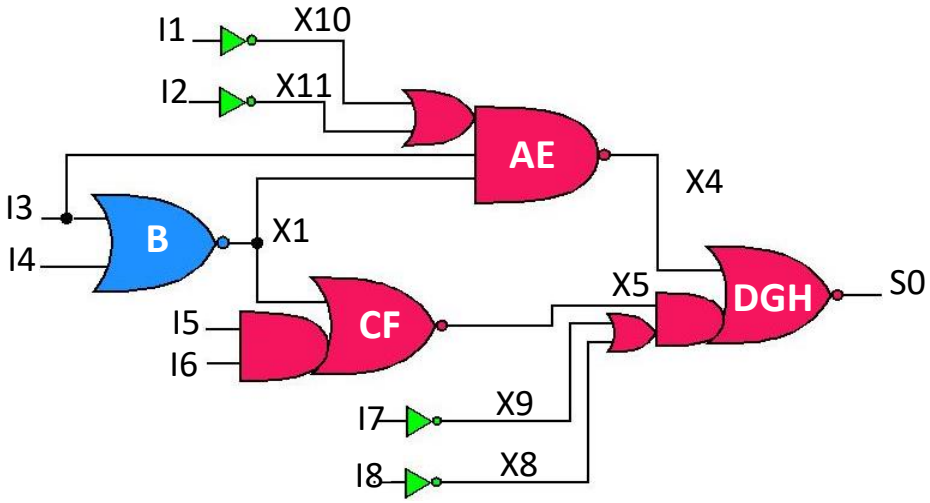


QTS (0,2) ≤ QMTS (2,3)

Fusão das portas GH e D



QTS (2,0) ≤ QMTS (2,3)



QTS (PMOS,NMOS) = Quantidade de Transistores em Série

NMI = Número Máximo de Inversores nas Entradas das Portas Aglutinadas

QMTS (PMOS,NMOS) = Quantidade Máxima de Transistores em Série

O processo de análise de nodos é mostrado na parte b) da figura 5.10. Primeiro o arquivo em formato spice é lido e somente a parte que contém o subcircuito, que descreve a equação, é selecionada. Pois, analisando os subcircuitos das portas lógicas descritos em formato Spice, com exceção dos nodos de Vcc e GND, os nodos que fazem a ligação entre transistores em série possuem apenas duas ocorrências em cada subcircuito. Observando a descrição Spice da equação $S2 =!((A4 * A5) + ((A1 + A2) * A3))$ na figura 5.10 se nota que o nodo NP01 aparece apenas duas vezes. Olhando para o diagrama de transistores, é possível ver que o nodo NP01 conecta os transistores MP00 e MP01. Essa mesma situação ocorre com o nodo NN00 que conecta o dreno do transistor MN00 ao terminal fonte do transistor MN01.

Deste modo, o processo de análise de nodos possui duas etapas. A primeira identifica o número de ocorrências de cada nodo no circuito. A segunda consiste em analisar, para os nodos de duas ocorrências (exceto Vcc e GND), os terminais de dreno e de fonte que estes nodos conectam e identificar todos os transistores que estão em sequência, conectados por nodos de duas ocorrências até encontrar um nodo com um número maior de ocorrências ou os nodos de GND e Vcc. Essa situação é ilustrada pelo caminho, na rede de NMOS, que começa no nodo OUT, passa pelo transistor MN01, pelo nodo NN00, pelo transistor MN00 e termina no nodo GND, presente no diagrama da figura 5.10.

Esse processo é repetido para todos os caminhos que contém transistores em série. No final, o caminho que possui o maior número de transistores em série é escolhido. No caso, do exemplo da figura 5.10, tanto na rede PMOS quanto na NMOS só há um caminho com transistores em série, logo, há 2 PMOS em série e 2 NMOS que também estão em série. Finalmente, esses valores são comparados com as quantidades máxima de transistores em série para cada rede.

A figura 5.11 mostra o processo de fusão utilizando o parâmetro de controle Quantidade Máxima de Transistores em Série (QMTS). A parte a) da referida figura apresenta o circuito que será minimizado para uma Quantidade Máxima de Transistores em Série de 2 transistores em série na rede de transistores PMOS e de 3 transistores em série na rede de transistores NMOS. Além disso, o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) foi definido em 2 neste exemplo. O processo de fusão é detalhado na parte b) da figura 5.11. Depois de verificadas as quantidades de transistores em série nas portas G e H, essas portas são aglutinadas para formar a porta complexa GH. Como a quantidade de transistores em série da porta GH é menor ou igual ao par QMTS, a porta não é descartada. Observando novamente o circuito, é notado que a porta D pode ser aglutinada com a porta GH. Mas para isso, deve ser submetida ao teorema de De Morgan. Primeiro o número de entradas da porta D é

comparado com o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI). Como esses números são iguais, o teorema de De Morgan é aplicado na porta D. Em seguida, a porta D é aglutinada com porta GH, formando a porta DGH. Esta nova porta complexa tem sua quantidade de transistores em série analisada do modo explicado anteriormente nesta seção. Como a quantidade de transistores em série, tanto na rede PMOS quanto na NMOS, da porta DGH é menor ou igual ao par QMTS definido no exemplo, a porta DGH substitui no circuito as portas D, G e H, como mostra o circuito da parte b) da figura 5.11. Analisando mais uma vez o circuito da parte da figura 5.11, é notado que as portas C e F também podem ser aglutinadas, então o processo realizado para a geração da porta DGH é repetido para produzir a porta CF e, como esta satisfaz os parâmetros de controle, ela substitui as portas C e F no circuito como pode ser visto na parte b) da figura 5.11.

Observando atentamente o circuito da figura 5.11, é percebido que a porta E poderia ser aglutinada com a porta H ou com a porta DGH, porém isso não ocorre porque a porta E precisa ser submetida ao teorema de De Morgan e isso produziria 3 inversores, ou seja, uma quantidade maior de inversores do que a definida no exemplo que foi 2 inversores (NMI = 2). Por isso somente a fusão da porta A com a porta E pode ser feita, uma vez que ao aplicar o teorema de De Morgan em A somente 2 inversores são gerados, o que é igual ao valor de NMI e também a Quantidade Máxima de Transistores em Série (2,3) é respeitada.

6 RESULTADOS

Os resultados compreendem o mensuramento do tempo de execução e uma análise quantitativa em termos de: número de transistores e de interconexões e quantidade de fusões realizadas. O objetivo desses experimentos, além de analisar a capacidade de redução no número de transistores e de interconexões presentes nos circuitos, é também explorar a capacidade de minimização que o LOMGAM possui.

Nos experimentos realizados, foram utilizados os circuitos do ITC 99 e do ISCAS 89, a saber: b01, b02, b03, b04, b05, b06, b07, b08, b09, b10, b11, b12, b13, b14, b15, b21, b22 (ITC'99,), s1196, s1423, s1488, s1494, s9234_1, s13207, s35932, s38417, s38584 e s15850 (ISCAS'89,). Os arquivos de netlist no formato Verilog estruturado foram obtidos através do uso do RTL Compiler para uma tecnologia preditiva de 45nm. As netlists foram sintetizadas usando somente as portas lógicas simples presentes na biblioteca, ou seja, AND, NAND, OR e NOR. A exceção do estudo de caso apresentado a seguir, todos os circuitos foram submetidos às mesmas restrições durante o processo de Síntese Lógica na ferramenta mencionada anteriormente. É importante destacar que o método apresentado no presente trabalho é independente de tecnologia, logo, os netlists, em formato Verilog Estruturado dos circuitos mencionados anteriormente, poderiam ser gerados por qualquer ferramenta de Síntese Lógica. Isso é possível porque o algoritmo proposto preocupa-se apenas com as portas lógicas presentes no circuito, portanto, não considera parâmetros tecnológicos inerentes aos processos tecnológicos. Sendo assim, o único critério utilizado para a escolha da tecnologia de 45nm foi o fato desta estar disponível para download gratuito no site da empresa que a elaborou.

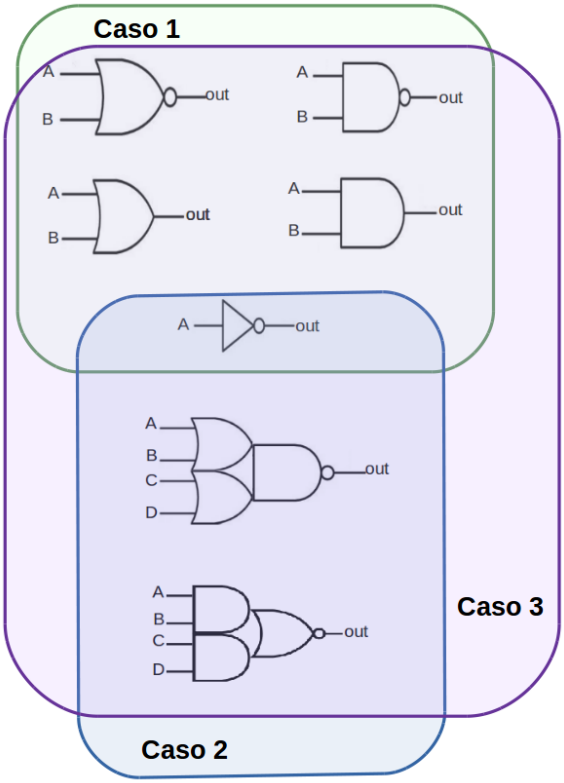
Este capítulo é organizado em 4 seções, a saber: seção 1 aborda um estudo de caso sobre o potencial da técnica; seção 2 apresenta uma análise quantitativa em termos de transistores, de interconexões e de tempo de execução para o parâmetro GMI; a seção 3 também aborda uma análise quantitativa de transistores e de interconexões, porém utilizando o parâmetro QMTS e a última seção apresenta uma comparação, em termos de transistores, entre a LOMGAM e a RTL Compiler.

6.1 Potencial de minimização: um estudo de caso

Visando analisar o potencial de minimização de equações booleanas pelo algoritmo proposto, foram implementadas as duas primeiras etapas do mesmo, ou seja, a identificação das interconexões e a identificação do fanout unitário. A partir da análise das funções lógicas dis-

poníveis na biblioteca de 45nm, foi observado a presença de três tipos de células: simples (ANDs, NANDs, NORs, ORs e Inversores), complexas (AIOs, OIAs, XOR, NXOR, MUX (Multiplexador), FA (Somador Completo) e HA (Meio Somador)). Como as funções XOR, NXOR, MUX, FA e HA podem ser representadas por portas lógicas simples, estas não foram incluídas nos casos estudados. Assim foram definidos os três casos a seguir: caso 1, circuitos formados por somente portas simples; caso 2, netlists constituídas por somente portas complexas, exceto XOR, NXOR, MUX, FA e HA; e caso 3, circuitos compostos por portas simples e complexas com exceção das células mencionadas anteriormente. Esses três casos são ilustrados na figura 6.1.

Figura 6.1 – Composição dos casos estudados.



Fonte: (SILVA; BONTORIN; REIS, 2015)

Neste experimento, foram utilizados os circuitos b14, b15, b21 b22 (ITC'99,), s35932, s38417 e s38584 (ISCAS'89,). Para cada um desses foi gerado três netlists, uma para cada caso. Isso foi possível através de restrição de quais portas lógicas da biblioteca de 45nm que o RTL Compiler poderia usar no processo de Síntese Lógica. Depois os arquivos, em Verilog estruturado, foram submetidos às primeiras etapas do algoritmo (identificação das interconexões e das portas combinacionais e identificação do fanout), que haviam sido implementadas. Em seguida as portas lógicas de fanout igual a um são selecionadas e contagem de células combinacionais e

interconexões que tem potencial para a minimização foi realizada (SILVA; BONTORIN; REIS, 2015).

Os resultados são apresentados nas tabelas 6.1 e 6.2. A primeira tabela é formada pelo total de portas lógicas que compõem o circuito, pela quantidade de portas que podem ser minimizadas e pelo percentual destas em relação ao total. Essas informações são mostradas para os três casos estudados. A segunda também está organizada do mesmo modo, porém apresenta as quantidades em termos de interconexões (SILVA; BONTORIN; REIS, 2015).

Tabela 6.1 – Quantidade de células que podem ser minimizadas pelo processo de fusão
Fonte: (SILVA; BONTORIN; REIS, 2015)

	Caso 1			**Caso 2**			**Caso3**		
	Células que podem ser minimizadas			**Células que podem ser minimizadas**			**Células que podem ser minimizadas**		
Circuito	**Total (unid)**	**(unid)**	**%**	**Total (unid)**	**(unid)**	**%**	**(unid)**	**Total (unid)**	**%**
b14	3629	2193	60,43	2248	1169	52,00	2248	1169	52,00
b15	6661	3962	59,48	3833	1706	44,51	3833	1706	44,51
b21	5001	5001	60,48	2754	2754	53,60	2754	2754	53,60
b22	12459	7401	59,40	7787	4121	52,92	7787	4121	52,92
s35932	7444	3090	41,51	4645	1332	28,68	4645	1322	28,46
s38417	6225	3011	48,37	4244	1448	34,12	4244	1448	34,12
s38584	6503	3054	46,96	4875	1571	32,23	4875	1571	32,23
Média			**57,83**			**46,31**			**46,37**

Tabela 6.2 – Interconexões de células que podem ser minimizadas pelo processo de fusão
Fonte: (SILVA; BONTORIN; REIS, 2015)

	Caso 1			**Caso 2**			**Caso3**		
	Interconexões que podem ser minimizadas			**Interconexões que podem ser minimizadas**			**Interconexões que podem ser minimizadas**		
Circuito	**Total (unid)**	**(unid)**	**%**	**Total (unid)**	**(unid)**	**%**	**(unid)**	**Total (unid)**	**%**
b14	4316	2193	50,81	2958	1169	39,22	2958	1169	39,52
b15	7649	3962	51,80	4906	1706	34,77	5042	1706	33,84
b21	9796	5001	51,05	6884	2754	40,01	6884	2754	40,01
b22	14834	7401	49,89	10217	4121	40,33	10217	4121	40,33
s35932	7299	3090	42,33	4541	1332	29,33	4561	1322	29,11
s38417	6652	3011	45,26	4690	1448	30,87	4690	1448	30,87
s38584	6600	3054	46,27	5011	1571	31,35	5011	1571	31,35
Média			**48,49**			**35,97**			**35,82**

Analisando as tabelas é percebido que os casos 2 e 3 possuem características semelhantes devido ao uso de portas complexas. Comparando os percentuais médios dos casos 2 e 3 é possível comprovar a afirmação, feita no início do presente parágrafo, estes casos apresentaram um potencial de minimização médio de células de 46,31% e de 46,37%, respectivamente. Os

mesmos casos também apresentam uma menor quantidade de portas quando comparados com o caso 1. Isso ocorre porque o caso 1 só utiliza portas simples, logo, usa mais portas para representar funções que poderiam ser descritas por apenas uma porta AOI, por exemplo. Além disso, e também por englobar os dois tipos de portas, foi concluído que o RTL Compiler prioriza o uso de portas complexas (SILVA; BONTORIN; REIS, 2015) quando se analisa os resultados do caso 3.

Situação semelhante é observada quando se compara os percentuais médios de redução no número de interconexões para os casos 2 e 3, ou seja, a média de redução na quantidade de interconexões para o caso 2 foi de 35,97% e para o caso 3 foi de 35,82%. Isso ocorre pelo fato dos dois casos utilizarem portas complexas e também porque o RTL Compiler deu prioridade para o uso de portas complexas. Como o caso 1 utiliza somente portas simples, há um maior potencial de redução no número de interconexões. Como dito no parágrafo anterior, tal situação acontece porque mais portas são necessárias para representar o circuito, uma vez que uma função booleana que poderia ser representada por uma única porta complexa, nesse caso, tem que ser representada por um conjunto de portas simples. Consequentemente, um número maior de interconexões são usadas no caso 1, como mostra o percentual médio do potencial de redução no número de interconexões do caso 1 apresentado na tabela 6.2.

