UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

CIPREDI : CONTRIBUIÇÃO INICIAL PARA UM MÉTODO DE CONCEPÇÃO DE CIRCUITOS INTEGRADOS PRÉ-DIFUNDIDOS

por

NEY LAERT VILAR CALAZANS

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

Prof. Dr. Dante Augusto Couto Barone
Orientador

Porto Alegre, julho de 1988.

CATALOGAÇÃO NA FONTE

Calazans, Ney Laert Vilar

CIPREDI: Contribuição inicial para um método de concepção de circuitos integrados pré-difundidos. Porto Alegre, PGCC da UFRGS, 1988.

1 v.

Diss. (mestr. Ci. Comp.) UFRGS - CPGCC, Porto Alegre, BR-RS, 1988.

Dissertação: pré-difundidos: gate arrays: sistemas de PAC: métodos de projeto: circuitos integrados

OFERECIMENTO

Dedico este trabalho à memória de minha irmã Neide Lúcia, cuja perda prematura mostrou a uma criança quão dura pode ser uma existência, mas que apontou maneiras de torná-la válida: respeitar, acima de tudo, a Natureza e os seres humanos, lutar com todas as forças pelo que se considera correto e não desistir jamais do que quer que se decida fazer. A você Neide, tudo que se pode dizer com palavras e mais, tudo que só se pode expressar com lágrimas.

AGRADECIMENTOS

Em primeiro lugar, quero agradecer a meus pais, Marilnha e Paulo, cuja dedicação e espírito de sacrifício pelos filhos é algo assombroso. Sem eles, nada teria sido possível. Agradeço a meus irmãos, Paulo e Nelson, pelo apoio concedido sempre que precisei e a meus sobrinhos, Neide, Marcus e Albert, pela alegria que acrescentam a minha vida. Agradeço a prima Mércia pelo alento em muitas horas difíceis.

Karin, obrigado pelo amor, pelos grandes esforços na construção de nossa vida a dois, pela própria ajuda na confecção deste trabalho e pelos pequenos gestos do dia a dia, que transformam os tantos momentos em que estamos juntos numa parte fundamental de minha vida.

Seria difícil enumerar todos aqueles que de alguma forma contribuiram para este trabalho. Devo ao meu orientador, Prof. Dr. Dante Barone as virtudes de organização desta dissertação, bem como a idéia primordial do trabalho. As falhas são obviamente devidas a mim. Agradeço ao Prof. Dr. Flávio Wagner pela chance de participar no projeto AMPLO, bem como pelos inúmeros comentários de valor inquestionável. Agradeço a Ricardo Jacobi, pela criteriosa posição em discussões sobre o assunto deste trabalho. Agradeço ainda a Fernando Osório, pela valiosa colaboração na implementação do EDGAR e a Carlos Franceschini pelo auxílio na implementação e caracterização da biblioteca de células de pré-difundidos.

Agradeço ao Renato Dillenburg, ao Renato Branco, à Carlota, ao Paulo Fernandes, ao Flavinho, ao Clovis, à Cristina, à Cínia, à Zeila, à Adriane, e ao Jacques, pelo simples fato de existirem como existem em minha vida.

Agradeço a todos aqueles que se sentirem

preteridos nesta lista de agradecimentos, pois o simples fato de se importarem com o conteúdo dela os faz merecer, revelando mais a minha incapacidade de lembrar que a justiça ou não de citá-los.

Agradeço finalmente, à Anna Luiza, por tudo de bom, e até pelo resto. É impossível negar a importância do que aconteceu.

"We, the willing, are doing the impossible for the ungrateful. We have done so much, with so little, for such a long time, we are now qualified to get anything from nothing."

SUMÁRIO

LISTA DE FIGURAS ---------- 010
LISTA DE TABELAS ---------- 013
RESUMO ---------- 014
ABSTRACT ---------- 015
1 INTRODUÇÃO ---------- 016
1.1 Proposta de uma Nova Taxonomia para Circuitos Integrados ---------- 017
2 HISTÓRICO DOS CIRCUITOS INTEGRADOS PRÉ-DIFUNDIDOS -- 024
2.1 Comparação de Estilos de Concepção ---------- 027
2.1.1 Comparação de circuitos não-personalizáveis e personalizáveis --- 031
2.1.2 Comparação entre os diversos estilos de projeto usando circuitos personalizáveis ---------- 033
2.2 Pré-difundidos - Evolução ---------- 039
2.3 Pré-difundidos - Concepção Tradicional --------- 042
2.4 Pré-difundidos - Aspectos Econômicos ---------- 045
2.5 Pré-difundidos no Contexto Nacional ---------- 049
3 MÉTODOS DE CONCEPÇÃO PARA CIRCUITOS PRÉ-DIFUNDIDOS - 055
3.1 Métodos e Ferramentas ---------- 056
3.1.1 Ótica pragmática para interpretação de métodos de projeto de CIs pré-difundidos ---------- 057
3.1.2 Ferramentas de "hardware" no projeto de pré-difundidos ---------- 071
3.1.3 Ferramentas de "software" no projeto de pré-difundidos ---------- 076
3.2 Tecnologias, Células de base, Matrizes e Biblioteca de Células ---------- 091
3.2.1 Tecnologias ---------- 092
3.2.2 Células de base ---------- 093
3.2.3 Matrizes ---------- 101
3.2.4 Biblioteca de células ---------- 102
3.3 A Relação Cliente/Fornecedor ---------- 104

3.3.1 Formas de interação ---------- 106
3.3.2 Documentos formais ---------- 110
4 O MÉTODO CIPREDI ---------- 114
4.1 Teoria do Sistema Alvo ---------- 116
4.1.1 Tecnologias e matrizes ---------- 116
4.2 Modos de Especificação ---------- 119
4.2.1 Linguagens de entrada ---------- 120
4.2.2 Linguagens de saída ---------- 123
4.2.3 A relação cliente/fornecedor ---------- 124
4.3 Processos de Projeto ---------- 125
4.4 Infraestrutura de Projeto ---------- 126
4.4.1 Ferramentas de "hardware" ---------- 126
4.4.2 Ferramentas de "software" ---------- 128
5 IMPLEMENTAÇÃO DE SUPORTE À CONCEPÇÃO ---------- 133
5.1 GME - Uma Nova Célula de Base ---------- 133
5.2 Implementação e Caracterização de uma Biblioteca de Células Mínima ---------- 139
5.3 Circuitos de Teste ---------- 142
6 FERRAMENTAS DO MÉTODO CIPREDI ---------- 150
6.1 Integração das Ferramentas ---------- 150
6.1.1 Relação entre ambiente de concepção e ferramentas ---------- 150
6.1.2 Integração das ferramentas do CIPREDI ---- 153
6.2 O Simulador NILO ---------- 158
6.2.1 Mensagens ---------- 161
6.2.2 A implementação do simulador escravo NILO ---------- 163
6.2.2.1 Mensagens do mestre para o escravo ---------- 167
6.2.2.2 Avaliação de subredes ---------- 169
6.2.2.3 Algoritmo de relaxação para subredes bidirecionais ---------- 170
6.2.2.4 Avaliação de portas lógicas e transistores ---------- 170
6.2.2.5 Avaliação de atrasos ---------- 171
6.2.2.6 Manipulação de listas ---------- 171
6.2.3 A definição do simulador mestre NILO ----- 173

6.2.3.1 Mensagens da interface gráfica para o mestre ---------- 174
6.2.3.2 Mensagens do escravo para o mestre ---------- 176
6.2.3.3 Algoritmo geral do simulador mestre ---------- 177
6.3 O editor EDGAR ---------- 181
6.3.1 Diagrama de fluxo de dados do editor EDGAR ---------- 182
6.3.2 Ambiente de implementação do EDGAR ------- 184
6.3.3 Relação com descrições AMPLO ---------- 185
6.3.4 Criação de células de biblioteca --------- 186
6.3.5 Estruturas de dados internas ---------- 187
7 DIREÇÕES FUTURAS DO MÉTODO CIPREDI E CONCLUSÕES ---- 190
7.1 Ambiente Mínimo ---------- 190
7.1.1 Teoria do sistema alvo ---------- 190
7.1.2 Modos de especificação ---------- 191
7.1.3 Processos de projeto ---------- 194
7.1.4 Infraestrutura de projeto ---------- 194
7.2 Evolução ---------- 198
7.2.1 Níveis superiores de abstração ---------- 198
7.2.2 Síntese automatizada ---------- 199
7.2.3 Verificação de projeto e projeto visando o teste ---------- 200
7.2.4 Integração do ambiente ---------- 201
7.2.5 Interação com fundições de silício ------- 201
7.2.6 Novos circuitos ---------- 202
7.3 Conclusões Finais ---------- 202
Anexo 1 - Exemplo de Documentos Formais usados no Intercâmbio de Informações entre o Cliente e o Fornecedor de Pré-difundidos ---------- 205
Anexo 2 - Proposta de Documentação para Biblioteca de Células de Pré-difundidos ---------- 214
Anexo 3 - Principais Rotinas e Estruturas de Dados do Simulador Mestre NILO ---------- 218
BIBLIOGRAFIA ---------- 224

LISTA DE FIGURAS

Figura 1.1 - Taxonomia do estado da arte em CIs 021
Figura 1.2 - "Layout" típico de CI pré-difundido 022
Figura 1.3 - "Layout" típico de CI pós-difundido 022
Figura 1.4 - "Layout" típico de CI dedicado 023
Figura 2.1 - Comparação volume x custo unitário dos estilos de concepção 027
Figura 2.2 - Comparação volume x total de portas equivalentes dos CIs personalizáveis 028
Figura 2.3 - Caracterização de mercado dos estilos de concepção 030
Figura 2.4 - Isolação por porta de poli 036
Figura 2.5 - Pré-difundido com blocos configuráveis ... 038
Figura 3.1 - Célula de base típica de pré-difundidos .. 059
Figura 3.2 - Exemplo de matriz de pré-difundido 060
Figura 3.3 - Exemplo de informações contidas em uma biblioteca de células 064
Figura 3.4 - Diagrama de Gajski ou Diagrama Y 068
Figura 3.5 - Processo de projeto aplicado no CPqD da Telebrás 070
Figura 3.6 - Estrutura geral de métodos de projeto para CIs pré-difundidos 071
Figura 3.7 - Processo de projeto típico para pré-difundidos 077
Figura 3.8 - Diagrama Y do processo de projeto típico 078
Figura 3.9 - Processo de projeto empregado pelo sistema MEC-CAD da GE 085
Figura 3.10 - Processo de projeto empregado pelo sistema DASH-GA Compiler da Futurenet 088
Figura 3.11 - Planta baixa de matriz com células isoladas 095
Figura 3.12 - Planta baixa de matriz com filas de células 095
Figura 3.13 - Planta baixa de matriz com mar de células 096

Figura 3.14 - Alocação dinâmica de canais 097
Figura 3.15 - Conexão com uso de transparências 098
Figura 3.16 - Isolação por porta de poli 099
Figura 3.17 - Isolação por óxido espesso 100
Figura 3.18 - Matriz com blocos especializados 101
Figura 3.19 - Pontos possíveis de mudança da
responsabilidade de projeto 107
Figura 3.20 - Projeto de pré-difundidos no sistema
LDS da LSI Corporation 109
Figura 4.1 - Processo de projeto empregado pelo sistema
AMPLO 122
Figura 4.2 - Linguagens de entrada atuais do CIPREDI .. 123
Figura 4.3 - Processo de projeto atual do método
CIPREDI 132
Figura 5.1 - Formatos utilizáveis para a célula
de base GME 136
Figura 5.2 - Porta NÃO-E implementada sobre transistores
grandes da célula GME 138
Figura 5.3 - Bit de memória RAM construído com
oito transistores 139
Figura 5.4 - Duas formas de implementar inversores
sobre a célula GME 144
Figura 5.5 - Topologia do multiplicador 147
Figura 5.6 - "Layout" do multiplicador 147
Figura 6.1 - Proposta de interação entre ferramentas
e o ambiente de concepção CIPREDI 151
Figura 6.2 - Proposta de integração entre ferramentas
no método CIPREDI 153
Figura 6.3 - Processo de projeto descendente
automatizado 156
Figura 6.4 - Processo de projeto manual ascendente 158
Figura 6.5 - O ambiente de simulação AMPLO 159
Figura 6.6 - Uma célula de memória RAM MOS
modelada em NILO 161
Figura 6.7 - Diagrama de Hasse (Reticulado Hierárquico)
da álgebra do simulador NILO 165
Figura 6.8 - Rotina principal do simulador escravo 168

Figura 6.9 - Estrutura da LEN através de um exemplo ... 173
Figura 6.10 - Rotina ative_mestre 178
Figura 6.11 - Diagrama de fluxo de dados do EDGAR 182
Figura 6.12 - Conexão - a unidade básica de "layout"
no EDGAR 188
Figura 7.1 - Fluxograma de projeto no ambiente
mínimo 196
Figura 7.2 - Processo de projeto no ambiente mínimo ... 197

LISTA DE TABELAS

Tabela 2.1 - Comparação qualitativa dos estilos de concepção 037
Tabela 2.2 - Comparação pré-difundido x pós-difundido 046
Tabela 2.3 - Comparação pré-difundido x dedicado 047
Tabela 2.4 - Custo do encapsulamento de um CI 048
Tabela 3.1 - Características da família de matrizes com canal de poli de 2 microns da AMI 060
Tabela 3.2 - Análise de necessidades de recursos pelas ferramentas de projeto 074
Tabela 3.3 - Matrizes disponíveis na família MEC4U da GE/Intersil 113
Tabela 5.1 - Células de biblioteca disponíveis no CIPREDI 140
Tabela 5.2 - Características do multiplicador 146
Tabela 7.1 - Tarefas e prioridades de implementação para o ambiente mínimo 198

RESUMO

Este trabalho constitui a contribuição inicial para o desenvolvimento de um método de concepção de circuitos integrados pré-difundidos, também denominados "gate arrays", no âmbito do CPGCC/UFRGS. Uma nova taxonomia para o estado da arte dos circuitos integrados é proposta, visando situar o escopo do método. Após a elaboração de um breve histórico dos circuitos pré-difundidos, desenvolve-se um estudo genérico sobre métodos de projeto e elabora-se uma proposta de método para este estilo de concepção. Ferramentas implementadas e atividades de suporte à concepção são descritas, bem como as diretivas para a evolução futura do método.

