

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

**Extração de Elementos Parasitas em
Circuitos CMOS Submicrônicos**

por

FÁBIO KLEIN FERREIRA

Dissertação submetida à avaliação,
como requisito parcial para a obtenção do grau de Mestre
em Ciência da Computação

Prof. Dr. Ricardo Augusto da Luz Reis
Orientador

Prof. Dr. Fernando Gehm Moraes
Co-orientador

Porto Alegre, maio de 2000

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Ferreira, Fábio Klein

Extração de Elementos Parasitas em Circuitos CMOS Submicrônicos / por Fábio Klein Ferreira. — Porto Alegre: PPGC da UFRGS, 2000.

89 f.: il.

Dissertação (mestrado) — Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2000. Orientador: Reis, Ricardo Augusto da Luz; Co-orientador: Moraes, Fernando Gehm.

1. Capacitâncias e resistências 2. Interconexões. 3. Extração Elétrica. 4. Concepção de circuitos integrados. 5. Microeletrônica. I. Reis, Ricardo Augusto da Luz. II. Moraes, Fernando Gehm. III. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitora: Profa. Wrana Panizzi

Pró-Reitor de Pós-Graduação: Prof. Franz Rainer Semmelmann

Diretor do Instituto de Informática: Prof. Philippe Olivier Alexandre Navaux

Coordenadora do PPGC: Profa. Carla Maria Dal Sasso Freitas

Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

Agradecimentos

Em primeiro lugar, gostaria de agradecer aos meus pais, Mário e Lira, por todo o apoio que me deram ao longo da minha vida, não medindo esforços para que eu chegasse até aqui.

À Adriana, que sempre esteve ao meu lado e me compreendendo, mesmo nos momentos mais difíceis.

Agradeço ao meu orientador, Ricardo Reis, pelos seus conselhos, suas palavras otimistas e pelo seu incentivo no desenvolvimento deste trabalho.

Ao Prof. Fernando Moraes, meu co-orientador, pelo seu incansável auxílio e dedicação, não poupando esforços em esclarecer as inúmeras dúvidas que apareceram durante o trabalho. Por fornecer o seu extrator EXT, que serviu de base para o desenvolvimento deste trabalho, diminuindo os meus esforços iniciais no desenvolvimento da estrutura de dados, o qual não tinha muita experiência.

Aos professores da área de Microeletrônica do PPGC, que possibilitam a formação de especialistas, à um nível de qualidade muito superior.

Ao pessoal do GME, pela sua ajuda e companherismo.

Aos funcionários da biblioteca do Instituto de Informática, dos quais não vou citar nomes, com medo de cometer faltas. Pelo apoio e soluções dos mais diversos problemas.

Ao pessoal da administração da rede, em especial, à Cristina e Margareth, que sempre se dispuseram a resolver os problemas, que não foram poucos.

Ao CNPq por financiar parte deste trabalho.

E finalmente, à Deus, pela vida, sem a qual nada seríamos.

À todos, Muito Obrigado!

Sumário

Lista de Abreviaturas	6
Lista de Figuras	8
Lista de Tabelas	9
Resumo	10
Abstract	11
1 Introdução	12
1.1 Tecnologias Submicrônicas	16
2 TROPIC - Geração Automática de Leiaute	19
2.1 EXT - Extrator de Conexões do TROPIC	21
3 Modelos de Extração	24
3.1 Modelos RC	27
3.2 Comparação entre Técnicas de Extração	31
4 Extração das Conectividades	32
4.1 Algoritmo <i>Bin-Based</i>	34
4.2 Avaliação do Algoritmo Implementado	36
4.3 Regras de Conectividades	42
4.4 Verificação das Conectividades	44
5 Extração das Capacitâncias	46
5.1 Metodologia de Extração das Capacitâncias	46
5.1.1 Algoritmo de Análise das Capacitâncias	49
5.2 Modelos de Capacitância	50
5.2.1 Capacitância Intrínseca	51
5.2.2 Capacitância de Acoplamento entre Conexões do Mesmo Nível	53
5.2.3 Capacitância de Acoplamento entre Conexões de Níveis Adjacentes	56
6 Extração de Resistências	58
6.1 Modelamento das Resistências	59
6.2 Renomeação dos nodos	60
6.3 Consideração das Capacitâncias de Acoplamento nos Modelos RC	62
7 Resultados	65
7.1 Tempo de Extração	65
7.2 Precisão da extração de circuitos	67
7.3 Taxa de distribuição dos polígonos	69
7.4 Curvas de Simulação	70
8 Conclusões	72

Anexo 1	Instruções de Uso do Programa	74
A.1	Especificação do diretório de trabalho	74
A.2	Chamada do extrator	74
A.3	Arquivos de saída	76
Anexo 2	Arquivo de Descrição da Tecnologia	79
B.1	Regras Geométricas	79
B.2	Regras CIF	81
B.3	Regras Elétricas	82
B.4	Exemplo de um Arquivo de Tecnologia	83
Bibliografia		86

Lista de Abreviaturas

SOC	System on-chip
CAD	Computer Aided Design
ATPG	Automatic Test Pattern Generator
DRC	Design Rule Check
ERC	Electric Rule Check
LVS	Layout versus Schematic
μm	micrometro ($10^{-6} m$)
CIF	Caltech Interchange Format
CMOS	Complementary MOS
MOS	Metal Oxide Semiconductor
IP	Intellectual Properties
VLSI	Very Large Scale Integration
RC	Produto Resistência e Capacitância
ASIC	Application Specific Integrated Circuit
GND	Ground (potencial de referência)
Vdd	potencial positivo
CLA	Carry lookahead adder
RIP	Ripple adder

Lista de Figuras

FIGURA 1.1 – Fluxo de projeto de síntese automática de circuitos digitais . .	15
FIGURA 1.2 – Contribuição no atraso devido as interconexões para tecnologias submicrônicas [SIA 97]	16
FIGURA 1.3 – Vista de uma interconexão submicrônica	17
FIGURA 2.1 – Fluxo de projeto para abordagem de bibliotecas virtuais	19
FIGURA 2.2 – Parâmetros necessários para o cálculo das capacitâncias parasitas	21
FIGURA 2.3 – Histograma do comprimento das interconexões para o circuito C7552	23
FIGURA 3.1 – Cruzamento de duas conexões representado através de um modelo (a) 2D (b) $2\frac{1}{2}$ D e (c) 3D [FER 99]	24
FIGURA 3.2 – Corte Vertical das Interconexões de um Circuito Integrado [DAI 97].	25
FIGURA 3.3 – Fluxo de Extração utilizando metodologia 3D	26
FIGURA 3.4 – Modelo (a) <i>L lumped</i> (b) π <i>lumped</i> (c) <i>T lumped</i>	27
FIGURA 3.5 – Modelo RC Distribuído	27
FIGURA 3.6 – Resposta a um salto de tensão [RAB 96]	29
FIGURA 3.7 – Comparação entre os modelos <i>L</i> , π e <i>T</i> [SAK 83a]. (a) conexões muito longas (b) conexões curtas	29
FIGURA 4.1 – Algoritmo <i>Scanline</i>	32
FIGURA 4.2 – Algoritmo <i>Corner-Stitching</i>	33
FIGURA 4.3 – Particionamento do circuito segundo algoritmo <i>quad-trees</i> e sua representação através da árvore binária	34
FIGURA 4.4 – Algoritmo <i>bin-based</i>	35
FIGURA 4.5 – Histograma da taxa de ocupação dos <i>bins</i> para o circuito C1908	37
FIGURA 4.6 – Histograma do número de <i>bins</i> ocupados por um polígono . .	38
FIGURA 4.7 – Circuito maior do que a grade	38
FIGURA 4.8 – Histograma da taxa de ocupação dos <i>bins</i> para o circuito C7552	39
FIGURA 4.9 – Histograma do número de <i>bins</i> ocupados por um polígono . .	40
FIGURA 4.10 – Alterações no algoritmo (a) implementação inicial (b) deslocamento do circuito (c) adaptação da grade ao tamanho do circuito .	41
FIGURA 4.11 – Histograma da taxa de ocupação dos <i>bins</i> para o circuito C7552	41
FIGURA 4.12 – Conexão indireta entre dois níveis	42
FIGURA 4.13 – Polígono definido pelas coordenadas dos cantos	43
FIGURA 4.14 – Relação entre polígonos para estarem conectados	43
FIGURA 4.15 – Exemplo para extração das conectividades	44
FIGURA 4.16 – Extração das conectividades passo-a-passo.	45
FIGURA 5.1 – Níveis $i \pm 2$ como planos terra	47
FIGURA 5.2 – Interação entre os níveis $i - 1$ e $i + 1$	47
FIGURA 5.3 – Variação na largura das conexões	48
FIGURA 5.4 – Níveis $i - 1$ e $i + 1$ considerados independentemente	48
FIGURA 5.5 – Algoritmo de análise $2\frac{1}{2}$ D das capacitâncias [CON 96]	49

FIGURA 5.6 – (a) estrutura para exemplificar o algoritmo (b) para cada lado de comprimento L (c) determinar o comprimento efetivo (L1) e o espaçamento (S1) dos vizinhos (d) analisar os dois lados (e) encontrar <i>crossovers</i> e determinar a largura deles	50
FIGURA 5.7 – Modelo de Capacitâncias	51
FIGURA 5.8 – Diferenças na espessura do óxido	52
FIGURA 5.9 – Estrutura 3D para simulação da capacitância intrínseca no FastCap	53
FIGURA 5.10 – Estrutura 3D para simulação da capacitância lateral no FastCap	55
FIGURA 5.11 – Máxima distância entre conexões vizinhos	56
FIGURA 5.12 – Estrutura para simulação da capacitância entre níveis no FastCap	57
FIGURA 6.1 – Modelamento das Interconexões	59
FIGURA 6.2 – Particionamento de uma curva para o cálculo da resistência	60
FIGURA 6.3 – Fluxo para extração de resistências	61
FIGURA 6.4 – Exemplo de renomeação de nodos	62
FIGURA 6.5 – Distribuição da capacitância de acoplamento nos modelos L e π <i>lumped</i>	63
FIGURA 6.6 – Distribuição da capacitância de acoplamento no modelo T <i>lumped</i>	63
FIGURA 6.7 – Cruzamento de duas interconexões e o modelo extraído	64
FIGURA 7.1 – Comparação do tempo de CPU, numa SUN Ultra Sparc 10	66
FIGURA 7.2 – Tempo de extração para o modelo C	67
FIGURA 7.3 – Comparação do atraso para circuitos extraídos pelo LASCA e Diva	68
FIGURA 7.4 – Comparação do atraso para circuitos extraídos pelo LASCA	69
FIGURA 7.5 – Ampliação da simulação do circuito Rip16	71
FIGURA 7.6 – Simulação dos modelos RC para o circuito Rip16	71
FIGURA A.1 – Histogramas (a) do comprimento das conexões (b) da taxa de ocupação dos <i>bins</i>	76
FIGURA B.1 – Regras relativas às camadas de conexões	79
FIGURA B.2 – Regras relativas aos contatos e vias	80
FIGURA B.3 – Regras para implementação dos transistores	81

Lista de Tabelas

TABELA 1.1 – Estimativa da evolução da tecnologia de semicondutores [SIA 97]	18
TABELA 2.1 – Distribuição do comprimento das interconexões para o circuito C7552	22
TABELA 3.1 – Pontos de interesse na resposta a um salto de tensão [RAB 96]	28
TABELA 3.2 – Comparação entre metodologias e modelos de extração [SYN 99]	31
TABELA 4.1 – Distribuição da taxa de ocupação dos <i>bins</i> para o circuito C1908	36
TABELA 4.2 – Distribuição do número de <i>bins</i> ocupados por um polígono para o circuito C1908	37
TABELA 4.3 – Distribuição da taxa de ocupação dos <i>bins</i> para o circuito C7552	39
TABELA 4.4 – Comparação do número de <i>bins</i> vazios	42
TABELA 4.5 – Comparação do tempo de CPU	42
TABELA 4.6 – Resumo das regras de conectividades implementadas internamente.	44
TABELA 5.1 – Influência do plano terra e das conexões vizinhas	47
TABELA 5.2 – Resultados comparativos para a capacitância intrínseca (fF) .	52
TABELA 5.3 – Resultados comparativos para a capacitância lateral	55
TABELA 5.4 – Resultados comparativos entre avaliações de capacitâncias entre níveis	57
TABELA 6.1 – Resultados comparativos do modelamento do circuito CLA (16bits)	64
TABELA 7.1 – Comparação do tempo de CPU para extração	65
TABELA 7.2 – Comparação do atraso de circuitos extraídos pelo LASCA e Diva	68
TABELA 7.3 – Comparação do atraso para extração RC realizada pelo LASCA	69
TABELA 7.4 – Tempo de descida para o circuito Rip16	71

Resumo

O desempenho dos circuitos integrado tem aumentado exponencialmente devido a diminuição do tamanho dos dispositivos e interconexões, juntamente com o surgimento de novas arquiteturas e metodologias de projeto. Com a diminuição do tamanho físico dos dispositivos, o atraso devido às interconexões passaram a ser equivalentes ou maiores do que o atraso gerado pelos próprios dispositivos ativos.

O aumento da importância dos elementos parasitas das interconexões na simulação de um circuito integrado, e maiores níveis de integração, demandam ao desenvolvimento de novas ferramentas.

Este trabalho apresenta uma ferramenta que realiza a extração dos elementos parasitas das interconexões de um circuito integrado, a partir da descrição geométrica do leiaute. Os elementos parasitas extraídos são as resistências das interconexões, capacitâncias ao plano terra (substrato) e as capacitâncias de acoplamento. Utilizando uma metodologia $2\frac{1}{2}D$ e equações empíricas, obtivemos um extrator rápido, mantendo uma precisão aceitável. E o usuário pode escolher o modelo (C, L, π , T *lumped*) no qual será representado as redes do circuito.

O *netlist* Spice gerado pode ser simulado por um simulador elétrico, depois que os estímulos foram adicionados, ou ser diretamente utilizado por um analisador de *timing*.

Palavras-chave: Capacitâncias, resistências, interconexões, extração elétrica, concepção de circuitos integrados, microeletrônica

TITLE: “PARASITIC EXTRACTION IN CMOS DEEP SUBMICRON IC DESIGN”

Abstract

The performance of integrated circuits has been increasing exponentially due to the device and interconnection scaling, new micro-architecture and design methodologies. With the physical feature sizes decrease, the time delay of electrical signals traveling in the interconnect is equivalent or greater than the gate for deep submicron technologies.

The emerging importance of interconnect parasitics in IC simulation, along with increasingly higher levels of system integrations, presents new challenges for IC design tool developers.

This work presents a tool that extracts an electrical description suitable for simulation from the layout of integrated circuit. The parasitics extracted are the interconnect resistances, interconnect ground capacitances (substrate) and coupling capacitances. Using a $2\frac{1}{2}$ D methodology and empiric equations, we get a fast extractor with acceptable accuracy. And the user can choose the model (C, L, π or T lumped) to represent the circuit net.

A Spice net list is generated and it can be simulated with an electric simulator, after appropriate stimuli have been added, or used by a timing analyzer.

Keywords: Capacitances, resistances, interconnection, electric extraction, integrated circuits conception, microelectronic

1 Introdução

Atualmente, o projeto de circuitos integrados complexos tornou-se impossível de ser realizado sem a ajuda de computadores e de metodologias de projeto específicas. Onde a metodologia de projeto pode ser definida como sendo uma seqüência de transformações que parte de uma descrição e/ou especificação inicial, o mais abstrata possível, para chegar a uma descrição final à nível de máscaras relativas ao processo de fabricação deste sistema no menor espaço de tempo possível.

Com o crescimento acelerado do número de transistores, necessita-se paralelamente, desenvolver ferramentas que auxiliam os projetistas. Estas ferramentas, denominadas ferramentas de CAD (*Computer Aided-Design*) para projetos de Circuitos Integrados, propiciam ao projetista um ambiente de desenvolvimento integrado, onde todo o processo de confecção do circuito pode ser monitorado adequadamente com segurança, rapidez e confiabilidade.

Estas ferramentas que automatizam as várias etapas de um projeto, podem ser divididas, de forma genérica, em [FER 98]:

- **Ferramentas de análise e verificação** - utilizadas para analisar o comportamento do circuito e verificar se a resposta está de acordo com o desejado. Exemplos: extrator lógico e elétrico, verificação formal, simulador lógico e elétrico, analisador de caminhos críticos, ferramenta de estimação de potência e outras.
- **Ferramentas de síntese e de implementação** - responsáveis pela geração e otimização do circuito e do seu respectivo leiaute. Exemplos: mapeador tecnológico, otimização lógica, decomposição lógica, síntese automática de leiaute, particionamento, posicionamento e roteamento.
- **Ferramentas de teste** - responsáveis em testar o circuito, validando sua funcionalidade e o processo de fabricação, como ATPG (*Automatic Test Pattern Generator*).

Aliado à utilização de ferramentas específicas a cada etapa do projeto, o projetista pode escolher entre as várias formas da descrição do circuito, de acordo com o nível de abstração. Desde uma descrição a nível de sistema até uma descrição à nível de máscaras do leiaute. Entre estas, temos por exemplo, a descrição comportamental, estrutural e à nível lógico.

Com isto o projetista está habilitado a realizar projetos cada vez mais complexos, entretanto é necessário o desenvolvimento contínuo de ferramentas de CAD, já que elas tornam-se obsoletas com o avanço da tecnologia.

Antes de chegar na última etapa do fluxo de projeto, a fabricação, é de vital importância conhecer o real comportamento do circuito, já que a confecção das máscaras e a própria fabricação dos circuitos integrados são processos demasiadamente custosos para que sejam realizados sem uma prévia verificação do projeto.

Além disto, os efeitos analógicos, como ruídos, flutuações do terra, *crosstalk*, causam distorções nos sinais de forma a dificultar a identificação dos níveis lógicos "0" e "1". Deste modo a integridade dos sinais tornou-se um importante fator numa análise pós-leiaute.

As ferramentas de análise e de verificação podem se situar em vários níveis de abstração. Entretanto, a verificação sempre é realizada extraindo parâmetros

característicos do projeto que está sendo executado e comparando-os com os valores desejados.

No nível de leiaute, podemos classificar os erros encontrados através destas ferramentas de verificação e análise, com sendo [OHT 86]:

- **erros de regras geométricas** - cada tecnologia tem suas próprias tolerâncias geométricas, como o mínimo espaço entre conexões, a largura mínima das conexões e outros. A violação destas regras pode levar ao decréscimo do desempenho do circuito.
- **erros topológicos ou erros lógicos** - incluem erros em conexões elétricas entre elementos do circuito e/ou elementos mal formados. Estes erros geralmente resultam em mau funcionamento. Podem ser encontrados através de ferramentas como o ERC (*electric rule check*), LVS (*layout versus schematic*) e simulação lógica.
- **erros de desempenho elétrico** - o circuito é projetado para funcionar de acordo com as características elétricas requeridas na etapa de especificação, como dissipação de potência, especificação de *timing* e outros. Este tipo de erro é causado por dimensionamento inadequado dos dispositivos e desconsideração dos efeitos parasíticos, e resulta em funcionamento lógico correto, porém com desempenho não satisfatório. Podem ser encontrados através de ferramentas como os simuladores elétricos, analisadores de *timing* e estimadores de potência.

No nível mais baixo de abstração, que antecede a etapa de fabricação do circuito, temos a descrição geométrica das máscaras do leiaute. Neste nível é necessário realizar verificações do leiaute e do funcionamento elétrico do circuito.

O DRC (*design rule check*) é a ferramenta que verifica as regras geométricas do projeto, que são limitações impostas na geometria do leiaute de um circuito integrado, de modo a garantir que o circuito possa ser fabricado corretamente. Por exemplo, uma conexão que é feita muito estreita, age como sendo um fusível e poderia se romper durante uma operação normal do circuito. Apesar de diferentes processos tecnológicos possuírem diferentes regras de projetos, as verificações destas regras podem ser descritas por uma série de operações básicas. Operações típicas são operações booleanas, operações de verificação, operações topológicas e operações geométricas.

Entretanto, para realizar a verificação do funcionamento elétrico é necessário a utilização de um extrator elétrico, que deve obter a descrição elétrica do circuito a partir do leiaute.

Quando efetuamos a extração de um leiaute de um circuito integrado, existem dois tipos de elementos elétricos que pretendemos extrair:

- **Dispositivos** - são elementos elétricos criados intencionalmente pela composição das várias camadas disponíveis (geralmente são transistores ou diodos).
- **Elementos Parasitas** - são elementos elétricos (tipicamente resistências e capacitâncias) que representam efeitos indesejáveis resultantes das interações entre as diversas camadas que constituem o circuito integrado, por estas não apresentarem características ideais. A extração destes elementos parasitas somente é necessária quando deve ser feita uma verificação detalhada dos tempos de propagação e análise a nível elétrico do desempenho do circuito integrado.

A qualidade de uma extração mede-se pela fidelidade com que são extraídos e modelados os elementos parasitas, dado que a extração dos dispositivos é mais simples.

O processo de extração consiste basicamente nas seguintes etapas:

- **Extração das Conectividades** - deve identificar as diversas camadas para caracterizar a forma como estão interligados os componentes.
- **Extração dos Dispositivos** - deve identificar todos os dispositivos existentes no leiaute.
- **Extração dos Parâmetros** - a extração dos dispositivos não é útil se não for acompanhada pela extração de suas propriedades físicas e parâmetros de funcionamento, de forma a possibilitar uma análise realista do comportamento do circuito. Exemplos típicos destes parâmetros são a largura (W) e o comprimento (L) do canal dos transistores MOS, e a área e o perímetro das regiões de fonte e dreno.
- **Extração dos Elementos Parasitas** - permite medir várias características da topologia de um circuito e a partir delas gerar elementos elétricos parasitas (resistências, capacitâncias e indutâncias) e interligá-los de forma correspondente.

Após ser feita a extração do leiaute, e consequente caracterização dos dispositivos, pode ser realizada uma comparação entre o *netlist* gerado pelo extrator e o *netlist* do circuito para verificar se são equivalentes. Com isto podemos identificar conexões erradas e dispositivos mal formados. Isto é feito pelo LVS (*layout versus schematic*).

Entretanto, erros que envolvem a funcionalidade ou a lógica do circuito são detectados somente através de simulações. Estas simulações podem considerar vários níveis de detalhes, indo desde simulações elétricas detalhadas, incluindo efeitos parasitas (resistências, capacitâncias e indutâncias), passando por níveis de chaves, nível lógico, até o nível de transferência entre registradores. A descrição adequada para cada nível pode ser obtida a partir da descrição do circuito à nível de transistores, obtida através de um extrator.

Na figura 1.1 temos um fluxo de projeto para síntese de circuitos digitais [FER 98], utilizando algumas ferramentas desenvolvidas localmente e outras comerciais. As várias etapas do fluxo podem ser resumidas em:

- Especificação do sistema
- Síntese Funcional
- Síntese Lógica
- Síntese Física
- Etapa de Verificação
- Etapa de Fabricação
- Etapa de testes e validação

Neste fluxo de projeto é indicada a ferramenta EXTRIBO [STE 89] para realizar a extração elétrica, que foi desenvolvida anteriormente no grupo de microeletrônica da UFRGS. Entretanto esta ferramenta tornou-se menos adequada para tratar dos atuais efeitos das tecnologias submicrônicas.

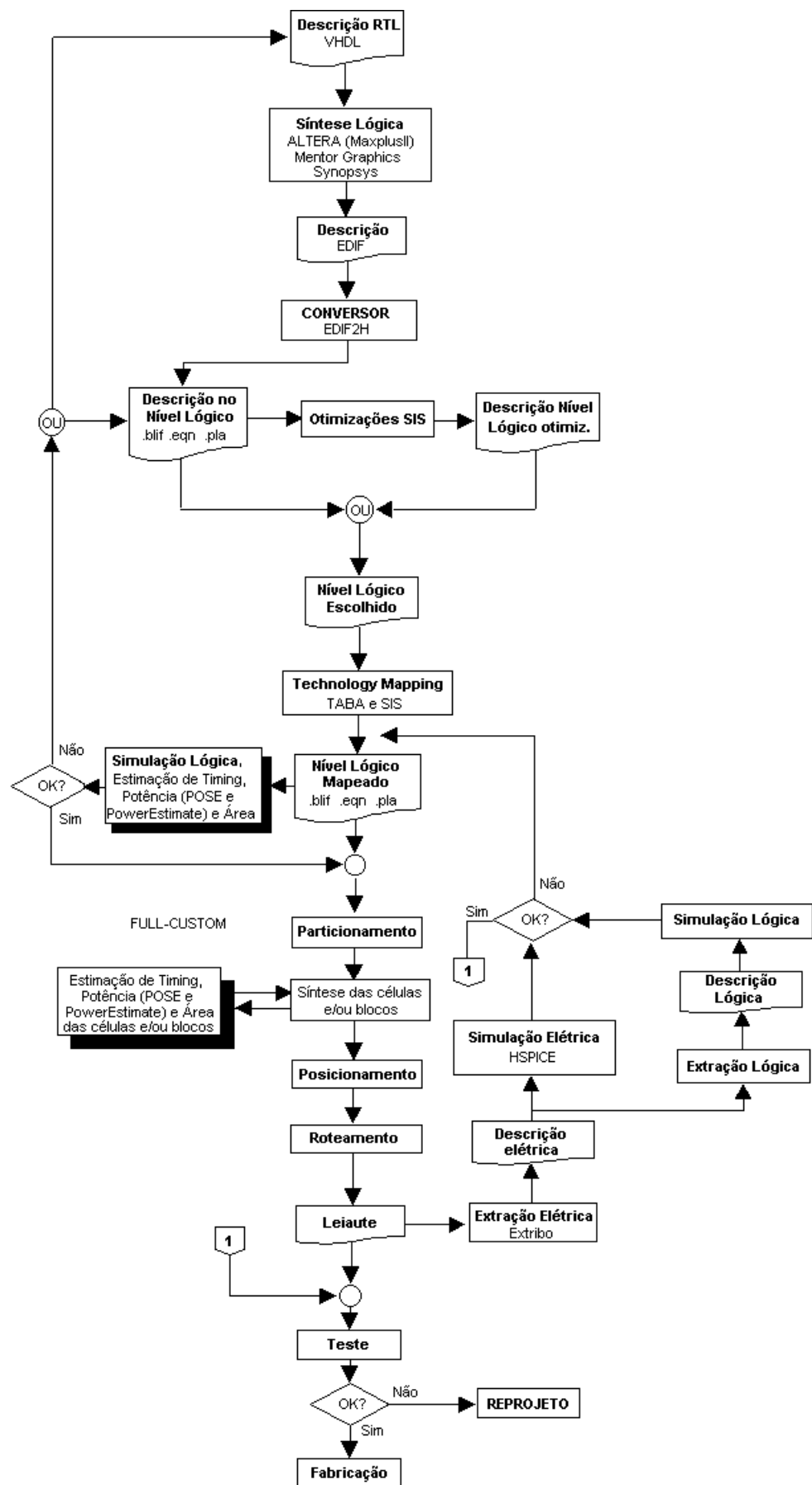


FIGURA 1.1 – Fluxo de projeto de síntese automática de circuitos digitais

1.1 Tecnologias Submicrônicas

A técnica clássica para determinar a velocidade de chaveamento dos blocos digitais é baseada na consideração de que as cargas são puramente capacitivas e concentradas num nodo (modelo *lumped*). Modelos simplificados de atraso para portas lógicas com uma carga capacitiva no nodo de saída podem facilmente determinar o comportamento dinâmico do circuito, uma vez que seja estimada a capacitância de carga.

Podemos classificar três componentes principais da carga de um porta lógica:

- Capacitância interna do transistor
- Capacitância das linhas de interconexão
- Capacitância de *gate*

Destas três componentes, a análise da capacitância das linhas de interconexão passa a ser uma função complexa de vários parâmetros do leiaute.

Cada linha de interconexão é uma estrutura tridimensional em metal e/ou polisilício, com significativas variações de forma, espessura e distância vertical ao plano terra (substrato). Além disto, cada interconexão é tipicamente cercada por um número de outras linhas, no mesmo nível ou em níveis adjacentes.

Com a diminuição do tamanho dos transistores para dimensões submicrônicas, o tamanho das interconexões passaram a contribuir de maneira significativa para o atraso do circuito. Isto pode ser visto na figura 1.2, onde nas tecnologias de $0,25\mu\text{m}$, as capacitâncias de interconexão, formadas por metalização de alumínio e dielétrico de dióxido de silício, contribuem em 50% do atraso total. Já para as tecnologias $0,18\mu\text{m}$, a contribuição é de 70%, e para tecnologias $0,15\mu\text{m}$, espera-se que a capacitância das interconexões contribuam 80% do atraso total [SIA 97].