As tabelas 6.1 e 6.2 também mostram que o uso de portas simples aumenta a quantidade de células que podem ser simplificadas pelo método proposto. Isso também significa uma maior redução no número de transistores. Isto foi comprovado por uma metodologia gulosa baseada no algoritmo Flood Fill (Algoritmo do jogo Campo Minado) (CONCEIÇÃO; POSSER; REIS, 2016) que obteve uma redução, em média de 17,5% na quantidade de transistores quando as netlists eram compostas apenas por portas simples e de 9,8% quando também são usadas portas complexas (CONCEIÇÃO; POSSER; REIS, 2016).

6.2 Análise quantitativa e tempo de execução para o parâmetro GMI (Índice de Fusão de Portas)

Nesta análise quantitativa, primeiramente foi gerado a netlist de cada um dos circuitos (b01, b02, b03, b04, b05, b06, b07, b08, b09, b10, b11, b12, b13, b14, (ITC'99,), s1196, s1423, s1488, s1494, s9234_1, s13207, s35932, s38417, s38584 e s15850 (ISCAS'89,)). Essas netlists foram carregadas na LOMGAM, várias vezes, porque foram usados diferentes índices de fusão de portas (GMI): 1, 2, 3, 4, 5, 6, 8, 10 e 50. Uma vez que o objetivo era avaliar o impacto deste parâmetro na quantidade de transistores e de interconexões, bem como no tempo de execução

do algoritmo e no número de fusões realizadas. Como saída o LOMGAM produz uma netlist minimizada no formato eqn e informa a quantidade de transistores, número de interconexões e de fusões que foram feitas.

Tabela 6.3 – Quantidade de Transistores por GMI para NMI = 0

		GMI					
	Circuito não minimizado	3		5		10	
	Número de transistores	**Número de transistores**		**Número de transistores**		**Número de transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	126	-4,55	126	-4,55	126	-4,55
b02	94	90	-4,26	90	-4,26	90	-4,26
b03	248	238	-4,03	238	-4,03	238	-4,03
b04	1300	1244	-4,31	1244	-4,31	1244	-4,31
b05	1694	1588	-6,26	1588	-6,26	1588	-6,26
b06	122	106	-13,11	104	-14,75	104	-14,75
b07	1196	1102	-7,86	1102	-7,86	1102	-7,86
b08	336	322	-4,17	322	-4,17	322	-4,17
b09	428	384	-10,28	384	-10,28	384	-10,28
b10	446	410	-8,07	410	-8,07	410	-8,07
b11	1910	1760	-7,85	1760	-7,85	1760	-7,85
b12	2670	2534	-5,09	2534	-5,09	2534	-5,09
b13	700	652	-6,89	652	-6,89	652	-6,89
b14	11698	10960	-6,31	10960	-6,31	10960	-6,31
s1196	1832	1710	-6,66	1710	-6,66	1710	-6,66
s1423	1866	1764	-5,47	1764	-5,47	1764	-5,47
s1488	2068	1906	-7,83	1906	-7,83	1906	-7,83
s1494	2108	1934	-7,99	1934	-7,99	1934	-7,99
s9234_1	2966	2586	-11,67	2586	-11,67	2586	-11,67
s13207	2762	2596	-6,01	2596	-6,01	2596	-6,01
s35932	29618	26944	-9,03	26944	-9,03	26944	-9,03
s15850	1578	1476	-6,46	1476	-6,46	1476	-6,46
Média			**-7,34**		**-7,57**		**-7,47**

GMI = Índice de Fusão de Portas

As tabelas 6.3, 6.4, 6.5 e 6.6 possuem a quantidade de transistores para os Índices de Fusão de Portas (GMI) de 3, 5 e 10 para os Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) 0, 2, 3 e 4. A segunda coluna de cada uma dessas tabelas contém a quantidade de transistores no circuito combinacional antes do processo de minimização e as demais colunas apresentam o número de transistores após o processo de fusão. Além disso, as referidas tabelas apresentam os percentuais de redução na quantidade de transistores em relação ao número de transistores presentes nas versões não minimizadas dos circuitos.

Analisando a tabela 6.3 é notado que para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) = 0 a variação no valor do Índice de Fusão de Portas (GMI) não significou uma maior redução na quantidade de transistores. A exceção do circuito b06, no qual o número de transistores caiu de 106 para GMI=3 para 104 com GMI = 5. Além disso,

esse circuito apresentou o maior percentual de redução na quantidade de transistores quando comparado com os demais circuitos testados. Em conta partida, o circuito b03 obteve o menor percentual de redução na quantidade de transistores. É importante destacar que com um NMI=0, nenhuma porta foi submetida ao teorema de De Morgan, logo, nenhum inversor foi adicionado ao circuito. Isso significa que somente portas não negadas, como ANDs e ORs, por exemplo, passaram pelo processo de fusão.

Tabela 6.4 – Quantidade de Transistores por GMI para NMI = 2

		GMI					
	Circuito não minimizado	3		5		10	
	Número de transistores	**Número de transistores**		**Número de transistores**		**Número de transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	126	-4,55	124	-6,06	124	-6,06
b02	94	90	-4,26	88	-6,38	88	-6,38
b03	248	234	-5,65	232	-6,45	232	-6,45
b04	1300	1260	-3,08	1258	-3,23	1258	-3,23
b05	1694	1604	-5,31	1602	-5,45	1602	-5,45
b06	122	104	-14,75	100	-18,03	100	-18,03
b07	1196	1098	-8,19	1098	-8,19	1098	-8,19
b08	336	344	2,28	342	1,79	342	1,79
b09	428	378	-11,68	372	-13,08	370	-13,11
b10	446	416	-6,73	414	-7,17	414	-7,17
b11	1910	1788	-6,39	1774	-7,12	1764	-7,64
b12	2670	2564	-3,97	2544	-4,72	2534	-5,09
b13	700	676	-3,43	672	-4,00	670	-4,29
b14	11698	10860	-7,16	10716	-8,39	10602	-9,37
s1196	1832	1734	-5,35	1718	-6,22	1708	-6,77
s1423	1866	1814	-2,79	1806	-3,22	1806	-3,22
s1488	2068	1916	-7,35	1888	-8,70	1868	-9,67
s1494	2108	1944	-7,52	1908	-9,23	1896	-9,80
s9234_1	2966	2586	-9,44	2628	-11,40	2604	-12,20
s13207	2762	2650	-4,06	2648	-4,13	2648	-4,13
s35932	29618	27610	-6,78	27324	-7,75	27324	-7,75
s15850	1578	1510	-4,31	1500	-4,94	1498	5,07
Média			**-6,21**		**-7,24**		**-7,51**

GMI = Índice de Fusão de Portas

Os dados da tabela 6.4 foram obtidos definindo-se o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) em 2 e para o Índice de Fusão de Portas foi usado a mesma variação da tabela 6.4. Um NMI = 2 significa que portas lógicas negadas com duas entradas podem ser submetidas ao teorema de De Morgan, logo, cada aplicação do teorema pode acrescentar dois inversores no circuito por fusão realizada. Observando a tabela 6.4 é percebido que novamente o circuito b06 apresentou o maior percentual de redução na quantidade de transistores. Porém, desta vez, o circuito s1423 foi que teve a menor redução na quantidade de transistores. Além disso, analisando os percentuais médios de redução na quantidade de

transistores, é percebido que quanto maior for o valor do Índice de Fusão de Portas maior é o percentual médio de redução no número de transistores.

Tabela 6.5 – Quantidade de Transistores por GMI para NMI = 3

		GMI					
	Circuito não minimizado	3		5		10	
	Número de transistores	**Número de transistores**		**Número de transistores**		**Número de transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	126	-4,55	124	-6,06	124	-6,06
b02	94	88	-6,38	86	-8,51	86	-8,51
b03	248	234	-5,65	232	-6,45	232	-6,45
b04	1300	1272	-2,15	1270	-2,31	1270	-2,31
b05	1694	1616	-4,60	1614	-4,72	1612	-4,84
b06	122	104	-14,75	100	-18,03	98	-19,67
b07	1196	1102	-7,86	1100	-8,03	1100	-8,03
b08	336	340	1,19	342	1,79	342	1,79
b09	428	386	-9,81	380	-11,21	378	-11,68
b10	446	430	-3,59	428	-4,04	428	-4,04
b11	1910	1810	-5,24	1790	-6,28	1780	-6,81
b12	2670	2584	-3,22	2562	-4,04	2552	-4,42
b13	700	672	-4,00	648	-4,57	666	-4,86
b14	11698	10880	-6,99	10738	-8,21	10624	-9,18
s1196	1832	1768	-3,49	1750	-4,48	1738	-5,13
s1423	1866	1810	-3,00	1802	-3,43	1802	-3,43
s1488	2068	1940	-6,19	1910	-7,64	1892	-8,51
s1494	2108	1992	-5,23	1962	-6,66	1948	-7,33
s9234_1	2966	2690	-9,31	2630	-11,33	2602	-12,27
s13207	2762	2644	-4,27	2642	-4,34	2642	-4,34
s35932	29618	27612	-6,77	27326	-7,74	27326	-7,74
s15850	1578	1526	-3,30	1518	-3,80	1518	-3,80
Média			**-5,67**		**-6,67**		**-7,03**

GMI = Índice de Fusão de Portas

A tabela 6.5 possui a quantidade de transistores para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 3 e Índices de Fusão de Portas (GMI) 3, 5 e 10. A partir das informações desta tabela, é notado que o circuito b04 apresentou a menor redução na quantidade de transistores quando comparado com os demais circuitos. Novamente, o b06 apresentou o maior percentual de redução na quantidade de transistores. Porém o b06 apresentou os mesmos percentuais obtidos para NMI=2 (tabela 6.4). Isso mostra que submeter portas com até 3 entradas ao teorema de De Morgan, nesse caso, não causou impacto na redução da quantidade de transistores. Mas, assim como foi visto na tabela 6.4, um aumento no valor de GMI permitiu um maior percentual médio de redução no número de transistores.

Para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 4 e para os Índices de Fusão de Portas (GMI) 3, 5 e 10 foram obtidos os dados contidos na tabela 6.6. O circuito b06 novamente apresentou o maior percentual de redução no número

de transistores, mantendo os mesmos valores apresentados para os NMIs 2 e 3, isso significa que esse circuito atingiu o seu percentual máximo de fusão e que um aumento no valor de NMI não proporcionará uma redução maior na quantidade de transistores. O circuito s15850 obteve o menor percentual de redução no número de transistores.

Tabela 6.6 – Quantidade de Transistores por GMI para NMI = 4

		GMI					
	Circuito não minimizado	3		5		10	
	Número de transistores	**Número de transistores**		**Número de transistores**		**Número de transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	126	-4,55	124	-6,06	124	-6,06
b02	94	88	-6,38	86	-8,51	86	-8,51
b03	248	234	-5,65	232	-6,45	232	-6,45
b04	1300	1272	-2,15	1270	-2,31	1270	-2,31
b05	1694	1624	-4,13	1622	-4,25	1620	-4,34
b06	122	104	-14,75	100	-18,03	98	-19,67
b07	1196	1102	-7,86	1100	-8,03	1100	-8,03
b08	336	344	2,38	342	1,79	342	1,79
b09	428	388	-9,35	386	-9,81	386	-9,81
b10	446	418	-6,18	416	-6,73	416	-6,73
b11	1910	1834	-3,98	1814	-5,03	1808	-5,34
b12	2670	2586	-3,15	2564	-3,97	2554	-4,34
b13	700	672	-4,00	668	-4,57	666	-4,86
b14	11698	11380	-2,72	11242	-3,90	11178	-4,45
s1196	1832	1776	-3,06	1758	-4,04	1746	-4,69
s1423	1866	1832	-1,82	1828	-2,04	1828	-2,04
s1488	2068	2012	-2,71	1990	-3,77	1982	-4,16
s1494	2108	2070	-1,52	2058	-2,09	2050	-2,47
s9234_1	2966	2814	-5,12	2772	-6,54	2754	-7,15
s13207	2762	2654	-3,91	2652	-3,98	2652	-3,98
s35932	29618	27612	-6,77	27326	-7,74	27326	-7,74
s15850	1578	1556	-1,39	1554	-1,52	1552	-1,65
Média			**-4,71**		**-5,60**		**-5,86**

GMI = Índice de Fusão de Portas

Analisando a quantidade de transistores no circuito b08 nas tabelas 6.3, 6.4, 6.5 e 6.6, é notado que só houve redução na quantidade de transistores quando NMI era 0 e que nos demais casos ocorreu um aumento no número de transistores de 1,79% para GMI = 10. Isso mostra que a quantidade de transistores adicionados pelo acréscimo de inversores no circuito foi maior do que a quantidade de transistores reduzida pela criação das portas complexas.

Quando é realizada uma comparação entre todos os percentuais médios de redução no número de transistores mostrados nas tabelas 6.3, 6.4, 6.5 e 6.6, é notado que esses percentuais de redução na quantidade de transistores aumentam até NMI (Número Máximo de Inversores nas Entradas das Portas Aglutinadas) = 3 e que decaem em NMI=4. Deste modo, é possível

concluir que provavelmente um Número Máximo de Inversores nas Entradas das Portas Aglutinadas maiores que 4, não seja mais possível reduzir a quantidade de transistores, pois a medida que as portas complexas diminuam a quantidade de transistores, a necessidade de inserção de inversores no circuito, para manter a sua funcionalidade lógica, cause um aumento no número de transistores no circuito.

Entende-se por interconexão o "fio"que conecta a saída de uma porta às entradas de outras portas lógicas. Se a porta possuir fanout maior que 1, este "fio"terá ramificações, caso contrário, ligará a saída de uma porta à entrada de outra. As tabelas 6.7, 6.8, 6.9 e 6.10 possuem a quantidade de interconexões para os Índices de Fusão de Portas (GMI) de 3, 5 e 10 para os Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) 0, 2, 3 e 4. A segunda coluna de cada uma dessas tabelas contém o número de interconexões presentes no circuito combinacional antes do processo de minimização e as demais colunas apresentam a quantidade de interconexões após o processo de fusão. Além disso, essas tabelas possuem os percentuais de redução no número de interconexões em relação a quantidade de interconexões presentes nas versões não minimizadas dos circuitos.

Tabela 6.7 – Quantidade de interconexões por GMI para NMI = 0

		GMI					
	Circuito não minimizado	3		5		10	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	34	-8,11	34	-8,11	34	-8,11
b02	26	24	-7,69	24	-7,69	24	-7,69
b03	66	61	-7,58	61	-7,58	61	-7,58
b04	362	334	-7,63	334	-7,63	334	-7,63
b05	454	402	-11,45	402	-11,45	402	-11,45
b06	34	28	-17,65	27	-20,59	27	-20,59
b07	332	286	-13,86	286	-13,86	286	-13,86
b08	84	77	-8,33	77	-8,33	77	-8,33
b09	128	106	-17,19	106	-17,19	106	-17,19
b10	117	99	-15,38	99	-15,38	99	-15,38
b11	509	434	-14,73	434	-14,73	434	-14,73
b12	707	639	-9,62	639	-9,62	639	-9,62
b13	202	178	-11,88	178	-11,88	178	-11,88
b14	3133	2764	-11,88	2764	-11,88	2764	-11,88
s1196	469	408	-13,01	408	-13,01	408	-13,01
s1423	494	443	-10,32	443	-10,32	443	-10,32
s1488	518	437	-10,32	437	-10,32	437	-10,32
s1494	526	442	-15,97	442	-15,97	442	-15,97
s9234_1	814	663	-18,55	652	-19,90	652	-19,90
s13207	741	658	-11,20	658	-11,20	658	-11,20
s35932	8215	6878	-16,28	6878	-16,28	6878	-16,28
s15850	427	376	-11,94	376	-11,94	376	-11,94
Média			**-13,14**		**-13,34**		**-13,34**

GMI = Índice de Fusão de Portas

Observando a tabela 6.7 é notado que, para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) = 0, a variação no valor do Índice de Fusão de Portas (GMI) não significou uma grande redução no percentual médio de diminuição no número de interconexões. A exceção do circuito b06, no qual o número de interconexões caiu de 28 para GMI=3 para 27 com GMI = 5. Por outro lado, o circuito b03 obteve o menor percentual de redução no número de interconexões. É importante destacar que com um NMI=0, nenhuma porta foi submetida ao teorema de De Morgan, logo, nenhum inversor foi adicionado ao circuito. Isso significa que nenhuma interconexão foi adicionada ao circuito por conta do acréscimo de inversores no mesmo.