ABSTRACT

This work constitutes a first contribution to the development of a design methodology for gate array integrated circuits in the CPGCC/UFRGS. A novel taxonomy of the state of the art on integrated circuits is proposed, aiming the definition of the scope of the work. After a brief review of gate array evolution, a general approach of design methods is developed, together with the proposal of a specific design method adequate for this design style. The tools implemented, as well as the elaborated design support activities are described. Finally, further directions for the evolution of the design method are presented.

1 INTRODUÇÃO

Este trabalho constitui a contribuição inicial para o desenvolvimento de um método de projeto de circuitos integrados digitais denominado CIPREDI. O método CIPREDI visa o estilo de concepção conhecido como "gate array", "masterslice" ou "uncommitted logic array" (ULA), entre outras denominações.

Nesta introdução são esboçados os capítulos do trabalho e uma taxonomia do estado da arte de circuitos integrados. No capítulo 2, apresenta-se um breve histórico dos "gate arrays", uma comparação dos diferentes estilos de concepção de CIs e uma justificativa para o próprio trabalho. O capítulo 3 revisa os métodos de projeto dedicados a este tipo de circuitos integrados (CI), discutindo aspectos tecnológicos, sistemas de Projeto auxiliado por computador (PAC) e a relação cliente/ fornecedor para "gate arrays". Ainda no capítulo 3 é definida uma ótica de abordagem de métodos de projeto adequada para CIs em geral e pré-difundidos em particular.

A partir do capítulo 4, mostra-se o que constitui a contribuição principal do presente trabalho, ou seja, a proposição do método CIPREDI. O capítulo 4 estabelece as diretrizes gerais do método. A seguir são mostrados, no capítulo 5, a estratégia usada para fornecer suporte à implementação. O capítulo 6 descreve as ferramentas de "software" implementadas neste trabalho, além de apresentar uma forma geral de integração das ferramentas no método proposto. Finalmente, o capítulo 7 explora as possibilidades de evolução para o método CIPREDI, propondo um conjunto de atividades consideradas essenciais para a obtenção de um sistema produtivo a partir do presente trabalho.

1.1 Proposta de uma Nova Taxonomia para Circuitos Integrados

Os circuitos integrados semi-dedicados ("semi-custom") englobam a modalidade de projeto alvo desta dissertação. Eles encontram-se, do ponto de vista de projeto, na divisa das áreas de atuação de dois profissionais: o engenheiro de sistemas e o projetista de CIs. A formação distinta destes profissionais pode causar problemas de comunicação quando os mesmos cooperam na execução de um projeto, uma vez que termos idênticos podem ter conotações distintas para um e para outro.

A nomenclatura é um ponto chave para que a concatenação de esforços de áreas distintas alcance os resultados esperados. Assim, o objetivo de estabelecer uma taxonomia do estado da arte dos circuitos integrados em geral e obter uma notação objetiva, coerente e não ambígua é a primeira meta desta dissertação. O intuito desta atividade é ajudar a esclarecer, para o principiante na área, o papel de cada elemento de um ambiente de concepção. Ao mesmo tempo, esta notação fornece, ao especialista, uma base sobre a qual organizar suas atividades.

Para fins de classificação, dois itens foram considerados na confecção da taxonomia: a dinâmica do comportamento do componente integrado e as características estruturais de seu projeto e fabricação.

Considerando o grau de intervenção do projetista de sistemas em suas características, identifica-se dois tipos de CIs: os não-personalizáveis ou fixos, e os personalizáveis. Os primeiros, também chamados CIs de prateleira ("off the shelf") ou de catálogo, são os que apresentam comportamento imutável, isto é, sabe-se de antemão quais pinos são entradas, quais são saídas e que funções o CI realiza. Nesta categoria encontram-se os CIs

de catálogo de baixa e média escala de integração (SSI - "Small-Scale Integration" e MSI - "Medium-Scale Integration"). Exemplo destes são a série TTL 7400, descrita em /TEX 76/ e a série CMOS 4000 /NAT 80/.

Os microprocessadores e outros circuitos de alta escala de integração (LSI - "Large-Scale Integration"), tais como controladores de acesso direto à memória e controladores de interrupções /INT 79/, também são CIs não-personalizáveis. Contudo, estes circuitos podem ser programados logicamente. A categoria dos não-personalizáveis pode, com base nesta característica, ser subdividida em CIs não-programáveis ou rígidos e CIs programáveis.

Os personalizáveis, por sua vez, são aqueles onde o projetista pode especificar, dentro de certos limites, qual será o comportamento do componente no sistema final. Estes, se dividem ainda em duas sub-categorias: os CIs personalizáveis pós-fabricação e os personalizáveis por-fabricação. A primeira sub-categoria engloba aqueles CIs que o projetista obtém já encapsulados, sendo personalizáveis através do uso de "hardware" e "software" específicos. Exemplos são os FPLAs ("Field Programmable Logic Arrays"), os PLDs ("Programmable Logic Devices") e os diversos tipos de PROMs ("Programmable Read-Only Memories").

Os CIs personalizáveis por-fabricação constituem o campo de atuação dos engenheiros de componentes. Estes profissionais atuam dentro da área de Microeletrônica, embora não necessariamente vinculados a uma manufatura de semicondutores. Esta sub-categoria divide-se ainda em dois grupos : o dos parcialmente personalizáveis e o dos totalmente personalizáveis.

Um circuito pertencente ao último grupo citado é

obtido após a passagem por um grande número de etapas de projeto. O formato final consiste de um conjunto de descrições de máscaras fotolitográficas (entre 8 e 13, tipicamente), enviadas para o fornecedor de componentes para manufatura. Em última análise, cada detalhe de cada uma das máscaras é de responsabilidade do projetista. A probabilidade de ocorrência de erros de projeto é maximizada, o mesmo ocorrendo com o tempo de projeto e o desempenho final do componente. A área de semicondutor ocupada é a menor possível e a verificação de projeto é a mais demorada. Os circuitos neste grupo são também denominados de dedicados ou "full-custom".

Os parcialmente personalizáveis são CIs onde o processo de projeto é facilitado pelo uso de estruturas previamente caracterizadas e dimensionadas, denominadas células. Por outro lado, esta abordagem reduz de maneira drástica a flexibilidade de implementação. Nomes alternativos para estes CIs são pré-caracterizados e semi-dedicados ("semi-custom"). Este grupo divide-se em dois subgrupos: os pré-difundidos e os pós-difundidos.

Os componentes presentes no último grupo são aqueles onde as células (exemplo de célula: um "flip-flop"), já têm o "layout" caracterizado e validado individualmente. Ao ser desenvolvido um novo projeto, contudo, deve-se passar por todas as etapas de fabricação novamente, pois não apenas a interconexão das células, mas também a posição relativa destas no substrato semicondutor podem ser escolhidas pelo projetista. Tal liberdade faz com que todas as descrições de máscaras se alterem de um projeto a outro. Neste sub-grupo, encontram-se os circuitos fabricados segundo o estilo comumente denominado "standard cell" ou "cell array".

Finalmente, os CIs personalizáveis por-fabricação, de forma parcial, pré-difundidos, objeto de

estudo deste trabalho, caracterizam-se por possuirem todos os níveis de máscaras previamente definidos, com exceção daqueles que especificam a interconexão dos elementos de "layout" reponsáveis pela execução das funções no circuito. O ponto de partida para o projeto, neste caso, é um conjunto finito de elementos não conectados, previamente caracterizados e dimensionados. Estes elementos apresentam-se dispostos de forma regular sobre um substrato semicondutor, formando um arranjo. Desta disposição origina-se a denominação tradicional de "gate array".

A personalização de um "gate array" se faz especificando as máscaras de interconexão entre os elementos do arranjo. Estas máscaras correspondem, em geral, a camadas de metal, sendo por isso denominadas também de metalizações. Os elementos de "layout" são tipicamente componentes de circuito, como transistores e resistores, excepcionalmente encontrando-se portas lógicas como elementos primitivos. A denominação genérica destes elementos no presente trabalho será célula de base.

O projetista trabalha com uma biblioteca de células colocada a sua disposição pelo fabricante, conforme descrito por /GAG 83/ e documentado em /GEN 83/ e /GOU 85/. Esta biblioteca de células consiste de primitivas de níveis mais altos de abstração que os elementos primitivos do arranjo, tais como o nível lógico ou o nível de blocos funcionais. A cada primitiva da biblioteca de células, o fabricante associa um padrão de metalização que implementa a função da primitiva sobre uma ou mais células de base.

Todas as etapas fotolitográficas anteriores às metalizações, dada sua generalidade, permitem a pré-fabricação e a estocagem da pastilha do circuito. Assim, após concluída a fase de projeto, o fabricante recebe a descrição das interconexões gerada pelo projetista e conclui o processo de fabricação utilizando as lâminas de

silício pré-difundidas armazenadas.

A figura 1.1 sumariza a taxonomia descrita.

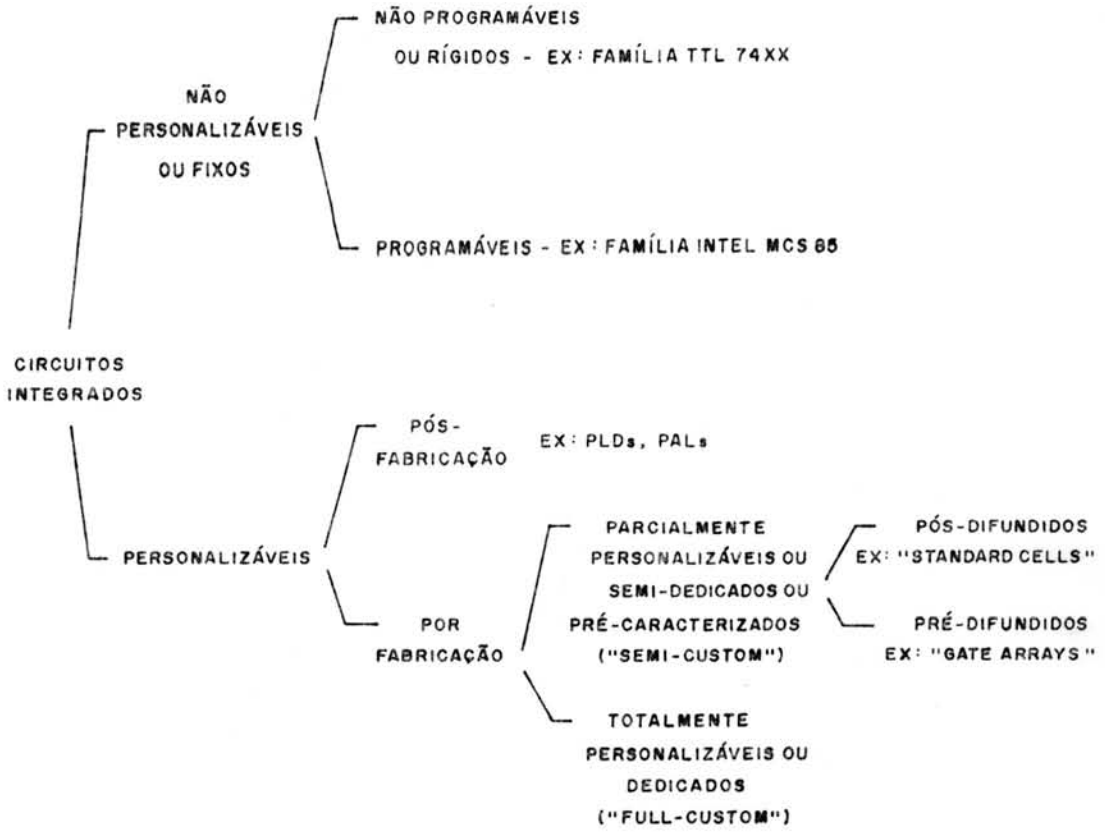


Figura 1.1 - Taxonomia do estado da arte em CIs

Nas figuras 1.2, 1.3 e 1.4 os três estilos de projeto personalizáveis por-fabricação são ilustrados, mostrando um "layout" típico para cada caso.