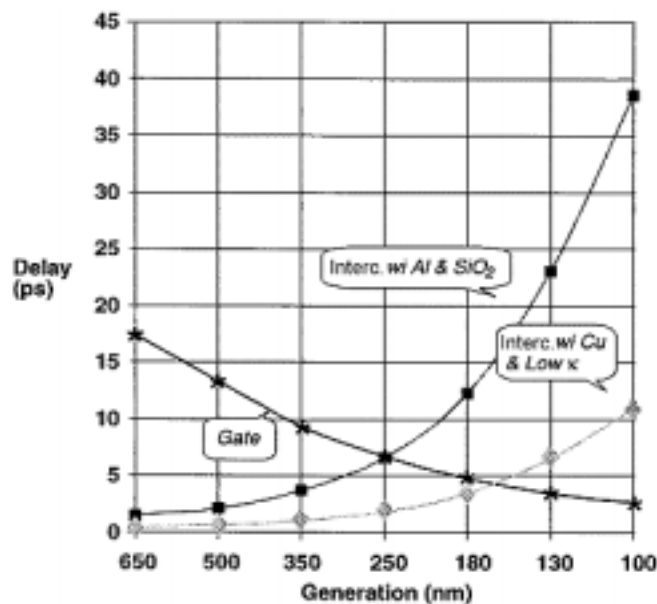


FIGURA 1.2 – Contribuição no atraso devido às interconexões para tecnologias submicrônicas [SIA 97]

Com isto, tem sido estudado a utilização de outros materiais para as interconexões, que representem uma menor contribuição no atraso do circuito. Por exemplo, a utilização do cobre juntamente com um isolante de baixa constante dielétrica, apresenta melhores resultados (figura 1.2). Para tecnologias $0,25\mu\text{m}$, a contribuição das interconexões é de apenas 25% do atraso total, chegando a 45% nas tecnologias $0,18\mu\text{m}$, e 55% nas tecnologias $0,15\mu\text{m}$.

Além disto, a crescente diminuição na largura das interconexões, fez com que a espessura se mantivesse constante ou fosse diminuída de um fator muito menor, para limitar a resistência. Então, a maior dimensão das conexões passa a ser a altura, como mostrado na figura 1.3. Para tecnologias $0,25\mu\text{m}$, a razão entre a altura e a largura é de 1,8, enquanto que espera-se que atinja 2,7 nas tecnologias $0,07\mu\text{m}$. Com isto o efeito *fringing*, que é a capacitância de perímetro da conexão ao substrato, passou a dominar as capacitâncias de interconexão, causando erros grosseiros na estimação, quando for utilizado um simples modelo aproximado de placas paralelas, que considera apenas o perímetro e a área das conexões.



FIGURA 1.3 – Vista de uma interconexão submicrônica

A quantidade de níveis de interconexão, 5-6 em 1998 e 7-8 em 2003 [SIA 97], e o fato do aumento da razão entre a espessura e a separação entre conexões, provoca um efeito que aumenta substancialmente a capacitância de acoplamento entre conexões próximas, tornando-as mais significantes do que as capacitâncias ao plano terra (substrato). Isto provoca o efeito *crosstalk*, que é a interferência no nível de um sinal que está trafegando em um conexão devido a uma outra conexão vizinha, degradando a integridade dos sinais e podendo levar ao mal funcionamento do circuito. Com tudo isto, o problema de avaliação das capacitâncias de interconexões passa a ser uma função complexa com vários parâmetros do leiaute.

Na tabela 1.1 temos uma estimativa da evolução da tecnologia nos próximos anos, onde atualmente estamos com tecnologias $0,18\mu\text{m}$ e $0,15\mu\text{m}$ disponíveis, e circuitos integrados com mais de 20 milhões de transistores, chegando a frequências na faixa de gigahertz. Nota-se o enorme comprimento na soma de todas as conexões do circuito, podendo chegar na faixa de quilômetros.

Logo, uma análise pós-leiaute é necessária para considerar os efeitos das interconexões, principalmente em processos submicrônicos. Já que nestes processos, as limitações de desempenho, como atraso e integridade dos sinais, tem sido determinadas muito mais pelos efeitos das interconexões do que pelas características dos dispositivos ativos.

Com isto, foi desenvolvido uma ferramenta que avalia as interações entre as diversas camadas de um circuito integrado, resultando na extração das capacitâncias de acoplamento, capacitâncias ao plano terra (substrato) e resistências. O extrator foi

TABELA 1.1 – Estimativa da evolução da tecnologia de semicondutores [SIA 97]

Tecnologia(nm)	250	180	150	130	100	70
Ano	1997	1999	2001	2003	2006	2009
Número de xtors	11M	21M	40M	76M	200M	520M
<i>clock</i> (MHz)	750	1200	1400	1600	2000	2500
Área(mm^2)	300	340	385	430	520	630
Máximo comprimento conexão (metros/chip)	820	1480	2160	2840	5140	10000
Níveis	6	6-7	7	7	7-8	8-9

escrito em linguagem C, sendo compilado em sistemas UNIX, estando associado a um conjunto de ferramentas para geração automática de leiaute, TROPIC [MOR 99], descrito no capítulo 2.

É utilizado como entrada do extrator, além da descrição CIF do leiaute, um arquivo contendo informações características de uma determinada tecnologia, deste modo consegue-se uma independência da tecnologia utilizada. Estas informações são descritas de forma textual, através de uma linguagem específica. Neste arquivo de tecnologia são definidos as camadas de interconexões, características elétricas de resistência e capacitância e a mínima distância em que serão desconsideradas as capacitâncias de acoplamento. A sintaxe da linguagem de descrição da tecnologia é apresentada ao longo do texto, e de forma mais detalhada no anexo 1.

O trabalho está organizado como segue. O capítulo 2 faz uma breve introdução sobre a metodologia para utilização de bibliotecas virtuais, e comenta sobre o extrator de conectividades desenvolvido para o TROPIC. O capítulo 3 apresenta as formas de modelamento das conexões, através de circuitos RC (resistências e capacitâncias). No capítulo 4 é apresentado a estrutura de dados utilizada, e uma referência a outros algoritmos existentes para operações sobre polígonos.

O cálculo dos elementos parasitas, capacitâncias e resistências, é tratado nos capítulos 5 e 6, respectivamente. E o capítulo 7 apresenta resultados comparativos entre circuitos extraídos através da ferramenta LASCA, desenvolvida em nosso trabalho, e pela ferramenta comercial Diva [CAD 98], Cadence.

O anexo 1 faz referências às opções disponíveis para o uso do programa. Enquanto que o anexo 2 apresenta uma descrição detalhada do arquivo de tecnologia e um exemplo para tecnologia CMOS 0,25 μ m.

2 TROPIC - Geração Automática de Leiaute

A utilização de bibliotecas virtuais é uma alternativa ao uso de métodos convencionais, tais como bibliotecas de células, geradores de módulos e leiautes *full-custom*.

O TROPIC [MOR 99] é um gerador de leiaute, onde a síntese do leiaute, avaliação de desempenho e otimização são associados em um único passo de projeto como representado na figura 2.1. Utiliza-se o conceito de biblioteca virtual, que se baseia no uso de células disponíveis no gerador de leiaute, ao contrário do uso de um conjunto de células pré-caracterizadas. O conjunto de células disponíveis no momento da geração é uma condição definida pelo usuário (como por exemplo, número de transistores em série e máximo *fanout*).

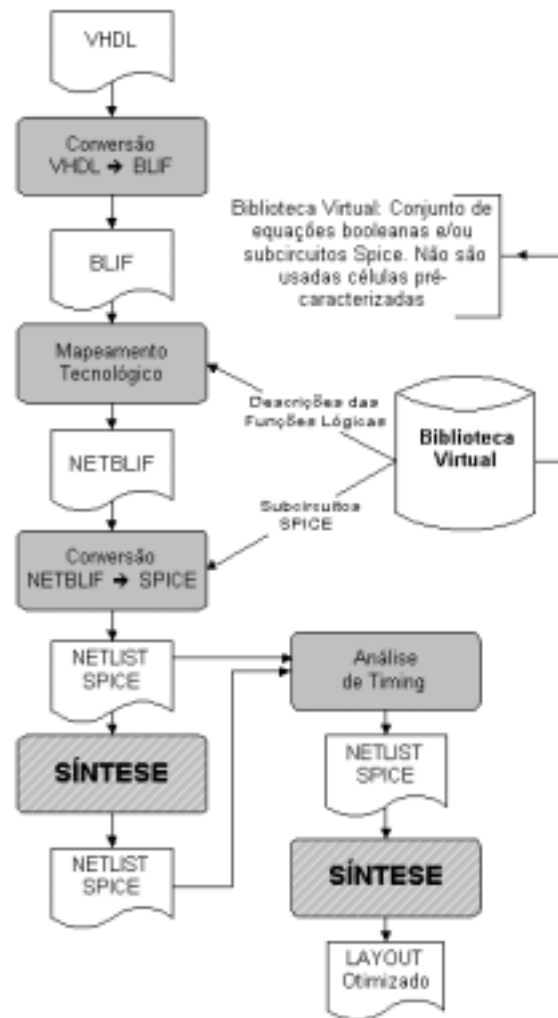


FIGURA 2.1 – Fluxo de projeto para abordagem de bibliotecas virtuais

O processo de mapeamento tecnológico tem como entrada uma descrição Booleana e o resultado é um *netlist Spice* (após a conversão de NetBlif em Spice), com o uso de portas CMOS complexas.

Este *netlist* pode ser utilizado para extrair as capacitâncias de roteamento e de difusão, conforme será descrito na seção 2.1. A partir disto pode ser feita uma análise de *timing* e um redimensionamento dos transistores. Com isto um segundo passo de síntese pode ser feito para a geração do leiaute final.

Usando uma ferramenta de síntese de layout [CAD 91], é possível quebrar as principais limitações da abordagem *standard-cell*, ou seja, um número fixo de funções e tamanho fixo dos transistores das células.

Entretanto, a abordagem de bibliotecas virtuais pode apresentar as seguintes limitações [MOR 99]:

- As células não são pré-caracterizadas. Com isto o desempenho da célula (atraso, potência e área) somente pode ser obtido após a síntese do leiaute, já que cada célula será gerada de acordo com o seu comportamento.
- A topologia das células é fixa. Geralmente, apenas portas complementares (mesmo número de transistores N e P) e *transmission gates* são implementados.
- O fluxo de projeto com síntese automática exige um número maior de passos que o necessário em projetos *standard-cells*, como os passos de compactação de layout, extração detalhada do circuito e simulação elétrica.

Por estas razões, ferramentas de síntese automática de leiaute não são ainda largamente utilizadas na indústria de semicondutores. Entretanto, a metodologia de bibliotecas virtuais tem crescido em importância, já que as ferramentas para a geração automática podem acelerar a geração de blocos IP para o processo de migração de tecnologia. E para os atuais processos submicrônicos, a importância de utilizar bibliotecas pré-caracterizadas é reduzida, porque a contribuição das interconexões no atraso total do circuito é equivalente ou maior do que o atraso devido às células.

O principal objetivo da ferramenta TROPIC é sintetizar rapidamente o leiaute inteiro de um bloco, começando de uma descrição no nível de portas lógicas, sem compactação de layout e ainda mantendo razoável densidade de transistores.

Se uma compactação de layout é empregada, mais de 3 horas são necessárias para transcrever uma descrição simbólica em uma descrição final de máscaras para um bloco de 5000 transistores, em uma Ultra-Sparc10. TROPIC pode sintetizar o leiaute e calcular as capacitâncias parasitas em menos de 2 minutos, para este exemplo.

Assim sendo, o tempo de CPU não é mais crítico, e várias iterações podem ser utilizadas para realizar otimizações no leiaute. A ferramenta de síntese lógica pode executar iterações iniciais para conseguir informações mais precisas sobre o comprimento do roteamento e a área das células. Isto irá gerar facilidades para a inserção de buffers e repetidores, pois a carga de cada nodo é calculada durante o processo de geração do leiaute.

A especificação da tecnologia é bastante simples, requerendo apenas regras básicas de distâncias, separações e extensões das camadas. Para a estimação dos elementos parasitas, a informação requerida é a capacitância de área e de perímetro de cada camada. Adicionalmente à síntese rápida, a fácil migração tecnológica permite resintetizar o circuito em novos processos, sem a necessidade de espera por novas bibliotecas de células.

A tecnologia utilizada no TROPIC é CMOS, com três níveis de metais para roteamento, sem restrições para contatos e vias empilhados. Todos os circuitos *benchmarks* utilizados durante este trabalho foram gerados através do TROPIC, para uma tecnologia 0,25 μm .

2.1 EXT - Extrator de Conexões do TROPIC

EXT é o extrator de conexões desenvolvido para o TROPIC [MOR 99] com o objetivo de prover uma ferramenta que considere apenas as capacitâncias ao substrato, e não um preciso extrator elétrico. Não foram consideradas as capacitâncias de acoplamento e resistências, sendo o modelo *lumped* adotado.

Preferiu-se calcular o comprimento das redes após o roteamento, através da leitura do arquivo de leiaute, como um extrator elétrico convencional, ao invés de calcular o comprimento durante o roteamento. Apesar desta abordagem consumir um maior tempo de CPU, ela foi adotada com o objetivo de prover ao TROPIC uma ferramenta para verificar o leiaute gerado, dando informações sobre eventuais conexões abertas ou curto-circuitos, semelhante a um ERC (*Electrical Rule Checker*).

O extrator EXT *serviu de base para o desenvolvimento deste trabalho*, onde procurou-se implementar funções de modo a torná-lo uma ferramenta mais eficiente para tratar circuitos submicrônicos. Esta subseção deve apresentar resumidamente as características do extrator EXT.

Na figura 2.2 temos a representação de um leiaute e todos os parâmetros necessários para o cálculo das capacitâncias parasitas realizado pelo extrator.

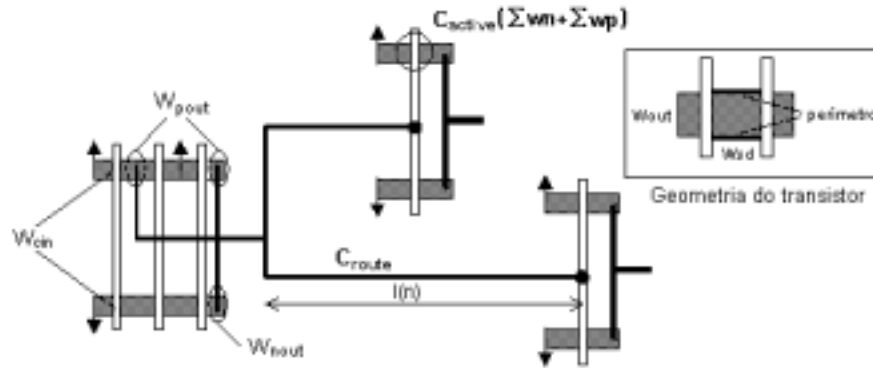


FIGURA 2.2 – Parâmetros necessários para o cálculo das capacitâncias parasitas

A capacitância parasita de cada rede, C_{load} , é dada pela seguinte fórmula

$$C_{load} = C_{active} + C_{route} + C_{diffusion}$$

onde C_{active} é função da largura de todos os *gates* conectados à respectiva rede, sendo calculado por

$$C_{active} = \left(\sum W_n + \sum W_p \right) \times l_{min} \times Cox \quad (2.1)$$

C_{route} é função do comprimento de cada camada que implementa a rede, sendo dada por

$$C_{route} = \sum_{n=poly, m1, m2, m3} \left(C_{area(n)} \times W(n) + 2 \times C_{perimetro(n)} \right) \times l(n)$$

Aqui é importante notar que são subtraídos os comprimentos dos polisilícios que estão sobre os transistores.

E $C_{diffusion}$ corresponde ao número de regiões de dreno e fonte que estão conectados a rede, sendo dado por

$$F_{diff} = \frac{(W_{nout} \cdot C_{jn} + 2 \cdot C_{jns w}) \cdot W_{sd} \cdot drain_N + (W_{pout} \cdot C_{jp} + 2 \cdot C_{jps w}) \cdot W_{sd} \cdot drain_P + W_{nout} \cdot C_{jns w} + W_{pout} \cdot C_{jps w}}{F_{diff}}$$

onde W_{sd} corresponde ao comprimento médio das regiões de dreno e source

$C_{jns w}$ e $C_{jps w}$ correspondem a capacitância de perímetro para a difusão

C_{jn} e C_{jp} correspondem a capacitância de área para a difusão

$drain_P$ e $drain_N$ é o número de regiões de dreno e fonte da rede

W_{n_cin} e W_{p_cin} corresponde a largura média do *gate* N e P, respectivamente.

drive é o número de pares de transistores que estão chaveando juntos.

E a capacitância de entrada de cada *gate* é dada por:

$$C_{in} = (W_{n_cin} + W_{p_cin}) \times l_{min} \times Cox \times drive$$

Desta forma, podemos determinar o fator de carga, como sendo a razão entre as capacitâncias de carga e de entrada (C_{load}/C_{in}). Quando este fator for superior a um limite, definido no arquivo de tecnologia, é gerado um aviso ao usuário, informando que o *driver* da rede deve ser redimensionado. Além disto, o extrator fornece os seguintes relatórios:

- *Netlist Spice* com as capacitâncias parasitas, podendo ser usado diretamente por um simulador elétrico.
- Capacitâncias parasitas:
 - Capacitâncias relativas: C_{active}/C_{in} , C_{route}/C_{in} , C_{diff}/C_{in} e C_{load}/C_{in} .
 - O valor absoluto das capacitâncias: C_{active} , C_{route} , C_{diff} , C_{load} e C_{in} .
 - O comprimento total por camada e o número de contatos em cada rede.
 - Dados topológicos de cada *gate*: número de entradas, fanin, fanout, número de transistores em série e largura média dos transistores.
 - Número estimado de *buffers* (quando C_{load}/C_{in} ultrapassar o fator limite de carga, definido no arquivo de tecnologia).
- Histograma do comprimento das interconexões. Na figura 2.3 temos um exemplo do histograma gerado para o circuito C7552 (ISCAS), e na tabela 2.1 temos a distribuição do comprimento das interconexões.

TABELA 2.1 – Distribuição do comprimento das interconexões para o circuito C7552

Tamanho das conexões	Porcentagem
0-100 μ m	80,5%
100-200 μ m	9,9%
200-500 μ m	7,4%
$\geq 500\mu$ m	2,2%

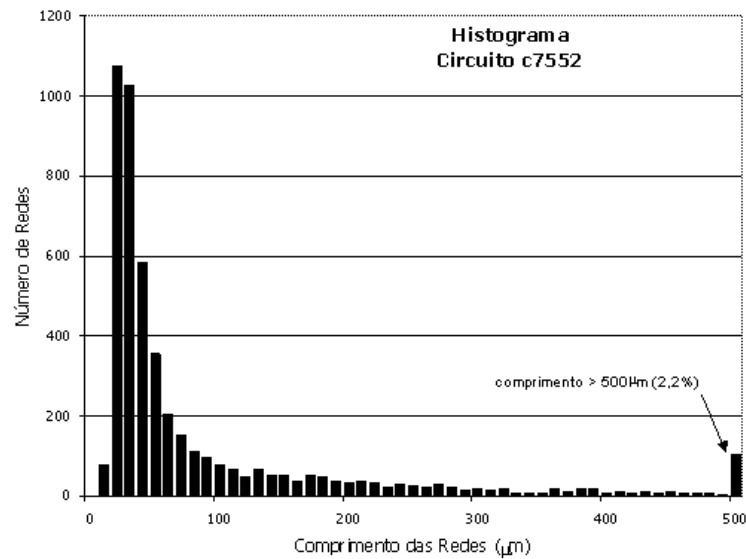


FIGURA 2.3 – Histograma do comprimento das interconexões para o circuito C7552

Este circuito C7552 tem 14376 transistores e 4764 redes. Nota-se que somente uma pequena parcela de redes possuem tamanho superior a $200\mu\text{m}$, que representa, para tecnologia $0,25\mu\text{m}$, um fator de carga igual a 2,5 (com uma largura média dos transistores igual a $2\mu\text{m}$).

Este relatório fornece importantes informações ao projetista, de modo a guiá-lo a um novo redimensionamento de algumas células do circuito.

3 Modelos de Extração

Em circuitos integrados CMOS VLSI, o atraso das conexões torna-se um importante fator na determinação do atraso total do sistema. Particularmente, o correto dimensionamento das linhas de *clock* ou barramentos são essenciais para a velocidade de operação do chip.

Para determinar o atraso de uma interconexão, o primeiro passo é estimar a capacitância e a resistência da interconexão. A precisão das estimativas está diretamente relacionada a metodologia empregada na extração.

O segundo passo é calcular a resposta do sistema. Aqui a questão de modelamento das interconexões é vital para estabelecer a viabilidade computacional no cálculo do atraso. Já que o aumento na complexidade das interconexões cria um grande número de vias e cruzamentos de conexões, fazendo com que um apropriado modelamento RC seja gerado por inúmeros elementos.

Baseado no grau de simplificação feita, o problema de extração de elementos parasitas pode ser dividida em três metodologias: 2D, quase 3D ou $2\frac{1}{2}$ D e 3D .

A extração bidimensional assume que as geometrias que estão sendo modeladas são uniformes em uma direção, geralmente a direção de propagação do sinal. Com isto os efeitos tridimensionais são ignorados. Por exemplo, um simples cruzamento de duas conexões será modelada como três estruturas bidimensionais (figura 3.1a). Onde uma das seções representa a área de sobreposição e as outras duas seções representam as áreas de não sobreposição.

Este método simplificado divide a estrutura original em um conjunto de seções 2D, mas gera resultados imprecisos porque não considera nenhum efeito tridimensional [DAI 97].

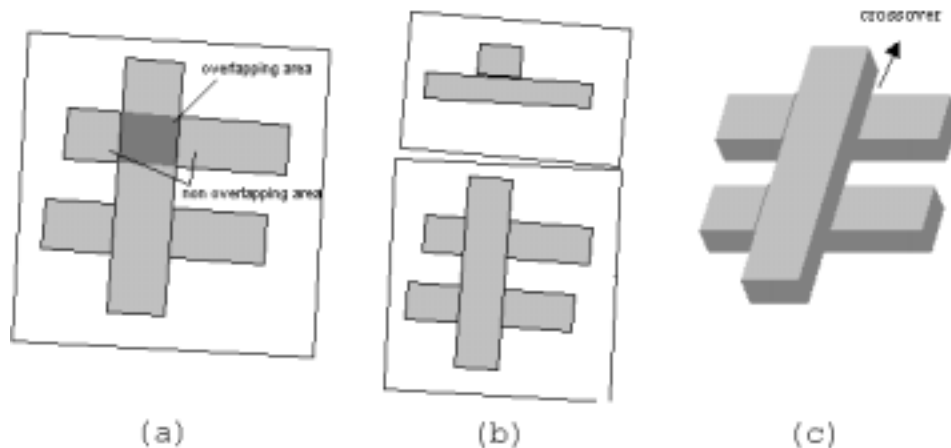


FIGURA 3.1 – Cruzamento de duas conexões representado através de um modelo (a) 2D (b) $2\frac{1}{2}$ D e (c) 3D [FER 99]

O modelo quase 3D ou $2\frac{1}{2}$ D representa qualquer estrutura por dois planos perpendiculares (figura 3.1b), e faz uso de sobreposição de geometrias para modelar mais precisamente os efeitos tridimensionais.

Somando-se os resultados e subtraindo a área de sobreposição, a extração $2\frac{1}{2}$ D pode modelar mais precisamente as estruturas tridimensionais, entretanto devido a divisão em

dois planos, ainda não modela todos os efeitos.

A extração 3D modela qualquer estrutura em três dimensões (figura 3.1c), através de um análise de elementos finitos das redes ou das condições de contorno. Enquanto que a análise por elementos finitos é uma tecnologia originalmente desenvolvida para objetos mecânicos, onde o objeto é decomposto em uma rede de elementos conectados, a aplicação das condições de contorno de todos os elementos da rede permite a solução de equações das propriedades dos materiais.

Em resumo, a extração 2D é mais rápida, mas menos precisa, a extração 3D é mais lenta mas mais precisa, enquanto que a extração $2\frac{1}{2}$ D encontra um equilíbrio entre precisão e a velocidade.

O aumento da importância das interconexões faz com que uma análise 3D seja crítica para o sucesso de um projeto [WAL 2000]. Entretanto, mesmo acurados extratores podem incorrer em penalidades de desempenho quando secções irregulares realísticas são introduzidas [DEN 97]. A figura 3.2 mostra um corte vertical de um circuito integrado. Pode-se notar a completa irregularidade na geometria do último nível de metal. Estes aspectos de processo apresentam um importante efeito nos valores dos elementos parasitas [WRI 93].

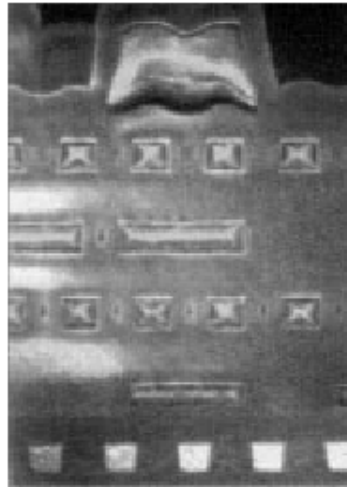


FIGURA 3.2 – Corte Vertical das Interconexões de um Circuito Integrado [DAI 97].

Torna-se proibitivamente custoso extrair todas as redes de um circuito em 3D, e de fato não é necessário [DAI 97], porque temos um grande número de redes que não são críticas para a determinação da velocidade máxima de operação do circuito.

Neste caso a combinação de metodologias pode ser eficiente. A figura 3.3 mostra um fluxo típico de extração através de uma metodologia 3D, proposto em [DAI 97].

Primeiramente, um simples extrator 2D ou $2\frac{1}{2}$ D pode ser utilizado para extrair capacitâncias com o modelo *lumped* de todas as redes. Com esta informação, pode-se identificar redes críticas através de três fatores:

- Comprimento da conexão
- Condições de carga
- Dimensionamento da porta (capacidade de *drive*)

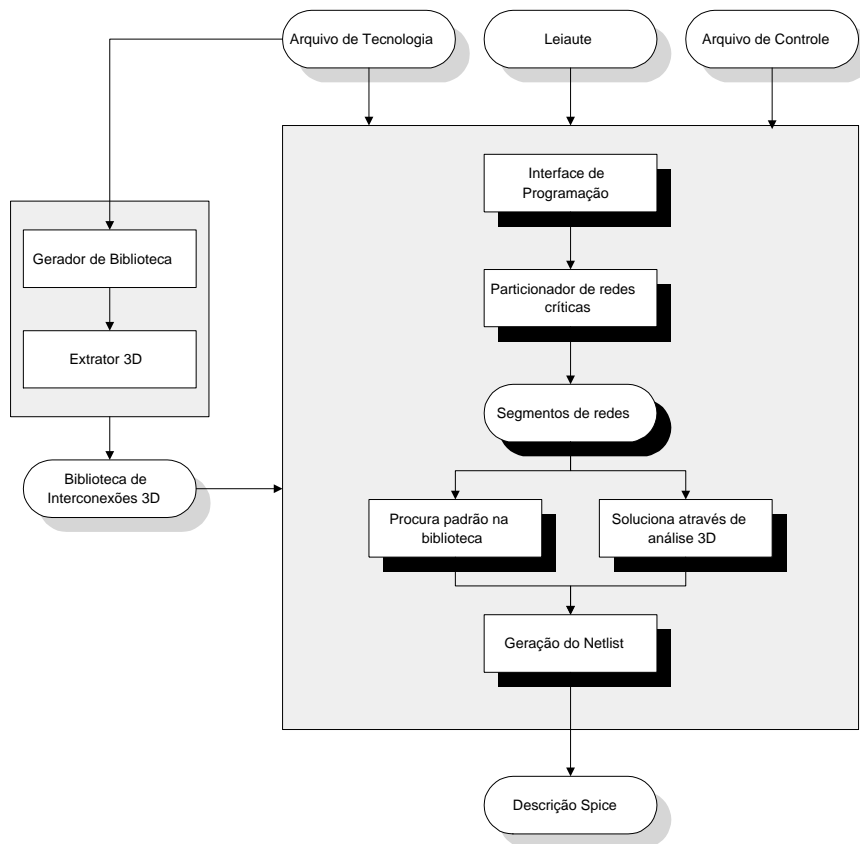


FIGURA 3.3 – Fluxo de Extração utilizando metodologia 3D

Erros de 20% a 40% são admitidos nesta fase. A seguir realiza-se a extração das redes críticas, através de um modelamento RC distribuído. Baseado nas restrições de *timing* e margens aceitáveis de *crosstalk*, pode-se determinar quais redes necessitam ser extraídas através de uma análise 3D.

Estas redes podem ser tratadas de duas maneiras. A primeira, seria tratar a rede crítica inteira e seus vizinhos como sendo uma imensa estrutura 3D e tentar modelar todas as redes simultaneamente. A outra técnica, seria decompor a rede crítica e seus vizinhos em secções menores.