Tabela 6.8 – Quantidade de interconexões por GMI para NMI = 2

		GMI					
	Circuito não minimizado	3		5		10	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	34	-8,11	33	-10,81	33	-10,81
b02	26	24	-7,69	23	-11,54	23	-11,54
b03	66	59	-10,61	58	-12,12	58	-12,12
b04	362	342	-5,52	341	-5,80	341	-5,80
b05	454	410	-9,69	409	-9,91	409	-9,91
b06	34	27	-20,59	25	-26,47	25	-26,47
b07	332	284	-14,46	284	-14,46	284	-14,46
b08	84	88	4,76	87	3,57	87	3,57
b09	128	103	-19,53	100	-21,88	99	-22,66
b10	117	102	-12,82	101	-13,68	101	-13,68
b11	509	448	-11,98	441	-13,36	436	-14,34
b12	707	654	-7,50	644	-8,91	639	-9,62
b13	202	190	-5,34	188	-6,93	187	-7,43
b14	3133	2714	-13,37	2642	-15,67	2585	-17,49
s1196	469	420	-10,45	412	-12,15	407	-13,22
s1423	494	468	-5,64	464	-6,07	464	-6,07
s1488	518	442	-14,67	428	-17,37	418	-19,31
s1494	526	447	-15,02	429	-18,44	423	-19,58
s9234_1	814	696	-14,50	673	-17,32	661	-18,80
s13207	741	685	-7,56	684	-7,69	684	-7,69
s35932	8215	7211	-12,22	7068	-13,96	7068	-13,96
s15850	427	393	-7,96	388	-9,13	387	-9,37
Média			**-10,99**		**-12,86**		**-13,37**

GMI = Índice de Fusão de Portas

Os dados da tabela 6.8 foram obtidos através da definição do Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) em 2, e para o Índice de Fusão de Portas foi usado a mesma variação da tabela 6.7. Observando a tabela 6.8 é percebido que o circuito b06 apresentou o maior percentual de redução no número de interconexões. Porém, desta vez, foi o circuito s1423 que obteve a menor redução no número de interconexões. Além disso, analisando os percentuais médios de redução no número de interconexões, é notado que quanto maior for

o valor do Índice de Fusão de Portas maior é o percentual médio de redução na quantidade de interconexões.

A tabela 6.9 apresenta o número de interconexões para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 3 e Índices de Fusão de Portas (GMI) 3, 5 e 10. A partir das informações desta tabela, é notado que o circuito b04 apresentou a menor redução no número de interconexões quando comparado com os demais circuitos. Novamente, o b06 obteve o maior percentual de redução no número de interconexões.

Para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 4 e para os Índices de Fusão de Portas (GMI) 3, 5 e 10, foram obtidas informações apresentadas na tabela 6.10. O circuito b06 novamente apresentou o maior percentual de redução na quantidade de interconexões, mantendo os mesmos valores apresentados para os NMI 2 e 3. Por tanto, o circuito b06 atingiu o seu percentual máximo de fusão e um aumento no valor de NMI não proporcionará uma redução maior no número de interconexões. Em contrapartida, o circuito s15850 apresentou menor percentual de redução na quantidade de interconexões.

Tabela 6.9 – Quantidade de interconexões por GMI para NMI = 3

		GMI					
	Circuito não minimizado	3		5		10	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	34	-8,11	33	-10,81	33	-10,81
b02	26	23	-11,54	22	-15,38	22	-15,38
b03	66	59	-10,64	58	-12,12	58	-12,12
b04	362	348	-3,87	347	-4,14	347	-4,14
b05	454	416	-8,37	415	-8,59	414	-8,81
b06	34	27	-20,59	25	-26,47	25	-26,47
b07	332	286	-13,86	285	-14,16	285	-14,16
b08	84	88	4,76	87	3,57	87	3,57
b09	128	107	-16,41	104	-18,75	103	-19,53
b10	117	109	-6,84	108	-7,69	108	-7,69
b11	509	459	-9,82	449	-11,79	444	-12,77
b12	707	664	-6,08	653	-7,64	648	-8,35
b13	202	188	-6,83	186	-7,92	185	-8,42
b14	3133	2724	-13,05	2653	-15,32	2596	-17,14
s1196	469	437	-6,82	428	-8,74	422	-10,02
s1423	494	466	-5,57	462	-6,48	462	-6,48
s1488	518	454	-12,36	439	-15,25	430	-16,99
s1494	526	471	-10,46	456	-13,31	449	14,64
s9234_1	814	698	-14,25	674	-17,20	660	-18,92
s13207	741	682	-7,96	681	-8,10	681	-8,10
s35932	8215	7212	-12,21	7069	-13,95	7069	-13,95
s15850	427	401	-6,09	397	-7,03	397	-7,03
Média			**-9,86**		**-11,77**		**-12,44**

GMI = Índice de Fusão de Portas

Comparando o número de interconexões contidas no circuito b08 nas tabelas 6.7, 6.8, 6.9 e 6.10, é percebido que só ocorreu uma redução no número de interconexões, quando NMI foi definido em 0. Nos demais casos aconteceu um aumento na quantidade de interconexões de 3,57% para GMI = 10. Isso mostra que a quantidade de inversores adicionados causou um considerável aumento na quantidade de interconexões, uma vez que, os inversores acrescentados precisam ser interconectados ás entradas das portas lógicas.

Quando é realizada uma comparação entre todos os percentuais médios de redução no número de transistores mostrados nas tabelas 6.3, 6.4, 6.5 e 6.6 com os percentuais médios de redução na quantidade de interconexões apresentados nas tabelas 6.7, 6.8, 6.9 e 6.10, é percebido que, em média, o processo de fusão permite uma maior redução no número de interconexões do que na quantidade de transistores. Porque, em média, o número de transistores reduziu em 6,89% e a quantidade de interconexões em 11,60%. Deste modo, o LOMGAM é mais eficiente em reduzir o número interconexões do que diminuir a quantidade de transistores no circuito, quando o parâmetro GMI (Índice de Fusão de Portas) é usado.

Tabela 6.10 – Quantidade de interconexões por GMI para NMI = 4

		GMI					
	Circuito não minimizado	3		5		10	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	34	-8,11	33	-10,81	33	-10,81
b02	26	23	-11,54	22	-15,38	22	-15,38
b03	66	59	-10,64	58	-12,12	58	-12,12
b04	362	348	-3,87	347	-4,14	347	-4,14
b05	454	420	-7,49	419	-7,41	418	-7,93
b06	34	27	-20,59	25	-26,47	24	-29,41
b07	332	286	-13,86	285	-14,16	285	-14,16
b08	84	88	4,76	87	3,57	87	3,57
b09	128	108	-15,63	107	-16,41	107	-16,41
b10	117	103	-11,97	102	-12,82	102	-12,82
b11	509	471	-7,47	461	-9,43	458	-10,02
b12	707	665	-5,94	654	-7,50	649	-8,20
b13	202	188	-6,83	186	-7,92	185	-8,42
b14	3133	2974	-5,08	2905	-7,28	2873	-8,30
s1196	469	441	-5,97	432	-7,89	426	-9,17
s1423	494	477	-3,44	475	-3,85	475	-3,85
s1488	518	490	-5,41	479	-7,53	475	-8,30
s1494	526	510	-3,04	504	-4,18	500	-4,94
s9234_1	814	760	-6,63	745	-8,48	736	-9,58
s13207	741	687	-7,29	686	-7,42	686	-7,42
s35932	8215	7212	-12,21	7069	-13,95	7069	-13,95
s15850	427	416	-2,58	415	-2,81	414	-3,04
Média			**-8,14**		**-9,75**		**-10,23**

GMI = Índice de Fusão de Portas

Tabela 6.11 – Quantidade de fusões realizadas por GMI e NMI

NMI	0			2			3			4		
GMI	3	5	10	3	5	10	3	5	10	3	5	10
Circuito	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**	**Unid.**
b01	3	3	3	10	11	11	10	11	11	10	11	11
b02	2	2	2	6	7	7	7	8	8	7	8	8
b03	5	5	5	23	24	24	23	24	24	23	24	24
b04	28	28	28	62	69	69	61	68	68	61	68	68
b05	53	53	53	86	87	87	89	90	91	90	91	92
b06	8	9	9	11	13	13	12	14	15	12	14	15
b07	47	47	47	83	83	83	84	85	85	84	85	85
b08	7	7	7	18	19	19	19	20	20	19	20	20
b09	22	22	22	39	42	43	36	39	40	33	34	34
b10	18	18	18	29	30	30	28	29	20	31	32	32
b11	75	75	75	130	138	143	123	133	138	117	126	129
b12	68	68	68	185	221	226	185	222	227	186	223	228
b13	24	24	24	47	49	50	52	54	55	52	54	55
b14	369	369	369	724	802	859	721	798	855	595	669	701
s1196	61	61	61	123	131	136	124	133	139	126	135	141
s1423	51	51	51	135	140	140	139	144	144	135	138	138
s1488	81	81	81	148	163	163	146	166	175	133	146	150
s1494	84	84	84	146	164	170	149	164	171	126	134	138
s9234_1	165	165	165	244	271	283	247	275	289	221	227	236
s13207	85	85	85	200	201	201	210	211	211	217	218	218
s35932	1337	1337	1337	2175	2318	2318	2175	2318	2318	2175	2318	2318
s15850	51	51	51	124	129	130	125	129	129	117	118	119
Média	**126**	**126**	**126**	**226**	**243**	**248**	**227**	**245**	**249**	**217,62**	**233**	**236,19**

NMI = Número Máximo de Inversores nas Entradas das Portas Aglutinadas
GMI = Índice de Fusão de Portas

A tabela 6.11 contém a quantidade de fusões realizadas para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 0, 2, 3 e 4 e valores de Índice de Fusão de Portas (GMI) iguais a 3, 5 e 10. Observando as médias de fusões realizadas, é percebido que para NMI=0 não ocorreu nenhuma variação nas quantidades médias de fusões mesmo com o aumento do valor do GMI. Porém, como para os demais valores de NMI diferentes de 0, o teorema de De Morgan pode ser usado durante o processo de fusão, é notado que a medida que o valor de GMI é aumentado mais fusões são feitas. Portanto, para NMIs diferentes de 0, quanto maior for o valor do GMI maior é a quantidade de fusões realizadas.

O tempo de execução da LOMGAM foi mensurado em um computador de 64 bits com sistema operacional Ubuntu 14.04 LTS, processador Intel Core 2 Quad CPU Q9400 e 6 gigabytes de memória RAM. A tabela 6.12 contém o tempo medido para diferentes índices de fusão de portas (GMI). Observando a tabela 6.12, é visto que o tempo de execução não aumenta exponencialmente com o aumento do GMI. Porém, o tempo de execução do LOMGAM, para o parâmetro GMI, está diretamente relacionado ao tamanho do circuito. Por exemplo, analisando a tabela 6.12 é observado que o circuito s35932 possui o maior tempo de execução e também é o circuito com o maior número de transistores entre os circuitos analisados, como é possível ob-

servar a partir da comparação entre as quantidades de transistores presentes na primeira coluna da referida tabela.

Quando as tabelas 6.7, 6.8, 6.9 e 6.10 são comparadas com o caso 3 da tabela 6.2, é possível notar que a redução na quantidade de interconexões obtidas após o desenvolvimento do método é bem menor do que o previsto no estudo de caso. Isso ocorre devido a limitação imposta pelos parâmetros de controle ao processo de fusão, o que em alguns casos, pode impedir que portas de fanout unitário sejam intercaladas. Outro fator que contribui para isso é o acréscimo de inversores para garantir a funcionalidade lógica do circuito.

6.3 Análise quantitativa para o parâmetro QMTS

O parâmetro Índice de Fusão de Portas não controla a quantidade de transistores em série nas portas complexas produzidas pelo LOMGAM, por isso foi desenvolvido o parâmetro Quantidade Máxima de Transistores em Série (QTMS). Como explicado no capítulo anterior, o QMTS é um par de valores, onde o primeiro informa a quantidade máxima de transistores em série permitida na rede composta por transistores PMOS, já o segundo valor mostra o número máximo de transistores em série permitidos na rede formada por transistores NMOS. Deste modo, o parâmetro Quantidade Máxima de Transistores em Série é definido como QMTS (PMOS, NMOS).

Assim como na análise quantitativa para o parâmetro GMI, primeiro é gerada a netlist em Verilog estruturado para os circuitos, que serão otimizados, utilizando o RTL Compiler. Depois, essas netlists foram carregadas no LOMGAM para os seguintes Números Máximos de Inversores nas Entradas das Portas Aglutinadas (NMI): 3, 4 e 7. Para cada NMI, foram usados os seguintes pares QMTS: QMTS (2,3), QMTS (3,5) e QMTS (4,4). Para cada parâmetro e netlist, o algoritmo produziu uma netlist minimizada nos formatos eqn e Spice, além de informar a quantidade de fusões e o total de transistores e de interconexões do circuito após a minimização.

As tabelas 6.13, 6.14 e 6.15 apresentam a quantidade de transistores para os pares QMTS mencionados anteriormente e para os Números Máximos de Inversores nas Entradas das Portas Aglutinadas (NMI) 3, 4 e 7. A segunda coluna de cada uma dessas tabelas contém a quantidade de transistores no circuito combinacional antes do processo de minimização e as demais colunas apresentam o número de transistores após o processo de fusão. Além disso, as referidas tabelas apresentam os percentuais de redução na quantidade de transistores em relação ao número de transistores presentes nas versões não minimizadas dos circuitos.