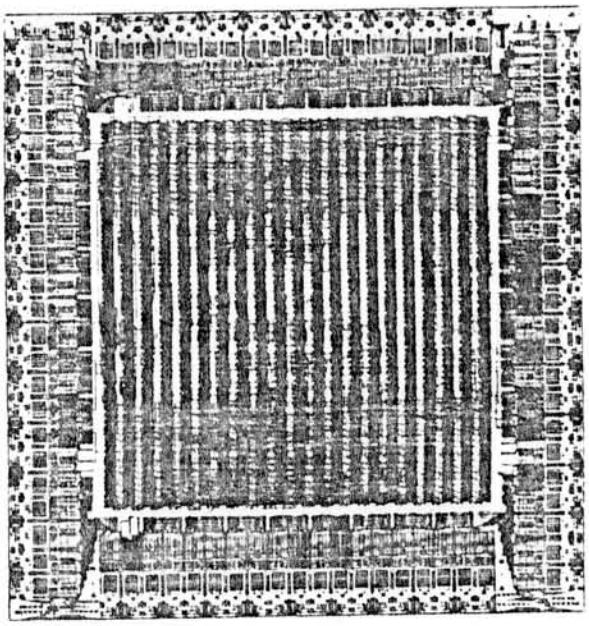


Figura 1.2 - "Layout" típico de CI pré-difundido
(extraído de /BER 83/)

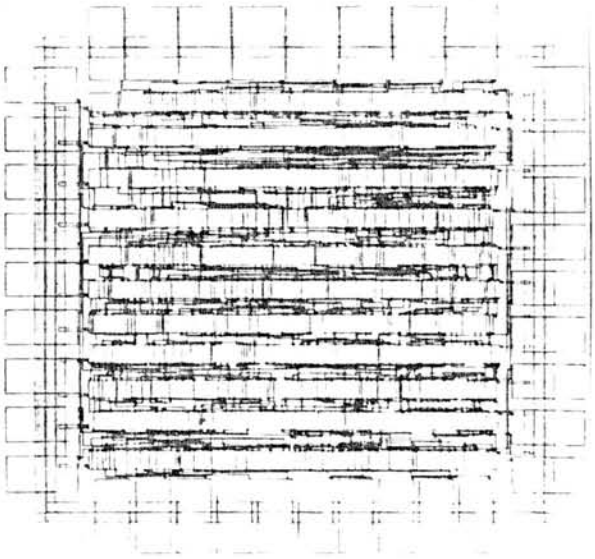


Figura 1.3 - "Layout" típico de CI pós-difundido
(extraído de /BER 83/)

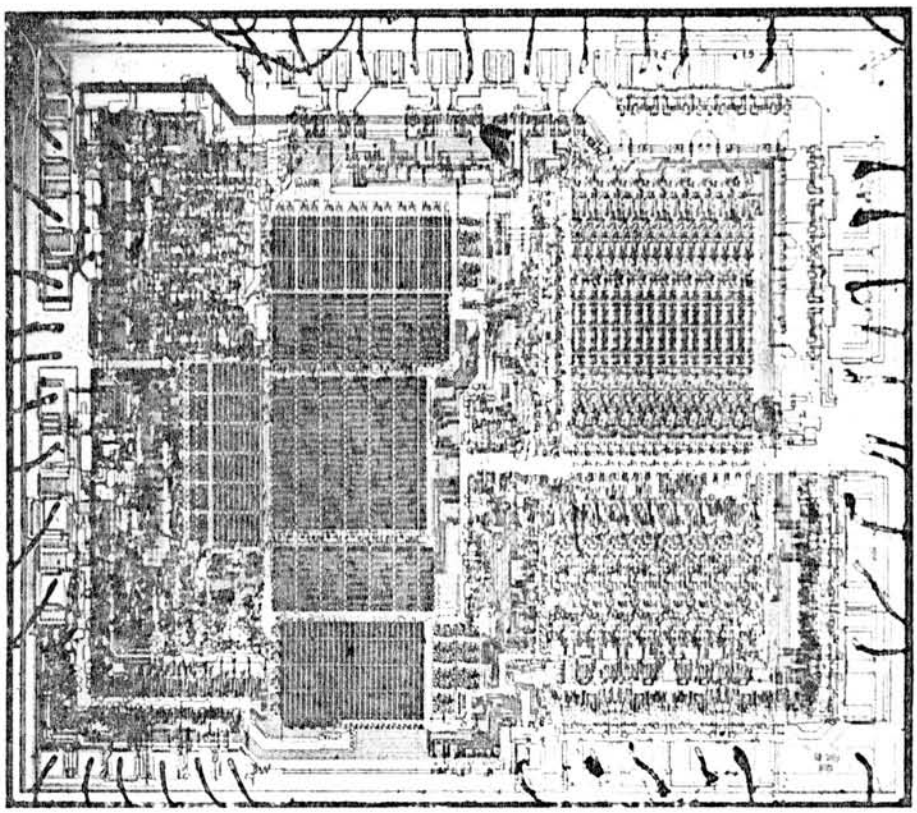


Figura 1.4 - "Layout" típico de CI dedicado
(o 18085 da Intel, extraído de /REI 83/)

2 HISTÓRICO DOS CIRCUITOS INTEGRADOS PRÉ-DIFUNDIDOS

Durante algumas décadas, a Microeletrônica constituiu um campo de atuação exclusivo para grandes conglomerados empresariais, cujo potencial econômico permitia o domínio de uma quantidade de tecnologias de caráter multidisciplinar. Estas tecnologias envolviam desde processos físico-químicos e metalúrgicos até aplicações da física de estado sólido, da eletrônica e da arquitetura de sistemas. O aperfeiçoamento destas tecnologias consumiu grande quantidade de recursos humanos, gerando o que Mead & Conway /MEA 80/ consideram o mais refinado processo de produção já desenvolvido.

Estes conglomerados produziam, empregavam e comercializavam um grande volume de CIs e/ou produtos contendo estes componentes. O volume de produção envolvido permitia um projeto cuidadoso de cada dispositivo, objetivando alcançar o melhor resultado possível do ponto de vista de desempenho e área de substrato semicondutor ocupada.

Três tipos de empresa, atualmente, usam esta forma de desenvolvimento de produtos e componentes:

. grandes fabricantes de computadores, como a International Business Machines Inc (IBM) e a Digital Equipment Corporation (DEC);

. grandes fabricantes de equipamentos, como a Hewlett-Packard Inc e a Tektronix Inc;

. grandes fabricantes de circuitos integrados de catálogo, como Motorola Inc e Intel /MUR 82b/.

Embora o nicho de mercado destas empresas

continue existindo, nas últimas duas décadas tem ocorrido uma diversificação considerável nas formas de obtenção de CIs para o projeto de sistemas eletrônicos. Isto ocorreu devido à evolução das tecnologias mencionadas, o que causou um barateamento significativo dos custos de projeto e fabricação destes elementos.

Considerações de volume de produção e área de substrato semicondutor ocupada por um "chip" ainda são, entretanto, críticas na avaliação da viabilidade econômica de um projeto de componente. Circuitos integrados são produzidos em escala: em cada passo da fabricação, de 50 a 100 "wafers" são processados simultaneamente, significando a implementação de 20000 ou mais CIs em paralelo /REA 85/. O custo fixo por passo de processamento dos "wafers" é uma grande percentagem do custo total de fabricação, o que torna necessário o processamento simultâneo do maior número possível de CIs para minimizar custos.

Muroga /MUR 82a/ menciona o parâmetro "yield", definido como a taxa percentual de circuitos funcionais (testados e aprovados, segundo o conjunto de critérios empregado pelo fabricante) por número total de circuitos fabricados. Este parâmetro, ainda segundo Muroga, varia exponencialmente com a área do circuito integrado. Isto significa dizer que mesmo pequenas economias de área no "layout" do CI podem conduzir a reduções significativas no custo final do componente. O "yield" varia entre 90% para circuitos SSI e 10% para circuitos LSI, aproximadamente. Circuitos pré-difundidos desperdiçam área em comparação com circuitos pós-difundidos e dedicados. Assim, considerações de área ocupada em silício são de grande importância na escolha do estilo de projeto mais adequado à implementação de componentes integrados.

As ordens de grandeza dos parâmetros acima mencionados sofrem, contudo, constantes alterações. Os

circuitos integrados específicos para uma dada aplicação ("Application Specific Integrated Circuit" - ASIC), surgiram como resultado da alteração destes valores, viabilizando uma nova gama de CIs de alta complexidade e volume de produção relativamente baixo. Não é incomum, hoje, a obtenção de ASICs a um custo aceitável para um consumo esperado de 1000 componentes/ano, um valor duas ou três ordens de grandeza abaixo do mínimo aceitável para os circuitos de catálogo tradicionais.

Dentro da taxonomia apresentada na seção 1.1 associa-se os ASICs aos circuitos personalizáveis. Estes dois termos serão usados como sinônimos ao longo deste trabalho.

Nas seções seguintes serão vistos os seguintes tópicos:

. uma comparação dos estilos constantes da taxonomia apresentada;

. a evolução histórica dos CIs pré-difundidos;

. uma primeira visão do método tradicional usado na concepção deste tipo de CIs;

. o conjunto de aspectos econômicos que influenciam a escolha dos pré-difundidos para a implementação de um componente;

. uma avaliação da situação brasileira em microeletrônica, salientando o possível papel dos ASICs no desenvolvimento deste setor.

2.1 Comparação de Estilos de Concepção

O projetista de sistemas eletrônicos tem hoje a seu dispor uma extensa gama de opções de implementação de circuitos, gerando a necessidade de constantes tomadas de decisão ao longo da concepção de sistemas. Estas tomadas de decisão envolvem questões de natureza técnica, financeira e de mercado.

Nesta seção e na seguinte objetiva-se situar os circuitos pré-difundidos no âmbito da concepção de sistemas. Além disso, pretende-se analisar o papel destes circuitos em relação a CIs obtidos a partir do uso de estilos de projeto diferentes, apontando as vantagens e as desvantagens da aplicação de pré-difundidos na implementação de projetos eletrônicos.

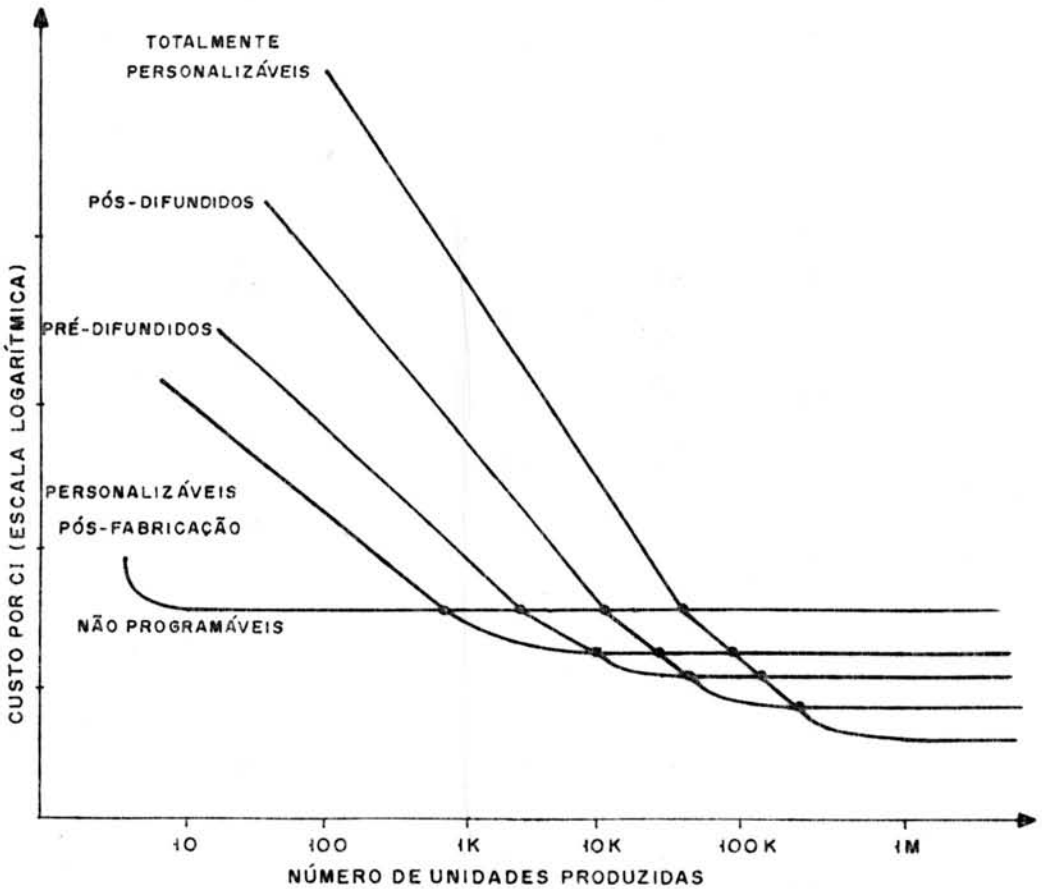


Figura 2.1 - Comparação volume x custo unitário dos estilos de concepção

A comparação de estilos de concepção de CIs se faz, via de regra, a partir de considerações do tipo volume de produção versus custo unitário do componente ou do tipo volume de produção versus número de portas lógicas equivalentes. Comparações desta natureza podem ser encontradas em /MUR 82b/, /BER 83/, /WOL 85/ e /MCM 85/. Duas das curvas obtidas aparecem nas Figuras 2.1 e 2.2, baseadas em similares de /MUR 82b/ e /WOL 85/.

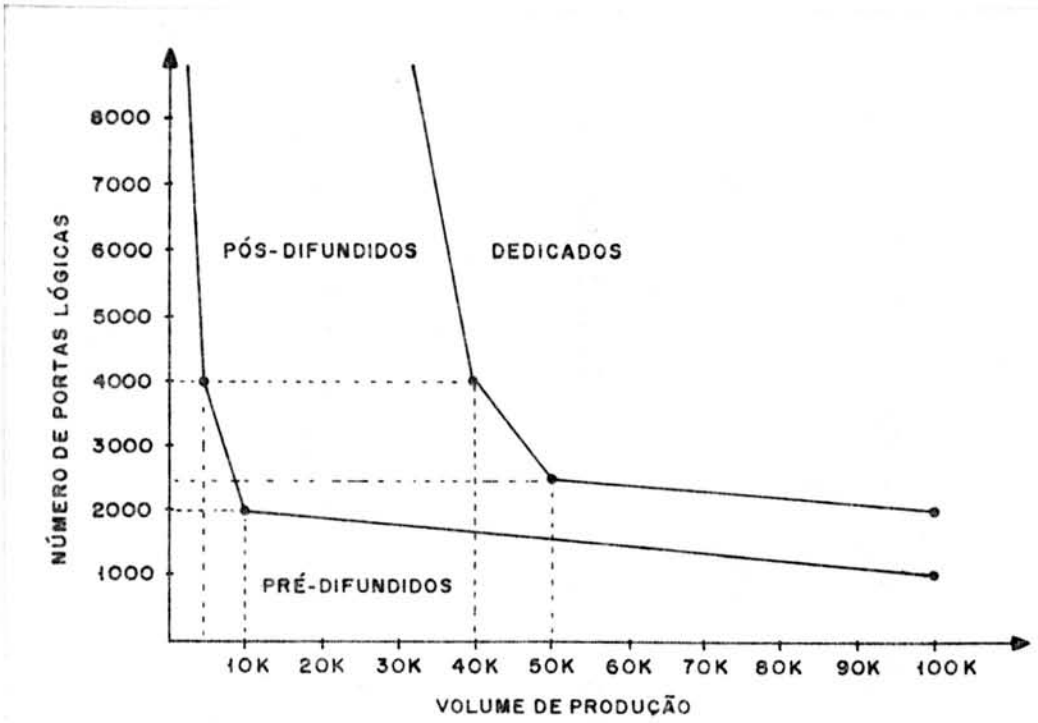


Figura 2.2 - Comparação volume x total de portas equivalentes dos CIs personalizáveis

O que se pode afirmar é que, em ambos os gráficos, está definida uma região onde se mostra mais viável o uso de um dos estilos de concepção. A interpretação das curvas, entretanto, é distinta nos dois casos. Na figura 2.1 tem-se a comparação convencional de custos para implementações usando dispositivos de catálogo, PLAs, pré-difundidos, pós-difundidos e dedicados. Esta é vista hoje como uma forma simplista de analisar o problema de custos, pois assume um CI com uma complexidade

específica conforme esclarece /WOL 85/. Ao empregar este gráfico deve-se, a exemplo do que é feito em /MCM 85/, especificar complexidades equivalentes em cada uma das implementações, a fim de se obter uma comparação válida.