A primeira técnica consegue um grau mais alto de precisão em termos de valores de capacitâncias e resistências da rede crítica. Entretanto, redes críticas são geralmente redes globais, como por exemplo barramentos e redes de *clock*. Deste modo, o problema se torna imenso, sendo computacionalmente proibitivo.

A decomposição da rede crítica e seus vizinhos em secções menores pode destruir a precisão caso não seja realizado um particionamento considerando as condições de contorno de todas as pequenas secções.

Ocorre que haverá muitas secções iguais ou similares, como por exemplo, *crossovers* com a mesma topologia mas com espaçamentos diferentes. Para solucionar este problema, é utilizada uma biblioteca de segmentos de conexões pré-caracterizados, com possíveis geometrias resultantes do particionamento.

Deste modo, a etapa crítica desloca-se da fase de extração para a fase de pré-caracterização.

3.1 Modelos RC

Existem muitas opções para modelar o comportamento de uma conexão. A primeira seria reunir a resistência total da rede numa única resistência e similarmente combinar a capacitância total num único capacitor. Este modelo simples é conhecido por "modelo L *lumped*" (figura 3.4a).

Outra opção seria dividir a capacitância total da rede de forma simétrica, e adicioná-la ao modelo entre os terminais da resistência total. Isto é conhecido como "modelo π *lumped*" (figura 3.4b). Da mesma forma, podemos dividir a resistência total da rede, formando o "modelo T *lumped*" (figura 3.4c).

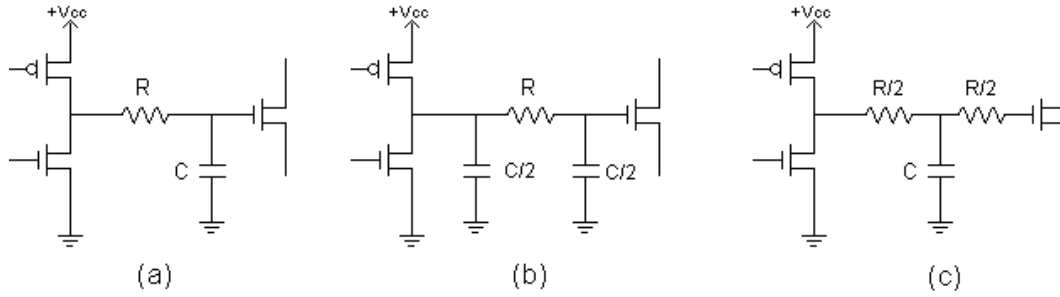


FIGURA 3.4 – Modelo (a) L *lumped* (b) π *lumped* (c) T *lumped*

Como demonstrado em [SAK 83a], ambos modelos π e T são acurados, não tendo vantagens particulares na adoção de um dos modelos. Entretanto o modelo L *lumped* é impreciso e gera resultados pessimistas, quando modela longas conexões [MAN 96][SAK 83a].

Um modelo que represente melhor o comportamento de uma interconexão pode ser conseguido quando a subdividimos em segmentos e substituímos cada segmento por um modelo π ou T. Este modelo é conhecido como RC Distribuído (figura 3.5).

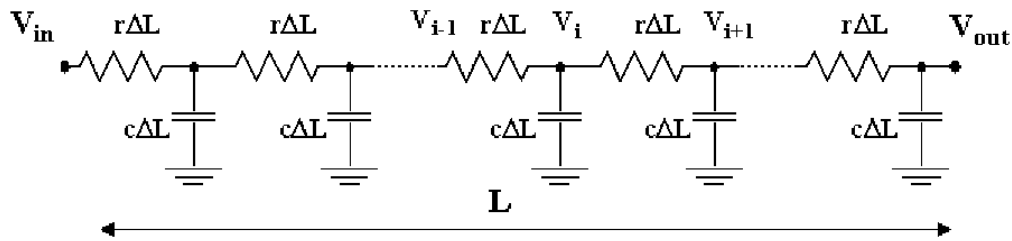


FIGURA 3.5 – Modelo RC Distribuído

Um modelamento mais acurado precisa considerar pequenas redes, ou segmentos como linhas de transmissão, especialmente em altas frequências. Isto é conseguido com o modelo RC Distribuído.

Geralmente uma complexa rede RC é construída de forma a modelar detalhadamente os efeitos das resistências distribuídas e das capacitâncias distribuídas de uma interconexão. Então são aplicadas técnicas para redução da rede, de forma a

obter uma rede final com um número mínimo de elementos e nodos, mas representando aproximadamente o mesmo comportamento. Para garantir equivalência entre a rede inicial e final, todos os métodos preservam o valor da constante de tempo de Elmore(τ) [ELM 48], que para a figura 3.5 é dada por:

$$\tau_N = \sum_{i=1}^N R_i \sum_{j=1}^N C_j$$

Na prática, existe alguns problemas relacionados com modelamento distribuído [GEN 92]:

1. pode ser difícil encontrar segmentos que são corretamente modelados através de uma secção π ou T, devido às irregularidades dos polígonos que formam a conexão, e os efeitos resistivos e capacitivos distribuídos desuniformemente.
2. o número de secções que são requeridos para obter um acurado modelamento pode ser grande para permitir uma eficiente verificação do comportamento do circuito

Entretanto, a tendência da indústria é uma transição da adoção de modelos RC *lumped* para modelos RC distribuídos. Isto pode ser visto nas citações a seguir:

“Nós estamos movendo-nos em direção a produzir um cálculo efetivo da capacitância da rede, preferivelmente a uma capacitância *lumped* baseada na carga total e no comprimento das conexões”, Richard F. Paul, gerente de metodologia de projeto de ASIC's da IBM Microeletronics (Essex Junction, VT), citado em [MAN 96].

“Para assuntos relacionados à integridade de sinais, uma simples capacitância e modelos RC não são suficientemente precisos”, Jon Powell, gerente de produtos da Quad Design (Camarillo, CA), citado em [MAN 96].

Em altas frequências, a resposta de um modelo *lumped* é dada por apenas um pólo simples, enquanto que o modelo de linha de transmissão apresenta uma resposta mais complexa.

Além disto, o modelo RC *lumped* não é suficientemente preciso nas altas frequências, mesmo com uma extração precisa do comprimento da rede. O principal problema é que a resposta à altas frequências é diferente entre um modelo *lumped* e um modelo distribuído, porque a capacitância efetiva é diferente da capacitância total. Entretanto, um modelamento *lumped* gera um menor número de nodos no circuito extraído.

O modelo RC *lumped* de uma longa conexão resulta numa estimacão pessimista do tempo de propagação. Alguns importantes pontos de referência na resposta a um salto de tensão são tabulados em tabela 3.1.

TABELA 3.1 – Pontos de interesse na resposta a um salto de tensão [RAB 96]

	Variacão de tensão	RC <i>lumped</i>	RC Distribuído
tempo de propagação	0 \rightarrow 50%	0,69RC	0,38RC
constante de tempo RC	0 \rightarrow 63%	RC	0,5RC
tempo de subida	10% \rightarrow 90%	2,2RC	0,9RC

Por exemplo, o atraso de propagação, definido como 50% do valor final, num modelo RC *lumped* é igual a 0,69RC, enquanto que um modelo distribuído tem um atraso de 0,38RC, sendo R e C a resistência e a capacitância total da conexão.

Na figura 3.6 é comparado a resposta a um salto de tensão para uma rede modelada como RC *lumped* e RC distribuído [RAB 96].

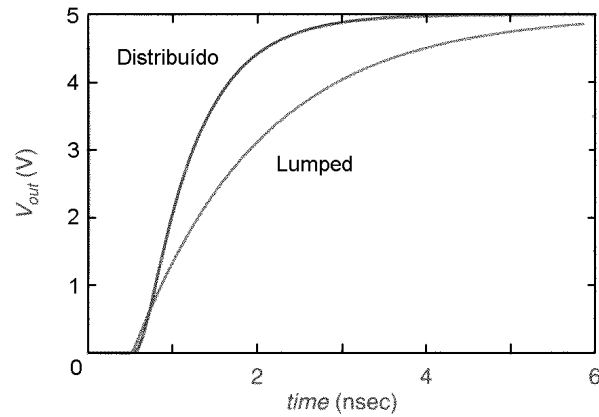


FIGURA 3.6 – Resposta a um salto de tensão [RAB 96]

O gráfico da figura 3.7 representa o erro relativo do mínimo pólo da função de transferência de um circuito *lumped* em relação a um circuito RC distribuído.

Sabendo que os pólos da função de transferência estão completamente separados, então o pólo mínimo pode ser considerado como a constante de tempo do circuito. Desde que o atraso é inversamente proporcional a constante de tempo, então o erro relativo do pólo mínimo é quase igual ao erro relativo do atraso [SAK 83a].

Na figura 3.7a é considerado o caso em que a resistência e a capacitância da conexão (R_{rot} e C_{rot}) são muito maiores que a resistência da porta MOS e a capacitância de carga (R_{drive} e C_{drive}). Este é o caso de conexões muito longas.

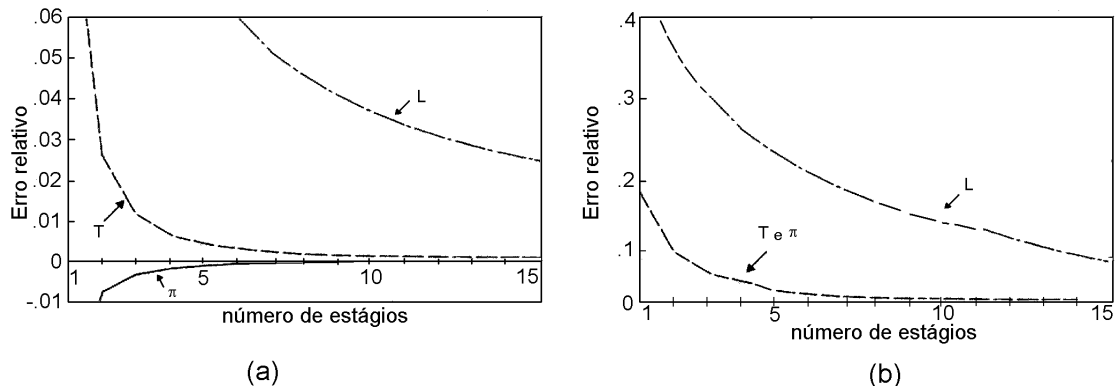


FIGURA 3.7 – Comparação entre os modelos L, π e T [SAK 83a]. (a) conexões muito longas (b) conexões curtas

Analisando este gráfico pode ser visto que a aproximação por um circuito L *lumped* apresenta um erro de cerca de 30% [SAK 83a], mesmo quando são utilizados três estágios em série, enquanto que os modelos π e T são satisfatórios.

Também deve ser notado que a precisão dos modelos é melhorada com o aumento do número de estágios L, π ou T em série. Mas é saturado em aproximadamente três ou quatro estágios, quando usado um modelo π ou T.

Na figura 3.7b temos o caso em que $R_{rot} \gg R_{drive}$ e $C_{rot} \approx C_{drive}$, ou $R_{rot} \approx R_{drive}$ e $C_{rot} \gg C_{drive}$, representando as conexões curtas do circuito.

Neste caso, a resposta torna-se mais lenta e as aproximações são melhores. Isto pode ser facilmente entendido, já que a medida que os valores dos elementos parasitas devido as interconexões diminuem, é natural que a parcela de atraso devido a eles também diminua, não havendo muita diferença entre os modelos. Entretanto, o modelo L é mais deficiente que os modelos π e T.

O erro com três estágios π ou T é menor que 3%, mas não é prático empregar este circuito para todas as interconexões de um circuito. Visando um menor custo computacional, o uso de circuitos *lumped* mais simples, dentro de uma tolerância de erro, é recomendado.

Em [SAK 83a] são apresentadas duas tabelas, com tolerância de erros de 10% e 3%, para a escolha do circuito *lumped* adequado. São empregados dois fatores para a determinação do circuito:

- a relação entre a capacitância de carga e a capacitância total da rede e
- a relação entre a resistência do transistor MOS do *drive* e a resistência total da rede.

A resistência do transistor MOS tende a levar em consideração as condições do dimensionamento da porta. O valor desta resistência está entre (1/máxima condutância do dreno) e (1/mínima condutância do dreno). Sabendo-se que as aproximações se tornam melhores quando a relação entre resistência do transistor e resistência da conexão aumenta, é seguro adotar (1/máxima condutância do dreno).

As ferramentas comerciais diferem no uso dos modelos. Por exemplo, o extrator xCalibre, da Mentor Graphics [MEN 99], utiliza o modelo T devido a três razões [PON 91]:

- a constante de tempo de Elmore [ELM 48] pode ser calculada quando conectados modelos T formando uma árvore RC.
- a redução do *netlist* é simplificada
- sendo um modelo para linhas de transmissão, os efeitos distribuídos podem ser precisamente representados, e indutores podem ser facilmente incluídos no modelo.

Já no extrator Dracula, da Cadence [CAD 98], é utilizado o modelo π . Como vantagem, ele cria um nodo a menos que o modelo T. Considerando que posteriormente à extração, o circuito será simulado, e que o tempo de simulação é proporcional ao número de nodos do circuito, a redução de nodos torna-se um problema relevante.

A escolha da metodologia e do modelamento das redes RC é dependente do projeto, e deve considerar o compromisso entre velocidade e precisão. No LASCA foi utilizado uma metodologia $2\frac{1}{2}D$, baseada em formulações empíricas, para cálculo dos elementos parasitas.

Também foram implementados os modelos L, π e T *lumped* de um estágio, de forma que o usuário possa escolher entre os modelos que representarão todas as redes do seu circuito.

3.2 Comparação entre Técnicas de Extração

Na tabela 3.2 é realizada uma comparação entre a utilização de metodologias para a extração e a forma de modelamento das redes RC apresentadas, levando em consideração fatores como precisão e velocidade.

TABELA 3.2 – Comparação entre metodologias e modelos de extração [SYN 99]

Habilidade	Descrição	Precisão	Velocidade	Uso
Extraí CI inteiro <i>lumped C</i> e <i>lumped R</i> formulação empírica	Utilização de equações $2\frac{1}{2}D$ para obter valores de cap. e identificar grandes redes RC	boa	excelente	identificar redes críticas para uma análise mais aprofundada
RC distribuído formulação empírica	utiliza formulação empírica, provê um maior detalhamento das redes	muito boa a excelente	muito bom	modo preferido para um detalhamento preciso e análise através de simulação
RC distribuído análise 3D	Combina uma análise 3D com uma biblioteca de padrões de segmentos de redes	excelente	bom a muito bom	análise de redes críticas, onde é requerida alta precisão
Análise 3D	soluciona equações usando as condições de contorno dos materiais	cerca de 5% dos valores obtidos pela caracterização do circuito	bom	caracterização de biblioteca de células

4 Extração das Conectividades

A extração das conectividades é uma das primeiras tarefas realizadas por um Extrator Elétrico. Ela tem a função de identificar as diferentes camadas dos polígonos e consequentemente, estabelecer a forma como os dispositivos estão interligados, atribuindo-lhes números de nodos para regiões equipotenciais.

Dada uma descrição do leiaute a partir de um conjunto de retângulos, a solução mais simples seria, para cada retângulo, pesquisar todos os outros de forma a identificar quais estão eletricamente conectados. Entretanto, uma comparação exaustiva, onde é verificado todos os polígonos que compõem o circuito, tem tempo de computação de $O(N^2)$, onde N é o número de polígonos, tornando-se ineficiente em termos de tempo de computação.

Vários algoritmos foram desenvolvidos para realizar tarefas sobre geometrias de polígonos de forma mais eficiente [OUS 84][BRO 86][CAR 87]. Todos levam em consideração a propriedade de proximidade entre dois polígonos, que considera que polígonos somente se relacionam com outros na sua vizinhança. Deste modo, eles reduzem drasticamente o conjunto de polígonos a serem pesquisados. Estes algoritmos que consideram a propriedade de proximidade entre polígonos, tem tempo de computação de $O(N)$ ou $O(N \cdot \log(N))$.

Um dos algoritmos mais utilizado em Extratores Elétricos é o *Scanline* [CAR 87]. O algoritmo *Scanline* consiste numa linha imaginária vertical (ou horizontal) que corta o leiaute no eixo X (ou Y). A estrutura de dados do *Scanline* contém todos os polígonos que são cortados pela linha, sendo armazenados em ordem decrescente da coordenada X do canto esquerdo. Partindo do leiaute (figura 4.1), uma linha varre o leiaute da esquerda para a direita, adicionando os polígonos na lista quando passar pela borda esquerda, e deletando quando passar pela borda direita. Neste caso, o tempo de computação é proporcional a área do circuito, sendo um dos algoritmos que requer menos memória para ser executado [MEI 92]. Como ilustrado na figura 4.1, somente parte do leiaute e o circuito extraído que está sobre a linha de varredura é mantido na memória do computador.

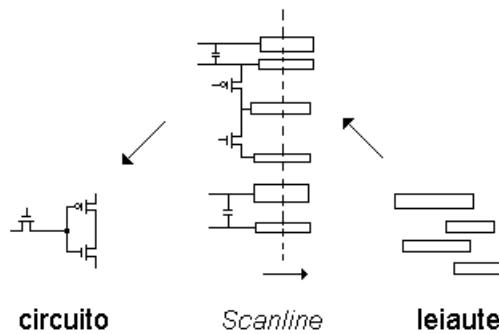


FIGURA 4.1 – Algoritmo *Scanline*

Outro algoritmo eficiente na operação sobre polígonos é o *corner-stitching* [OUS 84]. Ele consiste em manter, para cada retângulo normalizado, uma lista dos apontadores para os seus vizinhos. Entende-se como retângulo normalizado a forma como o leiaute é dividido, sendo que cada retângulo pode conter mais de um polígono do circuito.

Um vantagem da estrutura de dados *corner-stitching* está no fato de que os espaços vazios também são representados, sendo importante para ferramentas de compactação de leiaute.

É realizada uma divisão do leiaute em retângulos normalizados, onde cada um destes retângulos terá quatro ponteiros, chamados de *corner stitches*, dois no canto superior direito e dois no canto superior esquerdo. Os ponteiros destes dois cantos são suficientes para realizar todas as operações sobre o leiaute.

O extrator Space [MEI 92] é um exemplo do uso combinado dos algoritmos *Scanline* e *Corner-Stitching*. A figura 4.2 mostra uma representação do leiaute através do algoritmo *Corner-Stitching*.

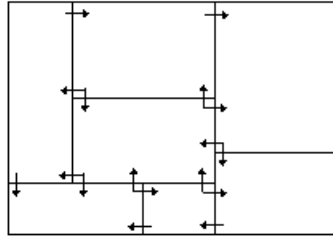


FIGURA 4.2 – Algoritmo *Corner-Stitching*

O algoritmo *quad-trees* [BRO 86], divide continuamente o circuito em quadrantes. Ele cria uma estrutura auxiliar em forma de árvore binária bidimensional, onde a raiz desta árvore é um retângulo que engloba todo o circuito. Este retângulo é repartido verticalmente e horizontalmente, dividindo o circuito em quadrantes. Esta subdivisão continua até que os quadrantes sejam pequenos suficientes para conter um pequeno número de polígonos. Desta forma, regiões com maior número de polígonos requerem uma maior subdivisão.

A árvore é obtida associando um nodo a cada quadrante e interligando cada nodo ao nodo do qual foi gerado (figura 4.3). Cada polígono é armazenado em cada quadrante que interseccionar, havendo assim polígonos que são armazenados em mais de um quadrante. E de forma a não desperdiçar memória, somente ponteiros aos polígonos são armazenados, ao invés de armazenar o próprio polígono. Nota-se que somente os nodos folhas (quadrantes que não são divididos) podem armazenar os polígonos.

Como vantagem, a sua estrutura de dados é facilmente criada a partir da lista de polígonos que formam o circuito e o tempo de computação é de $O(N \cdot \log(N))$. A desvantagem é a quantidade de memória necessária, já que é criada uma estrutura de dados auxiliar em forma de árvore. O algoritmo *quad-trees* usa, em média, 25% mais memória do que uma simples lista de polígonos [BRO 86]. Este algoritmo foi utilizado no editor de leiaute, Sela [CAS 91], desenvolvido na UFRGS.

Entretanto, a forma mais simples de armazenar os dados de um leiaute é através de uma lista encadeada, onde cada nodo da lista representa um polígono. É claro que um ordenamento dos polígonos pode melhorar a complexidade do algoritmo para algumas operações. A complexidade de espaço para uma lista encadeada é $O(N)$, onde N é o número de polígonos.

Um exemplo da utilização de listas encadeadas é o extrator Extribo [STE 89]. Apesar do método de listas encadeadas não ser muito eficiente em termos de tempo de

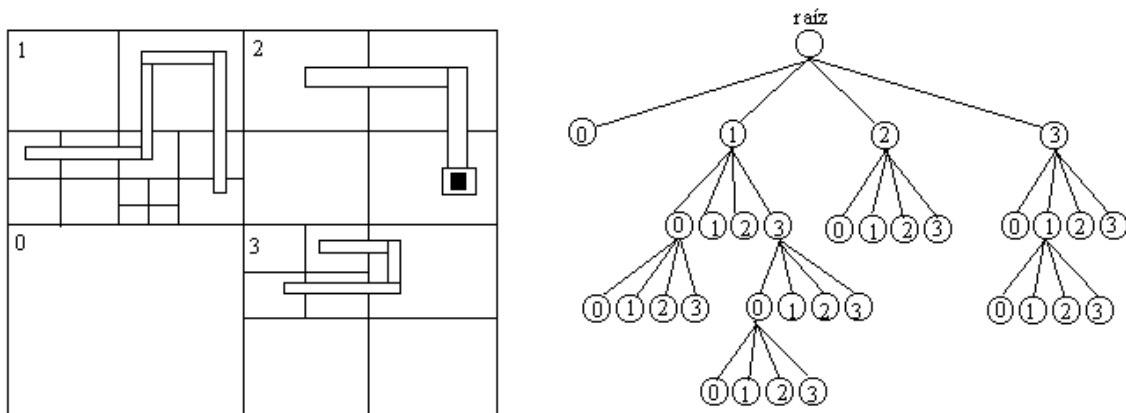


FIGURA 4.3 – Particionamento do circuito segundo algoritmo *quad-trees* e sua representação através da árvore binária

computação, ele requer uma menor quantidade de memória do que os outras estruturas de dados porque não necessita nenhuma estrutura de dados auxiliar. E isto foi determinante na escolha da estrutura de dados para o extrator Extribo, já que ele foi inicialmente criado para ser utilizado em computadores do tipo PC com 640Kb de memória disponível.

No extrator Extribo, os polígonos são definidos pelas coordenadas do canto superior esquerdo e do canto inferior direito. Então são formadas listas encadeadas para cada nível, ordenadas segundo a maior dimensão do circuito. Enquanto que a ordenação pode ser feita usando algoritmos de $O(N \cdot \log(N))$, as operações sobre máscaras são realizadas em $O(N^{3/2})$ [STE 89].

Operações como a extração das conectividades de mesmo nível são facilmente realizadas, já que os polígonos estão ordenados, mas a extração das conectividades entre níveis diferentes é menos eficiente, porque são mantidas listas encadeadas diferentes. Por exemplo, para verificar conectividade entre os níveis A e B, é necessário verificar as ligações de A em relação a B, e de B em relação a A. Logo, qualquer operação sobre níveis diferentes é pouco eficiente.

4.1 Algoritmo *Bin-Based*

A extração das conectividades realizada pelo extrator LASCA é feita pelo algoritmo *Bin-Based* [SHE 93]. Ele sobrepõe uma grade virtual sobre a área do leiaute, como mostrado na figura 4.4. Assim, esta grade divide a área em uma série de quadrados, chamados de "bins". Estes podem ser representados por uma matriz bidimensional de $N \times N$ quadrados. A complexidade de espaço desta estrutura de dados é dada por $O(bN)$, onde b é o número total de bins, e N é o número total de polígonos.

Cada elemento desta matriz irá conter todos os polígonos que estão interseccionando este quadrado, sendo armazenados através de uma lista encadeada. Poderá ocorrer que um polígono esteja contido em mais de um quadrado, e conseqüentemente estar incluído em cada lista encadeada destes quadrados. Pode parecer um grande desperdício de memória, mas assim como no algoritmo *quad-trees*, somente ponteiros para os polígonos são armazenados. Isto também evita redundância de informações, já que cada polígono deverá ter informações da sua posição e do nível a que

pertence. Na figura 4.4 temos o *bin* em destaque sendo interseccionado por sete polígonos que representam vias, metal1 e metal2.

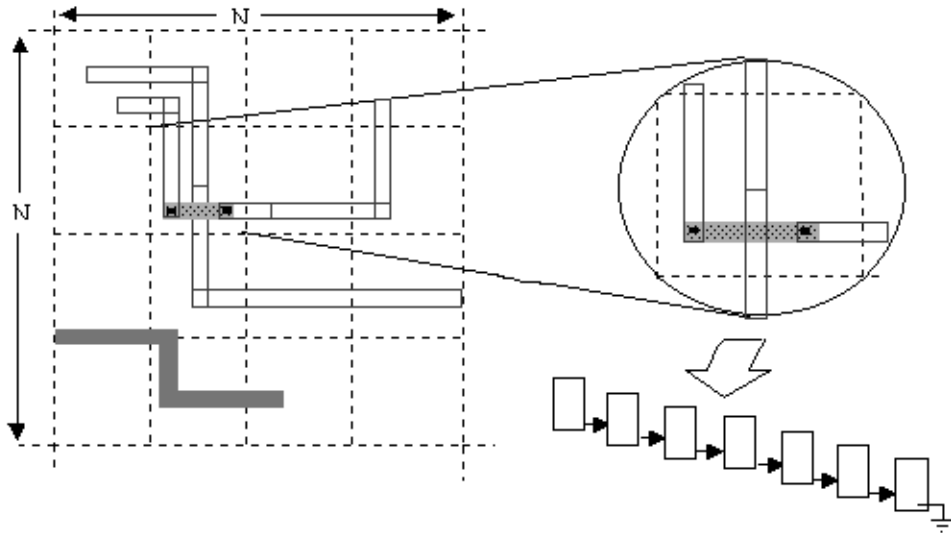


FIGURA 4.4 – Algoritmo *bin-based*

Nota-se algumas semelhança com o algoritmo *quad-trees*, como o particionamento do circuito em quadrados, mas eles diferem principalmente na estrutura auxiliar criada. Enquanto que o *quad-trees* utiliza um árvore binária, o *bin-based* utiliza uma matriz bidimensional. Além disto, o *bin-based* não difere entre regiões mais ou menos densas de polígonos, particionando o todo circuito de maneira igual. Isto pode representar uma menor eficiência, quando forem analisadas estas regiões, já que as listas encadeadas destes *bins* serão maiores do que a média geral dos *bins* do circuito. Mas isto não é tão significativo, como será mostrado na seção 4.2.

A estrutura de dados *bin-based* pode ainda ser vista como uma versão melhorada da estrutura de listas encadeadas, onde o equilíbrio entre tempo e espaço melhorou o desempenho do algoritmo. Entretanto, há exemplos que causam o pior caso de desempenho do algoritmo *bin-based*, podendo chegar a ser inferior a estrutura de listas encadeadas.

Isto é possível já que o tamanho dos *bins* são fixos, enquanto que o tamanho dos polígonos podem variar. Por exemplo, se um circuito tiver N polígonos, de modo que todos os polígonos pertençam ao mesmo *bin*, então o desempenho da estrutura de dados é semelhante ao de uma lista encadeada para quase todas as operações. Mas é pior para operações como, por exemplo, encontrar polígonos vizinhos. Isto ocorre porque mesmo os *bins* que não contiverem polígonos devem ser testados. O pior caso para estas operações é $O(b + N)$.

Em geral, a estrutura de dados *bin-based* é altamente sensível à relação entre tempo e espaço. Se os *bins* forem pequenos em relação ao tamanho dos polígonos, então os polígonos interseccionariam mais de um *bin*, e conseqüentemente aumenta a memória requerida. E ainda, muitos *bins* podem permanecer vazios, desperdiçando memória.

No caso dos *bins* serem muito grandes, o desempenho decairá, já que as listas encadeadas de cada *bin*, usadas para armazenar os polígonos, serão muito grandes. O

melhor caso seria quando cada *bin* contiver exatamente $\frac{N}{b}$ polígonos, e nenhum polígono ser armazenado em mais de um *bin*.

4.2 Avaliação do Algoritmo Implementado

O algoritmo *bin-based* foi inicialmente [MOR 99] implementado na sua forma mais simples. Foi definida uma grade fixa de 300×300 *bins* com um passo de $2\mu m$, ou seja, cada *bin* seria um quadrado de $2\mu m \times 2\mu m$. O tamanho do *bin* foi escolhido para tecnologias submicrônicas ($\leq 0,35\mu m$) de forma estatística para não formar longas lista encadeadas, o que poderia decair o desempenho do algoritmo.