Tabela 6.12 – Tempo de execução da LOMGAM

	Circuito não minimizado	**GMI**								
	Número de transistores	1	2	3	4	5	6	8	10	50
Circuito	**Unid.**	**Segundos**	**Segundos**	**Segundos**	**Segundos**	**Segundos**	**Segundos.**	**Segundos**	**Segundos**	**Segundos**
b01	132	7	7	5	7	6	6	7	8	5
b02	94	7	6	8	10	8	8	12	10	7
b03	248	5	8	8	9	10	10	9	7	9
b04	1300	7	7	6	12	10	8	8	8	6
b05	1694	8	9	8	9	7	5	5	10	8
b06	122	9	7	7	9	8	6	8	7	7
b07	1196	9	7	8	9	4	5	5	5	6
b08	336	8	9	6	8	8	6	6	5	6
b09	428	6	7	6	6	7	5	6	8	11
b10	446	4	6	6	5	5	5	7	6	7
b11	1910	6	4	7	5	6	6	10	6	5
b12	2670	5	8	5	5	6	5	6	7	6
b13	700	6	5	5	6	8	4	11	8	6
b14	11698	9	10	12	9	20	19	18	20	20
s1196	1832	8	6	12	7	7	8	8	6	6
s1423	1866	7	7	7	10	9	8	8	7	8
s1488	2068	9	7	10	7	8	5	7	8	7
s1494	2108	6	5	6	7	8	8	8	7	7
s9234_1	2966	12	9	11	14	12	8	8	13	11
s13207	2762	10	10	9	14	7	8	7	7	7
s35932	29618	40	37	37	39	41	40	39	40	43
s15850	1578	10	8	9	6	8	8	10	10	8
Média		**8,86**	**9,00**	**9,43**	**10,14**	**10,14**	**8,42**	**10,52**	**10,14**	**9,81**

Observando os percentuais médios de redução na quantidade de transistores na tabela 6.13 é notado que quanto maior for a quantidade de transistores em série permitida, maior é a o percentual médio de redução no número de transistores presentes nos circuitos. Analisando a tabela 6.13 é percebido que a maior redução na quantidade de transistores ocorreu para o circuito b06 e que o menor percentual de redução na quantidade transistores ocorreu para o circuito s35932. No caso do circuito s35932 há dois fatores que contribuíram para esse resultado: a quantidade de portas de fanout unitário que podem ser aglutinadas respeitando as quantidades máximas de transistores em série definidas e a quantidade de inversores disponíveis para serem "aproveitados"pelo teorema de De Morgan, fato que evita o acréscimo de inversores ao circuito.

Tabela 6.13 – Quantidade de transistores por QMTS (PMOS, NMOS) para NMI = 3

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não minimizado	2	3	3	5	4	4
	Número de Transistores	**Número de Transistores**		**Número de Transistores**		**Número de Transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	118	-10,61	110	-16,67	110	-16,67
b02	94	78	-17,02	72	-23,40	72	-23,40
b03	248	230	-7,26	222	-10,48	218	-12,10
b04	1300	1226	-5,69	1230	-5,38	1190	-8,46
b05	1694	1502	-11,33	1470	-13,22	1460	-13,81
b06	122	92	-24,59	92	-24,59	92	-24,59
b07	1196	1004	-16,05	996	-16,72	996	-16,72
b08	336	308	-8,33	314	-6,55	308	-8,33
b09	428	350	-18,22	350	-18,22	350	-18,22
b10	446	390	-12,56	378	-15,25	378	-15,25
b11	1910	1620	-15,18	1596	-16,44	1576	-17,49
b12	2670	2462	-7,79	2442	-8,54	2522	-5,54
b13	700	618	-11,71	604	-13,71	604	-13,71
b14	11698	10530	-9,98	10410	-11,01	10280	-12,12
s35932	29618	29458	-0,54	29454	-0,55	29454	-0,55
s15850	1578	1434	-9,13	1418	-10,14	1406	-10,90
Média			**-8,86**		**-10,04**		**-10,38**

QMTS = Quantidade Máxima de Transistores em Série

Na tabela 6.14 são mostradas as quantidades de transistores para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas igual a 4. Analisando os percentuais de redução na quantidade de transistores, apresentados na tabela 6.14, é possível perceber que, novamente, o maior percentual de redução no número de transistores ocorreu no circuito b06. Assim como foi observado na tabela 6.13, o circuito s35932 apresentou o menor percentual de redução na quantidade de transistores. Porém, para os pares QMTS (3,5) e QMTS (4,4), quando

Tabela 6.14 – Quantidade de transistores por QMTS (PMOS, NMOS) para NMI = 4

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não minimizado	2	3	3	5	4	4
	Número de Transistores	**Número de Transistores**		**Número de Transistores**		**Número de Transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	114	-13,64	106	-19,70	106	-19,70
b02	94	76	-19,15	72	-23,40	72	-23,40
b03	248	230	-7,26	220	-11,29	216	-12,90
b04	1300	1226	-5369	1160	-10,77	1160	-10,77
b05	1694	1496	-11,69	1434	-15,35	1414	-16,53
b06	122	92	-24,59	88	-27,87	88	-27,87
b07	1196	1004	16,05	996	-16,72	996	-16,72
b08	336	308	-8,33	310	-7,74	298	-11,31
b09	428	350	-18,22	338	-21,03	324	-24,30
b10	446	390	-12,56	380	-14,80	386	-13,45
b11	1910	1614	-15,50	1510	-20,94	1504	-21,26
b12	2670	2452	-8,16	2206	-17,38	2198	-17,68
b13	700	618	-1,71	602	-14,00	602	-14,00
b14	11698	10510	-10,16	9906	-15,32	9964	-14,82
s35932	29618	29458	-0,54	27346	-7,67	27346	-7,67
s15850	1578	1434	-9,13	1406	-10,90	1390	-11,91
Média			**-12,02**		**-15,93**		**-16,52**

QMTS = Quantidade Máxima de Transistores em Série

o NMI é igual a 4, os percentuais de redução do número de transistores, para o circuito s35932, passou para 7,67%.

A tabela 6.15 apresenta a quantidade de transistores para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 7. Novamente, o circuito b06 apresentou o melhor resultado. Pois obteve uma média de redução na quantidade de transistores de 26,78%. O circuito s35932 continuou tendo o menor percentual de redução na quantidade de transistores, cerca de 5,29%, em média. Comparando as tabelas 6.14 e 6.15 é possível notar que os circuitos b01, b03, b06, b07, b08, b10 e s35932 presentaram os mesmos percentuais de redução para os NMIs 4 e 7. Isso significa que, para as Quantidades Máximas de Transistores em Série definidas em QMTS (2,3), QMTS (3,5) e QMTS (4,4), foi atingida a quantidade máxima de fusões possíveis. Para tentar reduzir ainda mais a quantidade de transistores, neste caso, é necessário aumentar a quantidade máxima de transistores em série permitidos na rede de transistores PMOS e na rede de transistores NMOS, porém, mantendo o NMI = 7. O Número Máximo de Inversores nas Entradas das Portas Aglutinadas igual 7 deve ser mantido, porque, comparando os dados das tabelas 6.14 e 6.15, é observado que o aumento de NMI de 4 para 7 não causou uma maior redução no número de transistores, logo, somente aumentar o NMI não vai resultar em uma maior minimização dos circuitos analisados.

Analisando os percentuais médios de redução na quantidade de transistores, apresenta-

dos nas tabelas 6.13, 6.14 e 6.15, é observado que quanto maior for a Quantidade Máxima de Transistores em Série permitida, maior é o percentual médio de redução no número de transistores. No caso do Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI), foi obtido para NMI igual a 3 uma redução média de 9,76% na quantidade de transistores; para NMI igual a 4 a redução média no número de transistores foi de 14,82% e, finalmente, para o NMI igual a 7 houve uma redução média de 15,35% na quantidade de transistores. Deste modo, é possível concluir que quanto maior for o valor de NMI, maior será o médio de redução na quantidade de transistores. Considerando os valores utilizados para os parâmetros QMTS e NMI foi obtida uma redução média total de 13,31% no número de transistores nos circuitos analisados.

As tabelas 6.16, 6.17 e 6.18 apresentam o número de interconexões para os seguintes pares de Quantidade Máxima de Transistores em Série: QMTS (2, 3), QMTS (3, 5) e QMTS (4, 4). Os seguintes Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) foram usados: 3, 4 e 7. A segunda coluna de cada uma dessas tabelas contém a quantidade de interconexões presentes no circuito combinacional antes do processo de minimização e as demais colunas apresentam o número de interconexões após o processo de fusão. Além disso, essas tabelas possuem os percentuais de redução no número de interconexões em relação ao número de interconexões contidas nas versões não minimizadas dos circuitos.

Tabela 6.15 – Quantidade de transistores por QMTS (PMOS, NMOS) para NMI = 7

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não minimizado	2	3	3	5	4	4
	Número de Transistores	**Número de Transistores**		**Número de Transistores**		**Número de Transistores**	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	114	-13,64	106	-19,70	106	-19,70
b02	94	76	-19,15	70	-25,53	70	-25,53
b03	248	230	-7,26	220	-11,29	216	-12,90
b04	1300	1226	-5,69	1444	12,00	1444	-12,00
b05	1694	1496	-11,69	1420	-16,17	1400	-17,36
b06	122	92	-24,59	88	-27,87	88	-27,87
b07	1196	1004	-16,05	996	-16,72	996	-16,72
b08	336	308	-8,33	310	-7,74	298	-11,31
b09	428	350	-18,22	326	-23,83	330	-22,90
b10	446	390	-12,56	380	-14,80	386	-13,45
b11	1910	1614	-15,50	1486	-22,20	1478	-22,62
b12	2670	2452	-8,16	2188	-18,05	2178	-18,43
b13	700	618	-11,71	594	-15,14	594	-15,14
b14	11698	10422	-10,91	9452	-19,20	9478	-18,98
s35932	29618	29458	-0,54	27346	-7,67	27346	-7,67
s15850	1578	1434	-9,43	1406	-10,90	1384	-12,29
Média			**-12,07**		**-16,80**		**-17,18**

QMTS = Quantidade Máxima de Transistores em Série

Na tabela 6.16 são exibidos os números de interconexões para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas igual 3. O circuito b06 apresentou o maior percentual de redução na quantidade de interconexões, chegando a um percentual médio de redução no número de interconexões de 44,12%. A menor redução na quantidade de interconexões ocorreu com o circuito s35932. Comparando as tabelas 6.16 e 6.13 é possível notar que os circuitos b06 e s35932 obtiveram respectivamente o maior e o menor percentual de redução na quantidade de transistores.

Tabela 6.16 – Quantidade de interconexões por QMTS (PMOS, NMOS) para NMI = 3

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não minimizado	2	3	3	5	4	4
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	30	-18,92	26	-29,73	26	-29,73
b02	26	18	-30,77	15	-42,31	15	-42,31
b03	66	57	-13,64	53	-19,70	51	-22,73
b04	362	324	-10,50	327	-9,67	306	-15,47
b05	454	358	-21,15	342	-24,67	334	-26,43
b06	34	19	-44,12	19	-44,12	19	-44,12
b07	332	236	-28,92	232	-30,12	232	-30,12
b08	84	70	-16,67	73	-13,10	70	-16,67
b09	128	89	-30,47	89	-30,47	89	-30,47
b10	117	89	-23,93	83	-29,06	83	-29,06
b11	509	364	-28,49	352	-30,84	342	-32,81
b12	707	603	-14,71	593	-16,12	633	-10,47
b13	202	161	-20,30	154	-23,76	154	-23,76
b14	3133	2549	-18,64	2489	-20,56	2423	-22,66
s35932	8215	8135	-0,97	8133	1,00	8133	1,00
s15850	427	355	-16,86	360347	-18,74	341	-20,14
Média			**-21,19**		**-24,00**		**-24,87**

QMTS = Quantidade Máxima de Transistores em Série

As reduções nas quantidades de interconexões, para o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 4, estão expostas na tabela 6.17. Assim como ocorreu para o NMI igual a 3, novamente o circuito b06 apresentou o maior percentual de redução no número de interconexões e o circuito s35932 obteve o pior resultado mais uma vez. A partir da comparação entre as tabelas 6.16 e 6.17, foi notado que o circuito b06 aumentou seu percentual de redução no número de interconexões de 44,12% para QMTS (3, 5), com NMI igual a 3, para 50,00% quando o NMI é igual a 4. No caso do circuito s35932, para o NMI igual a 3 o percentual de redução na quantidade de interconexões variou de 0,97% a 1,00%. Já para o NMI igual a 4, essa situação melhorou consideravelmente, embora tenha mantido o percentual de redução no número de interconexões de 0,97% para QMTS (2,3), para QMTS (3,5) e QMTS (4,4) a porcentagem de redução na quantidade de interconexões subiu para 13,83%. Assim

como aconteceu quando as tabelas 6.16 e 6.13, a mesma situação foi observada para as tabelas 6.17 e 6.14, ou seja, os circuitos b06 e s35932 que obtiveram o melhor e o pior percentual de redução de interconexões, respectivamente, também apresentaram a maior e a menor redução na quantidade de transistores.

A tabela 6.18 apresenta os percentuais de redução na quantidade de interconexões para Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 7. Assim como nos casos anteriormente analisados nesta seção, o b06 apresentou a maior porcentagem de redução na quantidade de interconexões, em média, essa porcentagem foi de 48,04%. O circuito s35392 continuou apresentado o menor percentual de redução no número de interconexões, seu percentual médio foi de 9,54% menos interconexões. Através da comparação das tabelas 6.18 e 6.17 é percebido que os circuitos b01, b03, b06, b07, b08, b10 e s35932 obtiveram as mesmas quantidades de interconexões após o processo de fusão para os NMIs iguais a 4 e 7. Essa mesma situação ocorreu com esses circuitos quando, nesta seção, foram comparadas as suas quantidades de transistores (tabelas 6.17 e 6.18) após a minimização realizada pelo LOMGAM. Deste modo, quando a quantidade de transistores se estabiliza, a quantidade de interconexões também é estabilizada como mostra a comparação das tabelas 6.13, 6.14, 6.15, 6.16, 6.17 e 6.18.

Tabela 6.17 – Quantidade de interconexões por QMTS (PMOS, NMOS) para NMI = 4

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não otimizado	2	3	3	5	4	4
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	28	-24,32	24	-35,14	24	-35,14
b02	26	17	-34,62	15	-42,31	15	-42,31
b03	66	57	-13,64	52	-21,21	50	-24,31
b04	362	324	-10,50	288	-20,44	288	-20,44
b05	454	355	-21,81	324	-28,63	310	-31,72
b06	34	19	-44,12	17	-50,00	17	-50,00
b07	332	236	-28,92	232	-30,12	232	-30,12
b08	84	70	-16,67	71	-15,48	65	-22,62
b09	128	89	-30,47	83	-35,16	76	-40,63
b10	117	89	-23,93	84	-28,21	87	-25,64
b11	509	361	-29,08	309	-39,21	306	-39,88
b12	707	598	-15,42	475	-32,81	471	-33,38
b13	202	161	-20,30	153	-24,26	153	-24,26
b14	3133	2539	-18,96	2189	-30,13	2263	-27,77
s35932	8215	8135	-0,97	7079	-13,83	7079	-13,83
s15850	427	355	-16,86	341	-20,14	333	-22,01
Média			**-21,91**		**-29,20**		**-30,25**

QMTS = Quantidade Máxima de Transistores em Série

Tabela 6.18 – Quantidade de interconexões por QMTS (PMOS, NMOS) para NMI = 7

		QMTS					
		PMOS	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	Circuito não minimizado	2	3	3	5	4	4
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	37	28	-24,32	24	-35,14	24	-35,14
b02	26	17	-34,62	14	-46,15	14	-46,15
b03	66	57	-13,64	52	-21,21	50	-24,24
b04	362	324	-10,50	280	-22,65	280	-22,65
b05	454	355	-21,81	317	-30,18	303	-33,26
b06	34	19	-44,14	17	-50,00	17	-50,00
b07	332	236	-28,92	232	-30,12	232	-30,12
b08	84	70	-16,67	71	-15,48	65	-22,62
b09	128	89	-30,47	77	-39,84	79	-38,28
b10	117	89	-23,93	84	-28,21	87	-25,64
b11	509	361	-29,08	297	-41,65	293	-42,44
b12	707	598	-15,42	466	-34,09	461	-34,79
b13	202	161	-20,30	149	-26,24	149	-26,24
b14	3133	2495	-20,36	2010	-35,84	2019	-35,56
s35932	8215	8135	-0,97	7079	-13,83	7079	-13,83
s15850	427	355	-16,86	341	-20,14	330	-22,72
Média			**-22,00**		**-30,67**		**-31,48**

QMTS = Quantidade Máxima de Transistores em Série

Analisando os percentuais médios de redução na quantidade de interconexões, apresentados nas tabelas 6.16, 6.17 e 6.18 é notado que quanto maior for a Quantidade Máxima de Transistores em Série permitida, maior é o percentual médio de redução no número de interconexões. Para o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 3 houve uma redução média de 23,35% na quantidade de interconexões; quando o NMI é igual a 4 ocorre uma redução média no número de interconexões de 27,12%. No caso de um NMI igual a 7 há uma redução média de 28,05% na quantidade de interconexões. Assim como ocorreu com os percentuais de redução na quantidade de transistores, as porcentagens médias na redução do número de interconexões também crescem de acordo com o aumento dos valores de NMI. Considerando os valores utilizados para os parâmetros QMTS e NMI foi alcançado uma redução média total de 26,17% na quantidade de interconexões dos circuitos que foram submetidos ao LOMGAM.