A figura 2.2 traduz uma comparação mais significativa, pois define de forma clara a região onde a solução de projeto mais efetiva pode ser alcançada, segundo /WOL 85/. Uma conclusão quantitativa básica, observável nas figuras, é a de que um volume de produção acima de mil componentes já justifica, hoje, a aplicação de circuitos personalizáveis.

A última consideração acima parte do pressuposto de que a comparação entre estilos distingue os circuitos não-personalizáveis do tipo SSI/MSI dos LSI/VLSI, pois estes últimos ocupam uma posição no mercado onde os ASICs não conseguem penetrar atualmente e nem deverão fazê-lo no futuro, que é o mercado de aplicações genéricas, de alto consumo e alta competitividade entre os grandes fabricantes de semicondutores de catálogo. Um exemplo de componente com estas características são os microprocessadores e as famílias de periféricos associadas.

A separação explícita das características de custo e aplicação entre circuitos SSI/MSI e LSI/VLSI foi elaborada em estudos econômicos realizados por Fey e Paraskevopoulos /FEY 86/, onde, além das comparações convencionais, se mostra a relação existente entre o total de portas equivalentes implementadas, o custo por dispositivo e o custo total do sistema para circuitos LSI/VLSI de catálogo, circuitos dedicados, pós-difundidos, pré-difundidos e circuitos SSI/MSI de catálogo. Esta relação está ilustrada na figura 2.3 traduzida de /FEY 86/ com um levantamento para o ano de 1984 e uma projeção para 1990.

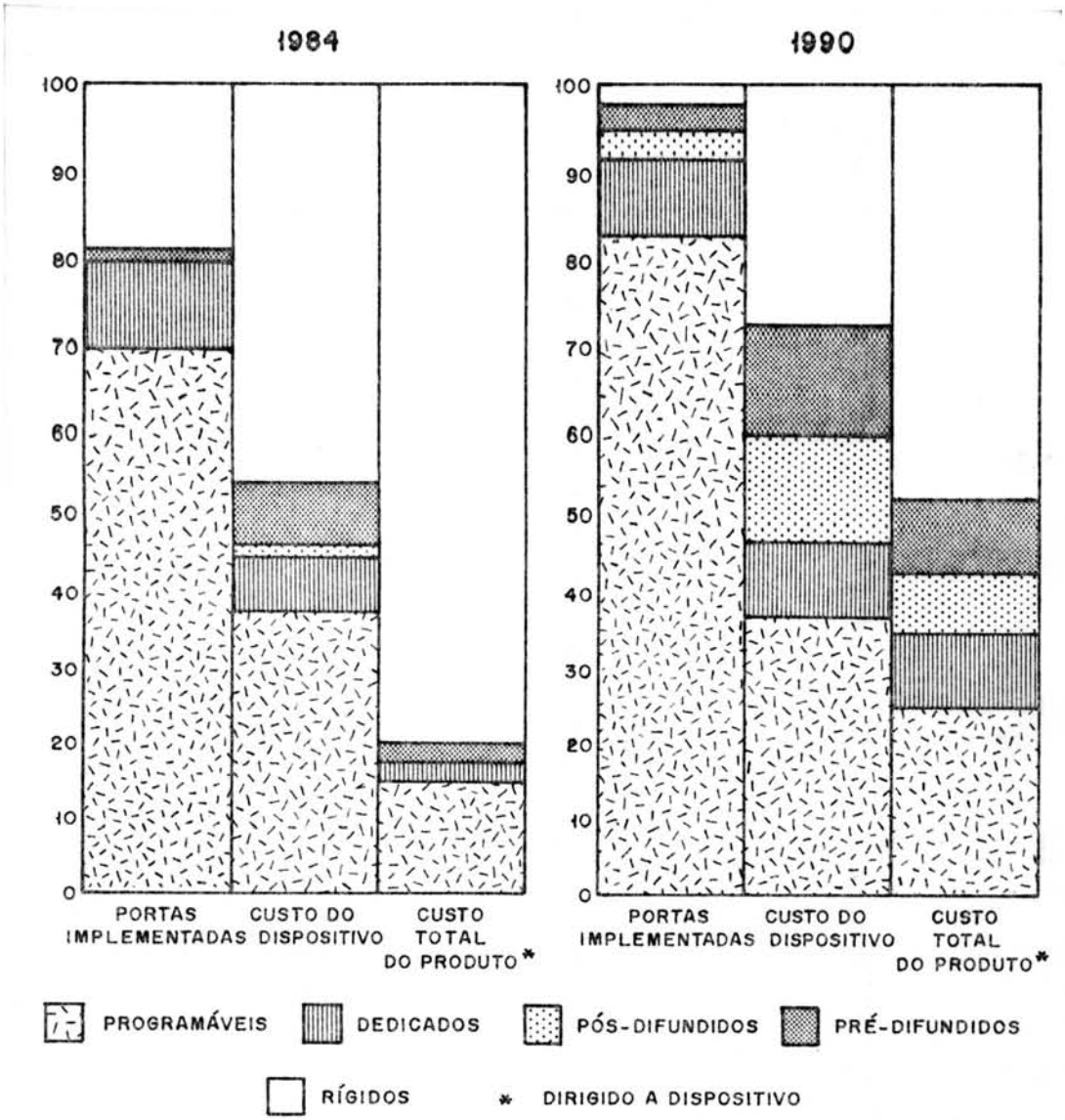


Figura 2.3 - Caracterização de mercado dos estilos de concepção

O termo porta equivalente, mencionado acima, é interpretado como um elemento de circuito com complexidade equiparável ao de uma porta lógica NÃO-E ("NAND") ou NÃO-OU ("NOR") de duas entradas. Nota-se, na figura 2.3, que:

. os circuitos não-personalizáveis LSI/VLSI estão em franca ascensão no mercado;

. os personalizáveis vão aos poucos se firmando como opção de projeto;

. os SSI/MSI encontram-se em visível declínio na sua participação em sistemas digitais eletrônicos.

Uma comparação de todos os estilos de projeto usando diferentes critérios como desempenho, tempo de projeto, e tamanho do CI resultante seria extremamente complexa. Tenta-se quebrar esta complexidade em segmentos nas sub-seções a seguir, começando com uma avaliação dos circuitos não-personalizáveis SSI/MSI em relação aos personalizáveis. A seguir, ensaia-se uma comparação dos diferentes estilos de projeto de circuitos personalizáveis, pós-fabricação, pré-difundidos, pós-difundidos e dedicados.

2.1.1 Comparação de circuitos não-personalizáveis e personalizáveis

A maior vantagem dos dispositivos não-personalizáveis é a facilidade de mudanças in-situ no projeto, seja para corrigir erros, seja para substituir peças falhas. Outra vantagem é a inexistência de despesas não recorrentes de engenharia no seu uso, como por exemplo aquelas envolvidas no projeto de um novo componente.

O preço pago por estas vantagens em relação aos dispositivos personalizáveis é, entretanto, alto. O sistema resultante possui baixa confiabilidade, resultante da proliferação de conexões entre CIs, dissipa maiores potências e ocupa mais espaço de placa. Além destas desvantagens, deve-se acrescentar uma acentuação nos custos de montagem, devido às necessidades maiores de inspeção e teste de placa, inserção de componentes, conexões, capacitores de desacoplamento, fiação, fonte de alimentação, arrefecimento e gabinetes, conforme enumerado por /MUR 82b/.

Uma última característica vantajosa dos circuitos não-personalizáveis é a facilidade de fornecimento

alternativo, não encontrada nos personalizáveis, exceto no caso dos personalizáveis pós-fabricação, como PROMs e PLAs, e ainda assim em bem menor escala, conforme cita /BRU 83/.

Os circuitos personalizáveis, por sua vez, possuem as seguintes vantagens em relação aos circuitos discutidos anteriormente:

. economia de espaço e peso;

. redução do custo do equipamento;

. aumento do desempenho;

. sigilo do projeto.

A primeira vantagem decorre do maior grau de integração. Em alguns casos, como instrumentos portáteis ou onde espaço e peso são um luxo, a exemplo de artefatos bélicos ou aeroespaciais, esta economia é indispensável /LEE 83/.

Quanto à segunda vantagem, embora a comparação direta entre o custo do circuito personalizável e o custo da implementação equivalente SSI/MSI favoreça quase sempre a última /FEY 86/, a soma final de economias nos custos de placa, testes e montagem supera por uma boa margem esta diferença, conforme mostram os estudos de /FEY 86/ e /BER 83/.

A terceira vantagem baseia-se na redução do comprimento das interconexões, causada pela integração. Esta redução diminui os atrasos de propagação do projeto, além de reduzir as capacitâncias parasitas em várias ordens de grandeza.

Embora não seja impossível extrair a

funcionalidade de um circuito personalizável, mediante o uso de técnicas de engenharia reversa, conforme ilustrado em /BIE 89/, esta tarefa revela-se muita mais árdua e consome muito mais tempo do que analisar uma placa de circuito impresso construída apenas com CIs de catálogo. A última vantagem deriva diretamente deste aspecto.

A despeito destas vantagens, a decisão entre usar ou não uma solução baseada em circuitos personalizáveis não é das mais simples. Existe uma série de riscos envolvidos no projeto e obtenção destes CIs. O mais crítico deles relaciona-se ao custo associado com o projeto do componente e o necessário relacionamento que deve haver entre o fornecedor de CIs personalizáveis e o cliente interessado em usá-los.

Este último requisito será tratado na seção 3.3, onde será discutida a relação cliente/fornecedor para o caso específico de circuitos pré-difundidos.

2.1.2 Comparação entre os diversos estilos de projeto usando circuitos personalizáveis

Após a opção por implementar parte ou todo um sistema eletrônico com circuitos personalizáveis ser tomada, resta um caminho razoável a ser percorrido ao longo das diversas possibilidades de implementação disponíveis no mercado. Duas características fundamentais do sistema alvo devem ser consideradas de início: o volume de produção esperado e o desempenho exigido.

Os circuitos dedicados oferecem a máxima frequência de operação e o CI mais compacto. Contudo, este estilo apresenta um investimento inicial alto, da ordem de milhões de dólares, além de um cronograma que muitas vezes se estende ao longo de anos /TEL 85/. O esforço dispendido

na geração das máscaras para este estilo é medido em vários homens-ano. A probabilidade de falha no projeto inicial é muito alta. A correção de erros de projeto é difícil e consome grandes quantidades de recursos, materiais e humanos. O uso de ferramentas de PAC é limitado pela necessidade de otimização global, ainda pouco viável com os recursos computacionais existentes.

Reserva-se assim este estilo para projetos com volume de produção esperado pelo menos acima de cem mil ou um milhão de componentes anuais, e cuja aplicação possa suportar o cronograma de desenvolvimento sem se tornar obsoleta, segundo Muroga /MUR 82b/.

No outro extremo de custo de personalização e rapidez para se obter protótipos encontram-se os personalizáveis pós-fabricação. As vantagens deste estilo residem na sua fácil obtenção por fontes alternativas e tempo de personalização muito curto, uma vez sabido o comportamento esperado. Este tempo é medido em horas ou mesmo minutos, facilitando a prototipação rápida. Por outro lado, sua densidade típica está muito abaixo daquela dos personalizáveis por-fabricação. Além disso, a quantidade de perdas no momento da personalização pode atingir até 20% dos componentes /BRU 83/. O desempenho comparativo destes circuitos é baixo, resultado de uma arquitetura genérica e inflexível.

Na região entre os dedicados e os personalizáveis pós-fabricação encontram-se os circuitos semi-dedicados, constituindo um compromisso de custo e desempenho no projeto de sistemas eletrônicos. Procede-se agora a uma comparação dos dois estilos de projeto para circuitos semi-dedicados: os pré-difundidos ou "gate arrays" e os pós-difundidos ou "standard cells".

Do ponto de vista de manufatura, a diferença

capital entre estes dois estilos reside no número de máscaras projetadas para a obtenção de um CI pré-caracterizado. No caso dos pré-difundidos este varia, normalmente, entre 1 e 4, enquanto que nos pós-difundidos o valor está entre 8 e 12, de acordo com /BER 83/. Esta diferença conduz às quatro conclusões abaixo, extraídas por Beresford em /BER 83/:

. pré-difundidos possuem custo de desenvolvimento mais baixo;

. pré-difundidos são fabricados mais rapidamente;

. pós-difundidos são mais flexíveis;

. pós-difundidos são mais densos.

A primeira conclusão se apóia no menor número de máscaras necessárias para o projeto de pré-difundidos e no maior número de iterações de geração automática das geometrias das máscaras ("layout") nos pós-difundidos. Este número adicional de iterações deriva do fato de ser sempre possível concluir o traçado de rotas automaticamente no estilo de concepção pós-difundido.

Ao lado do fato dos pré-difundidos necessitarem menos máscaras, o fornecedor pode manter lâminas de silício pré-processadas em estoque, economizando o tempo de muitos dos passos de manufatura. A segunda conclusão resulta desta característica.

Estruturas regulares densas como RAMs, ROMs e PLAs podem ser facilmente implementadas em pós-difundidos, ao contrário do que ocorre com "gate arrays", caracterizando a terceira conclusão.

Pré-difundidos necessariamente desperdiçam alguma

área do CI, ainda que o "layout" manual possa gerar aproveitamento de 95 a 98% dos componentes do arranjo. A última conclusão deriva destas considerações.