Com estas dimensões, o extrator estaria habilitado a analisar circuitos com até

$$\begin{aligned} n^0 bins \times passo grade &= 300 \times 2\mu m \\ tamanho do circuito &= 600\mu m \times 600\mu m \end{aligned}$$

Entretanto, quando analisarmos circuitos menores, surgem alguns problemas ao adotar-se uma matriz fixa:

- Desperdício de memória, já que haverá um grande número de *bins* vazios.
- Um pequeno aumento do tempo de computação, já que também teria que avaliar estes *bins* vazios.

Vamos tomar como exemplo dois circuitos *benchmarks* ISCAS85, o C1908 e o C7552, e analisar a distribuição dos polígonos na grade. Podemos realizar dois tipos de análise:

- sobre a taxa de ocupação dos *bins*, ou seja, a quantidade de polígonos em cada *bin*
- e sobre o número de *bins* em que um polígono está sendo armazenado, indicando uma relação entre o tamanho do *bin* e o tamanho dos polígonos.

O circuito C1908, gerado por [MOR 99] com três níveis de metais, é formado por 54471 polígonos (não considerando níveis como difusão, implantes e poço) e suas dimensões são de $237,1\mu m \times 203,8\mu m$.

Como o tamanho do circuito é menor que o tamanho da grade, espera-se uma certa quantidade de *bins* vazios, e isto é justamente o que pode ser visto na tabela 4.1. Na figura 4.5 temos o histograma da taxa de ocupação dos *bins* para este circuito.

TABELA 4.1 – Distribuição da taxa de ocupação dos *bins* para o circuito C1908

Número de polígonos dentro de cada <i>bin</i>	Número de <i>bins</i>	
0 (<i>bins</i> vazios)	78427	87,1%
1-5	2924	3,2%
6-10	3685	4,1%
11-15	3247	3,6%
≥ 16	1717	1,9%

Neste exemplo, temos cerca de 78427 *bins* vazios, que correspondem a 87,1% do total dos *bins*, isto porque o tamanho do circuito é inferior ao tamanho da grade. Apesar de haverem muitos *bins* vazios, os *bins* ocupados possuem poucos polígonos, o que poderá ser visto na segunda análise, facilitando enormemente a manipulação das listas encadeadas. Enquanto que o restante apresenta uma distribuição variando de 1 até aproximadamente 30 polígonos por *bin*. Podemos notar que mesmo não realizando um maior número de subdivisões do circuito nas áreas mais densas, como no algoritmo *quad-trees*, a maioria dos *bins* contém entre 5 e 15 polígonos.

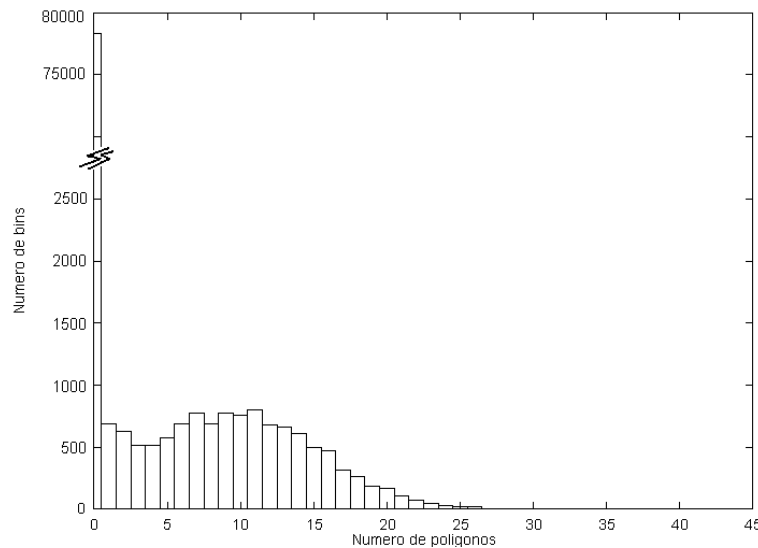


FIGURA 4.5 – Histograma da taxa de ocupação dos *bins* para o circuito C1908

A segunda análise pretende avaliar o tamanho dos *bins* em relação ao tamanho médio dos polígonos, considerando a quantidade de *bins* ocupados por um polígono. O ideal seria que cada polígono fosse armazenado em somente um *bin*, isto permitiria o menor consumo de memória. Entretanto isto não ocorre, como pode ser visto na figura 4.6, que demonstra a distribuição do número de *bins* ocupados por um polígono para o circuito C1908.

Na tabela 4.2 temos as informações retiradas do gráfico. Neste circuito, grande parte dos polígonos são menores que $2\mu\text{m}$ (tamanho de cada *bin*), ocupando somente um *bin*. O último ponto do gráfico representa a acumulação do número de polígonos que ocupam 100 ou mais *bins*, no caso são 42 polígonos.

TABELA 4.2 – Distribuição do número de *bins* ocupados por um polígono para o circuito C1908

Número de <i>bins</i> ocupados por um polígono	Número de polígonos	
1	27004	49,6%
2-5	26266	48,2%
6-10	958	1,8%
≥ 11	243	0,4%

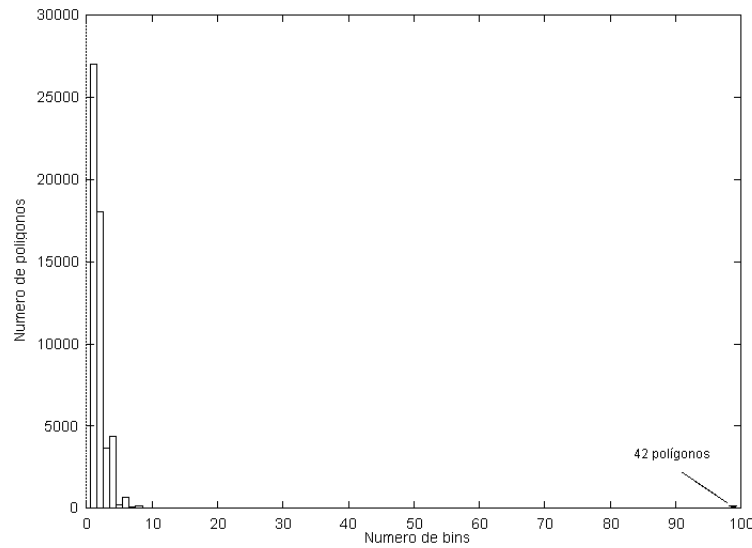


FIGURA 4.6 – Histograma do número de *bins* ocupados por um polígono

Agora vamos analisar o caso em que um circuito seja maior que o tamanho da grade ($600\mu m \times 600\mu m$). Neste caso, o algoritmo foi implementado de forma que os polígonos que estão fora da grade, sejam armazenados nos *bins* mais próximos, ou seja, nos *bins* das extremidades da grade.

Na figura 4.7, o polígono P1 será armazenado nos *bins* (3,3), (3,2) e (3,1). O mesmo ocorre quando temos polígonos com coordenadas negativas, que é o caso do polígono P2. O principal problema é que tende a superlotar os *bins* das extremidades, formando longas listas encadeadas. Isto pode ocasionar um aumento no tempo de computação das funções que devem percorrer estas listas.

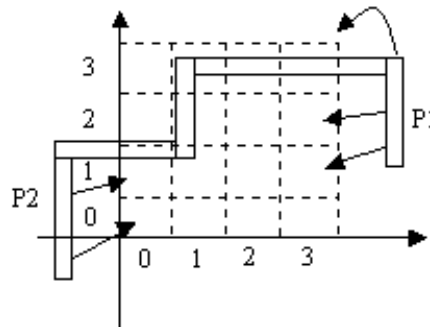


FIGURA 4.7 – Circuito maior do que a grade

Agora vamos tomar como exemplo o circuito C7552 e realizar novamente as duas análises feitas anteriormente. Este circuito é formado por 252.748 polígonos e seu tamanho é de $933,85\mu m \times 364,0\mu m$. Sendo a largura do circuito maior que a largura da grade, espera-se que todos os *bins* de coordenadas (299,X) com $0 \leq X < 300$, ou seja, os *bins* da extremidade direita, armazenem uma quantidade maior de polígonos do que a média dos outros *bins*.

Na tabela 4.3 temos a distribuição da taxa de ocupação dos *bins* para este circuito. Novamente a maior parcela corresponde aos *bins* vazios, no caso 42,6% do total dos *bins*.

TABELA 4.3 – Distribuição da taxa de ocupação dos *bins* para o circuito C7552

Número de polígonos dentro de cada <i>bin</i>	Número de <i>bins</i>	
0 (<i>bins</i> vazios)	38377	42,6%
1-5	21484	23,9%
6-10	15335	17%
11-15	9423	10,5%
≥ 16	5381	6%

Na figura 4.8 temos o histograma da taxa de ocupação dos *bins* para o circuito C7552. O último ponto do histograma temos cerca de 265 *bins* acumulados, representando todos os *bins* que tenham 100 ou mais polígonos. É justamente esta última parcela que representa os *bins* da extremidade direita da grade. Analisando mais detalhadamente este último ponto, temos cerca de 100 *bins* com mais de 500 polígonos cada um, o que representa um grande variação no número de polígonos contidos em um único *bin*.

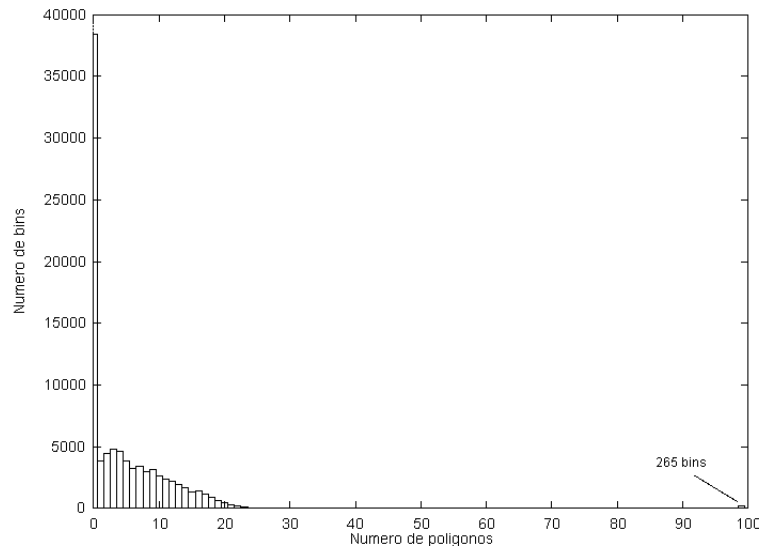


FIGURA 4.8 – Histograma da taxa de ocupação dos *bins* para o circuito C7552

Analisando o número de *bins* ocupados por um polígono para este circuito, podemos novamente verificar a correta escolha do passo da grade para esta tecnologia. Na figura 4.9 temos o histograma desta análise. Temos 127590 (50,5%) polígonos ocupando apenas 1 *bin*, 117890 (46,6%) polígonos ocupando de 2 a 5 *bins*, 5529 (2,2%) polígonos ocupando de 6 a 10 *bins*, e 1739 (0,7%) ocupando mais de 11 *bins*.

A partir destas análises podemos ressaltar alguns fatores que degradam o desempenho do algoritmo:

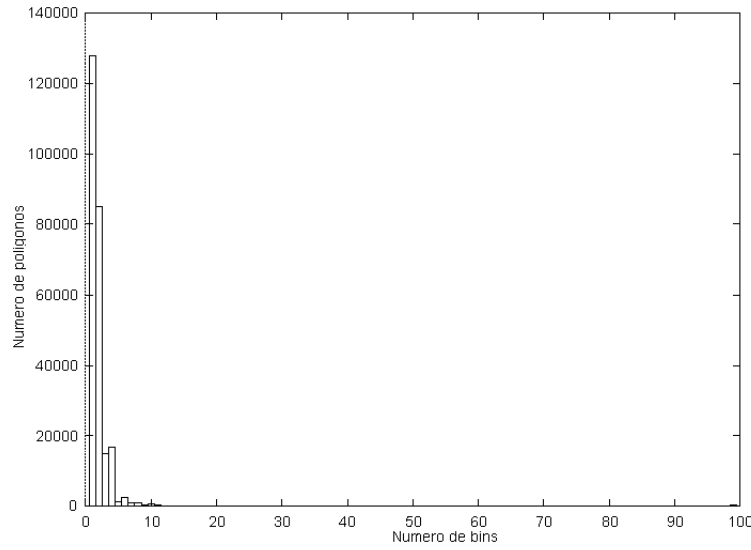


FIGURA 4.9 – Histograma do número de *bins* ocupados por um polígono

- grande número de *bins* vazios
- *bins* das extremidades da grade podem conter um grande número de polígonos, decorrente da relação dos tamanhos da grade e do circuito
- polígonos com coordenadas negativas são tratados como polígonos que excedem o tamanho da grade, e portanto colocados nos *bins* das extremidades esquerda ou inferior
- passo da grade é fixo, não importando a tecnologia utilizada

Baseado nestas observações realizamos modificações no algoritmo implementado, de modo a torná-lo mais eficiente. A primeira modificação refere-se ao tamanho da grade, que será adaptado ao tamanho do circuito.

Assim o tamanho da grade será dado por:

$$\begin{aligned} GRADE_X &= largura/passa \\ GRADE_Y &= altura/passa \end{aligned}$$

onde $GRADE_X$ representa o número de *bins* na horizontal, $GRADE_Y$ o número de *bins* na vertical, $passa$ é o tamanho do *bin* ($2\mu m$), e $largura$ e $altura$ são as dimensões do circuito.

Por exemplo, para o circuito C7552, cujo tamanho é $933,85\mu m \times 364,0\mu m$ teríamos

$$\begin{aligned} GRADE_X &= 933,85\mu m / 2\mu m = 466bins \\ GRADE_Y &= 364,0\mu m / 2\mu m = 182bins \end{aligned}$$

uma grade de $466 \times 182bins$.

Com isto devemos diminuir consideravelmente o número de *bins* vazios e a superlotação dos *bins* das extremidades.

Na figura 4.10a temos um exemplo com a utilização de uma grade fixa, apresentando todos os fatores que relatados anteriormente. Na figura 4.10b aplicamos um

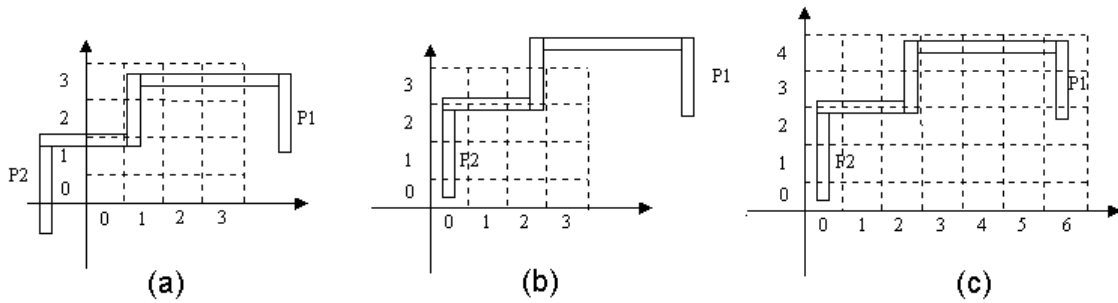


FIGURA 4.10 – Alterações no algoritmo (a) implementação inicial (b) deslocamento do circuito (c) adaptação da grade ao tamanho do circuito

deslocamento ao circuito, de modo que não tenhamos mais polígonos com coordenadas negativas. E na figura 4.10c adaptamos o tamanho da grade ao tamanho do circuito.

Na figura 4.11 temos o histograma da taxa de ocupação dos *bins* para o circuito C7552, no qual foi utilizada uma grade de $466 \times 182 \text{ bins}$.

Podemos notar duas alterações fundamentais: uma drástica redução no número de *bins* vazios, e uma melhor distribuição da taxa de ocupação dos *bins*.

Agora temos apenas 4199 *bins* vazios, correspondendo a 1,66% do total dos *bins*. Enquanto que a melhora na distribuição dos polígonos somente aconteceu devido à inexistência de polígonos fora da grade, devido à adaptação do tamanho. Enquanto que antes tínhamos *bins* com mais de 500 polígonos, agora não temos nenhum *bin* com mais de 45 polígonos.

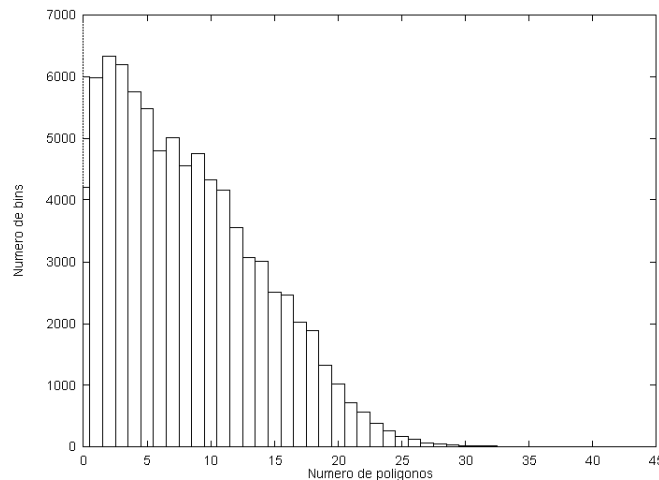


FIGURA 4.11 – Histograma da taxa de ocupação dos *bins* para o circuito C7552

Na tabela 4.4 temos uma comparação no número de *bins* vazios entre a utilização de uma grade fixa com tamanho de $300 \times 300 \text{ bins}$, e a utilização de uma grade adaptada ao tamanho do circuito, para os dois circuitos analisados anteriormente. O deslocamento é aplicado ao circuito, com a intenção de eliminar polígonos com coordenadas negativas, somente no caso da utilização da grade adaptada e se necessário. Em ambos os circuitos é evidente a redução na quantidade de memória utilizada, proporcionada pela alocação

dinâmica da matriz bidimensional.

TABELA 4.4 – Comparação do número de *bins* vazios

Circuito	Número de <i>bins</i> vazios	
	grade fixa	grade adaptada
C1908	78427	304
C7552	38377	4199

Quanto às alterações no tempo de computação, somente houveram quando o circuito era maior que o tamanho da grade inicial. Na tabela 4.5 temos uma comparação em termos de tempo de CPU, realizada numa SUN Ultra Sparc 10. Para o circuito C7552 houve uma redução de 2/3 no tempo, com a adaptação do tamanho da grade ao tamanho do circuito, enquanto que o circuito C1908 não apresentou modificações significativas.

TABELA 4.5 – Comparação do tempo de CPU

Circuito	Tempo ¹ (ms) de CPU numa Ultra Sparc 10	
	grade fixa	grade adaptada
C1908	4570	4660
C7552	81668	22481

Não foram feitas alterações no passo da grade, já que o tamanho dos *bins* de $2\mu m$ estava coerente com os tamanhos dos polígonos para as tecnologias utilizada ($0.25\mu m$).

4.3 Regras de Conectividades

As regras de conectividades entre os vários níveis são estabelecidas internamente ao extrator, de forma a simplificar o uso e minimizar a descrição do arquivo de tecnologia.

Como o extrator está apto a lidar com circuitos CMOS com até três níveis de metais, somente são descritas regras para conexão das seguintes camadas: polisilício, metal1, metal2, metal3 e *label*. Embora não sejam consideradas as camadas como difusão e poços, os parâmetros de área e perímetro das regiões de dreno e fonte podem ser fornecidos diretamente pela ferramenta de síntese, TROPIC [MOR 99].

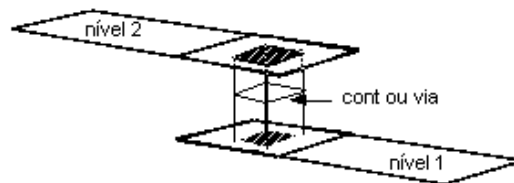


FIGURA 4.12 – Conexão indireta entre dois níveis

¹É considerado o tempo para extração das conectividades e cálculo das capacitâncias ao substrato

A conectividade dentro do mesmo nível é implícita. Enquanto que a conexão entre duas camadas dá-se de forma indireta através de contatos, via e via2, conforme a figura 4.12.

A partir da descrição CIF do circuito, os parâmetros dos polígonos são convertidos para coordenadas do canto inferior esquerdo (x_0, y_0) e para o canto superior direito (x_1, y_1), conforme a figura 4.13.

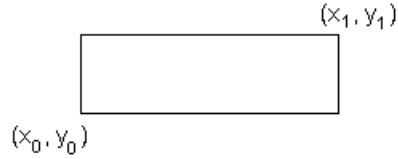


FIGURA 4.13 – Polígono definido pelas coordenadas dos cantos

Então, para que dois polígonos sejam considerados conectados, eles devem possuir níveis compatíveis e obedecer simultaneamente a uma das seguintes relações para o eixo x e para o eixo y.

$$Eixo\ x : \ P1.x0 \leq P2.x0 \leq P1.x1 \quad \text{ou} \quad (figura4.14a)$$

$$P1.x0 \leq P2.x1 \leq P1.x1 \quad (figura4.14b)$$

$$Eixo\ y : \ P1.y0 \leq P2.y0 \leq P1.y1 \quad \text{ou}$$

$$P1.y0 \leq P2.y1 \leq P1.y1$$

onde P1 e P2 representam os dois polígonos em que está sendo analisada a conectividade, e x_i e y_i ($i=0,1$) são as coordenadas dos cantos dos polígonos, conforme a figura 4.13.

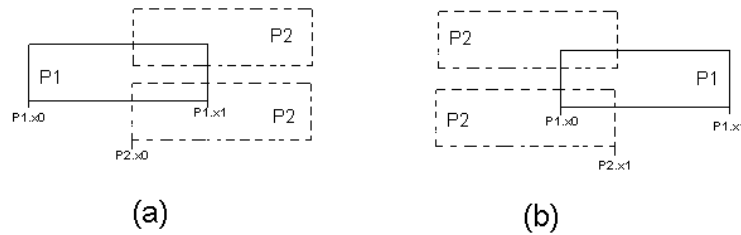


FIGURA 4.14 – Relação entre polígonos para estarem conectados

A compatibilidade entre níveis é implementada internamente ao extrator e indica quais níveis se conectam aos outros, sendo apresentados na tabela 4.6. Nota-se que a camada *label* deve sobrepor um contato, para que o extrator referencie a rede ao *label*.

TABELA 4.6 – Resumo das regras de conectividades implementadas internamente.

Camada	Conectado à
polissilício	contato
metal1	contato via
metal2	via via2
metal3	via2
label	contato

4.4 Verificação das Conectividades

Após formar a estrutura de dados através do algoritmo *Bin-Based*, inicia-se a fase de verificação das conectividades entre os diversos polígonos.

Dado um polígono qualquer que ainda não foi analisado, determina-se a quais *bins* ele está contido. Então, pesquisa-se nestes *bins* se há algum outro polígono que se encontre conectado.

A figura 4.15 será tomada para exemplificar a avaliação das conectividades no circuito, e demonstrar como a estrutura de dados *bin-based* auxilia nesta tarefa.

Na figura 4.15 temos uma grade 4×4 superposta ao circuito e duas redes a serem identificadas. Tomamos um primeiro polígono, correspondendo a ordem que estava descrito no arquivo do leiaute (.CIF). É importante notar que não é feito nenhum ordenamento dos polígonos, a pesquisa das conectividades sobre os polígonos é feita tomando-se os polígonos segundo a ordem em que estavam no arquivo do leiaute.

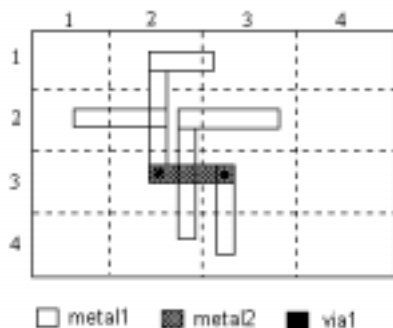


FIGURA 4.15 – Exemplo para extração das conectividades

Na figura 4.16a nomeamos este polígono como sendo P1. O polígono está contido nos *bins* (1,2) e (2,2). Segundo o algoritmo para extração das conectividades, devemos pesquisar todos os bins em que o polígono estiver contido.

Então começamos com o *bin* mais a direita, no caso (1,2), procurando algum outro polígono em contato e que satisfaça as regras de conectividade descritas na seção 4.3. No caso exemplo não temos nenhum, então passamos para o *bin* (2,2). Aqui teríamos os polígonos P2 e P7(figura 4.16b), mas a pesquisa na lista encadeada do respectivo *bin*

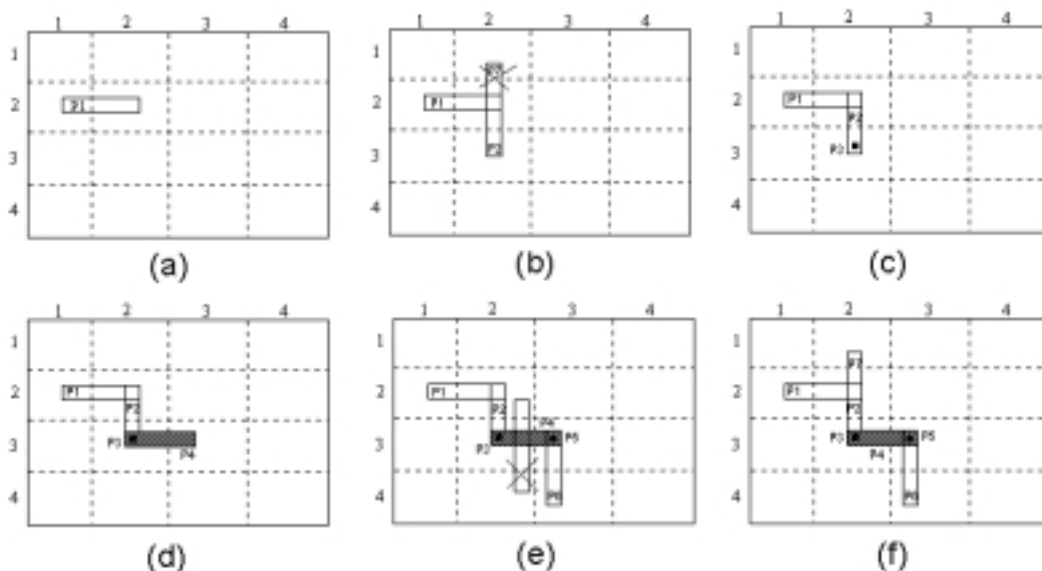


FIGURA 4.16 – Extração das conectividades passo-a-passo.

vai até o momento em que encontrarmos um primeiro polígono, no caso o P2. Então buscamos recursivamente todos os polígonos conectados a P2.

Quando pesquisarmos o *bin* (3,2) a procura de polígonos conectados a P2, encontraremos o polígono de metal2, entretanto, verificando as regras de conectividade, vemos que metal1 não se conecta diretamente a metal2, logo continuamos a pesquisa. Então encontramos uma via1, que nomeamos de P3 na figura 4.16c. Neste momento passamos a analisar os *bins* a que P3 pertence, em busca de contatos com P3, no caso temos somente o *bin* (3,2). Agora há possibilidade de conexão da via1 (P3) com o metal2 (P4).

Passamos a analisar o polígono P4, e vemos que no *bin* (3,2) há um polígono de metal1 que intersecciona P4, mas novamente não há possibilidade de conexão entre eles. Então passa a ser analisado o *bin* (3,3), figura 4.16d. E assim sucessivamente, até encontrarmos P6 (figura 4.16e).

Neste momento, em que P6 não se conecta a nenhum outro polígono, voltamos a analisar os *bins* a que o polígono anterior (P5) pertence. Nota-se que não serão novamente verificados todos os polígonos dos *bins* a que P5 pertence, mas somente os que ainda não haviam sido verificados, devido a termos encontrado o P6.

Se nenhum outro polígono for encontrado se conectando a P5, voltamos para P4, e assim sucessivamente até P1. Mas observamos que o polígono P2 se conecta a P7 (figura 4.16f), logo ele é incluído na lista dos polígonos da rede. Terminada a análise da rede, busca-se um próximo polígono, ainda não analisado, para formar outra rede, e assim até que todos os polígonos tenham sido analisados.

5 Extração das Capacitâncias

Um circuito VLSI pode ser visto como um denso sistema de vários níveis de metais isolados por um dielétrico, sob um substrato de silício. O cálculo das capacitâncias para este sistema multicondutor é uma tarefa complexa e difícil. Muitas técnicas foram desenvolvidas, como o método da diferenças finitas [SEI 88][UEB 86], método dos elementos finitos [RUE 73], métodos dos elementos de contorno [NIN 87] e os algoritmos multipólos [NAB 91]. Entretanto, por se tratarem de métodos específicos para extração 3D, eles apresentam um grande custo computacional quando aplicados a circuitos VLSI.