Quando se compara a porcentagem média de redução na quantidade de transistores (13,31% de redução no número de transistores) com o percentual médio de redução no número de interconexões (26,17% de redução na quantidade de interconexões), é possível concluir que o LOMGAM é mais eficiente em reduzir a quantidade de interconexões do que a quantidade de transistores, quando o parâmetro QMTS é considerado.

A tabela 6.19 contém a quantidade de fusões realizadas para um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 3, 4 e 7 e para os pares de Quanti-

dade Máxima de Transistores em Série QMTS (2, 3), QMTS (3, 5) e QMTS (4, 4). Comparando a quantidade de fusões feitas, para os Números Máximos de Inversores nas Entradas das Portas Aglutinadas (NMI) iguais a 3 e 4, é notado que praticamente todos os circuitos tiveram um aumento na quantidade de fusões com o incremento de 3 para 4 no valor do NMI. Porém quando o valor de NMI é incrementado de 4 para 7 é notado que, para os circuitos b01, b03, b06, b07, b08, b10 e s35932, o número de fusões permaneceu o mesmo. Assim como ocorreu para a quantidade de transistores e para o número de interconexões. Isso explica porque não houve alteração no número de transistores e de interconexões para os circuitos mencionados anteriormente neste parágrafo. Uma vez que não o número de fusões não aumentou quando o NMI foi modificado para 7, não há como reduzir ainda mais as quantidades de transistores e de interconexões. Deste modo, como foi dito anteriormente, nesta seção, para uma maior redução nas quantidades de interconexões e de transistores é necessário aumentar a quantidade máxima de transistores em série para ambas as redes, *Pull Up e Pull Down* e manter o valor de NMI em 7.

Analisando o número de fusões de cada circuito, na tabela 6.19, foi notado que quanto mais transistores e interconexões possuir o circuito, maior será a quantidade de fusões realizadas. A partir da tabela 6.19, é observado que o circuito s35932 possui o maior número de fusões. Entre os circuitos testados, s35932 possui originalmente a maior quantidade de transistores e de interconexões (29618 transistores e 8215 interconexões). Da mesma forma como ocorreu com as análises do número de transistores e de interconexões, foi notado que quanto maior for a quantidade de transistores em série permitida para a rede de transistores PMOS e para a rede de transistores NMOS maior será a quantidade média de fusões realizadas. Essa situação também é válida para o parâmetro NMI, pois o valor médio fusões passou de 175,53 fusões para um NMI igual a 3 para 274,21 fusões, em média, quando o NMI é igual a 7.

As figuras 6.2 e 6.3 mostram, respectivamente, os diagramas lógicos dos circuitos b02 e b06, antes e depois do processo de fusão realizado pelo LOMGAM. Neste experimento foi utilizado um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 7 e uma Quantidade Máxima de Transistores em Série (QMTS (4, 4)), ou seja, 4 transistores em série na rede de transistores PMOS e 4 transistores em série na rede de transistores NMOS. A parte a) de cada figura (figuras 6.2 e 6.3) apresenta o circuito antes do processo de fusão. Além disso, é possível notar que algumas portas lógicas da parte a) de ambas as figuras possuem uma numeração em romano. Observa-se também que essa numeração romana se repete em algumas portas. Quando os circuitos minimizados da parte b) de ambas figuras são analisados, é notado que as portas complexas criadas pelo processo de fusão também possuem a mesma numeração

romana. Porém, nesse caso, não ocorre repetição dos números. Isso acontece porque, por exemplo, as portas da parte a), de cada figura, que estão marcadas com o número *I* foram aglutinadas para formar a porta complexa marcada com o número *I* da parte b) das figuras 6.2 e 6.3. Esse processo ocorre com todas as portas que sofreram fusão para formar as portas complexas mostradas na parte b) de ambas figuras.

Quando as portas complexas da parte b) das figuras 6.2 e 6.3 são analisadas, é observado que a porta cuja saída é S0 no circuito b02 (figura 6.2) e que a porta cuja saída é S3 no circuito b06 (figura 6.3) são grandes. Isso pode levar a conclusão de que essas portas possuem mais do que 4 transistores em série em cada uma das redes de transistores. Fato que representaria uma inconformidade com o par QMTS (4, 4) escolhido para o presente experimento. Para evitar tal conclusão, as figuras 6.4 e 6.5 mostram os diagramas lógicos e de transistores, bem como a descrição Spice de cada porta, respectivamente. A parte a) de ambas as figuras mostra o diagrama lógico de cada uma das portas complexas produzidas no processo de fusão. Comparando a descrição Spice mostrada na parte b) de cada figura é notado que o dimensionamento dos transistores é igual para ambas as portas. Isso acontece porque, como foi dito no capítulo 5, o Elis não dimensiona os transistores (POSSER,). Por causa desse fato, os transistores utilizam o valor padrão da configuração do ELIS. Analisando o diagrama de transistores da parte b) da figura 6.4 é percebido que na rede *Pull Down* há 2 transistores NMOS em série e 2 transistores PMOS em série, logo, os valores do par QMTS (4, 4) são respeitados. A mesma situação é observada quando o diagrama de transistores presente na parte b) da figura 6.5 é analisado.

Tabela 6.19 – Quantidade de fusões por QMTS (PMOS, NMOS) e NMI

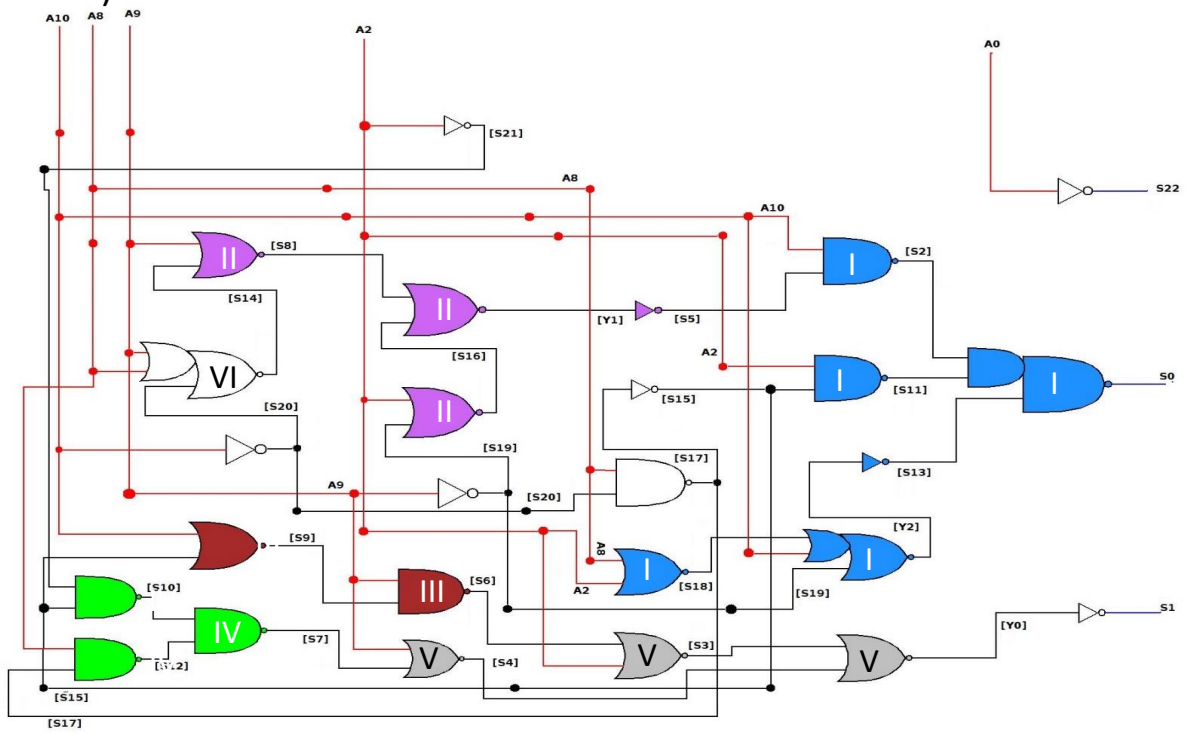
NMI	3						4						7					
QMTS	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**	**PMOS**	**NMOS**
	2	3	3	5	4	4	2	3	3	5	4	4	2	3	3	5	4	4
Circuito	**Unid.**		**Unid.**		**Unid.**		**Unid.**		**Unid.**		**Unid.**		**Unid.**		**Unid.**		**Unid.**	
b01	13		15		15		15		17		17		15		17		17	
b02	12		14		14		14		14		14		13		15		15	
b03	26		28		29		26		29		30		26		29		30	
b04	109		106		123		109		145		145		109		150		150	
b05	129		155		166		83		172		185		131		178		193	
b06	15		16		16		15		17		17		15		17		17	
b07	131		135		135		132		136		136		132		136		136	
b08	29		33		38		28		35		42		28		35		42	
b09	29		29		29		29		36		45		29		45		39	
b10	31		34		34		31		36		38		31		36		38	
b11	147		159		167		203		249		250		152		196		198	
b12	151		163		251		156		288		301		156		296		309	
b13	64		70		70		64		73		73		64		75		75	
b14	932		969		1199		941		1388		1423		985		1573		1624	
s35932	880		881		881		880		2127		2127		880		2127		2127	
s15850	131		146		150		131		162		170		131		162		168	
Média	**134,71**		**184,56**		**207,31**		**178,56**		**307,75**		**313,31**		**181,06**		**317,94**		**323,63**	

QMTS = Quantidade Máxima de Transistores em Série

NMI = Número Máximo de Inversores nas Entradas das Portas Aglutinadas

Figura 6.2 – Diagrama lógico do circuito b02 antes e depois do processo de fusão.

a) Circuito não minimizado:



b) Circuito minimizado:

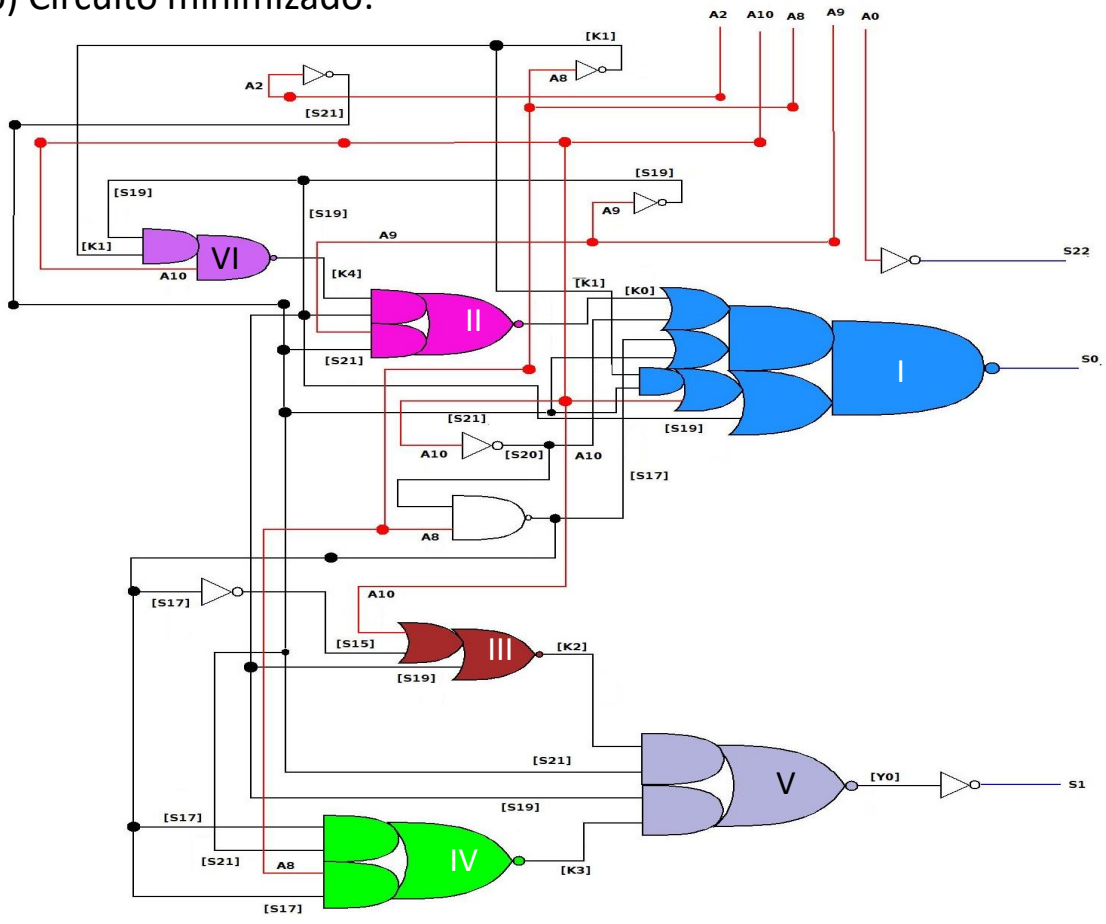
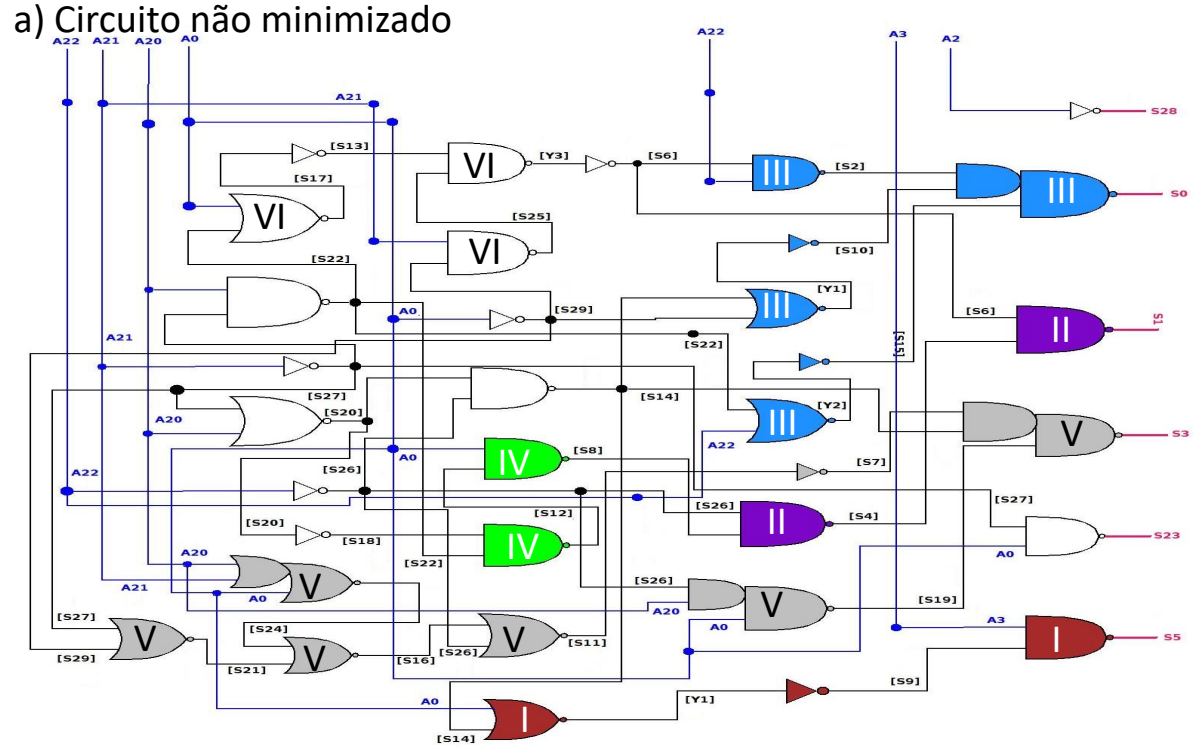