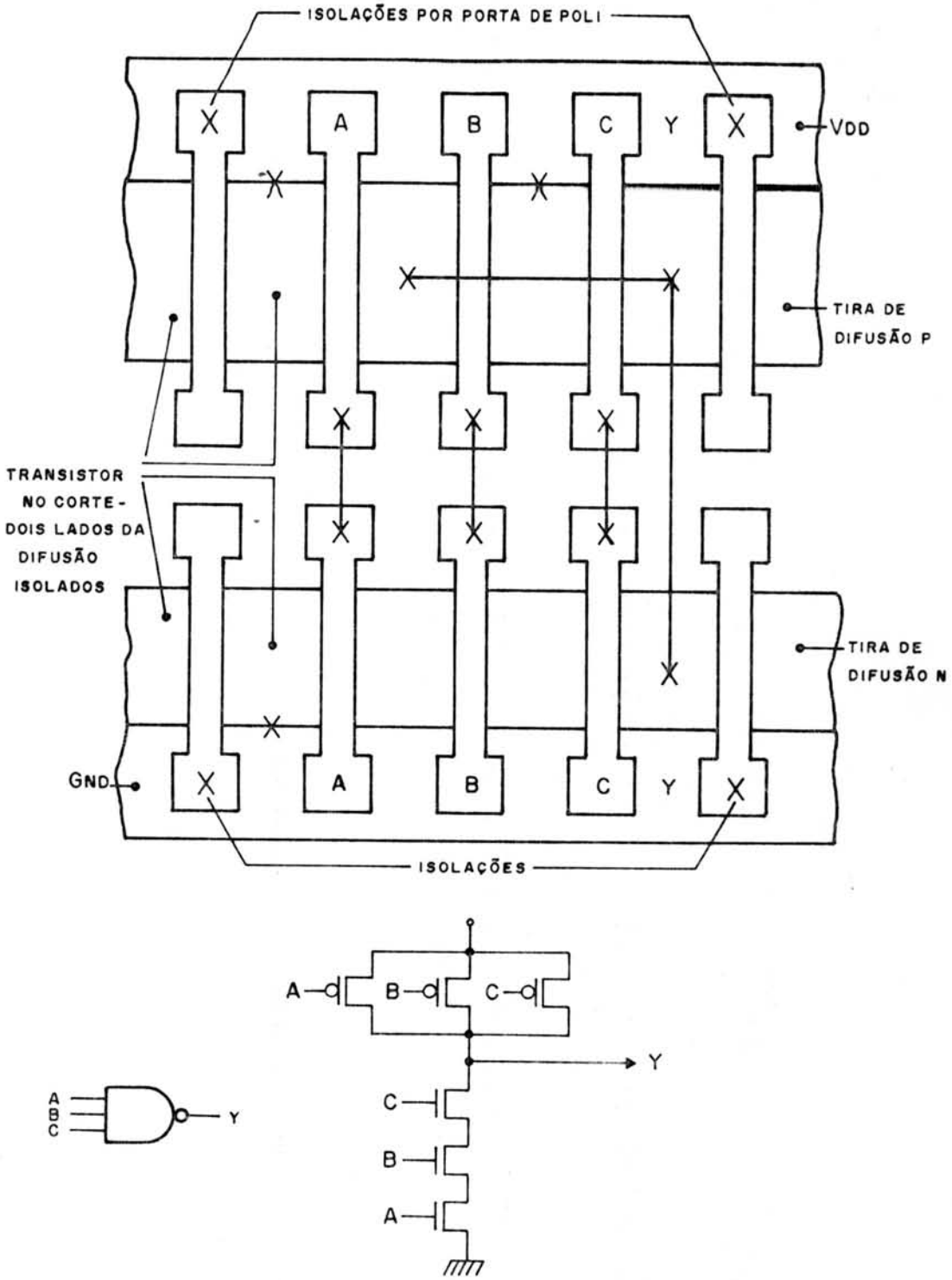


Figura 2.4 - Isolação por porta de poLI

Tabela 2.1 - Comparação qualitativa dos estilos de concepção

	TIPO DE CI					
	NÃO PERSONALIZÁVEIS		PERSONALIZÁVEIS			
CRITÉRIO	RÍGIDOS	PROGRA-MÁVEIS	PÓS-FABRICAÇÃO	PÓS-DIFUNDIDOS	PRÉ-DIFUNDIDOS	DEDICADOS
CUSTO COM BAIXO VOLUME DE PRODUÇÃO	BAIXO	BAIXO	BAIXO	ALTO	MÉDIO	MUITO ALTO
CUSTO COM ALTO VOLUME DE PRODUÇÃO	BAIXO	BAIXO	BAIXO	BAIXO	MÉDIO	MUITO BAIXO
DENSIDADE DE INTEGRAÇÃO	MUITO BAIXA	MUITO ALTA	MÉDIA	ALTA	MÉDIA OU ALTA	MUITO ALTA
APLICAÇÃO EM SISTEMAS	GENÉRICA	GENÉRICA	MAIS OU MENOS GENÉRICA	ESPECÍFICA	ESPECÍFICA	ESPECÍFICA
ALTERAÇÕES DE PROJETO DO SISTEMA	TRIVIAL	TRIVIAL	FÁCIL	DIFÍCIL	DIFÍCIL	MUITO DIFÍCIL
DESPESAS DE ENGENHARIA NÃO-RECORRENTES	NÃO EXISTEM	NÃO EXISTEM	BAIXAS	ALTAS	MÉDIAS	MUITO ALTAS
CONFIABILIDADE DO SISTEMA	BAIXA	MÉDIA	BAIXA	MUITO BOA	MUITO BOA	MUITO BOA
ECONOMIA DE ESPAÇO	NÃO PROPORCIONA	BOA	RAZOÁVEL	MUITO BOA	MUITO BOA	EXCELENTE
DESEMPENHO DO SISTEMA	BAIXO	BAIXO	BAIXO	ALTO	MÉDIO	MUITO ALTO
SIGILO DE PROJETO	NENHUM	BAIXO	RAZOÁVEL	ALTO	ALTO	ALTO
TEMPO DE PROJETO	MUITO CURTO	MUITO CURTO	CURTO	LONGO	MÉDIO	MUITO LONGO
FLEXIBILIDADE DE PROJETO	MÉDIA	BAIXA	BAIXA	ALTA	MÉDIA	MUITO ALTA

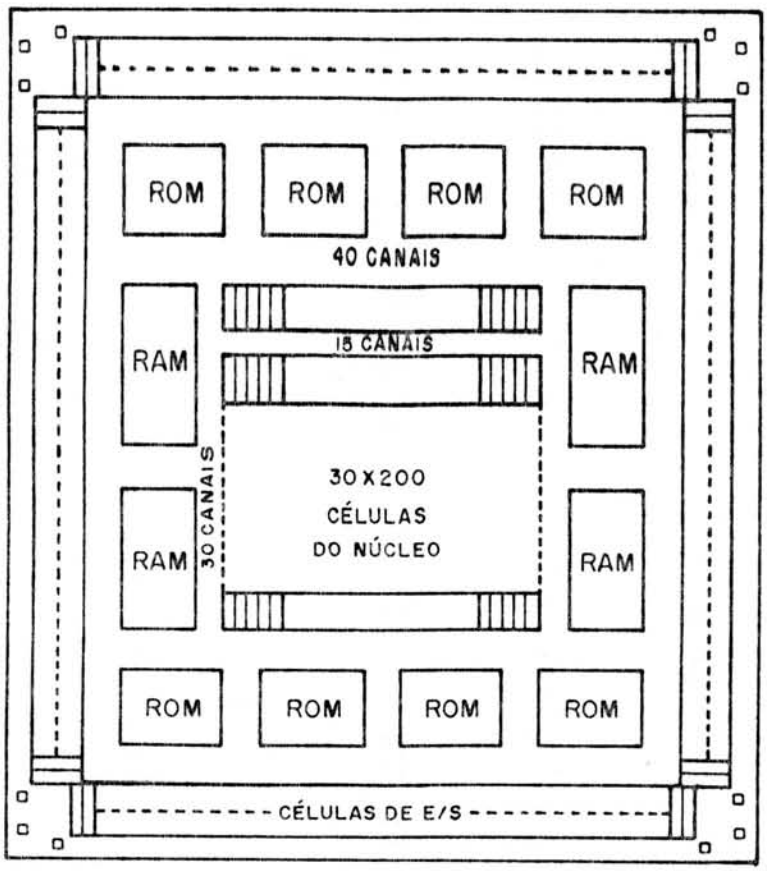


Figura 2.5 - Pré-difundido com blocos configuráveis

Na realidade, a evolução dos estilos de projeto vem tendendo a tornar mais difícil a decisão sobre qual a abordagem mais vantajosa, uma vez que os pontos altos de cada estilo vão sendo aos poucos incorporados aos outros, tornando menos distinguível o limite entre os estilos. Como ilustração de tal tendência tem-se, para o caso dos circuitos pré-difundidos, o uso da técnica de isolação por porta de polissilício descrito em /SAK 85/, com o objetivo de reduzir o desperdício de área no substrato semicondutor, e a presença de blocos configuráveis de alta densidade, como exemplificado por Miyahara em /MIY 86/. As figuras 2.4 e 2.5 ilustram a técnica de isolação por porta de polissilício e blocos configuráveis no "layout" de CIs pré-difundidos, respectivamente. A tabela 2.1 apresenta um resumo comparativo dos diferentes estilos de projeto, baseado nos critérios abordados.

2.2 Pré-difundidos - Evolução

Os circuitos pré-difundidos não constituem uma idéia nova. Os primeiros "gate arrays" comerciais foram introduzidos pela empresa Fairchild Inc., em 1967, empregando tecnologia DTL ("Diode-Transistor Logic") /MUR 82b/. Sua importância, contudo, não cresceu até meados da década de 70, quando a discrepância entre os níveis de integração dos dispositivos de memória e dos dispositivos lógicos se tornou marcante, conforme salienta /BRU 83/. Os primeiros alcançavam escalas de integração LSI e VLSI, enquanto que os últimos eram predominantemente compostos por CIs do tipo SSI/MSI. Assim, tornou-se necessário compatibilizar o nível de integração em projetos complexos, evitando que os dispositivos lógicos se tornassem um gargalo intransponível de custo e desempenho.

Em 1975, a Amdahl Corporation usou circuitos pré-difundidos de forma extensa pela primeira vez, nos computadores da série 470. Estas máquinas eram compatíveis com a família IBM 370, então considerada das mais rápidas em termos de computadores de grande porte ("mainframes") de uso geral. A série 470 superou os IBM 370 em velocidade, dobrando o desempenho de programas no caso do modelo mais avançado, o 470/V6, que continha 2000 "gate arrays" de 110 tipos diferentes.

Quatro anos após os lançamentos da Amdahl, a própria IBM anunciava o sistema 4300, com desenvolvimento baseado em "gate arrays". No mesmo ano, a IBM anunciou a integração da Unidade Central de Processamento (UCP ou, "Central Processing Unit" - CPU) do 370 em um único "chip" pré-difundido. Enquanto isso a Digital Equipment Corporation (DEC) noticiou o lançamento do mini-computador que se tornaria um padrão industrial, o VAX 11/750, com 90% de sua lógica implementada mediante uso de "gate arrays" /BRU 83/.

Considera-se útil apresentar a série de motivos que levou os fabricantes de computadores de grande e médio porte a serem os pioneiros no uso e na difusão das vantagens dos circuitos pré-difundidos:

. o uso de dispositivos de catálogo nestas aplicações tornava difícil aumentar o desempenho das máquinas em relação aos competidores, pois todos tinham acesso aos mesmos dispositivos;

. o uso de pré-difundidos garantia a natureza proprietária do projeto;

. o tamanho dos CIs projetados não precisava ser mínimo, pois outros componentes do sistema eram mais caros que toda a lógica (exemplos eram o subsistema de memória e os periféricos);

. o tamanho dos CIs não podia também ser muito grande (como era o caso quando se aplicavam circuitos de catálogo), sob pena de sacrificar a velocidade dos caminhos críticos;

. se cada CI fosse projetado mediante uso do estilo dedicado de concepção, o custo cresceria muito, pois estes sistemas precisavam de uma grande variedade de tipos de CIs, cada um com um pequeno volume de produção;

. devido à complexidade do projeto de um computador de grande/médio porte ser alta, o fabricante era obrigado a usar extensamente recursos de PAC, os quais proporcionavam ganhos significativos em termos de tempo e confiabilidade no caso do uso de circuitos pré-caracterizados;

. a grande quantidade de interconexões inter e

Intra CIs tornava imprescindível o uso de ferramentas de PAC para particionamento, posicionamento e traçado de rotas, cujos algoritmos funcionam com grande eficiência se aplicados às estruturas regulares presentes nos pré-difundidos /MUR 82b/.

Atualmente, uma gama bastante variada de tecnologias encontra-se à disposição dos projetistas de CIs pré-difundidos:

. tecnologias de alto desempenho, como TTL, ECL, CML, IIL, TRL e GaAs;

. tecnologias com alta capacidade de integração, como NMOS, CMOS e CMOS/SOS.

Em termos de exemplos de limites atuais da tecnologia, encontra-se por exemplo, "gate arrays" com 1000 a 2000 portas equivalentes implementados em GaAs. Estes circuitos apresentam velocidade de propagação abaixo de 200ps para uma porta lógica, além de imunidade à radiação da ordem de 10E8 rads, ou seja, 100 a 1000 vezes a imunidade obtida com tecnologias de silício /BUR 85/.

Do lado da densidade de integração, encontra-se pré-difundidos na tecnologia CMOS com até 130.000 portas equivalentes, das quais considera-se viável aproveitar 50.000 (40% de utilização) /CAR 86/.

Além das características de desempenho e densidade de integração, um conjunto de técnicas importadas de outros estilos de concepção aumenta hoje a flexibilidade de aplicação dos pré-difundidos:

. blocos otimizados em área, com arquiteturas especiais para uso específico, como RAMs, ROMs, multiplicadores, registradores, unidades lógico-aritméticas

e controladores conforme mostrado em /NAK 86/ e /MIY 86/;

. estruturas de teste como apresentadas por /RES 83/ e /NAK 86/;

. células de base com organização flexível capaz de suportar o projeto otimizado de lógica e memória num mesmo CI, como descrito em /TAK 85/ e /CAL 87a/.