Então, vários modelos empíricos foram propostos [SAK 83][CHE 92]. As formulações são geralmente validadas através de simulações numéricas, exibindo excelente precisão, e passaram a ser largamente utilizadas pela sua simplicidade e velocidade. Mas eles se limitam a poucas estruturas físicas para os quais foram desenvolvidos, não sendo extensíveis a outras estruturas.

Os procedimentos adotados, no extrator de conectividades desenvolvido, para a estimação das capacitâncias das interconexões estão baseados na metodologia descrita em [CON 96], no qual introduz simplificações no tratamento de estruturas multiníveis. Aliada a esta metodologia, foram utilizadas formulações proposta em [CHE 92] e formulações baseadas na área e perímetro das conexões, ambas as quais foram comparadas aqui em relação ao simulador numérico FastCap¹[NAB 91], apresentando resultados próximos.

5.1 Metodologia de Extração das Capacitâncias

Dado que nos modernos projetos VLSI pode-se assumir uma infinidade de interações entre os diversos níveis de interconexões, tornando o problema de extração muito complexo, em [CON 96] são descritos cinco fundamentos aplicáveis a processos submicrônicos. Eles visam a introdução de simplificações no tratamento do problema de extração de capacitâncias de interconexão em circuitos multiníveis, produzindo suficiente precisão. Estes fundamentos foram incluídas no produto *Cadence Silicon Ensemble 5.0* [CON 96] e estão baseadas no uso do extrator de capacitâncias 3D, FastCap [NAB 91]. Em [CON 96] foram utilizados circuitos nas tecnologias 0,5 μm , 0,35 μm e 0,18 μm para validar estas simplificações. Estes cinco fundamentos foram incorporados ao extrator desenvolvido e são descritos a seguir:

1. O plano terra e as conexões vizinhas do mesmo nível influenciam fortemente na capacitância de acoplamento, tendo que ser consideradas para um modelamento preciso.

Na tabela 5.1 é analisada a influência do plano terra e das conexões vizinhas, utilizando como parâmetro a relação $C_{i,i-2}/C_{i,i}$, onde $C_{i,i-2}$ é a capacitância entre conexões nos níveis i e $i-2$, e $C_{i,i}$ é a capacitância total da conexão do nível i . Nota-se que para o caso mais realista, onde temos no mínimo um plano terra (substrato) e duas conexões vizinhas, a capacitância de acoplamento é reduzida para 1,8%, denotando a importância de serem consideradas estas conexões vizinhas.

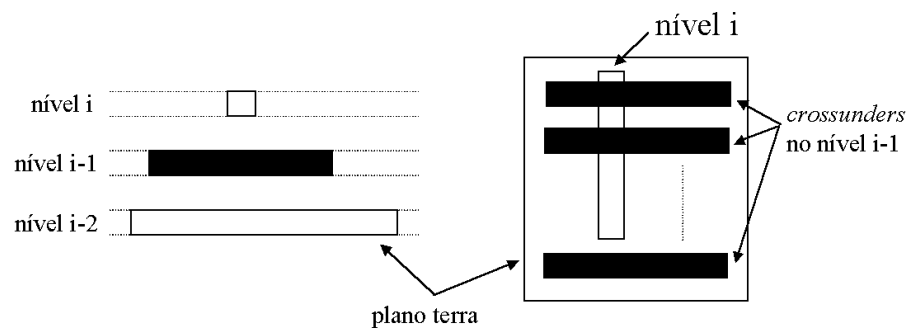
¹FastCap é um programa de domínio público disponível através de ftp *anonymous* em *rle-vlsi.mit/edu*. É utilizado em alguns produtos comerciais, como por exemplo, da Quantic e Ansoft.

TABELA 5.1 – Influência do plano terra e das conexões vizinhas

plano terra no nível $i - 3$	Número de vizinhos	$C_{i,i-2}/C_{i,i}$
Não	Nenhum	28,4%
Sim	Nenhum	16,3%
Sim	dois	1,8%

2. Durante a extração das capacitâncias no nível i , os níveis $i \pm 2$ serão considerados como plano terra, com erro negligenciável. Não há necessidade de analisar além destes níveis.

A diferença entre adotar os níveis $i \pm 2$ como planos terras é de apenas 0,5% para as capacitâncias de acoplamento entre vizinhos, e de aproximadamente 5% para os *crossovers* e *crossunders* [CON 96].

FIGURA 5.1 – Níveis $i \pm 2$ como planos terra

3. A capacitância entre as conexões de um nível $i - 1$ e o nível $i + 1$ é negligenciável quando a densidade de conexões no nível i ultrapassar um certo limiar².

FIGURA 5.2 – Interação entre os níveis $i - 1$ e $i + 1$

Em [CON 96] foi comprovado que, para processos $0,18\mu m$, a capacitância de acoplamento entre os níveis $i - 1$ e $i + 1$ representa apenas 0,4% da capacitância total

²Nos processos submicrônicos, a mínima área ocupada por um nível de metal é determinada pela *foundry* como sendo de 30%. Observa-se que o máximo de ocupação é de 50%, onde teríamos o espaçamento entre duas conexões igual a largura da conexão [CON 96].

da conexão do nível $i + 1$, para uma ocupação de 30% do nível i e com um plano terra no nível $i - 2$. E apenas 0,2% para o caso de haver dois vizinhos no nível $i + 1$. Dada a mínima área de ocupação dos níveis de metais em processos submicrônicos, a capacitância de acoplamento entre uma conexão no nível $i + 1$ e no nível $i - 1$ é insignificante, porque o nível i acaba funcionando como um blindagem.

4. A análise das capacitâncias entre conexões no mesmo nível considerará somente os vizinhos mais próximos de forma independente. A variação da largura é negligenciável.

O estudo na interação de vizinhos mais distantes do que os vizinhos imediatos, demonstra que eles podem ser ignorados sem prejuízo na precisão, causando um erros menores do que 3% [CON 96], enquanto que variações na largura dos vizinhos pode gerar erros de até 0,2%.

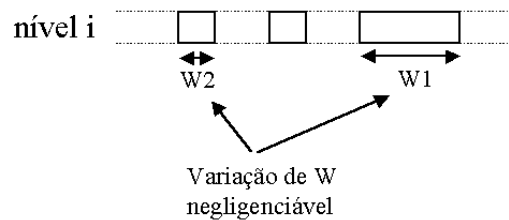


FIGURA 5.3 – Variação na largura das conexões

5. A interação dos níveis $i - 1$ em conjunto com o nível $i + 1$ no nível i é negligenciável, podendo ser considerados independentemente.



FIGURA 5.4 – Níveis $i - 1$ e $i + 1$ considerados independentemente

Os cinco fundamentos acima justificam uma simplificada metodologia $2\frac{1}{2}D$ de extração, onde para cada conexão num nível i , devemos analisar os vizinhos imediatos do mesmo nível, todos os *crossunders* no nível $i - 1$ e os *crossovers* no nível $i + 1$, tratando os níveis $i \pm 2$ como planos terra. Com isto estamos habilitados a tratar estruturas de múltiplos níveis de condutores, mantendo uma precisão aceitável. As seções seguintes mostram o algoritmo de análise das capacitâncias e as formulações utilizadas para o cálculo das capacitâncias.

5.1.1 Algoritmo de Análise das Capacitâncias

Uma situação típica que pode ocorrer em circuitos digitais *standard cells*, *gate-arrays* ou *full-custom* está demonstrada na figura 5.6a. Utilizamos esta estrutura para exemplificar o algoritmo de análise $2\frac{1}{2}$ D das capacitâncias. A conexão que está sendo analisada está no nível i , existem dois vizinhos no mesmo nível, as conexões 1 e 2, enquanto que as demais conexões estão no nível $i + 1$ (*crossovers*).

Primeiramente analisamos as conexões que estão no mesmo nível e que são vizinhos imediatos (figura 5.6b). Elas devem ser encontradas até uma certa distância da conexão em análise, fornecida através do arquivo de tecnologia, para que sejam consideradas. Em seguida são obtidos os parâmetros geométricos do leiaute, como o comprimento efetivo³(L) e a distância de separação em relação aos vizinhos (S). Na figura 5.6c temos os parâmetros L1 e S1 para a conexão 1, e na figura 5.6d, temos L2 e S2 para a conexão 2. Depois são analisadas as conexões dos níveis superior e inferior (figura 5.6d), extraíndo-se a largura (Wc) e a separação entre elas (Sc) (figura 5.6e). Além destas análises devemos verificar a existência de conexões paralelas ao lado de largura W.

Com todas estas informações podemos calcular as capacitâncias da conexão em análise, como mostra a figura 5.5.

Para cada conexão em análise com largura W e comprimento L

Para cada lado de comprimento L

Encontrar os vizinhos que estejam a uma distância menor que um dado limite

Tomar o comprimento efetivo(paralelo a conexão em análise) e o espaçamento

Para cada vizinho

Calcular as capacitâncias de fringing, lateral e de paralelismo

Fim

Fim

Para cada crossover de largura Wc

Encontrar a área em comum entre as conexões

Calcular a capacitância de crossover

Fim

Para cada crossunder de largura Wc

Encontrar a área em comum entre as conexões

Calcular a capacitância de crossunder

Fim

Para cada lado de largura W

Encontrar os vizinhos que estejam a uma distância menor que um dado limite

Tomar o comprimento efetivo e o espaçamento

Para cada vizinho

Calcular as capacitâncias de fringing, lateral e de paralelismo

Fim

Fim

Fim

FIGURA 5.5 – Algoritmo de análise $2\frac{1}{2}$ D das capacitâncias [CON 96]

³Comprimento efetivo de um vizinho para um dada conexão é o comprimento em que ambos são paralelos.

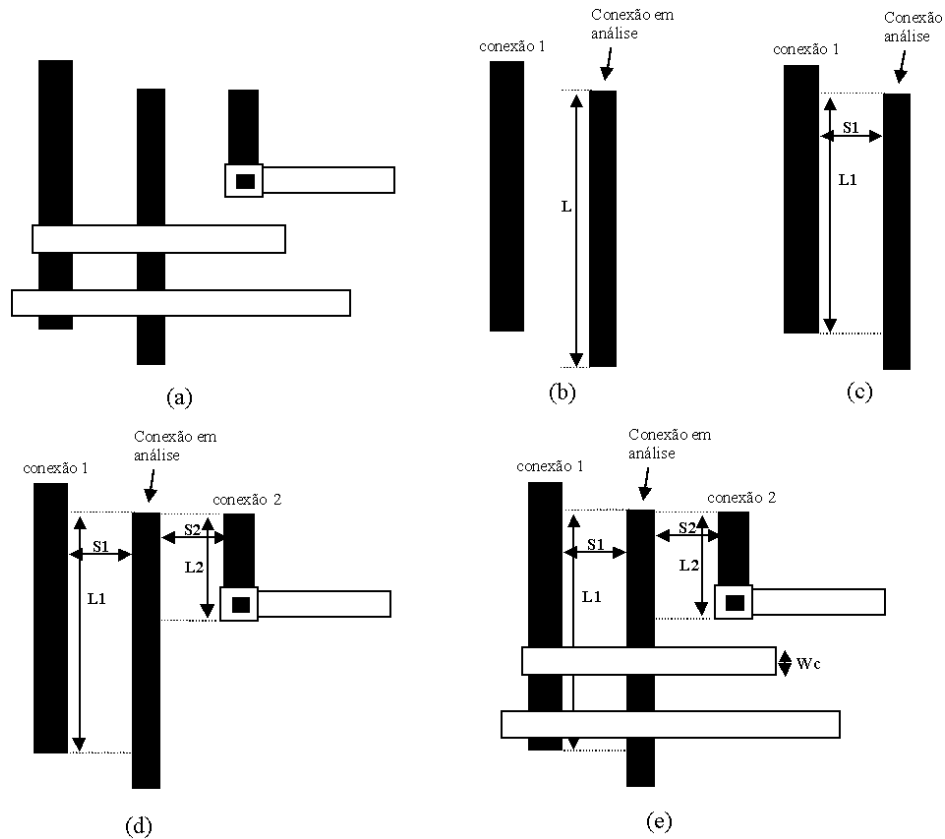


FIGURA 5.6 – (a) estrutura para exemplificar o algoritmo (b) para cada lado de comprimento L (c) determinar o comprimento efetivo ($L1$) e o espaçamento ($S1$) dos vizinhos (d) analisar os dois lados (e) encontrar *crossovers* e determinar a largura deles

5.2 Modelos de Capacitância

As capacitâncias de interconexão de cada nodo do circuito são calculadas seguindo o modelo da figura 5.7. Ele consiste de três níveis condutores, no qual o nível 1 pode ser considerado como o substrato, tratado como plano de referência (plano terra). São assumidos condutores perfeitos, com resistividade zero. As capacitâncias de interesse para qualquer nodo do circuito, em geral, consistem dos seguintes elementos:

- capacitância de *overlap* (C_a) - é devido à sobreposição de duas conexões em planos diferentes
- capacitância lateral (C_{lat}) - é a capacitância entre dois condutores de um mesmo plano
- capacitância de *fringing* (C_{fr}) - é o acoplamento entre a superfície de um condutor e a borda de outro condutor, em planos diferentes.

Então, a capacitância total de uma interconexão, pode ser dada por:

$$C_{tot} = C_a + C_{lat} + C_{fr}$$

Para a geometria da figura 5.7, a capacitância de *overlap* é dada pelo somatório da capacitância devido a sobreposição dos níveis 2 e 3 (C_{23a}), e dos níveis 1 e 2 (C_{21a}).

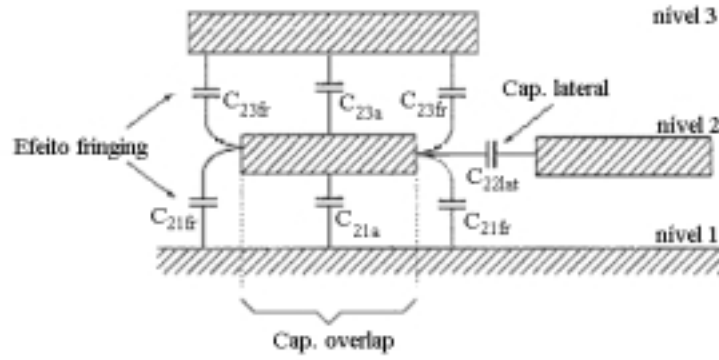


FIGURA 5.7 – Modelo de Capacitâncias

Similarmente, a capacitância de *fringing* é dada pela soma das capacitâncias C_{21fr} e C_{23fr} , enquanto que a capacitância lateral é dada por C_{22lat} . Note que C_{21fr} e C_{23fr} é o somatório da capacitância de *fringing* de ambos os lados da superfície do condutor, e que o valor das capacitâncias lateral e de *fringing* são alterados pela presença ou ausência dos condutores nos níveis 1 e 3 [ARO 96]. Deste modo, a capacitância total do condutor no nível 2 pode ser dada por:

$$C_{2tot} = C_{21a} + C_{23a} + C_{21fr} + C_{23fr} + C_{22lat}$$

Os valores das capacitâncias C_{ij} ($i, j = 1 \dots n$, onde n é o número de condutores) são calculados através de formulações empíricas, demonstrados nas seções a seguir, exibindo excelente concordância com o simulador numérico de capacitâncias, FastCap [NAB 92].

5.2.1 Capacitância Intrínseca

A capacitância intrínseca é a capacitância entre uma conexão e o plano terra, sendo formada por duas componentes: capacitância de *overlap* e a capacitância de *fringing*.

A capacitância de *overlap* pode ser modelada como um capacitor de placas paralelas. Se largura da conexão for substancialmente maior que a espessura do material isolante, pode-se assumir que as linhas do campo elétrico são ortogonais às placas do capacitor. Sob estas condições, e assumindo que o material isolante seja SiO_2 , a capacitância de *overlap* pode ser modelada como:

$$C_a = \frac{\epsilon_{ox}}{t_{ox}} WL$$

onde W e L são respectivamente a largura e o comprimento da conexão, ϵ_{ox} é a permissividade relativa do óxido, e t_{ox} é a espessura do óxido do *gate*. Entretanto, as conexões são geralmente roteadas acima do óxido que serve para separação dos dispositivos (*field oxide*), o qual é substancialmente mais espesso do que o óxido de *gate* (figura 5.8), resultando numa menor capacitância por unidade de área. Deste modo, a capacitância de *overlap* pode ser dada por:

$$C_{overlap} = C_{area} WL$$

onde C_{area} é a capacitância por unidade de área, dada em $fF/\mu m^2$, e WL representam a área de sobreposição.

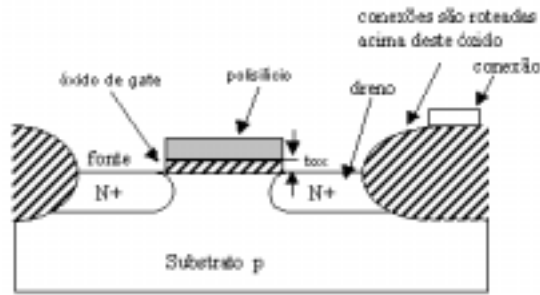


FIGURA 5.8 – Diferenças na espessura do óxido

A capacitância de *fringing* é formada entre a borda de um condutor e a superfície de um segundo condutor (no caso, o plano terra para a capacitância intrínseca). Ela somente é identificada entre dois condutores de planos diferentes. É dada por:

$$C_{fr} = 2 \times C_{fringing}L$$

onde $C_{fringing}$ é a capacitância de perímetro, dada em $fF/\mu m$. Assim a capacitância intrínseca pode ser dada pela equação 5.1, exatamente como a capacitância do modelo original, apresentado na equação 2.1.

$$C_{int} = (C_{area} \cdot W + 2 \cdot C_{fringing}) \cdot L \quad (5.1)$$

Para o correto cálculo da capacitância intrínseca, os dados necessários (C_{area} e $C_{fringing}$) devem constar no arquivo de tecnologia, através da diretiva “#CAPACITANCE INTRINSIC < layer > < n1 > < n2 >”, onde < layer > especifica a camada para a qual os dados serão utilizados, < n1 > e < n2 > correspondem respectivamente a C_{area} e $C_{fringing}$.

Note que estes dados devem ser fornecidos em relação ao plano terra, no qual o substrato será considerado como um plano terra para o nível i, somente se $i=1$ ou $i=2$, caso contrário, o plano terra será os níveis $i \pm 2$ (ver seção 5.1 item 2).

Foram realizadas comparações entre a equação 5.1 e simulações através do FastCap[NAB 91], utilizando-se a estrutura da figura 5.9. Nesta estrutura as dimensões foram normalizadas em relação à mínima largura da camada correspondente (W_{min}). A conexão apresenta um comprimento igual a $20W_{min}$ e o terra é um plano de $40W_{min} \times 40W_{min}$. Os resultados comparativos da capacitância intrínseca para uma conexão de polissilício, metal1, metal2 e metal3 são mostrados na tabela 5.2. Os valores apresentados estão fF , e os parâmetros utilizados, como largura (W_{min}), espessura (T) e distância ao plano terra (H), são referentes à tecnologia 0,25 μm .

TABELA 5.2 – Resultados comparativos para a capacitância intrínseca (fF)

	Poly	Metal1	Metal2	Metal3
FastCap	0,568	0,7441	1,155	0,9978
Equação 5.1	0,5683	0,7318	1,1277	0,9914
Sakurai[SAK 83]	0,5688	0,7317	1,1270	0,9921

Na tabela também foram incluídos os valores obtidos através de [SAK 83]. Observa-se uma excelente concordância entre os três métodos, onde a utilização da equação 5.1 apresentou um erro menor do que 2,5% em relação ao FastCap. Ressalta-se que para o cálculo da capacitância intrínseca do metal2 e metal3 foram utilizados, como plano terra, os níveis $i - 2$, ou seja, polisilício e metal1, respectivamente.

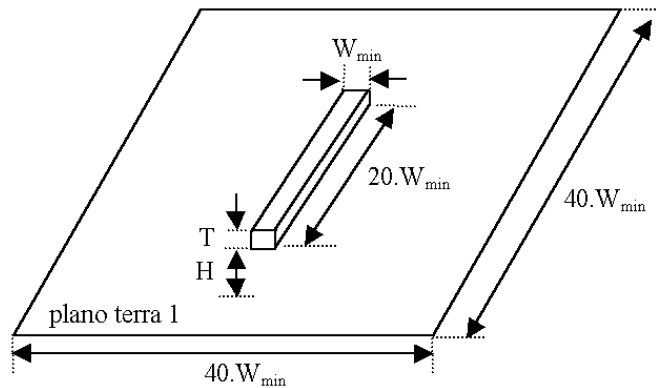


FIGURA 5.9 – Estrutura 3D para simulação da capacitância intrínseca no FastCap

5.2.2 Capacitância de Acoplamento entre Conexões do Mesmo Nível

Quando um sinal em uma conexão transiciona, ele gera uma interferência nas conexões adjacentes, devido ao acoplamento entre as conexões (podendo ser capacitivo ou indutivo), isto é chamado de efeito *crosstalk*. O distúrbio resultante age como uma fonte de ruído, podendo causar erros intermitentes, já que ele depende do valor transiente e da fase do sinal das conexões presentes na vizinhança. O problema é pior em processos submicrônicos, já que geometrias mais finas e numerosas camadas de metal significam um maior acoplamento entre conexões.

Este efeito pode afetar o desempenho do circuito de várias formas [SIC 93]: propagação de transições espúrias dentro da lógica do circuito (*glitches*), aumento e/ou decremento do atraso das portas lógicas, ruído dentro de células analógicas. Apesar de encontrarmos capacitâncias de acoplamento em qualquer lugar dentro de um leiaute, somente algumas configurações, não necessariamente correspondendo aos maiores acoplamentos, podem ser responsáveis por comportamentos falhos. Mesmo assim, determinar a exata quantidade de *crosstalk* é um problema difícil, já que além de depender da capacitância efetiva vista pela porta lógica, também depende do atraso, inclinação do sinal e do estado lógico das redes vizinhas [SCH 97]. O estado lógico, em particular, varia consideravelmente de um ciclo ao próximo. Deste modo, o comportamento de uma rede passa a ser função das redes vizinhas, além da própria rede. Devido a estas dependências, em processos submicrônicos, o atraso ou inclinação de um sinal passa a ser uma escala de valores, que variam de ciclo para ciclo, dependendo das redes adjacentes [SCH 97].

A capacitância de *crosstalk*, entre conexões do mesmo nível, pode ser reduzida se manter-mos a distância entre duas conexões grande o suficiente para introduzir uma conexão de blindagem (GND ou Vdd), entre os dois sinais. Com isto, substituímos a capacitância entre conexões para uma capacitância ao plano terra, eliminando a interferência [RAB 96]. Entretanto, isto nem sempre é possível, de modo que uma

avaliação das capacitâncias laterais é realizada pelo extrator de conectividades do LASCA.

A capacitância lateral, formada pelo paralelismo das bordas de dois condutores no mesmo plano, depende fundamentalmente de dois fatores:

- espessura da conexão
- e separação entre as conexões

Sendo a espessura de uma conexão um fator dependente da tecnologia utilizada e da camada ao qual está avaliando-se a capacitância, a separação entre as conexões passa a ser o principal fator dependente da estrutura analisada. Embora as regras da tecnologia informem o valor da capacitância lateral para cada camada, eles estão restritos a mínima distância de separação. Sabe-se que em circuitos realísticos, a taxa de ocupação de uma das camadas pode variar de 30% a 50%, fazendo com que a distância entre duas conexões não seja sempre igual a mínima distância da tecnologia. Com isto a utilização da capacitância lateral fornecida pelas regras da tecnologia podem gerar erros grosseiros.

Então optou-se por utilizar a formulação empírica descrita em [CHE 92]. Podemos ter dois casos de acordo com o número de planos terra existentes. Para o caso de apenas um plano terra, a equação dada a seguir é válida.

$$C' = 1,064 \left(\frac{T}{S} \right) (F_1)^{0,695} + (F_2)^{1,4148} (F_1)^{0,804} + 0,831 (F_2)^{0,055} \left(\frac{2H}{2H+0,5S} \right)^{3,542}$$

onde $F_1 = \left(\frac{T+2H}{T+2H+0,5S} \right)$

$$F_2 = \left(\frac{W}{W+0,8S} \right)$$

C' é a capacitância normalizada, por unidade de comprimento

ϵ é permissividade do dielétrico, para o SiO_2 é igual a $3,9 \times 0,008855 fF/\mu m$

W é a largura da conexão

T é a espessura da conexão

H é a distância de separação entre a conexão e o plano terra

S é a separação entre as duas conexões

L é o comprimento efetivo, em que as duas conexões são paralelas

E para o caso de dois plano terra, teremos:

$$C' = \frac{T}{S} \left(1 - 1,897e^{\frac{-H}{0,31S} - \frac{-T}{2,474S}} + 1,302e^{\frac{-H}{0,082S}} - 0,1292e^{\frac{-T}{1,326S}} \right) + 1,722 \left(1 - 0,6548e^{\frac{-W}{0,3477H}} \right) e^{\frac{-S}{0,651H}}$$

Considerando que o dielétrico seja óxido de silício(SiO_2), então a capacitância lateral pode ser dada, em fF , por

$$C_{lat} = C' \times \underbrace{3,9 \times 0,008855}_{\epsilon} \times L \quad (5.2)$$

Com o intuito de validar estas formulações, realizamos comparações entre a equação 5.2 e simulações no Fastcap, utilizando-se da estrutura da figura 5.10. Nesta estrutura temos todas as dimensões normalizadas em relação à largura mínima da conexão (W_{min}).

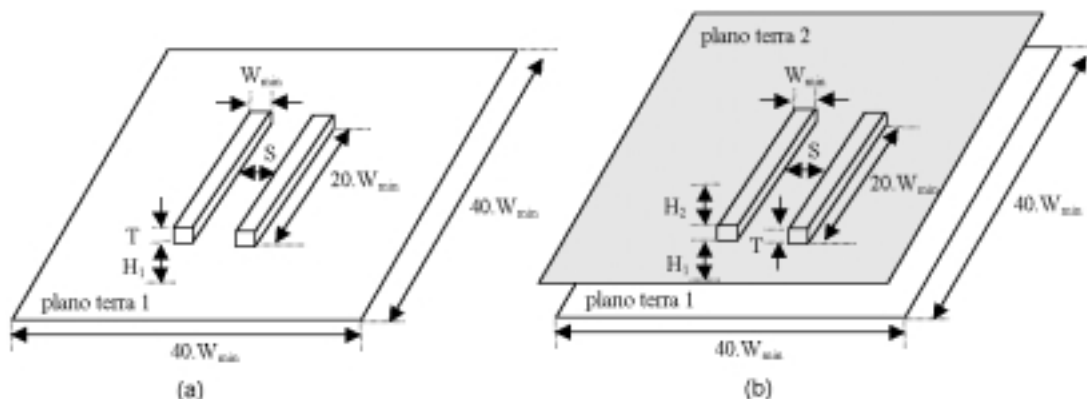


FIGURA 5.10 – Estrutura 3D para simulação da capacitância lateral no FastCap

Para o caso da figura 5.10b temos H_1 e H_2 como sendo as distâncias ao plano terra 1 e 2, respectivamente. Apesar de [CHE 92] não explicitar qual das distâncias deve ser utilizada para o cálculo da capacitância, foi tomada uma média entre os dois valores.

Na tabela 5.3 temos os resultados comparativos, variando o espaçamento entre as duas conexões (S), onde todas as dimensões representadas na figura 5.10 são referentes à camada de metal1 da tecnologia $0,25\mu m$, e considerando o substrato e o metal3 como planos terra (este último, apenas para o caso da figura 5.10b).

Os valores das capacitâncias para um plano terra mostraram-se concordantes com as simulações no FastCap, apresentando erros menores do que 2,5%, e na presença de dois planos terra, o erro é menor do que 10%.

Comparando-se os resultados entre os dois casos, nota-se a influência do segundo plano terra, diminuindo o valor das capacitâncias em até 20% (no caso de $S=4$).

TABELA 5.3 – Resultados comparativos para a capacitância lateral

		$S=1$	$S=1,5$	$S=2$	$S=3$	$S=4$
1 plano terra	FastCap	0,7076	0,5072	0,3954	0,2807	0,2095
	Equação 5.2	0,7127	0,5151	0,4052	0,2821	0,2129
	Erro (%)	+0,7	+1,55	+2,47	+0,5	+1,62
2 planos terra	FastCap	0,679	0,4758	0,3619	0,2328	0,1596
	Equação 5.2	0,6183	0,448	0,3534	0,2423	0,1754
	Erro (%)	-8,93	-5,84	-2,35	+4,08	+9,89

Os dados necessários para o cálculo da capacitância lateral devem ser fornecidos através do arquivo de tecnologia pela diretiva “#CAPACITANCE CROSSTALK < layer > < n1 > < n2 > < n3 >”, onde < layer > especifica a camada, < n1 > é a espessura da camada, < n2 > é a distância ao plano terra, e < n3 > é a máxima distância em que serão analisadas as conexões vizinhas.