Figura 6.3 – Diagrama lógico do circuito b06 antes e depois do processo de fusão.

a) Circuito não minimizado



b) Circuito minimizado

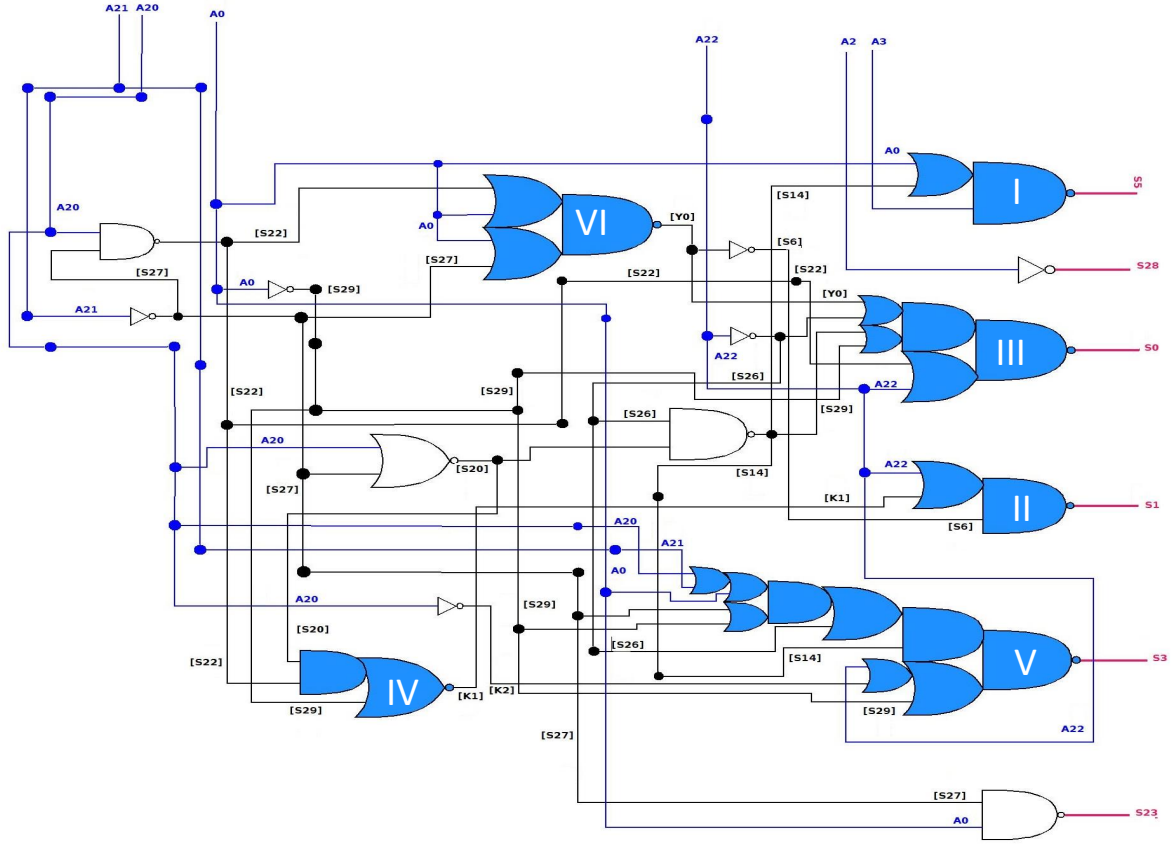
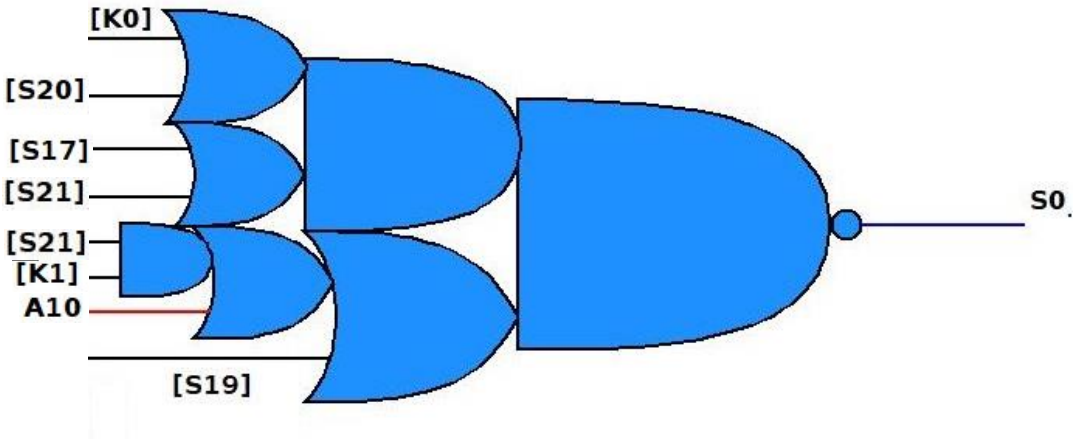


Figura 6.4 – Diagrama de transistores da porta complexa S0 do circuito b02

a) Porta complexa produzida pelo LOMGAM para o circuito b02.



b) Descrição spice e diagrama de transistores da porta S0.

```
.subckt S0 a b c d e f g out vcc gnd
MP00 NP00 f vcc vcc MODP W=3.6u L=0.3u
MP01 NP01 e NP00 vcc MODP W=3.6u L=0.3u
MP02 NP01 d NP00 vcc MODP W=3.6u L=0.3u
MP03 out g NP01 vcc MODP W=3.6u L=0.3u
MP04 NP02 b vcc vcc MODP W=3.6u L=0.3u
MP05 out a NP02 vcc MODP W=3.6u L=0.3u
MP06 NP03 d vcc vcc MODP W=3.6u L=0.3u
MP07 out c NP03 vcc MODP W=3.6u L=0.3u
MN00 NN00 c gnd gnd MODN W=2.4u L=0.3u
MN01 NN00 d gnd gnd MODN W=2.4u L=0.3u
MN02 NN01 a NN00 gnd MODN W=2.4u L=0.3u
MN03 NN01 b NN00 gnd MODN W=2.4u L=0.3u
MN04 out g NN01 gnd MODN W=2.4u L=0.3u
MN05 NN02 d NN01 gnd MODN W=2.4u L=0.3u
MN06 out e NN02 gnd MODN W=2.4u L=0.3u
MN07 out f NN01 gnd MODN W=2.4u L=0.3u
.ends S0

X1 [K0] [S20] [S17] [S21] [K1] A10 [S19]
 S0 vcc gnd S0
```

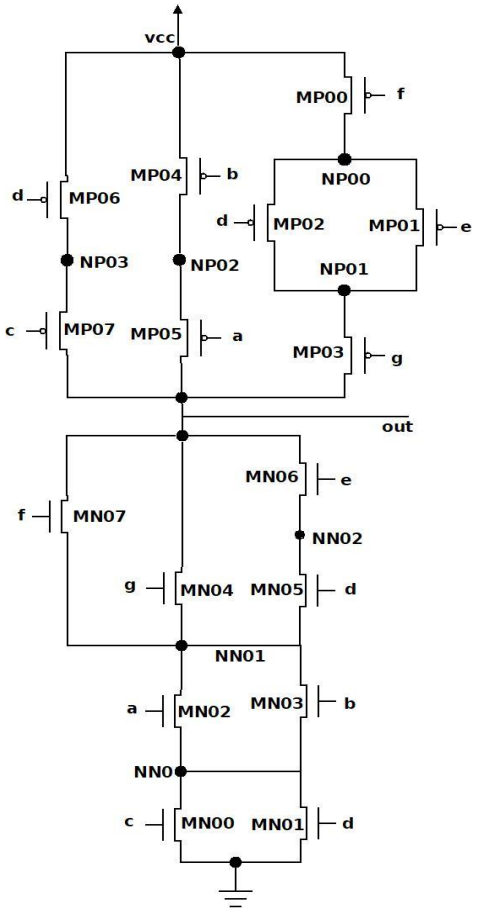
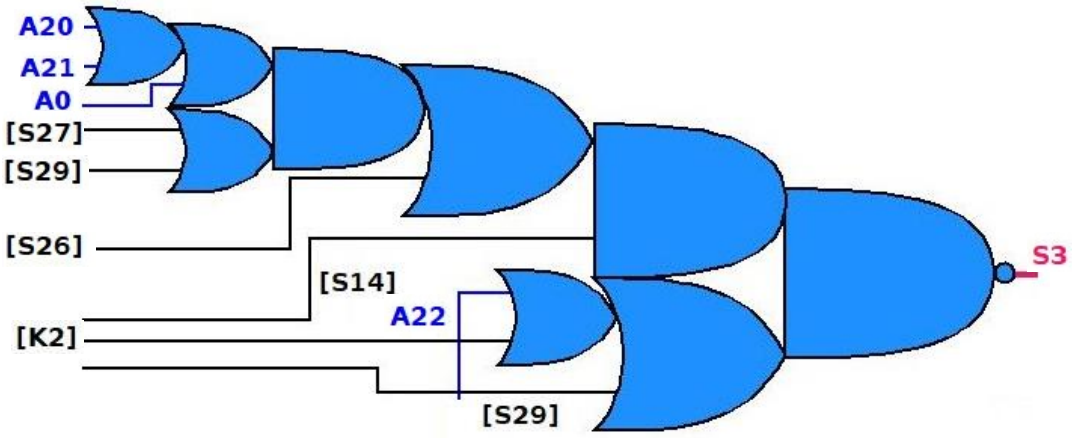


Figura 6.5 – Diagrama de transistores da porta complexa S3 do circuito b06

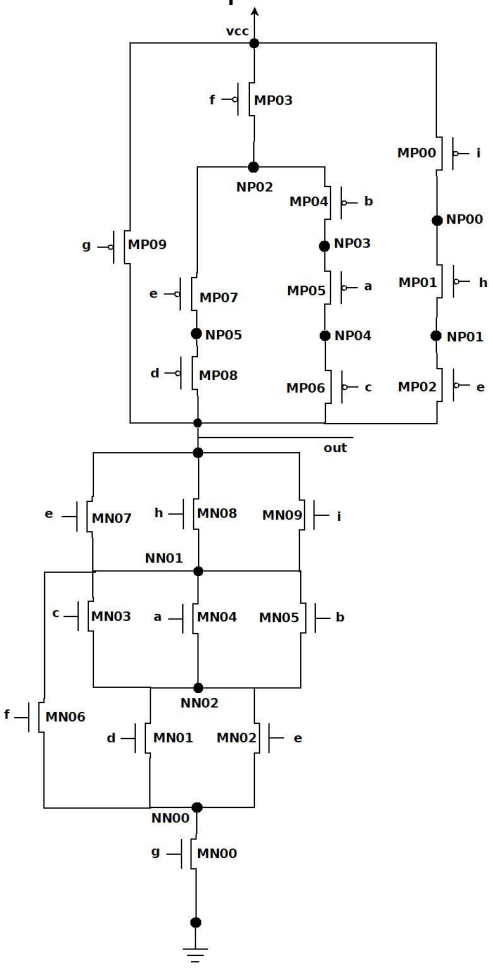
a) Porta complexa produzida pelo LOMGAM para o circuito b06.



b) Descrição spice e diagrama de transistores da porta S3.

```
.subckt S3 a b c d e f g h i out vcc gnd
MP00 NP00 i vcc vcc MODP W=3.6u L=0.3u
MP01 NP01 h NP00 vcc MODP W=3.6u L=0.3u
MP02 out e NP01 vcc MODP W=3.6u L=0.3u
MP03 NP02 f vcc vcc MODP W=3.6u L=0.3u
MP04 NP03 b NP02 vcc MODP W=3.6u L=0.3u
MP05 NP04 a NP03 vcc MODP W=3.6u L=0.3u
MP06 out c NP04 vcc MODP W=3.6u L=0.3u
MP07 NP05 e NP02 vcc MODP W=3.6u L=0.3u
MP08 out d NP05 vcc MODP W=3.6u L=0.3u
MP09 out g vcc vcc MODP W=3.6u L=0.3u
MN00 NN00 g gnd gnd MODN W=2.4u L=0.3u
MN01 NN02 d NN00 gnd MODN W=2.4u L=0.3u
MN02 NN02 e NN00 gnd MODN W=2.4u L=0.3u
MN03 NN01 c NN02 gnd MODN W=2.4u L=0.3u
MN04 NN01 a NN02 gnd MODN W=2.4u L=0.3u
MN05 NN01 b NN02 gnd MODN W=2.4u L=0.3u
MN06 NN01 f NN00 gnd MODN W=2.4u L=0.3u
MN07 out e NN01 gnd MODN W=2.4u L=0.3u
MN08 out h NN01 gnd MODN W=2.4u L=0.3u
MN09 out i NN01 gnd MODN W=2.4u L=0.3u
.ends S3

X3 A20 A21 A0 [S27] [S29] [S26] [S14] A22
 [K2] S3 vcc gnd S3
```



6.4 Análise comparativa entre a LOMGAM e a RTL Compiler

As netlists, geradas pelo RTL Compiler, que foram usadas neste estudo comparativo, utilizaram as restrições detalhadas no apêndice A. Essas netlists puderam usar todas as portas

lógicas disponíveis da biblioteca de 45nm utilizada, exceto meio-somadores, somadores completos, multiplexadores, XOR e NXOR. Deste modo, o RTL Compiler teve total liberdade para utilizar portas complexas e assim produzir circuitos minimizados. É importante relembrar que as netlists, que foram usadas pelo LOMGAM, são diferentes, apesar de também terem sido geradas pelo RTL Compiler. Pois, as netlists usadas pelo método proposto são compostas por portas lógicas simples, ou seja, AND, OR, NOR e NAND.

Na tabela 6.20 há a quantidade de transistores nos circuitos otimizados pelo RTL Compiler (terceira coluna). A quarta coluna da tabela em questão contém o percentual de redução no número de transistores em relação a versão não minimizada do circuito para o RTL Compiler. As demais colunas apresentam os melhores resultados, em termos de número de transistores, para o LOMGAM. No caso do LOMGAM, quando é considerado o parâmetro Quantidade Máxima de Transistores em Série, o melhor resultado foi obtido para o Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 7 para par QMTS (4,4).

Comparando (ver tabela 6.20) os percentuais de redução na quantidade de transistores entre o RTL Compiler e o LOMGAM para o Índice de Fusão de Portas, é notado que, o método proposto, para um GMI igual a 10 e um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 2, conseguiu reduzir o número de transistores em todos os circuitos. Mesmo assim, em média, seu percentual de redução no número de transistores ficou, aproximadamente, 3% a menos do que o percentual de redução na quantidade de transistores obtido pelo RTL Compiler.