2.3 Pré-difundidos - Concepção Tradicional

Além do uso de CIs pré-difundidos mencionado na seção anterior, uma aplicação teve papel fundamental na difusão deste estilo de concepção entre os projetistas de sistemas: a substituição de conjuntos de "chips" SSI/MSI de uma placa de circuito impresso por um único ou alguns poucos CIs pré-difundidos que implementam o mesmo comportamento desta `lógica de ligação' ("glue logic").

Esta aplicação é, em última análise, a maior responsável pela popularidade atual dos "gate arrays" /PIT 81/. Isto se deve à rapidez de sua implementação, uma vez que se trata de incrementar as qualidades de um projeto já pronto, muitas vezes já testado e aprovado, ocasionando um mínimo de custos de reprojeto.

O objetivo desta seção é discutir as formas de implementação tradicionais de pré-difundidos, tendo em vista as duas aplicações mencionadas, quais sejam, a implementação de sistemas computadores de médio e grande porte e a substituição de conjuntos de CIs SSI/MSI em projetos já implementados. Estas são, hoje, as mais típicas do estilo de projeto "gate arrays".

A concepção tradicional de CIs pré-difundidos baseou-se, a princípio, no uso de técnicas de projeto

manual. Um mínimo de tarefas eram executadas com o auxílio do computador, restringindo-se à documentação e digitalização das descrições de máscaras como mencionado em /AMI 82a/, /AMI 82b/ e /AMI 82c/. A necessidade de encurtar o tempo de projeto levou ao desenvolvimento de um conjunto de ferramentas de automação das tarefas mais caras em termos de recursos, ou seja a geração de máscaras e a validação do esquema lógico.

Os sistemas de PAC eram tradicionalmente instalados em centros dotados de infraestrutura completa, construídos em torno de computadores de grande porte. Estes centros pertenciam a grandes fabricantes de computadores ou a grandes fornecedores de componentes como os citados no início deste capítulo. Os grandes fabricantes de computadores realizavam todas as tarefas de cada etapa de projeto e fabricação dos dispositivos de maneira interna ("in-house").

Os clientes que optavam pela substituição de lógica de ligação por circuitos pré-difundidos, contudo, eram empresas muitas vezes de pequeno ou médio porte, o que impossibilitava um grande investimento na aquisição da infraestrutura completa de projeto e fabricação. Tal fato criou a necessidade de interação entre os fornecedores de pré-difundidos e os clientes interessados nestes componentes.

Normalmente, o fornecedor colocava à disposição do cliente o seu centro de projetos, dando treinamento no uso das ferramentas de "hardware" e "software" necessárias. O envolvimento do cliente se limitava, em geral, à obtenção de um diagrama lógico validado do circuito, deixando os passos posteriores do projeto para realização pelo fornecedor. Este possuía o "know-how" necessário à manipulação dos níveis de abstração inferiores do projeto: geração de máscaras, fabricação e teste do componente.

Embora esta forma de interação seja ainda bastante difundida e defendida por alguns fabricantes como a LSI Corporation /LOB 83/ /MAD 83/, outras formas surgiram, sendo algumas delas descritas em detalhe na seção 3.3 adiante.

Quanto ao aspecto ferramentas, na concepção tradicional o cliente usava um conjunto fracamente acoplado de programas ("toolbox") e máquinas que executavam cada uma das tarefas de projeto de maneira mais ou menos hermética. As operações auxiliadas por estas ferramentas consistiam da captura do projeto lógico, verificação básica e geração de vetores de teste /MAD 83/. A captura do projeto lógico era obtida via uma descrição hierárquica de módulos, seja a partir de uma linguagem textual de descrição, seja através de uma ferramenta gráfica interativa de captura de esquemas.

A verificação básica, por sua vez, era alcançada via simulação hierárquica ascendente ("bottom up") dos módulos, iniciando nos módulos dos níveis inferiores da hierarquia e agregando módulos validados até atingir a raiz, esta correspondendo à lógica completa. Os vetores de teste para o componente eram extraídos dos estímulos gerados pelo cliente para exercitar o circuito durante a simulação.

Terminadas estas tarefas, cabia ao fornecedor completar os passos de concepção, o que compreendia geração das máscaras de personalização a partir de: esquema lógico validado, bibliotecas de células pré-definidas e ferramentas automáticas de posicionamento de módulos e traçado de rotas. Estas ferramentas nem sempre conseguiam gerar 100% das conexões, obrigando a intervenção de especialistas na geração interativa das conexões inacabadas.

A forma tradicional de concepção de pré-difundidos possui, no entanto, algumas falhas, que vão dia a dia sendo corrigidas, mediante o desenvolvimento de novos métodos de abordagem do problema. Por exemplo, a testabilidade é hoje vista como parte integrante de qualquer projeto de componente, sendo abordada desde a especificação do mesmo.

Outro aspecto hoje reforçado diz respeito à integração das diversas ferramentas de auxílio ao projeto, capaz de proporcionar uma transição suave entre os vários níveis de abstração presentes no projeto. Tal integração é alcançada por meio de bancos de dados especialmente desenvolvidos como os citados por Katz /KAT 83/, Morschel /MOR 88/ e Golendziner /GOL 88/.

Por último, os níveis superiores de abstração de projeto já contam hoje com várias ferramentas para sua manipulação, principalmente no que diz respeito à síntese automatizada de uma descrição de nível de abstração mais baixo a partir de uma descrição de nível mais alto /SAS 86/.

2.4 Pré-difundidos - Aspectos Econômicos

Como já foi anteriormente mencionado, a escolha de um estilo de projeto para a implementação de todo ou perte de um sistema digital não é das mais simples. As considerações envolvidas recaem sempre nos aspectos de viabilidade econômica da implementação. O objetivo desta seção é fornecer algumas informações quantitativas sobre os custos envolvidos em um projeto de CI pré-difundido.

O custo de um sistema digital possui vários componentes. Quando se usam circuitos personalizáveis, um

dos componentes assume grande valor, as chamadas despesas não recorrentes de engenharia ("non-recurring engineer expenses" - NRE). Este fator está vinculado a itens como custo inicial de projeto, tempo de UCP gasto no projeto, fabricação das máscaras e produção de protótipos /BER 83/. Baseado no NRE, comparações entre custos de implementação foram realizadas por Beresford /BER 83/ e Dicken /DIC 83/. As comparações são transcritas nas tabelas 2.2 e 2.3, relacionando projetos equivalentes implementados sobre pré-difundidos, de um lado, e pós-difundidos e dedicados do outro, respectivamente.

Tabela 2.2 - Comparação pré-difundido x pós-difundido

Comparação de Custos para um CI de 1000 Portas Equivalentes, Encapsulamento Plástico de 40 Pinos - Faixa de Temperatura Comercial, Totalmente Digital, Lógica Aleatória.

ITEM NRE	PRÉ-DIFUNDIDO *	PÓS-DIFUNDIDO *
PROJETO E "LAYOUT"	16	23
TEMPO DE COMPUTADOR	5	6
CONFECÇÃO DAS MÁSCARAS	5	13
PRODUÇÃO DE PROTÓTIPOS	5	5
TOTAL DE DESPESAS NRE	31	47

* QUANTIA (EM MILHARES DE DÓLARES)

O NRE também foi extensamente usado por Fey /FEY 86/ em suas comparações de custos de produtos. Além do NRE, Fey utilizou outra ferramenta de avaliação, denominada de análise "break-even". Esta se baseia na pesquisa do ponto exato onde os custos de duas implementações com estilos de projeto diferentes se equivalem, dadas condições distintas para os parâmetros de concepção, tais como: total de CIs construídos, número de pinos de E/S e complexidade das implementações.

Tabela 2.3 - Comparação pré-difundido x dedicado

Custos para um CI de 2000 Portas Equivalentes

	PASSOS	QUANTIA (EM MILHARES DE DÓLARES) CI DEDICADO TAMANHO 25 Kmils	CI PRÉ-DIFUNDIDO TAMANHO 80Kmils
CUSTOS DE PROJETO	DEFINIÇÃO DO PRODUTO	20	20
	PROJETO LÓGICO E SIMULAÇÃO	50	10 *
	PROJETO DE CIRCUITO E SIMULAÇÃO	35	—
	PROJETO DAS MÁSCARAS	30	5 *
	DIGITALIZAÇÃO E VERIFICAÇÃO	20	5 *
	"PROTOPLOT" E MÁSCARAS	15	5 *
SUBTOTAL DE CUSTOS DE PROJETO		170	45
CUSTOS DE PROTOTIPAÇÃO	FABRICAÇÃO DE "WAFERS" PROTÓTIPOS	10	5 *
	GERAÇÃO DO PROGRAMA DE TESTES	20	5 *
	DIAGNÓSTICOS E TESTE (SE FOR O CASO)	25	10
	INTERAÇÕES DE PROJETO (SE FOR O CASO)	30	10
	CARACTERIZAÇÃO DO PRODUTO	20	—
	MELHORAMENTO DO "YIELD"	25	—
SUBTOTAL DE CUSTOS DE PROTOTIPAÇÃO		130	30
TOTAIS GERAIS		300	75

* DESPESAS INICIAIS DA FUNDIÇÃO

Apoiado nestes estudos, pode-se particionar o custo de um CI pré-difundido em três porções: o custo de fabricação do componente, o custo de teste do componente e o lucro esperado pelo fornecedor. A parcela mais significativa é o custo de fabricação. Este depende de variáveis como: área do CI, "yield", ou seja, número de CIs funcionais por lâmina ("wafer") fabricada, maturidade do processo de fabricação, complexidade do processo, número de pinos do CI resultante e tipo de encapsulamento /PIT 81/.

Pitts /PIT 81/ afirma, com base em análises de área ocupada pelo CI, número de defeitos por lâmina e "yield", que o custo de um CI pré-difundido cresce exponencialmente com a área.

O número de pinos de E/S, junto com o quesito faixa de temperatura de operação, determina a escolha do encapsulamento do CI (plástico, cerâmico, "pin-grid-array" - PGA, etc). Esta escolha assume papel importante no cálculo final do preço do circuito, pois o encapsulamento pode custar desde alguns centavos de dólar até vários dólares, como aponta Dicken /DIC 83/. A tabela 2.4 ilustra o custo típico para os encapsulamentos mais usados com CIs pré-difundidos.

Tabela 2.4 - Custo do encapsulamento de um CI

Custo do Encapsulamento (em Dólares)

	NÚMERO DE PINOS				
TIPO DE ENCAPSULAMENTO	16	24	40	64	84
DIP PLÁSTICO COM PINOS DOURADOS	0,045	0,095	0,163	1,410	—
DIP PLÁSTICO COM PINOS PRATEADOS	0,026	0,076	0,083	1,260	—
CERDIP	0,167	0,402	0,762	—	—
"CHIP CARRIER" SEM PINOS - METÁLICO	1,120	1,660	2,270	3,700	5,900
"CHIP CARRIER" SEM PINOS - CERÂMICO	0,470	0,880	1,090	1,680	3,150
DIP CERÂMICO COM PINOS DE METAL	1,640	2,200	3,400	6,950	10,400

Um último fator econômico de difícil avaliação é mencionado por Fey, qual seja, o custo de cronograma. Este custo afeta dois aspectos da organização interessada em CIs personalizáveis: os lucros e a taxa de aprendizado da equipe de projeto. Fey estima que um aumento de 12 meses no

cronograma de um projeto equivale a aumentar entre 25 e 50% o custo unitário de fabricação ("unit manufacturing cost" - UMC) do produto final que irá empregar o "chip". Fey cita ainda que atrasos podem afetar cronogramas de três maneiras fundamentais:

. pela erosão de preços;

. causando atrasos nos fluxos de caixa;

. provocando aumentos nos custos de desenvolvimento.

Fatores econômicos desempenham um papel crítico na escolha do estilo de concepção a adotar. Levantamentos quantitativos cuidadosos devem sempre ser efetuados antes de se decidir por uma forma de implementação.

A existência de um método de concepção que leve em conta os aspectos de mercado e os dados quantitativos levantados é altamente desejável. Contudo, considera-se que o estudo deste problema foge ao escopo do atual trabalho, concebido com um caráter eminentemente técnico. A discussão de tais problemas será restrita, no restante deste volume, a considerações qualitativas. Estas considerações destinam-se a esclarecer decisões tomadas no decorrer do desenvolvimento do método CIPREDI.

2.5 Pré-difundidos no Contexto Nacional

A posição do Brasil no campo da alta tecnologia não é, definitivamente, das mais confortáveis. A escassez de recursos técnicos e a deficiência de profissionais qualificados se aliam para dificultar o desenvolvimento de uma indústria competitiva a nível internacional, ou mesmo uma indústria capaz de de suprir as necessidades de alta

tecnologia do mercado interno.

No campo da microeletrônica, a necessidade de desenvolvimento de múltiplas áreas de conhecimento de forma simultânea torna o problema ainda mais difícil de ser resolvido.

O principal insumo necessário para alcançar a capacitação em microeletrônica é, sem dúvida, a indústria de informática. A reserva de mercado para informática /TAV 85/ surge, hoje, como uma maneira de prover um ambiente propício ao estabelecimento de uma indústria de força para este setor. Contudo, a dificuldade de obtenção de equipamentos dotados de tecnologia de ponta, indispensáveis em microeletrônica, constitui ainda um entrave à evolução dos centros de pesquisa e empresas dedicados à exploração das possibilidades desta ciência.

Quatro instituições nacionais possuem, atualmente, capacitação para o projeto de circuitos integrados LSI comercializáveis, sendo três delas empresas privadas - SID Microeletrônica, Itaucom e Elebra Microeletrônica -, e uma estatal, a Telebrás, através de seu Centro de Pesquisa e Desenvolvimento (CPqD). A empresa SID Microeletrônica possui ainda a única fundição de silício operacional instalada no Brasil. Ao lado destas empresas, o Centro Tecnológico para Informática (CTI) e alguns centros universitários como a Universidade Federal do Rio Grande do Sul (UFRGS) /REI 87/ apresentam condições de projetar circuitos e implementá-los através de convênios com instituições no exterior.