Recomenda-se que o parâmetro < n3 > seja no máximo igual a 2,5 vezes a mínima distância de separação entre duas conexões (S_{min}), garantindo assim que somente os vizinhos imediatos sejam analisados (ver seção 5.1 item 4) e que a equação 5.2 seja apropriada. Na figura 5.11 temos a representação da escolha deste parâmetro, onde temos três conexões (C1, C2 e C3). De acordo com o algoritmo da figura 5.5, quando

analisarmos C1 devemos procurar somente as conexões que forem paralelas ao lado de comprimento L ou ao lado de largura W, e que estejam dentro da área tracejada. Assim, as conexões que estiverem dentro da área achurada (não são paralelas) ou fora da área tracejada (excederam a distância máxima) não serão consideradas como vizinhos. No caso da figura 5.11 considerariamos apenas a conexão C2 como vizinho de C1.

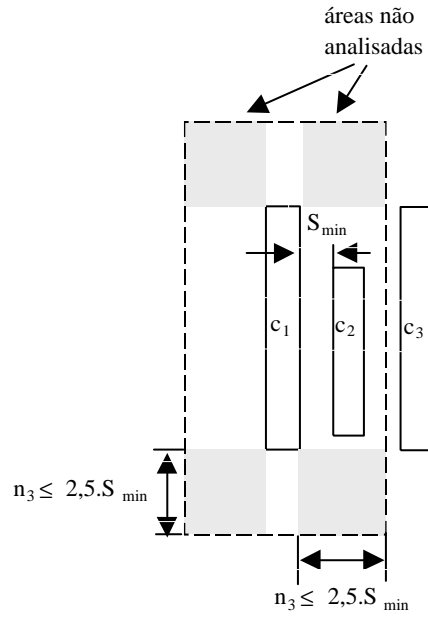


FIGURA 5.11 – Máxima distância entre conexões vizinhos

Para que o extrator utilize a equação específica ao número de planos terra existente, deve ser fornecido informações da ordem crescente em que as camadas estão dispostas no processo de fabricação. Com esta finalidade, deve ser incluída no arquivo de tecnologia a diretiva “#CAPACITANCE ORDER < layer₁ > ... < layer_n >”.

5.2.3 Capacitância de Acoplamento entre Conexões de Níveis Adjacentes

A capacitância entre níveis diferentes somente é calculada quando houver uma área em comum entre duas conexões. Esta área em comum pode ser obtida através da operação lógica AND entre duas conexões de níveis adjacentes.

Assim como a capacitância intrínseca, a capacitância entre níveis pode ser decomposta em duas componentes: capacitância de *overlap* e a capacitância de *fringing*. Considerando as conexões de níveis adjacentes como sendo ortogonais, o que é frequentemente verdadeiro num leiaute, a capacitância de *overlap*, devida a sobreposição de uma camada sobre outra, pode ser dada por

$$C_{overlap} = C_{area} \cdot W_1 \cdot W_2$$

onde C_{area} é a capacitância por unidade de área entre os níveis 1 e 2, dada em $fF/\mu m^2$, $W_1 \cdot W_2$ representa a área em comum entre as duas conexões. E a capacitância de *fringing* é dada por

$$C_{fr} = 2 \cdot C_{fringing} \cdot (W_1 + W_2) \quad (5.3)$$

onde $C_{fringing}$ é a capacitância de perímetro entre os níveis 1 e 2, dada em $fF/\mu m$. Note que agora o efeito *fringing* ocorre nas duas direções, ou seja, do nível 1 para o nível 2, e vice-versa, de modo que os perímetros W_1 e W_2 são somados na equação 5.3.

Logo, a capacitância entre níveis é dada por

$$C_{cross} = C_{area} \cdot W_1 \cdot W_2 + 2 \cdot C_{fringing} \cdot (W_1 + W_2) \quad (5.4)$$

Foram realizadas comparações entre os resultados obtidos através da equação 5.4 e simulações da estrutura da figura 5.12 no FastCap. As simulações consideraram o cruzamento de duas conexões de largura mínima (W_1 e W_2), submetidas a dois planos terra, reproduzindo as condições em que são encontradas num leiaute. O tamanho das conexões e dos planos terra foram normalizados em relação a mínima largura da camada de metal1 (W_{min}), e todas as simulações e cálculos através da equação 5.4 foram realizadas com parâmetros referentes a tecnologia $0,25\mu m$.

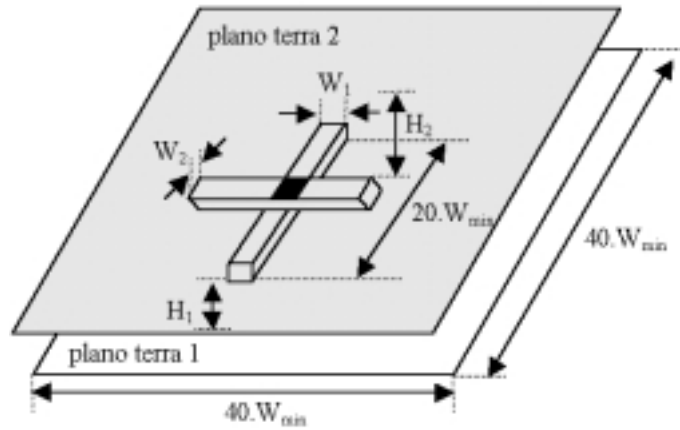


FIGURA 5.12 – Estrutura para simulação da capacitância entre níveis no FastCap

Na tabela 5.4 são mostrados os resultados comparativos. Mesmo apresentando um erro de 10% em relação a uma simulação 3D (no caso do cruzamento de metal1 e metal2), através da simples equação baseada na área e perímetro, consegue-se uma rápida avaliação da capacitância entre níveis.

Os dados necessários para o cálculo da capacitância entre níveis devem ser fornecidos através do arquivo de tecnologia pela diretiva “#CAPACITANCE CROSSOVER < layer > < n1 > < n2 >”, onde < layer > especifica a camada (i), < n1 > e < n2 > são respectivamente a capacitância de área e de perímetro em relação à camada superior ($i + 1$). Note que a ordem das camadas é especificada através da diretiva “#CAPACITANCE ORDER”.

TABELA 5.4 – Resultados comparativos entre avaliações de capacitâncias entre níveis

	poly - metal1	metal1 - metal2	metal2 - metal3
FastCap	0,0658	0,1168	0,1386
Equação 5.4	0,0649	0,1046	0,1283
Erro (%)	-1,36	-10,44	-7,43

6 Extração de Resistências

O valor da resistência de uma região é dependente de como a corrente flui através desta região, e as direções do fluxo de corrente são determinadas pela forma da região, e pela forma e localização dos terminais de contato. A resistência, em geral, pode ser obtida numericamente resolvendo a equação de Laplace $\nabla^2\psi = 0$ em duas ou três dimensões, com as condições de contorno derivadas das formas da região e dos contatos. Entretanto, esta técnica apresenta um alto custo computacional quando utilizadas em circuitos maiores.

Geralmente as geometrias encontradas em leiautes são suficientemente regulares, de forma que a equação de Laplace para duas dimensões pode ser resolvida para gerar expressões analíticas. Então, o problema de encontrar o valor da resistência, se reduz a:

- particionar uma região em componentes regulares, onde as expressões analíticas possam ser utilizadas
- determinar a direção do fluxo de corrente nestas componentes
- somar a resistência ao longo da direção do fluxo de corrente
- adicionar a contribuição das regiões de contatos e/ou vias

Muitos programas são baseados no conhecimento do fluxo de corrente [HOR 83][CHI 89], de modo que a resistência de uma região retangular é simplesmente calculada pela razão entre o comprimento e largura, e pela resistividade do material. Então, nestes métodos, o programa estima a direção do fluxo de corrente por vários métodos, como o algoritmo de Lee, e então divide a região em retângulos. Os valores individuais da resistência dos retângulos são somados ao longo do fluxo de corrente. Em muitos casos o fluxo de corrente através de curvas, regiões de contatos, e outros efeitos são considerados para melhorar a precisão. Entretanto, estes métodos são restritos a algumas formas de regiões, e pode ser formada uma biblioteca de formas padrões com suas resistências conhecidas.

Por outro lado, existem métodos mais gerais, que não ficam limitados a um conjunto de formas fixas. A maioria destes métodos calculam a resistência por análise numérica de equações físicas. Estes métodos numéricos incluem o método da diferença finita, o método dos elementos finitos (FEM), e a transformação de Schawartz-Cristoffel [CHA 70]. Dentre estes métodos, o FEM é o melhor método para resolver equações diferenciais e tem sido adotado por muitos programas [BAR 85][MIT 87], inclusive pelo Extribo [STE 89], que já foi desenvolvido na UFRGS.

O método FEM consiste na substituição dos parâmetros distribuídos por elementos discretos finitos de comportamento aproximadamente equivalente. No caso, o parâmetro distribuído é a resistividade do condutor, que é modelada por uma rede de resistores discretos. Esta rede de resistores é representada através de uma matriz de admitâncias, cuja dimensão é igual ao número de nós da rede. A principal desvantagem deste método é que esta matriz pode atingir dimensões muito grandes, exigindo grande capacidade computacional, e portanto deve ser restringida a pequenas regiões do circuito.

No extrator desenvolvido optou-se por utilizar um método mais simples, o cálculo da resistência através do número de quadrados, que apesar de ser menos preciso, obtêm-se uma maior velocidade de extração. Na seção seguinte é descrito em detalhes este método e as simplificações efetuadas.

6.1 Modelamento das Resistências

Em circuitos VLSI, há dois tipos de mecanismos físicos que causam a resistência elétrica. Uma é a resistência de contato, que ocorre na interface entre dois diferentes materiais, por exemplo, metal e silício. O outro é devido à própria resistência do material no qual é percorrido por um fluxo de corrente. Na figura 6.1 temos uma vista das conexões com seu respectivo modelamento. Nota-se que existem contatos empilhados, sendo representados pelas resistências de contato em série, e uma capacitância de acoplamento entre as duas conexões vizinhas.

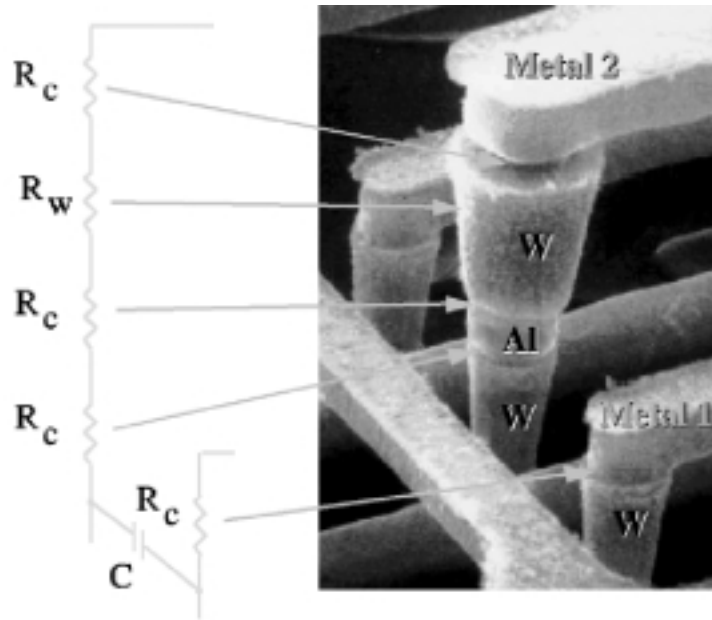


FIGURA 6.1 – Modelamento das Interconexões

A resistência de um pedaço uniforme de material condutor pode ser dada por

$$R = \frac{\rho}{T} \frac{L}{W}$$

onde ρ é a resistividade do material, T é a espessura, L é o comprimento e W é a largura do material. Dado que T é constante para cada camada de material condutor, dentro de uma mesma tecnologia, então a equação da resistência pode ser reescrita:

$$R = R_{\square} \frac{L}{W} \quad (6.1)$$

onde R_{\square} é a resistência por quadrado, dada em Ω/\square . Desta forma, para obter-se a resistência de uma conexão, simplesmente multiplicamos o comprimento da conexão pela resistência por quadrado e dividimos pela largura.

Então, a resistência pode ser relacionada a um fator puramente geométrico e a uma constante dependente da tecnologia. Para uma região retangular, o fator geométrico é o número de quadrados entre dois contatos, ou a razão comprimento/largura.

Assim, devemos reduzir uma rede a um conjunto de regiões retangulares, para aplicarmos a equação 6.1. Para realizar este particionamento, o extrator assume algumas simplificações:

- não considera-se geometrias irregulares, ou seja, somente linhas ortogonais são permitidas (geometria Manhattan)
- os contatos ou vias possuem o tamanho mínimo especificado pela tecnologia
- o particionamento de regiões mais complexas, como uma curva (figura 6.2), não é considerado o distúrbio na corrente, tratando como regiões retangulares separadas

O cálculo da resistência de contato e/ou vias é feito simplesmente multiplicando o número de contatos e/ou vias de uma rede pela resistência fornecida no arquivo de tecnologia, de modo que esta resistência está relacionada ao mínimo tamanho destes elementos.

A figura 6.2 mostra a principal propriedade do fluxo de corrente nos polígonos. Quando um polígono tem uma curva, ocorre um distúrbio na corrente somente nesta curva. Distanciando-se desta curva, o efeito é muito pequeno. Outras situações, como uma abrupta mudança na largura de uma conexão ou uma junção T, também podem causar um distúrbio semelhante. Embora estes distúrbios influenciam na resistência desta região, eles não são considerados pelo extrator desenvolvido, particionando em regiões retangulares distintas.

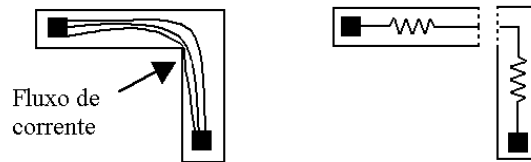


FIGURA 6.2 – Particionamento de um curva para o cálculo da resistência

O fator característico da tecnologia, R_{\square} , deve ser fornecido via arquivo de tecnologia pela diretiva “#RESISTANCE < layer > < n1 >”, onde *layer* é a camada, e *n1* é a resistência por quadrado, dado em Ω/\square .

6.2 Renomeação dos nodos

Para realizar a extração de resistências são necessários três arquivos de entrada, conforme a figura 6.3, os quais são:

- Arquivo de leiaute - é o principal arquivo, contendo a descrição do leiaute através de polígonos. A descrição das máscaras do circuito deve ser feita na linguagem CIF, e somente a geometria Manhattan é válida.
- Arquivo dos dispositivos - este arquivo deve conter a relação dos transistores, já que esta extração não é realizada pelo extrator de conectividades desenvolvido. Este arquivo somente é necessário para realizar a extração das resistências.
- Arquivo de saídas de células - este arquivo deve conter a relação dos nodos que são saídas de células. É definido como uma saída de célula, o encontro dos planos p e n. Somente é necessário para o caso de extração de resistências.

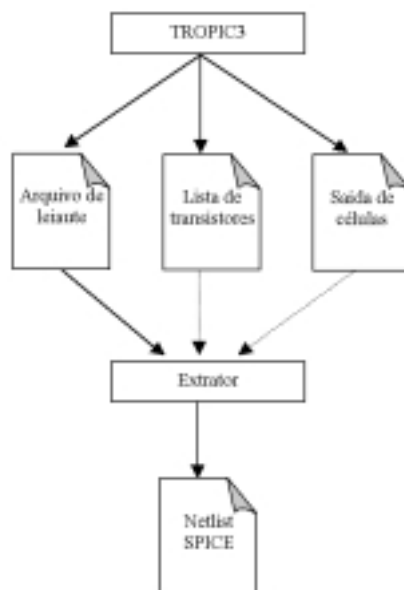


FIGURA 6.3 – Fluxo para extração de resistências

Os arquivos contendo a relação dos dispositivos e das saídas das células são necessários porque ao extrairmos as resistências do circuito, as redes são particionadas e novos nodos devem ser criados para acomodar as resistências. Desta forma, devemos renomear os nodos dos transistores, logo estes arquivos são necessários para manter a compatibilidade entre os nodos dos elementos parasitas extraídos e os nodos dos transistores. Note pela figura 6.3 que estes dois arquivos são gerados pelo TROPIC [MOR 99] juntamente com o leiaute, disponibilizando um ambiente integrado para geração e extração de leiaute.

Atualmente, o extrator LASCA está habilitado a extrair resistência somente a partir de circuitos gerados pelo TROPIC [MOR 99], porque não é realizado a extração de dispositivos. No entanto, a extração de capacitâncias pode ser realizada a partir do leiaute de qualquer circuito.

Podem ocorrer três situações em que devemos analisar a necessidade de renomeação dos nodos dos transistores:

1. Redes de saídas de células - são redes que conectam duas células entre si, ou a saída de uma célula a uma saída primária do circuito. Neste caso, devemos renomear o dreno e/ou fonte dos transistores de saída da célula. Note que deve ser informado ao extrator os nodos do circuito que são saídas de células (ver anexo 1).
2. Redes de entradas primárias - são redes que conectam as entradas do circuito com as entradas de células. Como uma entrada primária sempre está conectada ao *gate* de um transistor, então renomeamos todos os *gates* que não pertencem à lista de saídas de células.
3. Redes internas às células - estas redes não são renomeadas porque não são modeladas por resistências, já que geralmente são redes pequenas, causando um erro desprezível ao negligenciá-las.

Note a importância dos arquivos que devem ser fornecidos para a extração de resistências. O arquivo que contém a relação dos transistores indica quais nodos **podem** ser renomeados, e o arquivo que contém a relação dos nodos de saída de células, fornece informações para decidir quais nodos **devem** ser renomeados.

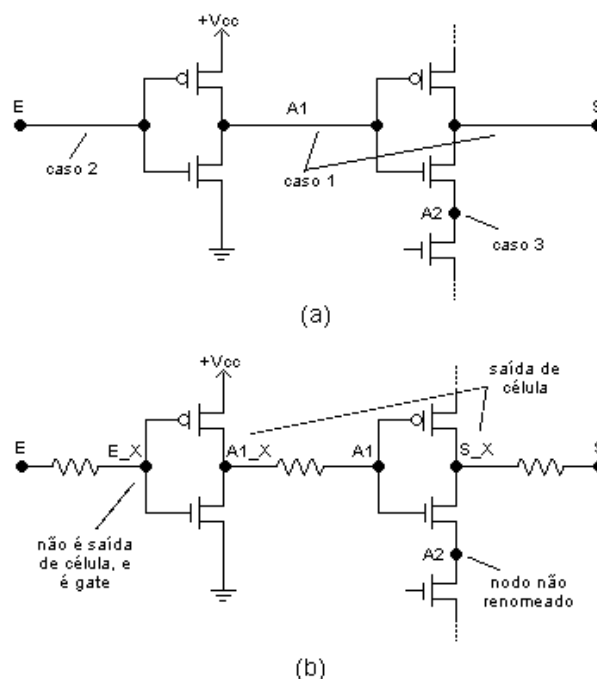


FIGURA 6.4 – Exemplo de renomeação de nodos

Na figura 6.4a temos um exemplo de duas células conectadas entre si, e os três casos assinalados. E na figura 6.4b temos o modelamento das redes através de resistências, sendo eliminadas as capacitâncias para melhor visualização. Estas resistências podem representar um modelo π ou L *lumped*, porém para o caso do modelo T *lumped* seria adicionada mais uma resistência a cada rede, e conseqüentemente, mais um nodo.

6.3 Consideração das Capacitâncias de Acoplamento nos Modelos RC

Ao incluirmos os efeitos resistivos das conexões, com uma segmentação da rede resistiva em mais de uma parte, esta rede que equivalia a um único nodo, passa a conter vários nodos interligados pelas resistências. Com isto, o modelamento das capacitâncias de acoplamento também é alterado. Apesar de não haverem muitas referências na bibliografia do melhor método para modelar estas capacitâncias de acoplamento, elas são representadas pelo extrator conforme [CAD 98] e [GEN 92]. Os modelos L e π são modelados igualmente, como mostra a figura 6.5, enquanto que a distribuição das capacitâncias de acoplamento do modelo T é mostrada na figura 6.6.

Foi realizada uma comparação do desempenho dos circuitos extraídos e dos tempos de simulação e extração para o circuito RIP (16 bits) entre os diversos modelos que podem ser extraídos, sendo mostrados na tabela 6.1.

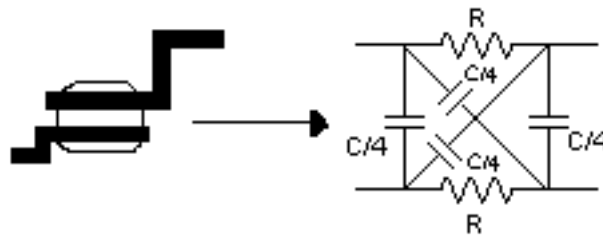


FIGURA 6.5 – Distribuição da capacitância de acoplamento nos modelos L e π *lumped*

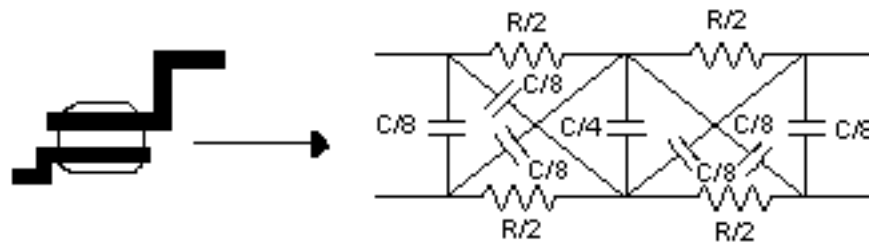


FIGURA 6.6 – Distribuição da capacitância de acoplamento no modelo T *lumped*

Para o caso em que extraímos apenas as capacitâncias, tomamos o extrator comercial Diva como referência (resultados apresentados na sétima coluna). Na segunda coluna temos os resultados obtidos pelo extrator original, EXT [MOR 99], onde são consideradas apenas as capacitâncias intrínsecas (C_{ground}). Nota-se que este caso gera o atraso mais pessimista, apresentando uma diferença no atraso do circuito extraído de 12,59%. Enquanto que ao considerarmos as capacitâncias de acoplamento e intrínseca (terceira coluna) temos um erro de apenas 1,81%. Entretanto, o número de capacitâncias aumenta de um fator de 5,2 e conseqüentemente, os tempos de extração e simulação também são maiores. Deste modo, temos um caso em que são extraídos apenas as capacitâncias ao plano terra, apresentando um erro maior, mas com menor número de elementos extraídos, podendo ser uma primeira aproximação do atraso do circuito. E outro caso em que foram extraídos as capacitâncias de acoplamento e ao plano terra, onde a extração efetuada pelo LASCA ainda é cerca de 25 vezes mais rápido do que o Diva, e com melhor precisão do que o caso anterior.

Ao modelarmos o circuito como RC *lumped*, temos uma equivalência entre os modelos π e T em termos de atraso, mas o modelo L gera uma sobreestimação do atraso do circuito, assim como descrito no capítulo 3. Os tempos de simulação são muito maiores quando é adotado um modelo que considere capacitâncias e resistências, o que é evidente devido ao tempo de simulação ser proporcional ao número de nodos do circuito.

O número de capacitâncias dos modelos L ou π aumenta por um fator de 13 em relação ao modelo C_{ground} , enquanto que para o modelo T o fator de aumento é de 20. Isto acontece no exemplo de um circuito pequeno com 9066 polígonos e 528 transistores, revelando que o *netlist* gerado pode facilmente “explodir” para circuitos maiores, sendo inútil para posteriores simulações. Com isto o usuário deve estar atento a relação precisão-número de nodos-tempo de simulação para que consiga o melhor desempenho da extração e simulação.

TABELA 6.1 – Resultados comparativos do modelamento do circuito CLA (16bits)

	EXT	LASCA				DIVA
	C_{ground}^1	$C\ lumped^2$	$L\ lumped$	$\pi\ lumped$	$T\ lumped$	
Nº resistores	0	0	80	80	160	0
Nº capacitores	162	853	2009	2089	3165	1378
Tempo de Extração ³ (ms)	1841	1749	1997	2032	2124	37900
Tempo de Simulação ⁴ (min)	2:28	2:57	8:30	8:27	9:24	10:08
Atraso (ns)	3,3713	3,7872	4,2228	4,1815	4,1793	3,8571

1 Foi considerado apenas as capacitâncias intrínsecas

2 Foram consideradas as capacitâncias intrínseca e de acoplamento

3 Simulações realizadas numa SUN Ultra Sparc 10

4 Simulações realizadas no Hspice [MET 92]

Na figura 6.7 temos um exemplo de um cruzamento de duas conexões com o seu respectivo modelamento. Note que foi utilizado o modelo $\pi\ lumped$, e que a extração de um simples *crossover* gerou 10 elementos parasitas.

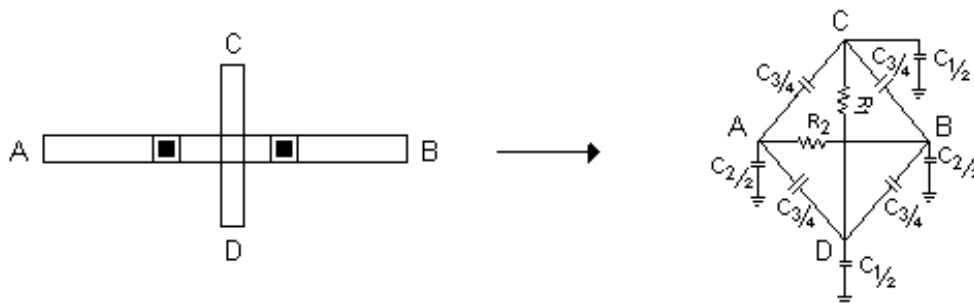


FIGURA 6.7 – Cruzamento de duas interconexões e o modelo extraído

7 Resultados

A avaliação do extrator LASCA foi efetuado segundo os seguintes critérios:

- **Tempo de extração** - visa avaliar o tempo de CPU consumido durante a extração.
- **Precisão da extração de circuitos** - visa avaliar o grau de precisão dos elementos parasitas extraídos.
- **Taxa de distribuição dos polígonos** - visa avaliar o desempenho do algoritmo *bin-based* implementado em relação a distribuição dos polígonos.
- **Curvas de Simulação** - visa comparar a resposta a um salto de tensão para os vários modelos extraídos.

Todos os leiautes dos circuitos utilizados para avaliar o extrator foram gerados através do TROPIC [MOR 99], utilizando tecnologia $0,25\mu m$ CMOS, com três níveis de metais e com contatos empilhados, e com transistores com $W=2\mu m$ e $L=0,25\mu m$.

7.1 Tempo de Extração

Na tabela 7.1 temos os tempos de extração para circuitos aritméticos (primeiro bloco) e *benchmarks* ISCAS85 (segundo bloco), numa SUN Ultra Sparc 10.

TABELA 7.1 – Comparação do tempo de CPU para extração

Circuito	xtors	polígonos	EXT	LASCA (ms)				DIVA (min)
			C_{int}	C	L	π	T	
Adder	28	449	689	716	807	811	811	0:12
Addergate	40	635	620	670	851	894	869	0:13
Alu	260	906	906	1056	1322	1434	1550	0:24
Alugate	432	6503	1113	1388	1620	1641	1633	0:40
Rip16	448	7270	1210	1551	1833	1816	2007	0:41
CLA16	528	9066	1841	1749	1997	2032	2124	0:46
Rip32	896	15503	1691	2455	3011	3008	3025	1:18
CLA32	1056	19051	2054	3001	3295	3354	3728	1:41
C17	24	386	752	767	877	887	803	0:12
C432	150	2564	928	1036	1150	1177	1160	0:20
C499	1556	26592	2904	4579	4710	4732	5013	2:26
C880	1750	30790	3207	4897	5320	5339	5655	1:44
C1355	2244	38841	4005	5824	6835	6851	7772	2:12
C1908	3146	54471	4863	7850	9050	9085	9397	3:07
C3540	7154	129918	10886	19542	22530	22824	23830	8:01
C5315	10656	196695	16427	30465	40090	40136	41175	14:06
C6288	10112	155885	12906	21852	26580	26604	27334	9:55
C7552	14376	265563	23791	42506	55130	55238	58685	19:20

A tabela está organizada, como segue: na três primeiras colunas temos o nome dos circuitos, o número de transistores e o número de polígonos, respectivamente. Na quarta coluna temos os tempos de extração para o caso de considerarmos apenas as capacitâncias intrínsecas (C_{int}). Na quinta coluna foram consideradas as capacitâncias de acoplamento e as capacitâncias intrínsecas (modelo C). Na três colunas seguintes, temos a extração RC, utilizando os modelos L, π e T, respectivamente. E na última coluna, temos os tempos de extração através do Diva. Todos os tempos são dados em milisegundos, salvo o caso do Diva, onde estão em minutos e segundos.