Observando, na tabela 6.20, apenas os percentuais da quantidade de transistores para a ferramenta RTL Compiler é possível notar que, para os circuitos b01, b03 e b10, houve um aumento na quantidade de transistores mesmo com o uso de portas complexas sendo permitido. Isso ocorre porque, provavelmente, o RTL Compiler considerou outras métricas como prioritárias, como por exemplo satisfazer restrições de tempo, ao invés de priorizar a redução no número de transistores. Assim como foi observado no estudo de caso, apresentado na primeira seção do presente capítulo, é possível concluir que o RTL Compiler, mesmo aumentando a quantidade de transistores em alguns circuitos, priorizou o uso de portas complexas quando teve oportunidade.

Voltando para a tabela 6.20, é possível observar que para o parâmetro Quantidade Máxima de Transistores em Série, o melhor resultado foi de 17,18%, em média, na redução na quantidade de transistores em série para o par QMTS (4,4) e um Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI) igual a 7. Quando os percentuais médios de redução no número de transistores entre o RTL Compiler e o LOMGAM (parâmetro QMTS)

são comparados é notado que este último conseguiu reduzir a quantidade de transistores em todos os circuitos. Para saber o real impacto que portas complexas produzidas pelo LOMGAM podem causar nos circuitos em termos de atraso, potência e área há a necessidade de realizar simulações elétricas e a síntese física desses circuitos através do uso de uma ferramenta com a capacidade de sintetizar qualquer rede de transistores como ASTRAN (ZIESEMER,), por exemplo. Infelizmente, tanto a síntese física quanto as simulações elétricas estão fora do escopo deste trabalho. Uma vez que, diferentemente do RTL Compiler, o LOMGAM não se preocupa com parâmetros como o atraso e a área do circuito.

Tabela 6.20 – Comparação em termos de número de transistores entre RTL Compiler e LOMGAM

		RTL Compiler		***LOMGAM***			
				NMI			
				7		2	
				QMTS		**GMI**	
	Circuito não minimizado			**PMOS** 4	**NMOS** 4	10	
Circuito	**Unid.**	**Unid.**	**%**	**Unid.**	**%**	**Unid.**	**%**
b01	132	144	9,09	106	-19,70	124	-6,06
b02	94	92	-2,13	70	-25,53	88	-6,38
b03	248	290	16,94	216	-12,90	232	-6,45
b04	1300	932	-28,31	1144	-12,00	1258	-3,23
b05	1694	1532	-9,56	1400	-17,36	1602	-5,43
b06	122	120	-1,64	88	-27,87	100	-18,03
b07	1196	1036	-13,38	996	-16,72	1098	-8,19
b08	336	306	-8,93	298	-11,31	342	1,79
b09	428	424	-0,93	330	-22,90	370	-13,55
b10	446	544	21,97	386	-13,45	414	-7,17
b11	1910	1362	-28,69	1478	-22,62	1764	-7,64
b12	2670	2434	-8,84	2178	-18,43	2534	-5,09
b13	700	606	-13,43	594	-15,14	670	-4,29
b14	11698	9226	-21,13	9478	-18,98	10602	-9,37
s35932	29618	23670	-20,08	27346	-7,67	27324	-7,75
s15850	1578	1382	-12,42	1384	-12,29	1498	-5,07
Média			**-10,49**		**-17,18**		**-7,51**

QMTS = Quantidade Máxima de Transistores em Série
NMI = Número Máximo de Inversores nas Entradas das Portas Aglutinadas
GMI = Índice de Fusão de Portas

7 CONCLUSÕES E TRABALHOS FUTUROS

O LOMGAM é um algoritmo de minimização lógica que visa reduzir a quantidade de transistores e de interconexões em um circuito. Para isso, o método realiza a fusão de portas lógicas com fanout unitário para produzir portas lógicas complexas. A metodologia proposta possui duas formas de controle: Índice de Fusão de Portas (GMI) e Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI); e Quantidade Máxima de Transistores em Série (QMTS) e Número Máximo de Inversores nas Entradas das Portas Aglutinadas (NMI).

Quando se compara o potencial de minimização lógica apresentado na seção 6.1 do capítulo 6 com os resultados obtidos após a implementação do método proposto (seções 6.2 e 6.3 do capítulo 6) é observado que esse potencial de minimização lógica não é atingido. Isso ocorre porque o LOMGAM possui parâmetros para controlar as fusões que podem ser feitas e também controla a quantidade de inversores que o processo de fusão pode adicionar ao circuito a cada execução.

Esses parâmetros de controle são necessários para evitar a produção de portas complexas com um grande no número de transistores em série e também para controlar a quantidade de inversores que são acrescentados ao circuito para manter a sua funcionalidade lógica. A partir dos experimentos realizados foi possível observar que os melhores percentuais de redução tanto na quantidade de transistores quanto no número de interconexões foram obtidos para o parâmetro de controle Quantidade Máxima de Transistores em Série, que, em média, reduziu em 13,31% a quantidade de transistores e em 26,17% o número de interconexões presentes nos circuitos analisados.

Diante dos resultados apresentados, conclui-se que o LOMGAM atingiu o seu principal objetivo, que era reduzir a quantidade de transistores, para os parâmetros utilizados nos testes, dentro do propósito de explorar o potencial de minimização do algoritmo apresentado. Porém foi constatado que o algoritmo aqui introduzido é mais eficiente em reduzir a quantidade de interconexões presentes nos circuitos. Como é observado quando os percentuais citados no parágrafo anterior são comparados. Assim como ocorreu com o parâmetro QMTS, para o parâmetro GMI, o método apresentado neste trabalho também foi mais eficiente em reduzir a quantidade de interconexões do que o número de transistores, uma vez que apresentou uma média de redução de 11,60% na quantidade de interconexões presentes nos circuitos.

Outro fato importante é que quando se realiza uma comparação entre o RTL Compiler e o LOMGAM, para o parâmetro QMTS, é notado que o método proposto foi mais eficiente em reduzir a quantidade de transistores do que a ferramenta comercial, em média, houve uma re-

dução na quantidade de transistores de 10,49% e de 13,31% respectivamente. Para o parâmetro Índice de Fusão de Portas (GMI), o LOMGAM mostrou-se menos eficiente. Pois, em média, conseguiu uma redução de apenas 6,98% na quantidade de transistores. Quando o seu melhor caso foi comparado com RTL Compiler (ver última seção do capítulo 6) o algoritmo proposto foi menos eficiente do que a ferramenta comercial.

Como o GMI não controla a quantidade de transistores em série, o seu uso é desaconselhado para a tecnologia CMOS tradicional. Na qual a quantidade de transistores em série influencia o comportamento elétrico de uma porta lógica. Por isso o Índice de Fusão de Portas deve ser utilizado em novas tecnologias que não sejam CMOS ou que explorem novos arranjos de transistores como o Kernel Finder (POSSANI et al., 2016) o qual realiza a minimização lógica explorando arranjos não-série-paralelo de transistores. Por isso, o parâmetro Quantidade Máxima de Transistores em Série (QMTS) é o mais adequado para a tecnologia CMOS. Felizmente, o algoritmo apresentou resultados satisfatórios em termos de redução na quantidade de transistores. O uso de portas complexas poderá trazer outros benefícios ao circuito, por exemplo, atender mais facilmente restrições de tempo. Para analisar os benefícios que as portas complexas produzidas pelo LOMGAM (usando o parâmetro Quantidade Máxima de Transistores em Série) podem trazer ao circuito é necessário realizar a síntese física independente de biblioteca de células e simulações elétricas dos circuitos modificados pelo processo de fusão.

Como trabalhos futuros, será desenvolvida uma nova versão do LOMGAM. Esta nova versão, começará lendo a descrição comportamental do circuito e fará um mapeamento da mesma para um grafo. Em seguida, usando o conceito de programação dinâmica, o grafo será dividido em subgrafos. Cada um desses subgrafos será mapeado para uma rede de transistores. Finalmente, cada uma dessas redes originará uma porta lógica complexa. Associando o método de fusão atual (processo de fusão para QMTS) ao conceito de programação dinâmica, o LOMGAM realizará mais uma tentativa de reduzir a quantidade de transistores e de interconexões. Durante esse processo as portas complexas produzidas terão seu atraso e consumo analisados. Para que, o impacto do atraso da nova porta, no caminho em que está inserida, possa ser analisado. Se o atraso do caminho aumentar o atraso do circuito, a nova porta será descartada. Isso será possível porque o LOMGAM terá uma interface com um simulador elétrico. Devido a isso um novo parâmetro será acrescentado ao programa: o Atraso Máximo Permitido (AMP). Este parâmetro servirá para o programa determinar se o impacto do atraso do caminho pode fazer o circuito ultrapassar o atraso máximo definido pelo usuário. Se o atraso do circuito ficar maior que o atraso definido pelo usuário, a nova porta complexa será descartada. Além disso, se durante o processo e fusão, a ferramenta constatar que não é possível reduzir ainda

mais a quantidade de transistores e de interconexões o processo será abortado, caso o usuário não especifique nenhum valor para o AMP. Além disso, as ferramentas que serão integradas ao LOMGAM, como por exemplo, o simulador elétrico, terão o seus códigos fontes alterados para que a passagem de parâmetros seja realizada de forma direta, sem o uso de arquivos. Pois atualmente a passagem de parâmetros de controle do LOMGAM para o ELIS ocorre por meio de arquivos. Deste modo, pretende-se reduzir ainda mais o tempo de execução da ferramenta.

REFERÊNCIAS

AKERS, S. B. Binary Decision Diagrams. **Computers, IEEE Transactions on**, Vol. C-27, n. 6, p. 509–516, 1978. ISSN 0018-9340.

ALEGRETTI, C. G. P. **Analytical Logical Effort Formulation for Local Sizing.** 2013.84f.,Tese (Doutorado em Microeletrônica) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2013. Disponível em: <http://hdl.handle.net/10183/97867>. Acesso em 15/08/2014.

AMARU, L.; GAILLARDON, P.-E.; MICHELI, G. D. Biconditional BDD: A Novel Canonical BDD for Logic Synthesis Targeting XOR-rich Circuits. **Design, Automation Test in Europe Conference Exhibition (DATE)**, Vol.C, n. 8, 2013.

AMARÚ, L.; GAILLARDON, P.; DE MICHELI, G. Majority-Inverter Graph: A Novel Data-Structure and Algorithms for Efficient Logic Optimization. **Proceedings of the 51st Annual Design Automation Conference (DAC)**, 2014.

AMARÚ, L.; GAILLARDON, P.-E.; De Micheli, G. Biconditional BDD: A Novel Canonical BDD for Logic Synthesis Targeting XOR-rich Circuits. **Design, Automation & Test in Europe Conference & Exhibition (DATE), 2013**, p. 1014–1017, 2013. ISSN 1530-1591.

AMARÚ, L.; GAILLARDON, P.-E.; DE MICHELI, G. A Sound and Complete Axiomization of Majority-n Logic. **IEEE Transactions on Computers**, 2015.

AMARÚ, L.; GAILLARDON, P.-E.; De Micheli, G. Boolean Logic Optimization in Majority-Inverter Graphs. **Proceeding of Design Automation Conference**, 2015. ISSN 0738-100X.

AMARÚ, L.; GAILLARDON, P.-E.; De Micheli, G. Majority-Inverter Graph: A New Paradigm for Logic Optimization. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, v. 35, n. 5, 2016.

AMARÚ, L.; GAILLARDON, P.-E.; MICHELI, G. D. Mixsyn: An efficient logic synthesis methodology for mixed xor-and/or dominated circuits. **Proceedings of the Asia and South Pacific Design Automation Conference, ASP-DAC**, p. 133–138, 2013.

ARORA, H.; BANEJIE, A.; JIDGE, R. R. A Novel Algoritmic Approach for logic Synthesis Engine Design. **IEEE**, 2013.

Berkeley Logic Synthesis and Verification Group. **ABC: A system for sequential synthesis and verification.** 2005. Disponível em: <http://www.eecs.berkeley.edu/~alanmi/abc/>. Acesso em 19/11/2015.

Berkeley Logic Synthesis and Verification Group. **Multi-Valued Logic Synthesis.** Disponível em: <http://embedded.eecs.berkeley.edu/Respep/Research/mvsis/mvlogic.html>. Acesso em 19/11/2015.

BRAYTON, R.; MISHCHENKO, A. ABC: An Academic Industrial-Strength Verification Tool. **Computer Aided Verification**, Springer, p. 24–40, 2010.

BRAYTON, R. K. et al. Mis: A multiple-level logic optimization system. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, IEEE, Vol. 6, n. 6, p. 1062–1081, 1987.

COHN, M.; LINDAMAN, R. Axiomatic Majority-Decision Logic. **IRE Transactions on Electronic Computers**, Vol. EC-10, n. 3, p. 530–530, 1961. ISSN 0367-9950.

CONCEIÇÃO, C. et al. A cell clustering technique to reduce transistor. **IEEE International Conference on Electronics, Circuits and Systems (ICECS)**, IEEE, 2017.

CONCEIÇÃO, C. M. de O.; POSSER, G.; REIS, R. Reducing the number of transistors with gate clustering. **VII Latin American Symposyum on Circuits and Systems (LASCAS)**, p. 163–166, 2016.

CORMEN, T. H. et al. Introduction to Algorithms. MIT PRESS, p. 1312, 2009. ISSN 15580768.

CORREIA, V.; REIS, A. Advanced technology mapping for standard-cell generators. **17th Symposium on Integrated Circuits and Systems Design (SBCCI)**, 2004.

DE MICHELI, G. Synthesis and optimization of digital circuits. McGraw-Hill, 1994.

DEVADAS, S.; GHOSH, A.; KEUZER, K. Logic synthesis. McGraw-Hill, 1994.

EBENDT, R.; FEY, G.; DRECHSLER, R. Advanced bdd optimization. Springer, 2005.

FIGUEIRÓ, T.; RIBAS, R. P.; REIS, A. AIG Rewriting Considering Multiple Objectives. **Proc. 25º South Symposium on Microeletronics**, 2010.

FIGUEIRÓ, T.; RIBAS, R. P.; REIS, A. Construtive AIG Optimization considering Input Weights. **12º International Symposium on Quality Eletronic Design (ISQED)**, 2011. ISSN 1948-3295.

GRANDHI, S. et al. Reliability Aware Logic Synthesis through Rewriting. **27th IEEE International System-on-Chip Conference (SOCC)**, p. 274–279, 2014.

ISCAS'89. **ISCAS'99 Benchmarks website.** 1989. Disponível em:<http://www.pld.ttu.ee/~maksim/benchmarks/iscas89/verilog/>. Acesso em 30/10/2014.

ITC'99. **ITC'99 Benchmarks website.** 1999. Disponível em: <http://www.cad.polito.it/downloads/tools/itc99.html>. Acesso em 25/10/2014.

KAGARIS, D. Moto-x: A multiple-output transistor-level synthesis cad tool. **IEEE Transactions on Computer-Aided Design of Integreted Circuits and Systens**, Vol. 35, n. 1, 2016. ISSN 0278-0070.

KATZ, R. H. Comtemporary logic design. Prentice Hall, 1994.

KEUTZER, K. Dagon: Technology binding and local optimization by dag matching. **24th ACM/IEEE Design Automation Conference**, p. 341–347, 1987.

KLEINBERG, J.; TARDOS, E. Algorithm design. Person Addison Wesley, 2006.

LIENIG, A. B. K. et al. VLSI Physical Design: From Graph Partitioning to Timing Closure. Springer, 2011. ISSN 0717-6163.

LOGICS. **ELIS website.** 2004. Disponível em: <http://www.inf.ufrgs.br/logics/>. Acesso em 14/10/2015.

MANES, G.; PELOSI, G. Enrico fermi's ieee milestone in florence. Firenze University Press, 2015.

MARQUES, F. d. S. **Technology mapping for virtual libraries based on cells with minimal transistor stacks.** 2008.,Tese (Doutorado em Ciência da Computação) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2008. Disponível em:<http://hdl.handle.net/10183/16130>. Acesso em 17/10/2015.