Estas implementações são obtidas através dos chamados Circuitos Multi-Projeto, uma técnica onde vários CIs distintos são difundidos sobre um único "wafer" de semicondutor, sendo o custo rateado entre os diversos projetos presentes na lâmina. Esta técnica viabiliza a

prototipação de circuitos e a validação de projetos de cunho acadêmico ou experimental.

A fundição de silício mencionada anteriormente foi implantada como uma "joint-venture" de duas multinacionais: a RCA e a Philco-Ford. A empresa resultante, denominada Phibrase, inviabilizou-se legalmente com o advento da Lei de Informática. O patrimônio físico da Phibrase foi então adquirido pela SID Microeletrônica. A compra de tecnologia não foi efetuada.

A fábrica compreende uma linha de difusão de circuitos bipolares analógicos. Os componentes atualmente produzidos são circuitos de catálogo aplicados em equipamentos de áudio e vídeo, como por exemplo, amplificadores operacionais e decodificadores AM/FM. Os componentes são comercializados no mercado nacional e também exportados.

Os insumos para o funcionamento da fundição são quase todos importados. Os produtos químicos são obtidos em parte de empresas nacionais, mas a maioria precisa ser importada. Os "wafers" de silício para produção são importados, já existindo, entretanto, empresas nacionais cujo produto está em nível de qualificação para uso na fábrica. As máscaras fotolitográficas são totalmente confeccionadas fora do País, não existindo perspectiva para a implantação de uma facilidade deste tipo no Brasil, pelo menos a curto prazo.

O mercado de componentes eletrônicos no Brasil aplica principalmente circuitos analógicos bipolares. Circuitos digitais construídos com tecnologia MOS são ainda pouco usados, mas sua participação vem crescendo gradativamente.

A evolução de equipamentos de projeto e

manufatura de componentes tem sido baseada no fornecimento de incentivos fiscais pelo Governo. As três empresas privadas atuando nesta área comprometeram-se, em troca destes incentivos, a instalar linhas de produção de circuitos CMOS com tecnologia avançada - pelo menos 2 microns de comprimento de canal - até 1990. A viabilidade destas metas e a utilidade de serem cumpridas dentro do prazo especificado parecem, ainda, pouco claras.

Cada uma das empresas privadas com capacitação para o projeto funciona vinculada a uma ou mais fundições de silício no exterior, sendo estas responsáveis pela manufatura dos projetos realizados nas empresas. SID, Elebra e Itaucom atuam ainda como intermediárias entre as fundições e usuários interessados na aplicação de ASICs. As empresas promovem cursos de treinamento em circuitos pré-caracterizados para engenheiros de sistemas, colocando à disposição destes suas ferramentas de "software" e "hardware" para o projeto de CIs.

As ferramentas de "software" usadas nestas empresas e no CPqD-Telebrás são de dois tipos básicos:

. fornecidas pela fundição para "hardware" genérico;

. fornecidas por terceiros compatíveis com os processos do fabricante usado.

Por exemplo, no CPqD-Telebrás usa-se o simulador lógico HILO-2 da empresa GenRad, funcionando em um minicomputador de uso genérico, o VAX 11/785 da Digital Equipment Corporation. Acoplado a este existem ferramentas de geração interativa do "layout" de CIs, executando sobre estações de trabalho Applicon. O "layout" de pré-difundidos é gerado com base em bibliotecas de células fornecidas pela fundição, no caso a AMI Inc /TEL 85/. Na SID, por outro

lado, os projetos são implementados usando estações de trabalho da Mentor Graphics, com ferramentas de captura de projeto fornecidas pela mesma empresa e bibliotecas compatíveis com as da fundição VTI.

Dos centros envolvidos diretamente com a concepção de circuitos integrados comercializáveis, o CPqD-Telebrás aparece como o melhor aparelhado e com maior experiência. Mais de 10 circuitos LSI projetados pela sua equipe foram implementados com sucesso. A maior parte destes circuitos foram implementados segundo o estilo dedicado, alguns sendo "standard cells". Os CIs foram concebidos nas tecnologias NMOS e CMOS digital e fabricados pela AMI Inc. São todos circuitos aplicados em telefonia (detetor de tons MFC, interface lógica para equipamento Multiplex, etc). Atualmente, está em fase de projeto um CI dedicado para ser usado na Rede Digital de Serviços Integrados.

Alguns circuitos pré-caracterizados foram concebidos por Itaucom, SID e Elebra, seja para aplicação em equipamentos da própria empresa, seja para uso em equipamentos de terceiros /SEG 86/. A falta de informações relativas ao uso de ASICs em equipamentos nacionais, consideradas em geral como classificadas, não permite qualquer conclusão definitiva sobre a viabilidade atual deste tipo de mercado. Entretanto, a existência de centros de projeto funcionais pertencentes a empresas privadas demonstra, pelo menos, a existência de um potencial de consumo de ASICs.

Uma observação qualitativa de mercado mostra, por outro lado, um baixo volume comercializado de equipamentos onde o uso de ASICs seria viável, do ponto de vista econômico. Este baixo volume é conseqüência das características do mercado nacional, ainda pouco desenvolvido. Para aproveitar as vantagens dos ASICs neste

tipo de mercado, é necessário o uso de um estilo de projeto viável em baixos níveis de produção. O estilo de projeto com circuitos pré-difundidos apresenta as características mais adaptadas a esta realidade, como já foi discutido. Considera-se assim justificada a opção de se estudar e implementar um método de projeto para circuitos pré-difundidos. O objetivo principal deste estudo é fornecer subsídios para a implementação de um sistema de projeto adaptado aos equipamentos e meios disponíveis ao usuário interessado na aplicação de ASICs. Acredita-se que este método pode suprir as necessidades atuais dos projetistas, além de poder evoluir junto com as tecnologias de microletrônica.

3 METODOS DE CONCEPÇÃO PARA CIRCUITOS PRÉ-DIFUNDIDOS

Neste trabalho, entende-se por método de concepção o conjunto de aspectos que caracteriza um ambiente de projeto de circuitos. No capítulo presente, as características relevantes de sistemas de PAC adequados para o projeto de circuitos pré-difundidos são apresentadas. O objetivo do capítulo é definir os requisitos básicos de um ambiente de concepção de "gate arrays". Estes requisitos são a base sobre a qual se apóiam os capítulos subseqüentes, que descrevem a implementação do método CIPREDI.

As vantagens do uso de "gate arrays", quando comparados com circuitos dedicados, compreendem a facilidade de projeto e a rapidez de implementação. A crescente aceitação dos circuitos personalizáveis no mercado mundial pode ser demonstrada, se considerarmos a afirmação de Rupp /RUP 87/ de que tem se verificado, durante esta década, um acréscimo de 37% ao ano na taxa de aplicação de CIs pré-difundidos.

Os projetistas de ASICs, contudo, continuam necessitando de métodos e ferramentas mais efetivos na elaboração do projeto destes componentes. Como prova desta necessidade, Rupp cita que metade de todos os projetos de CIs "gate arrays" iniciados nunca alcança a fase de produção. Esta cifra surge como conseqüência de uma série de fatores responsáveis pelo fracasso de projetos, tais como:

. custo final excedendo estimativas iniciais;
. problemas de teste e fabricação;
. especificações imprecisas ou incompletas;
. considerações de tempo necessário para que o produto final chegue ao mercado.

Atualmente, em todo mundo, quase 300 fundições de silício /RUP 87/ oferecem produtos pré-difundidos. Isto implica uma vasta gama de escolhas à disposição do cliente interessado em CIs personalizáveis. Cada uma destas fundições aplica diferentes princípios na confecção do seu sistema de PAC, gerando uma grande variedade de abordagens do problema. A falta de padronização dificulta a avaliação objetiva de qual seria o sistema ideal para a implementação de um dado projeto.

Na seção 3.1, comentam-se os métodos e as ferramentas empregados correntemente no projeto de "gate arrays". A esta, segue-se a seção 3.2, onde os problemas das tecnologias mais usadas, da célula de base, das configurações de CIs pré-difundidos e das bibliotecas de células são discutidos. Finalmente, a seção 3.3 aborda a relação cliente/fornecedor dentro do contexto da concepção de pré-difundidos.

3.1 Métodos e Ferramentas

Em seu trabalho sobre elaboração de métodos de projeto de sistemas, Rocha Costa /COS 82/ define técnica como "um conjunto de operações muito bem caracterizadas, as quais devem ser realizadas em uma determinada ordem, sobre determinados objetos, para que se possa obter, com esses objetos, um determinado resultado (novos objetos ou transformações dos objetos já existentes)". Ainda no mesmo trabalho, define-se método como "um conjunto de técnicas, organizadas para a obtenção de determinado resultado, quando da manipulação de determinados objetos". Embora se mostrem adequados para um trabalho de metodologia como /COS 82/, estas definições se revelam muito vagas do ponto de vista prático. Assim, neste trabalho, procura-se abordar métodos segundo uma ótica pragmática, respeitada a definição formal acima.

Na subseção 3.1.1, define-se a ótica pragmática para interpretação de métodos de projeto para CIs pré-difundidos, demonstrando sua utilidade através da descrição de sistemas constantes na bibliografia com o uso desta ótica. As subseções posteriores tratam das ferramentas de "hardware" e de "software" presentes em sistemas de PAC para circuitos pré-difundidos.

3.1.1 Ótica pragmática para interpretação de métodos de projeto de CIs pré-difundidos

Os requisitos de um método de projeto de sistemas, segundo /COS 82/ devem envolver, no mínimo, aspectos relacionados com:

a) a teoria subjacente ao sistema alvo do método;
b) o modo pelo qual é feita a especificação do sistema;
c) a teoria dos processos de projeto envolvidos;
d) a infraestrutura de projeto, ou seja, as ferramentas de projeto.

Definimos a seguir a ótica de interpretação, baseado em cada um dos itens acima.

Neste trabalho, a teoria subjacente (item a) é a de implementação de circuitos pré-difundidos. Nela, o sistema alvo é um circuito integrado contendo uma matriz regular bidimensional de transistores ou portas lógicas. Esta matriz é pré-processada sobre um substrato semicondutor. Os componentes de circuito são todos previamente caracterizados, posicionados e dimensionados. A personalização consiste em conectar componentes, com o fim de obter um determinado comportamento do CI.

Em "gate arrays", 90% do processamento /TEL 85/

já está efetuado antes do início da implementação de um projeto, garantindo tempos curtos para a obtenção de protótipos e eficiência de projeto.

Identificam-se quatro requisitos relevantes para a caracterização de um método, com relação ao item a:

. tecnologia de implementação;
. célula de base;
. matriz;
. biblioteca de células.

Em geral, cada tecnologia (ECL, CMOS, GaAs) de implementação implica um método de projeto distinto, devido aos diferentes compromissos presentes em cada um destes tipos de organização de semicondutores.

Por célula de base se entende a unidade de "layout" da matriz bidimensional que constitui o núcleo de qualquer "gate array". Uma grande quantidade de topologias propostas existe hoje comercialmente disponível. A capacidade de manipulação de diferentes topologias é uma característica desejável a qualquer método de projeto, uma vez que, dentro de uma mesma fundição, coexistem várias células de base usadas para a implementação de "gate arrays". Cada topologia é elaborada com o fim de servir à implementação de determinadas estruturas de forma otimizada (exemplo: lógica aleatória, memórias, PLAs, etc). Na figura 3.1 mostra-se um exemplo de célula de base típica, com o diagrama elétrico e o "layout" de máscaras pré-difundidas.

O termo matriz designa o arranjo bidimensional que constitui o núcleo de um CI "gate array" e, por extensão, cada configuração núcleo + pinos de entrada e saída ("pads"). Assim, matriz refere-se ao "layout" pré-difundido em silício, sem a personalização final, provida pelo projeto, como exemplificado pela figura 3.2 abaixo.

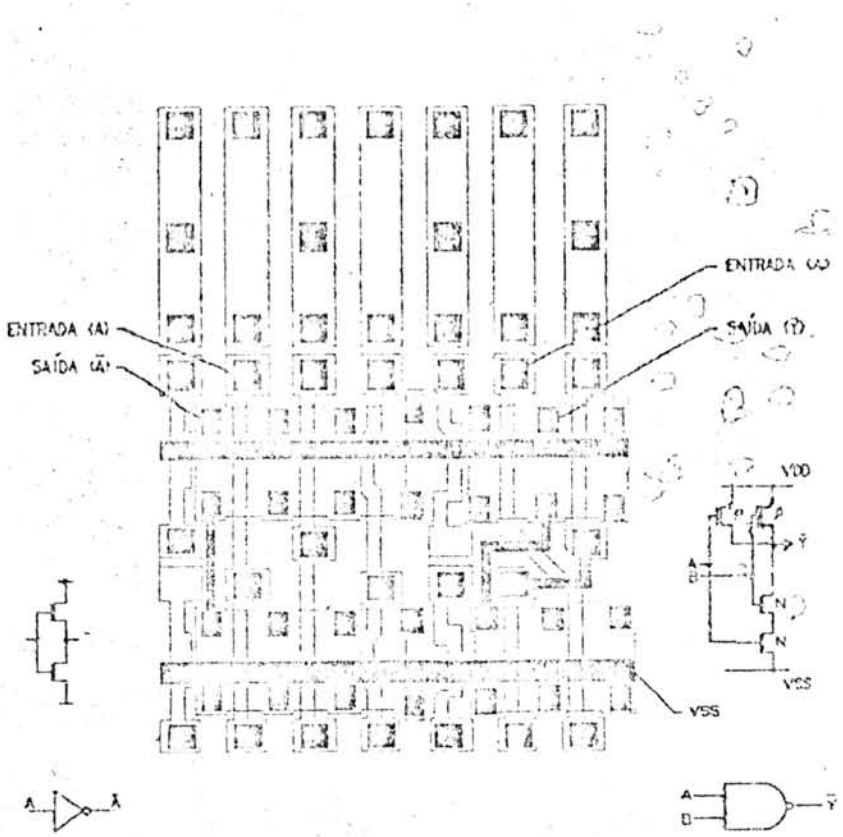


Figura 3.1 - Célula de base típica de pré-difundidos

Cada fundição possibilita a escolha de uma configuração de matriz dentre um conjunto finito de opções suportadas. Por exemplo, a AMI /TEL 85/ possui 4 famílias de "gate arrays" disponíveis, cada família usando um tipo diferente de célula de base. As famílias oferecidas apresentam, cada uma, entre 4 e 7 matrizes ou configurações de pré-difundidos. Destas, a família construída com uso de uma célula de base CMOS com porta de polissilício de 2 microns possui 7 matrizes distintas, a menor com 1300 portas equivalentes, 64 células de E/S e 68 "pads", e a maior com 10000 portas equivalentes, 216 células de E/S e 224 "pads". A tabela 3.1 ilustra os parâmetros que caracterizam esta família.