Comparando os tempos de extração através do Diva e do LASCA (modelo C), notamos que o LASCA é em média 25 vezes mais rápido do que o Diva. Para um circuito contendo 256563 polígonos, todas as capacitâncias são extraídas pelo LASCA em menos de 45 segundos, enquanto que o extrator Diva leva 19 minutos.

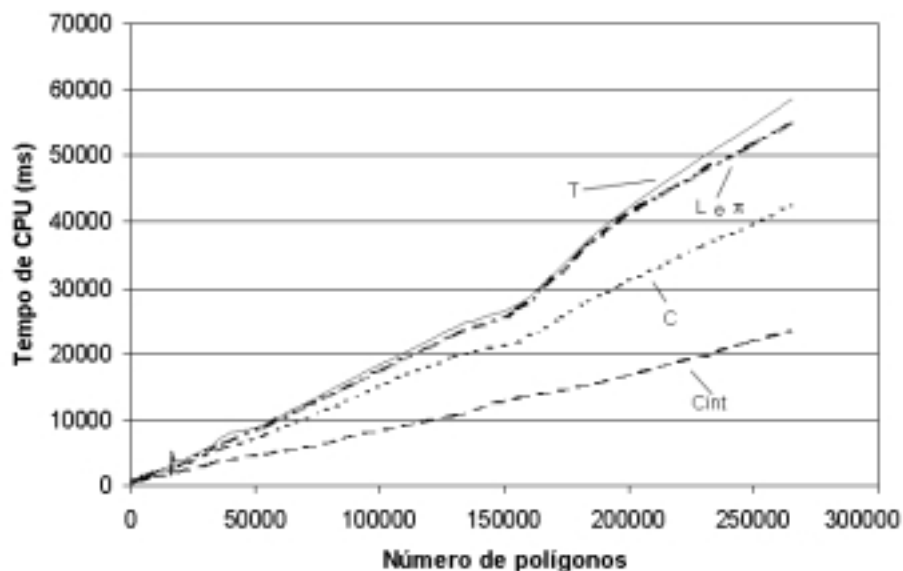


FIGURA 7.1 – Comparação do tempo de CPU, numa SUN Ultra Sparc 10

Na figura 7.1 foram plotados os resultados da tabela 7.1 para os circuitos extraídos através do LASCA.

Entre os modelos L, π e T não há grandes diferenças nos tempos de extração, entretanto, o tempo de extração para os circuitos modelados como T *lumped* é pouco maior, principalmente quando os circuitos são maiores. Isto ocorre porque existe um número maior de elementos parasitas extraídos.

Comparando-se os modelos de capacitância (C_{int} e C), temos um aumento médio no tempo de extração do modelo C de 50% em relação ao modelo C_{int} , devido a função de encontrar todos os vizinhos das conexões, que para circuitos muito densos e/ou maiores, pode provocar um aumento desta relação.

As funções de cálculo das resistências e renomeação de nodos, presentes na extração dos modelos RC, foram otimizadas e apresentaram um aumento médio de apenas 20% no tempo de extração, em relação ao modelo C.

Deste modo, ao incluirmos novas funções ao extrator original, EXT [MOR 99], representado pela quarta coluna na tabela 7.1, tornamos o extrator LASCA mais adequado a tratar dos atuais efeitos das tecnologias submicrônicas, sem exonerar o tempo de extração.

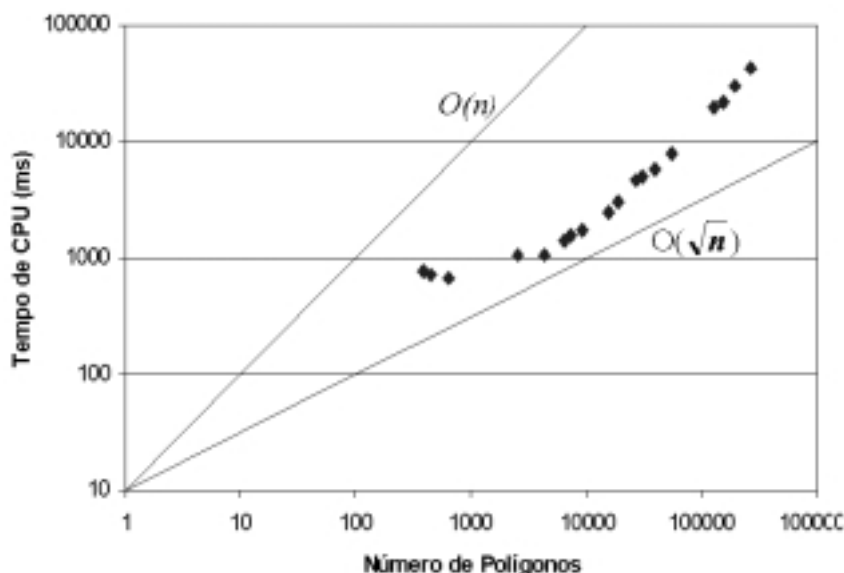


FIGURA 7.2 – Tempo de extração para o modelo C

Na figura 7.2 temos uma análise da complexidade do tempo de extração para o caso em que são consideradas as capacitâncias de acoplamento e intrínsecas. As linhas representam uma complexidade $O(n)$ e $O(\sqrt{n})$ de 100 *polígonos/segundo*. A principal vantagem do algoritmo *bin-based* implementado está no fato de que podemos rapidamente analisar as conectividades de uma grande quantidade de polígonos, e assim estimar as capacitâncias do circuito com uma boa precisão.

Entretanto, uma maior velocidade de extração requer um maior consumo de memória do que outros algoritmos como o *scanline*, porque todo o circuito é armazenado na memória.

Deste modo, temos uma complexidade de memória de $O(bn)$, e uma complexidade de tempo de CPU de $O(\sqrt{n})$, onde b é o número de *bins* e n é o número de polígonos.

7.2 Precisão da extração de circuitos

Para avaliar qualitativamente os valores das capacitâncias e resistências extraídas, os circuitos após serem extraídos foram simulados através do simulador elétrico HSPICE [MET 92] e foi tomado o atraso como fator de comparação.

Na tabela 7.2 temos a comparação do atraso para os circuitos aritméticos, no qual foram extraídas as capacitâncias através do LASCA e do Diva. Para todos os circuitos foi utilizada uma capacitância de carga de 35 fF, equivalente a um *fanout* igual a 5 para a tecnologia 0,25 μm . As simulações foram realizadas com vetores de entradas que sensibilizassem o caminho crítico dos circuitos.

Nas duas primeiras colunas temos os nomes dos circuitos e o número de transistores. Na terceira, quarta e quinta colunas é feita uma comparação do número de capacitâncias extraídas por cada método. E nas três colunas seguintes, temos os resultados das simulações elétricas. Enquanto que nas duas últimas colunas temos o erro relativo ao Diva.

TABELA 7.2 – Comparação do atraso de circuitos extraídos pelo LASCA e Diva

Circuito	xtors	Nº de capacitâncias			Atraso (ns)			Erro (%) ¹	
		LASCA		DIVA	LASCA		DIVA	C_{int}	C
		C_{int}	C		C_{int}	C			
Adder	28	12	33	61	0,3624	0,3847	0,3678	-1,47	+4,59
Addergate	40	14	57	79	0,4042	0,4304	0,4247	-4,83	+1,34
Alu	260	93	510	725	0,985	1,1167	1,1056	-10,9	+1
Alugate	432	116	764	1071	1,0984	1,2401	1,2208	-10,03	-0,88
Rip16	448	162	853	1377	3,3713	3,7872	3,8571	-12,59	-1,81
CLA16	528	214	983	1453	2,5539	2,8737	2,6234	-2,65	+9,54
Rip32	896	322	2063	3097	7,0158	8,2248	8,2730	-15,19	-0,58
CLA32	1056	426	2753	4012	5,3084	6,2971	5,7966	-8,42	+8,63

¹ Erro relativo ao DIVA

Nota-se que ao considerarmos apenas as capacitâncias intrínsecas (sexta coluna) temos um erro relativo médio de 8,25% para estes circuitos (o pior caso, 15%), gerando uma estimativa pessimista do atraso. E no caso de considerarmos as capacitâncias de acoplamento juntamente (sétima coluna), o erro médio diminui para apenas 3,63%. Isto evidencia a importância das capacitâncias de acoplamento para circuitos submicrônicos.

Na figura 7.3 temos os resultados destas simulações dos circuitos plotados num gráfico. Na tabela 7.2 são mostrados os resultados das simulações dos circuitos extraídos, onde as interconexões foram modeladas como RC.

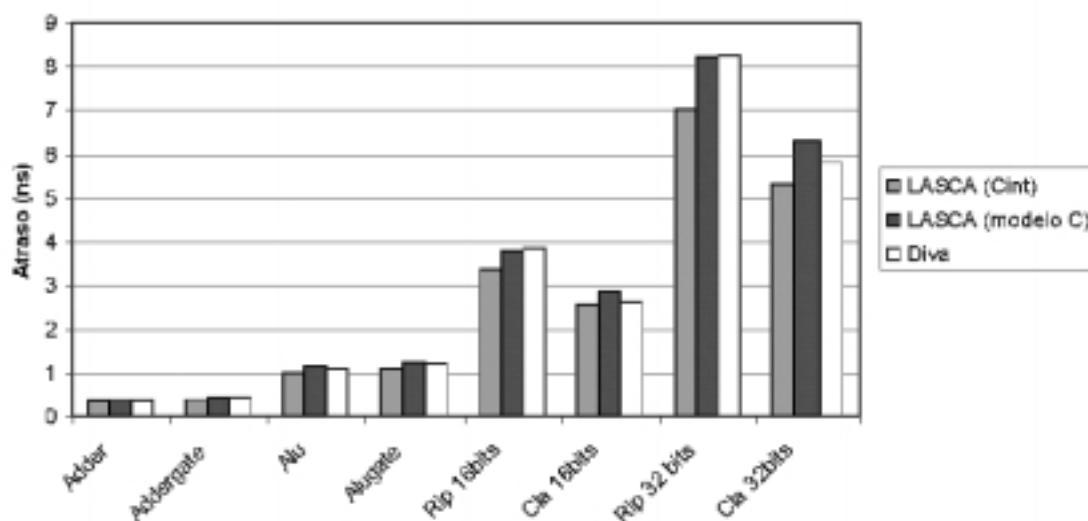


FIGURA 7.3 – Comparação do atraso para circuitos extraídos pelo LASCA e Diva

Foi realizada uma comparação do número de elementos extraídos em cada modelo (L, π e T), das resistências na segunda e terceira coluna, e das capacitâncias na quarta, quinta e sexta coluna. Note que a segunda coluna equivale para ambos os modelos, L e π ,

TABELA 7.3 – Comparação do atraso para extração RC realizada pelo LASCA

Circuitos	Nº de Resistências		Nº de capacitâncias			Atraso (ns)		
	L ou π	T	L	π	T	L	π	T
Adder	5	10	65	70	97	0,4451	0,4390	0,4410
Addergate	12	24	159	171	261	0,4983	0,4919	0,4922
Alu	50	100	1388	1438	2266	1,2755	1,2660	1,2622
Alugate	112	224	2457	2569	4150	1,4153	1,4003	1,3985
Rip16	80	160	2009	2089	3165	4,2228	4,1815	4,1793
CLA16	100	200	2345	2445	3707	3,3836	3,3278	3,3371
Rip32	160	320	5188	5348	8313	9,4074	9,3142	9,2680
CLA32	200	400	7291	7491	11829	7,4475	7,3089	7,3013

já que o número de resistências é igual.

Analisando estes resultados, vemos que os modelos π e T possuem uma resposta muito próxima, enquanto que o modelo L apresenta uma sobreestimativa do atraso, assim como descrito na seção 3.1. Também nota-se a enorme diferença no número de capacitâncias do modelo T em relação aos outros dois modelos. Na figura 7.4 foram plotados os resultados das simulações de atraso dos circuitos aritméticos.

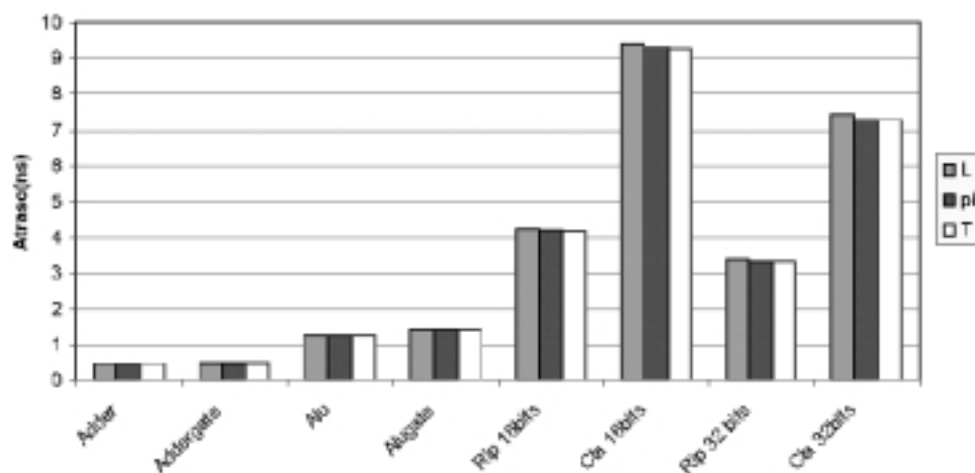


FIGURA 7.4 – Comparação do atraso para circuitos extraídos pelo LASCA

Não foram realizadas comparações com o extrator Diva, porque não tínhamos disponível um arquivo de tecnologia para a extração das resistências.

7.3 Taxa de distribuição dos polígonos

A taxa de distribuição dos polígonos visa avaliar as alterações referentes à adaptação do tamanho da grade virtual ao tamanho do circuito feitas no algoritmo *bin-based*.

Circuit name: C17						Circuit name: C432					
P	0	6.3	%	[4]		P	0	4.5	%	[19]	
P	1-5	9.5	%	[6]		P	1-5	1.2	%	[50]	
P	6-10	20.6	%	[13]		P	6-10	24.0	%	[101]	
P	11-15	31.7	%	[20]		P	11-15	33.6	%	[141]	
P	>16	31.7	%	[20]		P	>16	25.9	%	[109]	
		100	%	[63]		P		100	%	[420]	
Circuit name: C1355						Circuit name: C1908					
P	0	1.3	%	[101]		P	0	2.3	%	[270]	
P	1-5	15.9	%	[1262]		P	1-5	18.8	%	[2244]	
P	6-10	26.1	%	[2068]		P	6-10	29.4	%	[3506]	
P	11-15	28.2	%	[2231]		P	11-15	28.8	%	[3429]	
P	>16	28.6	%	[2269]		P	>16	20.7	%	[2469]	
		100	%	[7931]		P		100	%	[11918]	
Circuit name: C3540						Circuit name: C7552					
P	0	3.3	%	[1072]		P	0	3.0	%	[2290]	
P	1-5	25.3	%	[8300]		P	1-5	30.2	%	[23058]	
P	6-10	28.2	%	[9223]		P	6-10	26.9	%	[20544]	
P	11-15	25.2	%	[8239]		P	11-15	23.4	%	[17897]	
P	>16	18.1	%	[5923]		P	>16	16.8	%	[12856]	
		100	%	[32757]		P		100	%	[76375]	

Acima foram apresentados alguns relatórios da taxa distribuição dos polígonos para circuitos ISCAS85.

A emissão destes relatórios e do histograma da distribuição dos polígonos são disponibilizados pelo extrator LASCA.

Uma rápida análise destes relatórios, informa a existência de uma pequena quantidade no número de *bins* vazios, em média 3 a 4% do total dos *bins*. Isto confirma a eficiência das alterações implementadas, já que o extrator EXT [MOR 99] apresentava mais de 50% dos *bins* vazios para os circuitos analisados.

Outra análise importante é referente a escolha do tamanho dos *bins* ($2\mu m \times 2\mu m$). Nos exemplos analisados tivemos 80% dos *bins* em média, possuindo até 16 polígonos. Desta forma, temos pequenas listas encadeadas, e conseqüentemente, o tempo de pesquisa nestas listas é pequeno, garantindo um bom desempenho do algoritmo.

7.4 Curvas de Simulação

A seguir apresentamos as curvas de simulação do circuito Rip16 (*ripple carry additonneur 16 bits*), utilizando uma capacitância de carga de $35fF$, equivalente a um *fanout* igual a 5 para transistores com $W=2\mu m$ e $L=0,25\mu m$. Na figura 7.5 temos a resposta a um salto de tensão para o circuito modelado através dos cinco modelos disponíveis: C_{int} , C, L, π e T.

Nesta ampliação das curvas de simulação, podemos identificar o tempo de descida do circuito Rip16, conforme a tabela 7.4. E na figura 7.6 temos a ampliação das curvas de simulação dos modelos RC.

Como visto anteriormente, o modelo C_{int} apresenta uma estimativa pessimista do atraso do circuito, não sendo suficiente para circuitos submicrônicos. Mas ao considerarmos as capacitâncias de acoplamento juntamente com as capacitâncias intrínsecas, temos uma estimativa mais realista do atraso do circuito.

Comparando as curvas de simulação para o caso em que as interconexões são

modeladas como circuitos RC, podemos notar que o modelo L apresenta o maior tempo de descida, enquanto que os modelos π e T apresentam tempos similares, assim como descrito em [SAK 83a]. Entretanto, o erro do modelo L em relação aos modelos π e T é de apenas 1%.

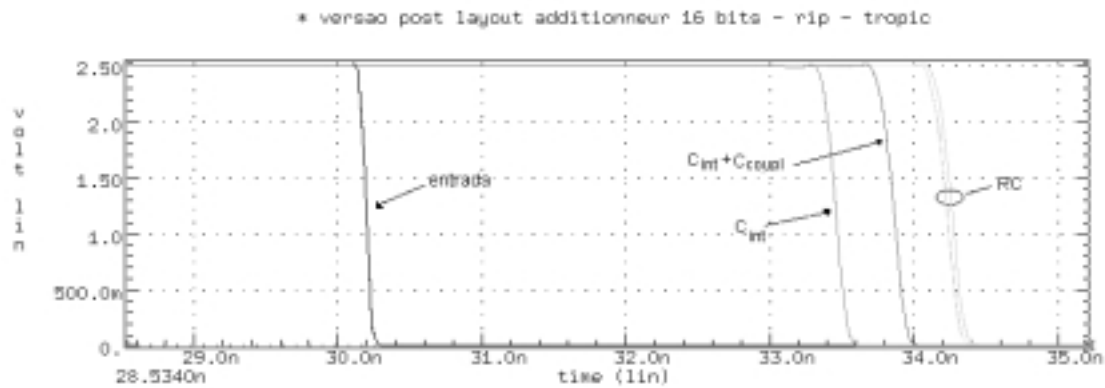


FIGURA 7.5 – Ampliação da simulação do circuito Rip16

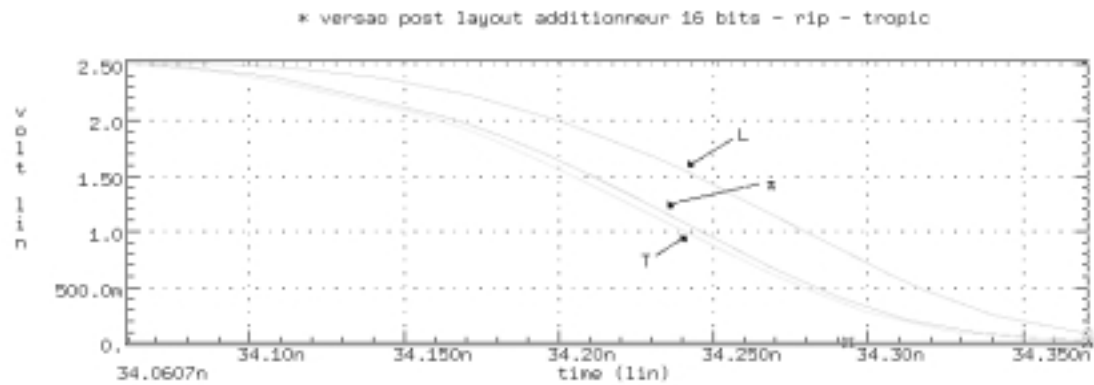


FIGURA 7.6 – Simulação dos modelos RC para o circuito Rip16

TABELA 7.4 – Tempo de descida para o circuito Rip16

	C_{int}	C	L	π	T
Rip16	3,3713	3,7872	4,2228	4,1815	4,1793

8 Conclusões

Este trabalho trata do desenvolvimento de uma ferramenta para extrair rapidamente as resistências e capacitâncias de acoplamento e as capacitâncias intrínsecas a partir da descrição do leiaute de um circuito integrado, com uma aceitável precisão. Este extrator está integrado a um gerador automático de leiaute, TROPIC, resultando num ambiente integrado para uma rápida implementação de macro-células e posterior extração dos elementos parasitas.

Tomando-se como base o extrator de conexões, EXT, que avalia as capacitâncias parasitas em relação ao substrato, implementamos o extrator LASCA, adequado para tratar circuitos submicrônicos.

Primeiramente otimizamos o algoritmo de extração de conectividades, *bin-based*, fazendo com que o tamanho da grade imposta sobre o circuito fosse igual ao tamanho do circuito, ao invés de utilizarmos uma grade fixa. Com isto evitamos um desperdício de memória quando analisamos circuitos pequenos, e conseguimos um aumento do desempenho, principalmente para circuitos grandes. Enquanto que no algoritmo inicialmente implementado no extrator EXT, podíamos ter mais de 90% dos *bins* vazios dependendo do tamanho do circuito, passamos a ter apenas 3 a 4% de *bins* vazios em média. E aumento de desempenho foi comprovado com alguns circuitos, chegando a reduzir o tempo de extração em até 2/3.

A análise das capacitâncias de acoplamento foi realizada utilizando uma simples metodologia $2\frac{1}{2}D$, onde foram consideradas as interações mais relevantes: os vizinhos imediatos, *crossovers* e *crossunders*, tratando os níveis $i \pm 2$ como planos terra. Aliada a esta metodologia foram empregadas formulações empíricas para o cálculo das capacitâncias. Os valores obtidos através destas formulações foram confrontados com simulações 3D no FastCap, apresentando uma ótima concordância.

Com a finalidade de validar as capacitâncias extraídas, tomou-se alguns circuitos aritméticos para realizar uma simulação elétrica do atraso. Então foram feitas comparações em relação ao extrator comercial Diva. Há duas possibilidades de extração de capacitâncias no LASCA: extrair apenas as capacitâncias intrínsecas, ou extrair juntamente as capacitâncias de acoplamento. No caso em que foram extraídos somente as capacitâncias intrínsecas, tivemos um erro médio de 8,25%, e o pior caso foi de 15%. Mas ao considerarmos todas as capacitâncias, o erro diminuiu para 3,63%. Isto demonstra a importância em considerar as capacitâncias de acoplamento, principalmente para circuitos submicrônicos, já que elas contribuem enormemente para o atraso total do circuito.

A extração de resistências foi feita através da relação entre o comprimento e a largura das conexões, multiplicada pela resistência por quadrado. Embora seja um dos métodos mais simples para extração das resistências, temos um ganho no tempo de CPU ao realizar esta tarefa.

Foram disponibilizados três modelos RC: L, π e T *lumped* de 1 estágio. Assim o usuário pode definir qual o melhor modelo que representará as redes do circuito. Comparações entre os três modelos, indicaram que ao modelar as redes como L *lumped*, temos uma sobreestimação do atraso do circuito, enquanto que os modelos π e T apresentaram uma resposta muito semelhante.

No extrator estão disponíveis 5 modelos: C_{int} , C, L, π e T. À medida que o tamanho do circuito a ser extraído aumenta, devemos considerar o número de elementos extraídos, para que o *netlist* gerado seja útil para uma ferramenta de simulação ou de análise de

timing.

Nos modelos RC, o número de capacitâncias cresce muito rapidamente, já que as capacitâncias de acoplamento são divididas entre os nodos criados. Isto ocorre principalmente no modelo T, onde uma capacitância de acoplamento é substituída por 7 capacitâncias, enquanto que nos modelos L ou π é substituída por apenas 4 capacitâncias.

Deste modo, a escolha do modelamento das redes passa a ser um fator fundamental, onde o projetista deve considerar o compromisso entre velocidade de extração e precisão, juntamente com a finalidade do *netlist* gerado pelo extrator LASCA.

Anexo 1 Instruções de Uso do Programa

A.1 Especificação do diretório de trabalho

Para a extração de capacitâncias é necessários apenas o arquivo contendo o leiaute do circuito, enquanto que para o caso da extração de resistências, também será necessário os seguintes arquivos:

- **flat** - contém a relação dos dispositivos ativos (transistores e diodos)
- **cin** - contém a relação dos nodos que são saídas de células

No arquivo de tecnologia existem duas diretivas que fornecem os diretórios de trabalho. A diretiva “#dirIN” não é utilizada pelo extrator, somente pelo gerador de leiaute TROPIC [MOR 99]. Enquanto que a diretiva “#dirOUT” contém o diretório em que os arquivos **.cif**, **flat** e **cin** se encontram, conforme o exemplo a seguir.

Exemplo:

```
⇒ #dirOUT benches
arquivo cif: ~user/benchs/file.cif
arquivo flat: ~user/benchs/cxfile/flat
arquivo cin: ~user/benchs/cxfile/cin
```

Note que os arquivos flat e cin devem estar no subdiretório */cxfile*, onde *file* é o nome do arquivo de leiaute. A primeira vista, pode parecer estranho a existência das duas diretivas (#dirIN e #dirOUT) e a utilização de apenas uma (#dirOUT). Mas o arquivo de tecnologia é utilizado tanto pelo extrator como pelo gerador de leiaute TROPIC. Isto também explica a existência das regras geométricas, existentes neste arquivo, mas não necessárias para a etapa de extração.

De acordo com o fluxo da figura 6.3, quando o leiaute é gerado pelo TROPIC, a diretiva #dirIN estará apontando para o diretório em que se encontra o arquivo de descrição do circuito, e o leiaute gerado será colocado no diretório especificado em #dirOUT. Ao realizar a extração, os arquivos necessários serão buscados de #dirOUT e o *netlist* Spice gerado será colocado em #dirOUT. Isto permite uma maior integração entre as duas ferramentas. Podemos, inclusive, gerar o leiaute de um circuito e realizar a extração juntamente, a partir da chamada do TROPIC, sem a necessidade de executarmos separadamente ambas as ferramentas.

A.2 Chamada do extrator

A extração dos parasitas é feita executando o programa EXT5, onde obtemos:

```
use: lasca [options] <techfile><file>
OPTIONS:
  [g] plot a histogram of wire lengths
  [a] plot the algorithm efficiency histogram
  [l] create a CIF file for each net
  [c] extract only ground capacitances
```

```

[rl] extract resistance of nets (L lumped model)
[rp] pi lumped model
[rt] T lumped model
read a CIF layout file, extracting all wire lengths,
routing capacitances and resistances

```

Por exemplo, para o circuito CLA (*carry lookahead adder*), faríamos:

⇒ lasca garp t25f4 cla

```

graph mode
Resistance mode (pi lumped model)
** file found in: /home/pinot/fklein/bin/t25f4

```

```

Reading the CIF file
----->end of cif file : cla

```

Numero de buffers 46 - Poligonos 9066

```

Extracting the conectivities
Finding all neighbors to calculate coupling capacitances
Number of neighbors=8097

```

```

----- Lenght Wire Distribution -----
# P 0-100um 90.7% [195]
# P 101-200um 8.4% [18]
# P 201-300um 0.0% [0]
# P 301-400um 0.0% [0]
# P 401-500um 0.0% [0]
# P > 500um 0.9% [2]
# 100.0% [215/215]
----- Resistance Distribution -----
# P 0-2k 94.4% [203]
# P 2k-4k 5.6% [12]
# P 4k-6k 0.0% [0]
# P 6k-8k 0.0% [0]
# P 8k-10k 0.0% [0]
# P > 10k 0.0% [0]
# 100.0% [215/215]
----- Polygons Distribution (pol/bins)-----
# P 0 0.2% [45]
# P 1-5 1.3% [271]
# P 6-10 2.8% [560]
# P 11-15 2.7% [551]
# P >16 2.0% [397]
# 9.0% [1824/1824]

```

```

Number (estimated) of buffers:32
Width/Height(um) = 115.85 x 64.40

```

```

Size grade = 57 X 32
Capacitances to ground = 332
Coupling capacitances = 2565
Total number of capacitances = 2897
Total number of resistances = 119
#CPU time (ms) :  user 2010 system 100 [TOTAL 4971]

```

As opções “g” e “a” conduzem o extrator ao modo gráfico, onde são gerados os histogramas da distribuição do comprimento das redes (figura A.1a) e da taxa de ocupação dos *bins* (figura A.1b). Juntamente com o histogramas, são emitidos relatórios referentes ao tamanho das conexões (0-100 μm , 101-200 μm ...) e do número de polígonos em cada *bin* (0, 1-5, 6-10 ...).