MARQUES, F. d. S. et al. Métodos e resultados de otimizações de circuitos implementados sobre o ambiente de síntese lógica elis. **Proceedings of Iberchip**, 2004.

MARQUES, F. S. et al. DAG Based Library-Free Technology Mapping. **Proceedings of the 17th ACM Great Lakes symposium on VLS (GLSVLSI)**, 2007.

MARTINS, M. G. A.; RIBAS, R. P.; REIS, A. Funtional composition: A New Paradigm for Performing Logic Synthesis. **13º International Symposium on Quality Eletronic Design (ISQED)**, 2012. ISSN 948-3295.

MATOS, J. M. et al. Deriving Reduced Transistors Count Circuits from AIGs. **27th Symposium on Integrated Circuits and Systems Design (SBCCI)**, 2014.

MATOS, J. M. A. D. **Graph-Based Algorithms for Transistor Count Minimization in VLSI Circuit EDA.** 2014.85f., Dissertação (Mestrado em Microeletrônica) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2014. Disponível em: <http://hdl.handle.net/10183/147759>. Acesso em 10/02/2015.

MISHCHENKO, A.; BRAYTON, R. Faster Logic Manipulation for Large Designs. **Proceedings of Annual Design Automation Conference (DAC)**, 2013.

MISHCHENKO, A. et al. Technical report FRAIGs: a unifying representation for logic synthesis and verification. University of California, 2005.

POSSANI, V. N. et al. Graph-based transistor network generation method for supergate design. **IEEE Transactions on Very Large Scale Integration Systems**, IEEE, Vol. 24, n. 2, p. 692–705, 2016.

POSSER, G. **Dimensionamento de Portas Lógicas Usando Programação Geométrica.** 2011.105f.,Dissertação (Mestrado em Ciência da Computação) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2011. Disponível em:<http://www.lume.ufrgs.br/handle/10183/29571>. Acesso em 20/01/2015.

RABEY, J. M.; CHANDRAKASAN, A.; NIKOLIC, B. Digital integrated circuits — a design perspective. Prentice Hall, 2003.

RANGANATHAN, K.; BALAKRISHNAN, R. A textbook of graph theory. Springer, 2012.

REIS, A. et al. Library free technology mapping. **International Conference on Very Large Scale Integration**, p. 303–314, 1997.

REIS, A. et al. Associating CMOS Transistors with BDD Arcs for Technology Mapping. **Eletrinics Letters**, Vol. 31, n. 14, 1995. ISSN 0013-5194.

REIS, A.; ROBERT, M.; REIS, R. Topological parameters for library free technology mapping. **Proceedings of XI Brazilian Symposium on Integrated Circuit Design**, p. 2–4, 1998.

REIS, R. Design automation of transistor networks, a new challenge. **2011 IEEE International Symposium of Circuits and Systems (ISCAS)**, p. 2485–2488, 2011. ISSN 0271-4302.

SCARTEZZINI, G. **Low-power design using networks of transistors.** 2014.84f.,Dissertação (Mestrado em Computação com ênfase em Microeletrônica) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2014. Disponível em: <http://hdl.handle.net/10183/127453>. Acesso em 20/06/2015.

SCHEFFER, L.; LAVAGNO, L.; MARTIN, G. EDA for IC Implementation, Circuit Design, and Process Technology. CRC Press, 2006.

SENTOVICH, E. M. et al. Technical reports sis: A system for sequential circuit synthesis. University of California, n. UCB/ERL M92/41, 1992.

SHANNON, C. E. A Symbollic Analysis of Relay and Switching Circuits. **Electrical Engineering**, Vol. 57, p. 713 – 723, 1938. ISSN 0095-9197.

SHELDON B. AKERS, J. On the Algebraic Manipulation of Majority Logic. IEEE, 1961.

SILVA, L. M. d.; BONTORIN, G.; REIS, R. Study on the Potential Simplification of a Customized Library Approach for Logical Synthesis. 2015.

SOEKEM, M. et al. Optimizing Majority-Inverter Graph with functional Hashing. **Design, Automation Test in Europe Conference Exhibition (DATE)**, 2016. ISSN 1558-1101.

STEMMER, M. A. **Extribo:um extrator hierárquico de circuitos.** 1989.,Dissertação (Mestrado em ciência da Computação) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,1989.Disponível em: <http://hdl.handle.net/10183/25181>. Acesso em 19/11/2015.

SUTHERLAND, I.; SPROULL, B.; HARRIS, D. Logical effort: Designig fast cmos circuits. Morgan Kaufmann Publishers, 1998.

WESTE, N. H. E.; HARRIS, D. M. Cmos vlsi design a circuits and systens perpective. Prentice Hall, 2011.

XUE, J.; AL-KHALILI, D.; ROZON, C. N. Technology mapping in library-free logic synthesis. **Proceedings of Microtechnologies for the New Millennium**, Vol. 5837, p. 919–928, 2005.

ZIESEMER, A. M. **Síntese Automática do Leiaute de Redes de Transistores.** 2014.125f.,Tese (Doutorado em Microeletrônica) - Instituto de Informática Universidade Federal do Rio Grande do Sul, Porto Alegre,2014. Disponível em: <http://hdl.handle.net/10183/97852>. Acesso em 10/01/2015.

APÊNDICE A — ARQUIVO DE RESTRIÇÕES UTILIZADO NA CONFIGURAÇÃO DO RTLCOMPILER PARA A SÍNTESE LÓGICA DOS CIRCUITOS

```
set sdc_version 1.0
current_design s38584_bench
# define o clock de 100MHz
create_clock -domain clock_domain period 10.0 name clock waveform 0.0 5.0 [get_port "blif_clk_net"]
# define o tempo de subida e descida do clock em 100ps
set_clock_transition rise 0.1 [get_clocks "blif_clk_net"]
set_clock_transition fall 0.1 [get_clocks "blif_clk_net"]
# modela o atraso das entradas dos módulos
set_input_delay clock clock 0.2 [all_inputs]
set_output_delay clock clock 0.2 [all_outputs]
# modela a incerteza do clock em 200ps para setup e 100ps para hold
set_clock_uncertainty 0.2 setup [ all_clocks ]
set_clock_uncertainty 0.1 hold [ all_clocks ]
# restringe a utilização de algumas células da biblioteca pela ferramenta.
set_dont_use [get_lib_cells XNOR2_X1]
set_dont_use [get_lib_cells XNOR2_X2]
set_dont_use [get_lib_cells XOR2_X1]
set_dont_use [get_lib_cells XOR2_X2]
set_dont_use [get_lib_cells FA_X1]
set_dont_use [get_lib_cells HA_X1]
set_dont_use [get_lib_cells MUX2_X1]
set_dont_use [get_lib_cells MUX2_X2]
```

APÊNDICE B — NETLIST DO CIRCUITO B02 NO FORMATO EQN

INORDER = A0 A1 A2 A3 A4 A5;

OUTORDER = S0 S1 S2 S6 S17;

$S0 = !([S3]);$

$S1 = !((([S12] + [S14]) * [S4]) * [S5]);$

$S2 = !((([S12] + A3) * [S8]) * [S5]);$

$S6 = !([S9] + A3);$

$S17 = !(A0);$

$[S3] = !(([S7] * A2) + ([S11] * [S15]));$

$[S4] = !(([S10] * [S14]) + (A3 * A2));$

$[S7] = !(([S13] + [S15]) * (A4 + A5));$

$[S8] = !(([S12] * A3) * [S18]);$

$[S10] = !([S13]);$

$[S11] = !(A4 * (A5 + A2));$

$[S18] = !(A2);$

$[S5] = !([Y0]);$

$[S9] = !([Y1]);$

$[S13] = !(A4 * A5);$

$[S14] = !(A3 + A2);$

$[S16] = !(A5);$

$[S12] = !(A4 * [S16]);$

$[S15] = !(A3);$

$[Y0] = !([S9] + [S15]);$

$[Y1] = !(A4 + [S16]);$

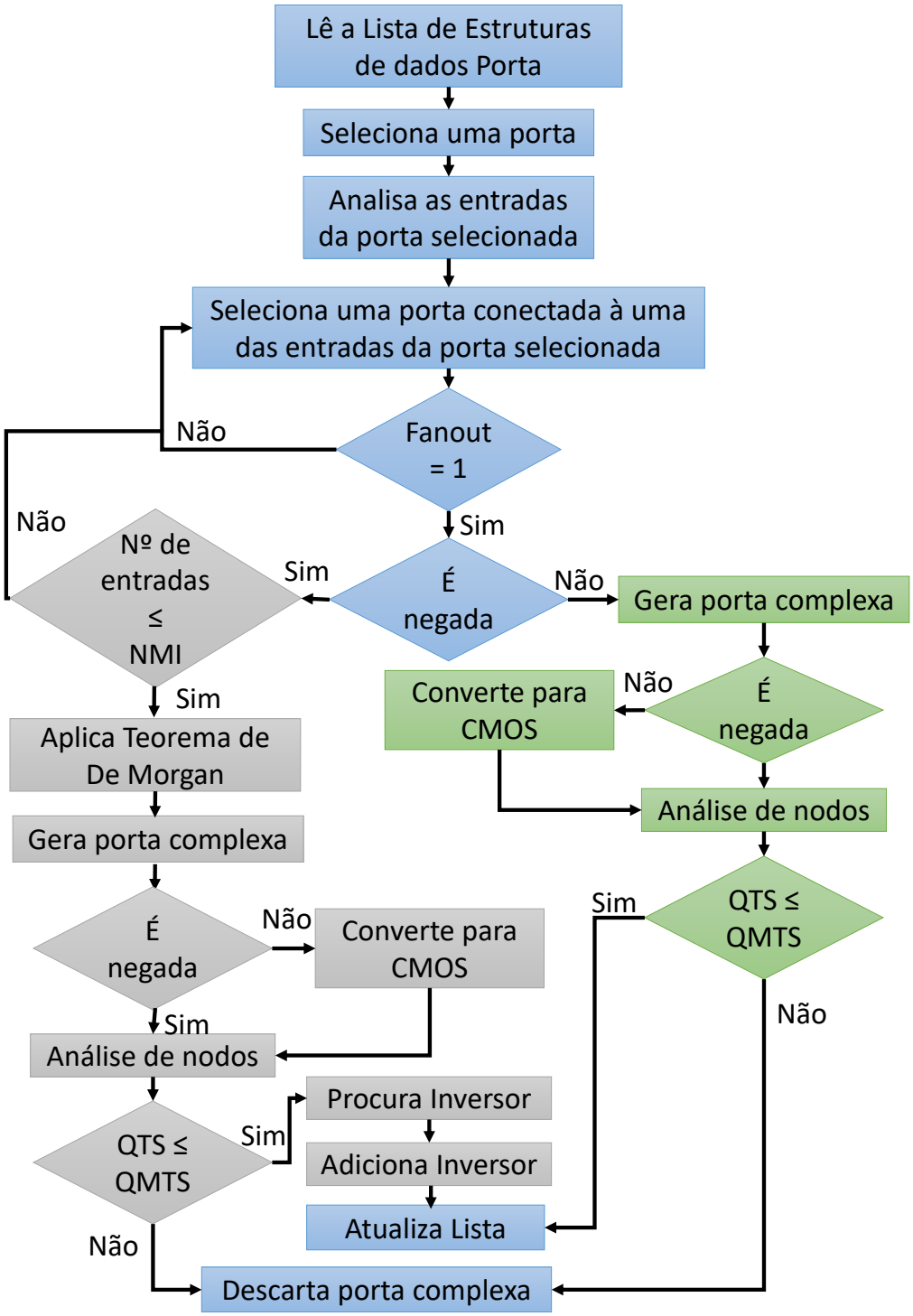
APÊNDICE C — PSEUDOCÓDIGO DO PROCESSO DE FUSÃO PARA GMI

Algoritmo 1: PROCESSO DE FUSÃO PARA GMI

```
1  início
2      para Lista = início até Lista = fim faça
3          porta = Lista_equação
4          out = Lista_saída
5          se Lista_fanout ≥ 0 então
6              se porta não for um inversor então
7                  para Lista = início até Lista = fim faça
8                      porta2 = Lista_equação2
9                      out2 = Lista_saída2
10                     se (out2 ≠ out)e(Lista_fanout2 = 1) então
11                         se (Status + Status2 + 1) ≤ GMI então
12                             se porta2 possui negação então
13                                 porta = fusão(porta2, porta)
14                                 Status = Status + Status2 + 1
15                                 Atualiza Lista
16                             fim
17                             se porta2 possui negação então
18                                 se nmero_de_entradas2 ≤ NMI então
19                                     Aplica De Morgan em porta2
20                                     porta = fusão(porta2, porta)
21                                     Status = Status + Status2 + 1
22                                     bool = Procura Inversor na Lista
23                                     se bool = falso então
24                                         Acrescenta Inversor na lista
25                                     fim
26                                     Atualiza Lista
27                                 fim
28                             fim
29                         fim
30                     fim
31                 fim
32             fim
33         fim
34     fim
35 fim
```

APÊNDICE D — FLUXOGRAMA E PSEUDOCÓDIGO DO PROCESSO DE FUSÃO PARA QUANTIDADE MÁXIMA DE TRANSISTORES EM SÉRIE (PMOS, NMOS)

Figura D.1 – Fluxograma do Processo de Fusão para QMTS(PMOS, NMOS).



Algoritmo 2: PROCESSO DE FUSÃO PARA QMTS(PMOS, NMOS)

```
1  início
2      para Lista = início até Lista = fim faça
3          porta = Lista_equação
4          out = Lista_saída
5          se Lista_fanout ≥ 0 então
6              se porta não for um inversor então
7                  se QTS(PMOS, NMOS) ≤ QMTS(PMOS, NMOS) então
8                      porta2 = Lista_equação2; out2 = Lista_saída2
9                      se porta2 não possui negação então
10                         porta = fusão(porta2, porta)
11                         se porta sem negação então
12                             Converte para CMOS
13                         fim
14                         chama ELIS
15                         Análise de nodos
16                         se QTS(PMOS, NMOS) ≤ QMTS(PMOS, NMOS)
                            então
17                             Atualiza Lista
18                         fim
19                     fim
20                     se porta2 possui negação então
21                         se nmero_de_entradas2 ≤ NMI então
22                             Aplica De Morgan em porta2
23                             porta = fusão(porta2, porta)
24                             se porta sem negação então
25                                 Converte para CMOS
26                             fim
27                             chama ELIS; Análise de nodos
28                             se
                                QTS(PMOS, NMOS) ≤ QMTS(PMOS, NMOS)
                                então
29                                 bool= Procura inversor na Lista
30                                 se bool = falso então
31                                     Acrescenta Inversor na Lista
32                                 fim
33                                 Atualiza Lista
34                             fim
35                         fim
36                     fim
37                 fim
38             fim
39         fim
40     fim
41 fim
```

APÊNDICE A — FERRAMENTAS ACADÊMICAS PARA SÍNTESE LÓGICA DISPONÍVEIS PARA DOWNLOAD

Tabela A.1 – Ferramentas acadêmicas para Síntese Lógica disponíveis para download

Ferramenta	Disponível para download	Download do código fonte	Download do executável
MIS	Não		
SIS	Sim	https://embedded.eecs.berkeley.edu/pubs/downloads/sis/index.htm	
ABC	Sim	https://bitbucket.org/alanmi/abc	http://people.eecs.berkeley.edu/∼alanmi/abc/abc.htm
TABA	Não		
ELIS	Sim		http://www.inf.ufrgs.br/logics/index.php?option=com_content&view=article&id=5:elis&catid=8&Itemid=103
VIRMA	Não		
MIXSyn	Não		
MOTO-X	Não		