Através da opção “l” podemos gerar um arquivo CIF para cada rede do circuito. A opção “c” desabilita a extração das capacitâncias de acoplamento, extraíndo apenas as capacitâncias intrínsecas. Note que a utilização desta opção faz com que o extrator LASCA execute as mesmas funções do extrator EXT [MOR 99], o qual serviu de base para o desenvolvimento deste trabalho.

Para extrair as resistências, estão disponíveis as opções “rl”, “rp” e “rt”, que geram os modelos L, π e T *lumped*, respectivamente. Quando for realizada a extração das resistências, será emitido um relatório com o número de redes com resistências próximas (0-2k Ω , 2k Ω -4k Ω ,...).

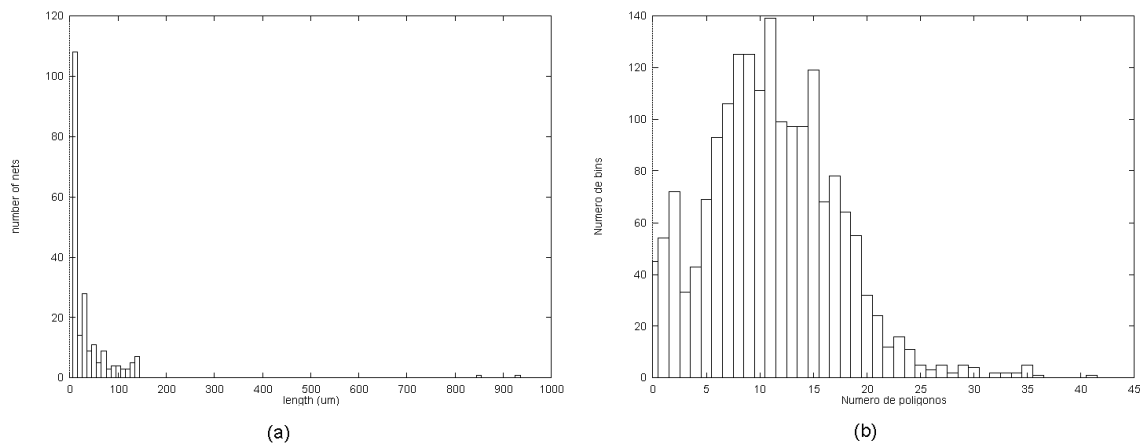


FIGURA A.1 – Histogramas (a) do comprimento das conexões (b) da taxa de ocupação dos *bins*

A.3 Arquivos de saída

O seguintes arquivos são gerados pelo extrator:

- *file.est* - este arquivo contém o *netlist* Spice gerado, com as capacitâncias, resistências e transistores. Estes dois últimos, somente se foi fornecido os arquivos *cin* e *flat*.
- */cxfile/wire.ps* - é o arquivo postscript do histograma do comprimento das conexões. Somente é gerado se for incluída a opção “g”.

- */cxfile/algo.ps* - é o arquivo postscript do histograma da taxa de ocupação dos *bins*. Somente é gerado se for incluída a opção “a”.
- */cxfile/res.ps* - é o arquivo postscript do histograma das resistências das redes. Somente é gerado se for incluída a opção “rx”, onde x pode ser l,p ou t.
- */cxfile/report* - este arquivo fornece um relatório contendo informações referentes a capacitâncias relativas, capacitâncias absolutas, dados topológicos de cada *gate* e o número estimado de *buffers*. Podemos dividir este relatório em cinco partes:

1. **Capacitâncias relativas a C_{in}** - nesta parte são descritos: o nome da rede (*net name*), o comprimento total da rede (*l_{total}*) e as capacitâncias relativas: C_{active}/C_{in} , C_{rot}/C_{in} , C_{dif}/C_{in} e C_{load}/C_{in} , onde $C_{load} = C_{active} + C_{rot} + C_{dif}$. A mensagem *buf* indica que relação C_{load}/C_{in} é maior do que o fator C_{limit} , definido no arquivo de tecnologia.

```
***
** Circuit Name:  alu
** Date:   Mon Feb 15 11:01:40 1999
**

--- RELATIVE CAPACITANCES-----
net name l_total act/Cin rot/Cin dif/Cin Cload/Cin
c4      11.8    0.15    0.56    0.71
equal   12.1    0.16    0.28    0.43
_4-X19   21.5    1.00    0.31    0.56    1.86
_4-X13   22.2    1.00    0.33    0.56    1.89
_4-X18   23.1    1.00    0.33    0.56    1.88
_4-X16   24.2    1.00    0.35    0.56    1.91
l3       80.9    3.00    1.10    1.25    5.35 buf
h3       86.4    5.00    1.27    1.25    7.52 buf
```

2. **Capacitâncias absolutas** - nesta parte são descritos: o nome da rede (*net name*), o comprimento total da rede (*l_{total}*), e os valores das capacitâncias C_{active} , C_{rot} e C_{dif} .

```
---CAPACITANCES-----
net name l_total Cact  Crot  Cdif  Cload  Cin  Cload/Cin
_1-X10    8.3          0.76    0.76
_5-X18    8.3          0.76    0.76
_5-X19    8.3          0.76    0.76
_1-X7     8.5          0.78    0.78
h2       189.6   48.27  8.53  8.64  75.44  6.90 10.94 buf
h0       197.8   48.27 19.50  8.64  76.41  6.90 11.08 buf
```

3. **Comprimento do roteamento** - nesta parte são descritos: o nome da rede (*net name*), o comprimento total da rede (*l_{total}*), o número de transistores conectados a rede (*tr(n,p)*), o comprimento das camadas de polisilício (*l_{poly}*), metal1 (*l_{m1}*), metal2 (*l_{m2}*), metal3 (*l_{m3}*), e o número de contatos, vias e vias2 da rede.

```

--- ROUTING -----
net name l_total tr(n,p) l_poly l_m1 l_m2 l_m3 ctos via1 via2
_l1-X10 8.3 0,0 0.0 5.2 3.1 0.0 4 2 0
_l5-X18 8.3 0,0 0.0 5.2 3.1 0.0 4 2 0
s3 94.4 4,4 32.8 5.4 7.2 49.0 10 8 12
a3 101.4 2,2 19.2 6.4 3.2 72.6 4 2 6
f2 135.9 1,1 11.7 11.4 9.4 103.4 9 4 12

```

4. **Topologia das células** - nesta parte são descritos: a função das células, número de entradas, número de pares de transistores conectados a entrada da célula (*fanin*), número de pares de transistores conectados na saída (*fanout*), dimensões médias da largura do canal dos transistores N e P (*avg(Wn)* e *avg(Wp)*), número de transistores em série nos planos N e P e o número de transistores conectados em série nos ramos N e P mais longos (*drainNP*).

```

--- drainN/drainP corresponds to the number of
--- drain/sources connected to the branch containing
--- the maximum number of serial transistors
--- (GENERATED BY TROPIC3)

```

```

net cell inputs fanin fanout avg(Wn) avg(Wp) serieNP drainNP
_l1 nand 4 1 1 2.00 2.00 4, 1 4, 4
_l3 nand 5 1 1 2.00 2.00 5, 1 5, 5
y aoi 7 1 1 2.00 2.00 4, 4 7, 7
x1 aoi 3 1 2 2.00 2.00 2, 2 3, 3

```

5. **Informações complementares** - esta parte contém informações do arquivo de tecnologia. Em negrito temos o número de vezes em que $C_{load}/C_{in} > C_{limit}$

Capacitances:

```

Cox: 6.895200
Cref: 1.206660
DiffN: Cjn:0.799200 Cjns:0.147900
DiffP: Cjp:0.818700 Cjps:0.151400
POLY: A:0.098700 P:0.044500
METAL1: A:0.024700 P:0.040800
METAL2: A:0.015000 P:0.038000
METAL3: A:0.008000 P:0.032800
Climate: 5.00

```

W source: 0.50

lmin:0.350000 wmin:0.250000

Widths: poly:0.250 metal:0.400 metal:0.600 metal:0.600

Number (estimated) of buffers:9

Anexo 2 Arquivo de Descrição da Tecnologia

O arquivo de descrição da tecnologia tem a função de tornar o uso do extrator independente do processo utilizado para a fabricação do circuito. Preferiu-se criar um arquivo de tecnologia com comandos específicos ao invés de utilizar-se algum arquivo relativo a um extrator comercial. Com isto consegue-se uma maior simplificação, menor número de operações e facilidade na criação do arquivo.

O arquivo de tecnologia é formado por uma série de diretivas iniciadas por “#”, que são interpretadas sequencialmente. Basicamente, ele pode ser dividido em três conjuntos de regras:

- regras geométricas (28 regras)- são regras específicas para a geração do leiaute, como a largura mínima, espaçamento e distância de contatos e vias para cada camada, distância e margem para o metal, regras de desenho do transistor e largura das linhas de metal. Estas regras não são utilizadas pelo extrator, apenas pelo TROPIC.
- regras CIF (14 regras) - é a definição dos nomes CIF para cada camada do leiaute.
- regras elétricas (6 regras) - estas são as regras utilizadas pelo extrator para o cálculo dos elementos parasitas. Estão incluídos as capacitâncias de perímetro e de área de cada camada, capacitância do óxido, limite de fator de carga e resistência das camadas.

B.1 Regras Geométricas

As regras geométricas podem ser divididas em:

- Regras relativas as camadas de interconexões - definem a distância de separação e a largura mínima das conexões. As diretivas que definem estes parâmetros são formadas pela identificação da camada, precedida da letra “L” para a largura mínima, ou pela letra “D” para a distância mínima (figura B.1). Por exemplo:

DPOLY é a distância mínima entre duas conexões de polisilício, e LPOLY é a largura mínima do polisilício.

As camadas são identificadas por “POLY” para polisilício, “DIF” para difusão, “M1” para metal1, “M2” para metal2 e “M3” para metal3.

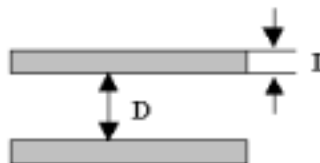


FIGURA B.1 – Regras relativas às camadas de conexões

- regras relativas aos contatos e vias - definem o comprimento e a distância de separação dos contatos ou vias, e a margem para o metal (figura B.2). As diretivas também são formadas pela identificação do elemento (contato ou via) precedidos pela letra “L” para comprimento, “D” para distância de separação ou “M” para a margem. Então teríamos:

LCTO é a largura do contato,

MCTO é a margem do contato para o metal 1, e

DCTO é a distância de separação entre dois contatos.

Os contatos são identificados por “CTO”, a via 1 por “VIA1”, e a via 2 por “VIA2”.

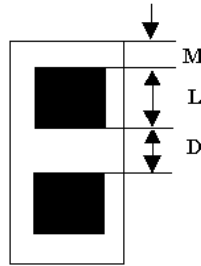


FIGURA B.2 – Regras relativas aos contatos e vias

- regras relativas aos transistores - definem os parâmetros para a implementação dos transistores. Alguns parâmetros estão definidos na figura B.3.

1. MINPWELL - é a distância entre o implante P e o poço N.
2. MDIFIMP - é a distância entre o implante N/P e a região ativa.
3. MTIEIMP - é a distância entre o implante N/P e a região ativa para contatos ao substrato (*body-ties*).
4. DTIEWELL - é a distância entre o implante N/P para contato ao substrato e poço N.
5. DIMPWELL - distância entre implante P e o poço N.
6. XTRPOLY - extensão do polisilício.
7. XTRPOLYDIF - margem da difusão do transistor.
8. DCTOPOLY - distância entre contato e o polisilício.

E “XTRW” representa a mínima largura do transistor e “XTRL” é o mínimo comprimento do transistor.

- regras relativas a alimentação - define apenas a largura das linhas de alimentação, dada pela diretiva “WSUPPLY”.

Todas as regras geométricas são dadas em $\mu m/1000(10^{-9}m)$, por exemplo, para o caso da distância entre dois contatos for de $0,5\mu m$, no arquivo de tecnologia teríamos “DCTO 500”.

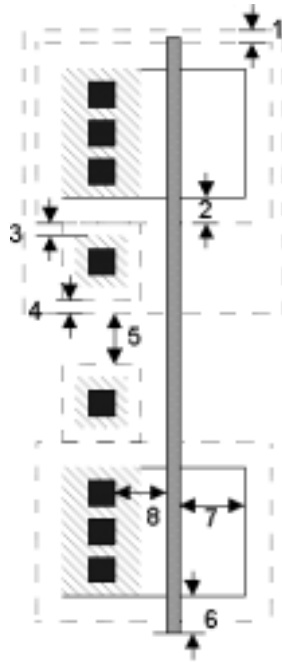


FIGURA B.3 – Regras para implementação dos transistores

B.2 Regras CIF

Estas regras servem para estabelecer uma correspondência entre os nomes dos níveis definidos no arquivo de leiaute e os nomes da descrição CIF. São formadas por duas diretivas:

- LABELCMD - define o comando para os *labels*.
- CIFLAYERS - é uma lista formada por duas colunas, que irá estabelecer a correspondência entre a descrição CIF e o arquivo de leiaute. Na primeira coluna, temos as camadas definidas internamente ao extrator, enquanto que a segunda coluna apresenta os nomes das camadas contidas no arquivo de leiaute.

A diretiva CIFLAYERS não é obrigatória, porém a sua falta faz com que seja assumida a seguinte correspondência:

metal1	CME1
metal2	CME2
metal3	CME3
poly	CPOL
cont	CCON
via	CVIA
via2	CVIA2
nwell	CNWI
pimp	CPPI
nimp	CNPI
ndif	CTOX
pdif	CTOX
label	CLAB

B.3 Regras Elétricas

Estas são as principais regras utilizadas pelo extrator, do qual depende, fundamentalmente, a sua precisão. São definidos os parâmetros necessários para o cálculo das capacitâncias e resistências. As seguintes diretivas devem ser definidas:

- CAPACITANCE ORDER - define a ordem em que as camadas estão dispostas no processo de fabricação. Isto é importante para verificar qual camada representa o primeiro plano terra e a existência de um segundo plano terra.
- CAPACITANCE CROSSTALK $\langle layer \rangle \langle n1 \rangle \langle n2 \rangle \langle n3 \rangle$ - define parâmetros para o cálculo da capacitância de *crosstalk*. Os parâmetros necessários são:

layer - é a camada para a qual os parâmetros a seguir são válidos.

n1 - é a espessura da camada

n2 - é a distância vertical ao substrato

n3 - é a máxima separação em que será calculada a capacitância.

Recomenda-se que seja no máximo 2,5 vezes o valor de *n1*.

- CAPACITANCE INTRINSIC $\langle layer \rangle \langle n1 \rangle \langle n2 \rangle$ - define os parâmetros para o cálculo das capacitâncias ao plano terra (substrato). Os parâmetros são:

layer - é a camada para a qual os parâmetros são válidos

n1 - é a capacitância de área entre o *layer* e o substrato, em $fF/\mu m^2$

n2 - é a capacitância de perímetro entre *layer* e o substrato, em $fF/\mu m$

- CAPACITANCE CROSSOVER $\langle layer \rangle \langle n1 \rangle \langle n2 \rangle$ - define os parâmetros para o cálculo da capacitância entre duas camadas diferentes. Os parâmetros são:

layer - é a camada para a qual os parâmetros são válidos

n1 - é a capacitância de área entre o *layer* e a camada superior

n2 - é a capacitância de perímetro entre o *layer* e a camada superior

Note que a camada superior é definida na diretiva “CAPACITANCE ORDER”.

- CAPAS - define parâmetros para o cálculo da capacitância de difusão, o fator de carga limite e permissividade relativa do óxido.

Cjn e *Cjp* - são as capacitâncias de área para a difusão, em $fF/\mu m^2$

Cjns e *Cjps* - são as capacitâncias de perímetro para a difusão, em $fF/\mu m$

Eox - é permissividade relativa do SiO_2

Tox - é a espessura do óxido, em metros

Climit - é o fator de carga limite

FDIF - é o divisor da capacitância de difusão

- RESISTANCE $\langle layer \rangle \langle n1 \rangle$ - define os parâmetros para o cálculo das resistências. São eles:

layer - é a camada na qual será calculada a resistência

n1 - é a resistência, dada em *Ohms* para o caso das vias e contatos, e em *Ohm/□* para as camadas de conexões (polisilício, metal1, metal2 e metal3)

B.4 Exemplo de um Arquivo de Tecnologia

```
*****
Rules file for tropic 3.1 - TECH HCMOS7 - 0.25um - 04/02/1999
*****
```

directory containing the input spice file
 #dirIN Extraction/tropic/benchs

directory wich will contain: (1) cif file (2) error file (3) extracted wires
 #dirOUT Extraction/examples

* L length - M margim - D distance

```
#DPOLY 450
#LPOLY 250
#DDIF 500
#DM1 400
#LM1 400
#DM2 400
#LM2 600
#DM3 400
#LM3 600
```

```
#LCTO 300
#MCTO 100
#DCTO 500
```

```
#LVIA1 400
#MVIA1 100
#DVIA1 500
```

```
#LVIA2 400
#MVIA2 100
#DVIA2 500
```

```
#XTRW 350
#XTRL 250
#XTRPOLY 350
#XTRPOLYDIF 200
```

```
#MDIFIMP 250
#MIMPWELL 600
#DIMPWELL 350
```

```
#MTIEIMP 250
#DTIEWELL 0
#WSUPPLY 1500
```

#LABELCMD 94

Interconnection order for calculate the capacitances

#CAPACITANCE ORDER poly metal1 metal2 metal3

Interconnection Capacitances

CAPACITANCE CROSSTALK layer n1 n2 n3

n1 - layer tchikness

n2 - distance to the ground plane

n3 - maxima distance between neighbors

#CAPACITANCE CROSSTALK poly 200 350 625

#CAPACITANCE CROSSTALK metal1 600 1400 1000

#CAPACITANCE CROSSTALK metal2 720 2300 1500

#CAPACITANCE CROSSTALK metal3 720 4320 1500

CAPACITANCE INTRINSIC layer n1 n2

n1 - area capacitance

n2 - perimeter capacitance

#CAPACITANCE INTRINSIC poly 0.0987 0.0445

#CAPACITANCE INTRINSIC metal1 0.0247 0.0408

#CAPACITANCE INTRINSIC metal2 0.0150 0.0380

#CAPACITANCE INTRINSIC metal3 0.0080 0.0328

CAPACITANCE CROSSOVER layer n1 n2

n1 - area capacitance layer i+1

n2 - perimeter capacitance layer i+1

#CAPACITANCE CROSSOVER poly 0.0432 0.0466

#CAPACITANCE CROSSOVER metal1 0.0384 0.0477

#CAPACITANCE CROSSOVER metal2 0.0384 0.0477

Interconnection Resistance

#RESISTANCE poly 10

#RESISTANCE metal1 0.25

#RESISTANCE metal2 0.055

#RESISTANCE metal3 0.055

#RESISTANCE cont 12

#RESISTANCE via 1.5

#RESISTANCE via2 1.5

#CIFLAYERS

nwell nwell

pimp pplus

nimp nplus

ndif active

pdif active

poly poly

metal1 metal1

metal2 metal2

metal3 metal3

cont contact

via v1

via2 v2

label text

END

#CAPAS - typical

Cjn 0.7992

Cjp 0.8187

Cjnsw 0.1479

Cjpsw 0.1514

Tox 5e-9

Eox 3.9

Climite 5.0

/* defines the load limit */

FDIF 2.0

END

*****end

Bibliografia

- [ARO 96] ARORA, Narain D.; RAOL, Kartik V.; SCHUMANN, Reinhard et al. Modeling and Extraction of Interconnect Capacitances for Multilayer VLSI Circuits. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.15, n.1, p.58–66, Jan. 1996.
- [BAR 85] BARKE, E. Resistance calculation from mask artwork data by finite element method. In: DESIGN AUTOMATION CONFERENCE, 22., 1985. **Proceedings...** Los Alamitos: IEEE Computer Society, 1985.
- [BRO 86] BROWN, Randy Lee. Multiple Storage quad trees: A simple faster alternative to bisector list quad trees. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.5, n.3, p.413–418, July 1986.
- [CAD 91] CADENCE. **Virtuoso layout synthesizer - LAS - user guide**. [S.l.: s.n.], 1991.
- [CAD 98] CADENCE. **Diva Interactive Verification Reference**. [S.l.: s.n.], 1998.
- [CAR 87] CARLSON, E.; RUTENBAR, R.A. A Scanline Data Structure Processor for VLSI Geometry Checking. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.6, n.5, p.780–797, Sept. 1987.
- [CAS 91] CASACURTA, Alexandre et al. Sela - Sistema de Edição de Layout. In: SEMINÁRIO INTERNO DE MICROELETRÔNICA, 7., 1991, Capão da Canoa, RS. **Anais...** Porto Alegre: CPGCC da UFRGS, 1991.
- [CHA 70] CHAWLA, B. R.; GUMMEL, H. K. A boundary technique for calculation of distributed resistance. **IEEE Transactions Electron Devices**, New York, v.ED-17, p.915–925, Oct. 1970.
- [CHE 92] CHERN, Jue Hsien; HUANG, Jean; ARLEDGE, Lawrence et al. Multilevel Metal Capacitances Models for CAD Design Synthesis Systems. **IEEE Electron Devices Letters**, New York, v.13, n.1, p.32–34, Feb. 1992.
- [CHI 89] CHIANG, Kuang Wei. Resistance Extraction and Resistance Calculation in GOALIE2. In: DESIGN AUTOMATION CONFERENCE, 26., 1989. **Proceedings...** New York: ACM, 1989.
- [CON 96] CONG, Jason; KAHNG, Andrew B.; NOICE, David; SHIRALI, Nagesh; YEN, Steve H. **Analysis and Justification of a Simple, Practical 2 1/2D Capacitance Extraction Methodology**. [S.l.]: UCLA Computer Science, 1996. (Technical Report 970013).
- [DAI 97] DAI, Wayne; SUN, Weikai. **3-D Parasitic Extraction for Deep Submicron IC Design**. Disponível por WWW em <http://www.colo.isdmag.com/Editorial/1997/ASIC9707.html> (1997).
- [DEN 97] DENG, E. Aykut; ROHRER, Ronald A. Hierarchical 2-D Field Solution for Capacitance Extraction for VLSI Interconnect Modeling. In: DESIGN AUTOMATION CONFERENCE, 33., 1997. **Proceedings...** Los Alamitos: IEEE Computer Society, 1997.

- [ELM 48] ELMORE, W.C. Transient response of damped linear networks with particular regard to wideband amplifiers. **IEEE Journal of Applied Physics**, New York, v.19, p.55–63, Jan. 1948.
- [FER 98] FERREIRA, Fábio Klein et al. **Fluxo de Projeto para Circuitos Digitais**. [S.l.: s.n.], 1998.
- [FER 99] FERREIRA, Fábio Klein; REIS, Ricardo. Parasitic Capacitance Extraction in Submicronic Technologies. In: SEMINÁRIO INTERNO DE MICROELETRÔNICA, 14., 1999, Pelotas, RS. **Anais...** Porto Alegre: CPGCC da UFRGS, 1999.
- [GEN 92] GENDEREN, A.J. van; MEIJS, N.P. van der. Reduced RC Models for IC Interconnections with Coupling Capacitances. In: EUROPEAN DESIGN AUTOMATION CONFERENCE, 29., 1992. **Proceedings...** Los Alamitos: IEEE Computer Society, 1992.
- [HOR 83] HOROWITZ, Mark; DUTTON, Robert W. Resistance Extraction from Mask Layout Data. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.CAD-2, n.3, p.145–150, July 1983.
- [MAN 96] MANIWA, R.T. **Focus Report - Signal Integrity Tools**. Disponível por WWW em <http://www.eedesign.com/Eedesign/FocusReport9607.html>, (1996).
- [MEI 92] MEIJS, N.P. van der; GENDEREN, A.J. van. Space - Efficient Extractions Algorithm. In: EUROPEAN DESIGN AUTOMATION CONFERENCE, 29., 1992. **Proceedings...** Los Alamitos: IEEE Computer Society, 1992.
- [MEN 99] MENTOR GRAPHICS. **ICverify Manual**. [S.l.: s.n.], 1999.
- [MET 92] META SOFTWARE. **HSPICE USER'S MANUAL**. [S.l.: s.n.], 1992.
- [MIT 87] MITSUHASHI, Takashi; YOSHIDA, Kenji. A Resistance Calculation Algorithm and Its Application to Circuit Extraction. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.CAD-6, n.3, p.337–345, May 1987.
- [MOR 99] MORAES, F.; ROBERT, M.; AUVERGNE, D. A Virtual CMOS Library Approach for Fast Layout Synthesis. In: VLSI, 1999. **Proceedings...** Kluwer Academic Publishers, 1999.
- [NAB 92] NABORS, K.; KIM, S.; WHITE, J.; SENTURIA, S. **Fastcap USER'S GUIDE**. [S.l.: s.n.], 1992.
- [NAB 91] NABORS, K.; WHITE, J. Fastcap: A Multipole Accelerated 3D Capacitance Extraction Program. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.10, n.11, p.1447–1459, Nov. 1991.
- [NIN 87] NING, Q.; DEWILDE, P. M.; NEERHOFF, F. L. Capacitance coefficients for VLSI multilevel metallization lines. **IEEE Transactions Electron Devices**, New York, v.ED-34, p.644–649, 1987.
- [OHT 86] OHTSUKI, T. **Layout Design and Verification**. [S.l.]: North Holland, 1986. v.4.

- [OUS 84] OUSTERHOUT, J.K. Corner Stitching: A Data Structuring Technique for VLSI Layout Tools. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.CAD 3, n.1, p.100–111, Jan. 1984.
- [PON 91] PONG, Teng-Sin; BROOKE, Martin A. A Parasitic Extraction and Network Reduction Algorithm for Analog VLSI. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.10, n.2, p.145–149, Feb. 1991.
- [RAB 96] RABAEY, Jan M. **Digital Integrated Circuits**. [S.l.]: Prentice Hall, 1996.
- [RUE 73] RUEHLI, A. E.; BRENNAM, P. A. Efficient capacitance calculations for three-dimensional multiconductor systems. **IEEE Transactions Microwave Theory Technology**, New York, v.MTT-21, p.76–82, 1973.
- [SAK 83] SAKURAI, T.; TAMARU, K. Simple Formulas for two and three dimensional capacitances. **IEEE Transactions Electron Devices**, New York, v.ED-30, n.2, p.183–185, Feb. 1983.
- [SAK 83a] SAKURAI, Takayasu. Approximation of Wiring Delay in Mosfet LSI. **IEEE Journal of Solid State Circuits**, New York, v.SC-18, n.4, p.418–426, Aug. 1983.
- [SCH 97] SCHEFFER, Lou. A Roadmap of Cadtool Changes for Submicron Interconnect Problems. In: INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN, 1., 1997. **Proceedings...** New York: ACM, 1997.
- [SEI 88] SEIDL, A.; SVOBODA, M.; OBERNDORFER, J. et al. CAPCAL: A 3D capacitance solver support of CAD systems. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, n.11, p.549–556, 1988.
- [SHE 93] SHERWANI, Naveed. **Algorithms for VLSI Physical Design Automation**. [S.l.]: Kluwer Academic Publishers, 1993.
- [SIA 97] SIA - Semiconductor Industry Association. **The National Technology Roadmap for Semiconductor**. Disponível por WWW em <http://notes.sematech.org/ntrs/PublNTRS.nsf> (1997).
- [SIC 93] SICARD, E.; DEMONCHAUX, T.; NOULLET, J.L.; RUBIO, A. Crosstalk Extraction from Mask Layout. In: EUROPEAN DESIGN AUTOMATION CONFERENCE, 1993. **Proceedings...** Los Alamitos: IEEE Computer Society, 1993.
- [STE 89] STEMMER, Marcos Augusto. **EXTRIBO - Um Extrator Hierárquico de Circuitos**. Porto Alegre: PPGC da UFRGS, 1989. Dissertação de Mestrado.
- [SYN 99] SYNOPSYS. **Arcadia Datasheet**. Disponível por WWW em http://www.synopsys.com/products/etg/arcadia_ds.html (1999).
- [UEB 86] UEBBING, R. H.; FUKUNA, Masao. Process Based three dimensional capacitance simulation - TRICEPS. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.CAD-5, p.215–220, 1986.

- [WAL 2000] WALKER, Martin G. Modeling the wiring of deep submicron ICs. **IEEE Spectrun**, New York, p.65–71, Mar. 2000.
- [WRI 93] WRIGHT, P.J.; SHIH, Y.C.A. Capacitance of top leads metal - comparison between formula, simulation and experiment. **IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems**, New York, v.12, n.12, p.897–902, Dec. 1993.