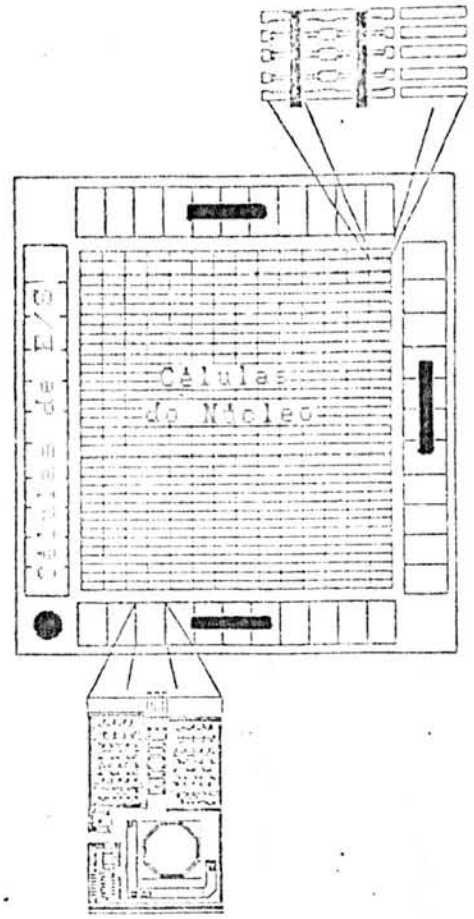


Figura 3.2 - Exemplo de matriz de pré-difundido

Tabela 3.1 - Características da família de matrizes com canal de poli de 2 microns da AMI

GATES EQUIVALENTES	CÉLULAS DE E/S	PADS
1,3 K	64	68
2,0 K	80	84
3,0 K	104	108
4,0 K	116	124
5,0 K	136	144
8,0 K	180	188
10,0 K	216	224

A gama de configurações é oferecida como forma de viabilizar economicamente o maior número possível de projetos. A cada projeto, a escolha da configuração recai sobre a menor possível dentre as oferecidas. Esta escolha é guiada pelo método e postergada até a obtenção de uma estimativa do tamanho do circuito (em portas equivalentes) estar disponível. Além do número de portas, uma noção da área necessária para realizar interconexões é fundamental na escolha da matriz mais adequada. Os métodos modernos permitem o experimento com configurações em tempo de geração automática do "layout", possibilitando maior segurança e flexibilidade na implementação. Obviamente, o mesmo método de projeto deve suportar todas as matrizes disponíveis.

A biblioteca de células é o conjunto de primitivas de alto nível de abstração oferecidas pelo fabricante de "gate arrays" ao projetista. Cada célula de uma biblioteca consiste de um padrão de personalização a ser superposto à matriz, com o intuito de implementar determinados componentes. A complexidade de cada célula é função do método escolhido, sendo duas abordagens comumente utilizadas. Na primeira, emprega-se uma biblioteca trivial de células contendo portas lógicas e transistores de passagem. Associadas a esta biblioteca, estão ferramentas capazes de concatenar células simples para a geração automática de módulos mais complexos. Na segunda abordagem, opta-se pelo uso de uma extensa biblioteca de células contendo primitivas de alto nível de abstração como contadores, multiplexadores e unidades lógico-aritméticas.

Por exemplo, a AMI /GOU 85/ emprega dois tipos de biblioteca CMOS: uma com diversos tipos de portas lógicas, onde o circuito mais complexo é um "flip-flop" com sinais "set" e "reset" assíncronos, e uma biblioteca denominada

Auxílios à Captura de Esquemáticos ("Schematic Capture Aids"), onde se encontra células tão complexas como um contador "up/dowm" de 8 bits. A LSI Corp /LSI 84/ usa uma abordagem similar, apenas com denominações distintas para os tipos de células que compõem as suas bibliotecas CMOS: macrocélulas, para as de baixa complexidade, e macrofunções, para as restantes. Já a General Eletric, por outro lado, usa uma biblioteca com primitivas cuja complexidade não ultrapassa a de um estágio de um contador "up/down" ou de um "flip-flop" JK com sinais "set" e "reset" assíncronos /GEN 83/.

Outros itens relacionados ao requisito biblioteca de células do método de projeto são:

. a independência da biblioteca de células com relação à célula de base;

. as informações armazenadas em cada célula da biblioteca.

O primeiro item exige um método bastante flexível, capaz de gerenciar diferentes personalizações associadas a uma célula de biblioteca, em função da célula de base e matriz usadas. Contudo, esta abordagem propicia um ambiente confortável quando conversões de projeto são valorizadas, como no caso de conversão de um "gate array" CMOS para ECL, visando aumentar o desempenho do circuito; ou no caso de conversão de um projeto "gate array" para "standard cells", como citado em /WOL 85/.

O último item envolve decisões sobre quais informações armazenar junto a cada célula da biblioteca. Além do "layout" das camadas de personalização, costuma-se acrescentar:

. informações gráficas usadas por ferramentas de captura de esquemas elétricos ou lógicos;

. informações geométricas identificando os pontos de entrada e saída do "layout" da célula (para fins de posicionamento e traçado de rotas automáticos);

. informações sobre parâmetros elétricos, como atrasos de propagação e atrasos devido ao carregamento capacitivo das saídas.

A decisão sobre o que inserir na biblioteca é tarefa do método. A figura 3.3 mostra, a título de exemplo, a reprodução das informações fornecidas por um fabricante, no caso a GE, sobre uma célula de biblioteca típica.

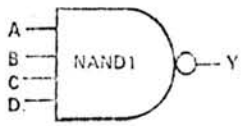
Uma discussão sobre as abordagens destes requisitos, existentes na bibliografia, pode ser encontrada na seção 3.2.

O segundo aspecto (item b) de um método de projeto de "gate arrays" a ser analisado é o da especificação do sistema. O método deve especificar os requisitos linguagens de entrada e linguagens de saída para cada nível de abstração suportado. Além disto, a utilização de linguagens textuais e/ou gráficas deve ser considerada cuidadosamente, levando em conta aspectos de facilidade de uso e poder da linguagem de especificação.

A existência de padrões, a nível internacional, para a confecção de descrições textuais e gráficas como EDIF /EDI 84/ e GKS /CUN 87/, deve ser considerada na elaboração do método. Se o intercâmbio de informações de projeto com outros centros for provável, o uso de padrões é essencial.

ND4
4 INPUT NAND

Formerly named NC, ND4 is a 4 input NAND constructed from four transistor pairs as shown in the circuit schematic.



TRUTH TABLE

A	B	C	D	Y
0	X	X	X	1
X	0	X	X	1
X	X	0	X	1
X	X	X	0	1
1	1	1	1	0

X = Don't Care

EQUIVALENT INPUT LOADS (L)

Input	L
A	1
B	1
C	1
D	1

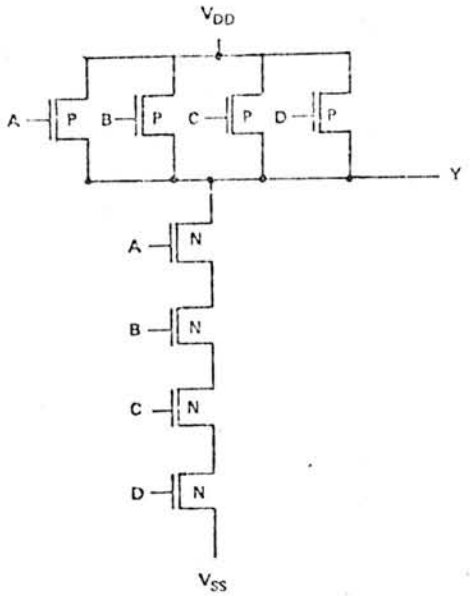
OUTPUT DRIVE CHARACTERISTICS

Output	Drive	
	Rise	Fall
Y	1X	.25X

Number of Cells: 1 1/3 Unused Cell Positions: 25

EDL Description: Occurrence-name:ND4(A,B,C,D),(Y)

TDL Description: Occurrence-name(Y)=ND4(A,B,C,D)$



PERFORMANCE DATA

MEASUREMENT	PARAMETER	OUTPUT	DELAYS (ns) MIN	TYP	MAX
Prop Delay	A,B,C,D, to Y	Rising	4 + .4L	7 + .8L	11 + 2L
		Falling	3 + .6L	7 + 1L	15 + 3L

L = Equivalent Loads

Figura 3.3 - Exemplo de informações contidas em uma biblioteca de células

Descrições de nível mais baixo de abstração possuem formatos consagrados pela indústria como a linguagem CIF de descrição geométrica de máscaras /MEA 80/, e o formato CALMA para a fita geradora de padrões. Algumas ferramentas de "software" tradicionalmente aplicadas em projetos de CIs empregam formatos hoje considerados padrões "de facto". Exemplos destes formatos são a entrada de dados textual para o simulador elétrico SPICE /BER 81/ e a linguagem textual TDL usada como entrada para o simulador lógico e analisador de temporização TEGAS /GEN 83/. As linguagens gráficas de especificação, apesar de largamente empregadas na atualidade, não apresentam sinais de constituir um consenso entre sistemas. Isto se deve às dependências de dispositivo sempre vinculadas a uma forma gráfica de apresentação de informações. Contudo, a forma gráfica de interação é, há muito tempo, um recurso indispensável em qualquer sistema de PAC.

Finalmente, o método deve especificar as escolhas do requisito relação projetista/fabricante, item discutido adiante, na seção 3.3.

O terceiro aspecto (item c) de um método de projeto para pré-difundidos envolve considerações sobre os processos de projeto. Três requisitos sobressaem neste aspecto:

. tomadas de decisão de projeto;
. flexibilidade do ambiente de projeto;
. níveis de abstração.

Gajski /GAJ 83/ relaciona três abordagens diferentes do requisito "tomadas de decisão de projeto" ao discutir ferramentas voltadas para a concepção de circuitos VLSI. A primeira abordagem, que ele denomina de "computer-aided design", ou CAD, consiste em deixar que todas as

decisões de projeto sejam tomadas pelo projetista humano. A segunda baseia-se na crença de que a capacidade humana pode ser capturada na forma de regras de projeto, e armazenada em uma base de conhecimentos. As ferramentas que usam esta abordagem denominam-se sistemas especialistas. A última abordagem é o oposto da primeira. Ela está baseada na idéia de que o conhecimento é algorítmico. Segundo esta abordagem, pode-se elaborar sistemas que gerem a solução de uma problema de projeto VLSI a partir de uma descrição formal em alto nível, sem intervenção humana significativa no processo de síntese. O método deve optar por uma ou por um conjunto coerente de abordagens aplicáveis ao longo do projeto.

O método deve caracterizar a flexibilidade do ambiente de projeto desejável, dadas as condições de contorno. Se o ambiente é o meio acadêmico, a flexibilidade deve ser máxima para permitir a aplicação do método a múltiplas tecnologias, linguagens de descrição e formas de representação de projetos. Se o ambiente for industrial, a flexibilidade deve ceder lugar à busca do desempenho máximo para as ferramentas de projeto. Ainda, estas ferramentas devem ser específicas para as condições restritas de implementação do fabricante.

Por último, o requisito níveis de abstração deve ser acessado pelo método. Abordagens ascendentes ("bottom-up") ou descendentes ("top-down") estão disponíveis como formas puras de evolução do projeto ao longo dos níveis de abstração. Em geral os métodos de projeto adotam uma abordagem mista usando o enfoque ascendente para a geração de células de biblioteca e simulação lógica, e o enfoque descendente para os níveis de abstração superiores.

O último aspecto (item d) de um método de projeto para CIs pré-difundidos trata da infraestrutura de projeto, ou seja, das ferramentas de "hardware" e "software"

usadas. Do ponto de vista prático, as ferramentas de "software" caracterizam o método. Explicar o funcionamento de cada uma destas ferramentas e o algoritmo de aplicação delas é apresentar o método de projeto de maneira informal.

Nas seções 3.1.2 e 3.1.3, os requisitos ferramentas de "hardware" e ferramentas de "software", respectivamente, são discutidos.

Antes de passar à discussão das ferramentas, um último mecanismo metódico será apresentado, o diagrama Y ou de Gajski /GAJ 83/. As ferramentas de "software" são o requisito mais importante dos métodos de projeto em estudo. Assim, justifica-se o uso de um mecanismo de visualização das relações entre os níveis de abstração de projeto e as ferramentas responsáveis pela evolução deste projeto ao longo dos níveis. O diagrama de Gajski representa justamente este mecanismo.

O diagrama Y divide os níveis de abstração em três representações distintas:

. representação funcional, uma forma de mostrar o que o projeto faz e como ele é elaborado;

. representação geométrica, a qual ignora, tanto quanto possível, o comportamento do projeto, preocupando-se com a disposição espacial dos seus componentes.

. representação estrutural, uma ponte entre as duas outras representações, ou seja, um mapeamento de uma representação funcional em componentes e conexões sob restrições de área, custo e tempo;

Na figura 3.4 é mostrado o diagrama Y. Os círculos representam os níveis de abstração e são normalmente omitidos. Os 3 segmentos de reta concorrentes correspondem às diferentes representações de um projeto.

de representação específica.

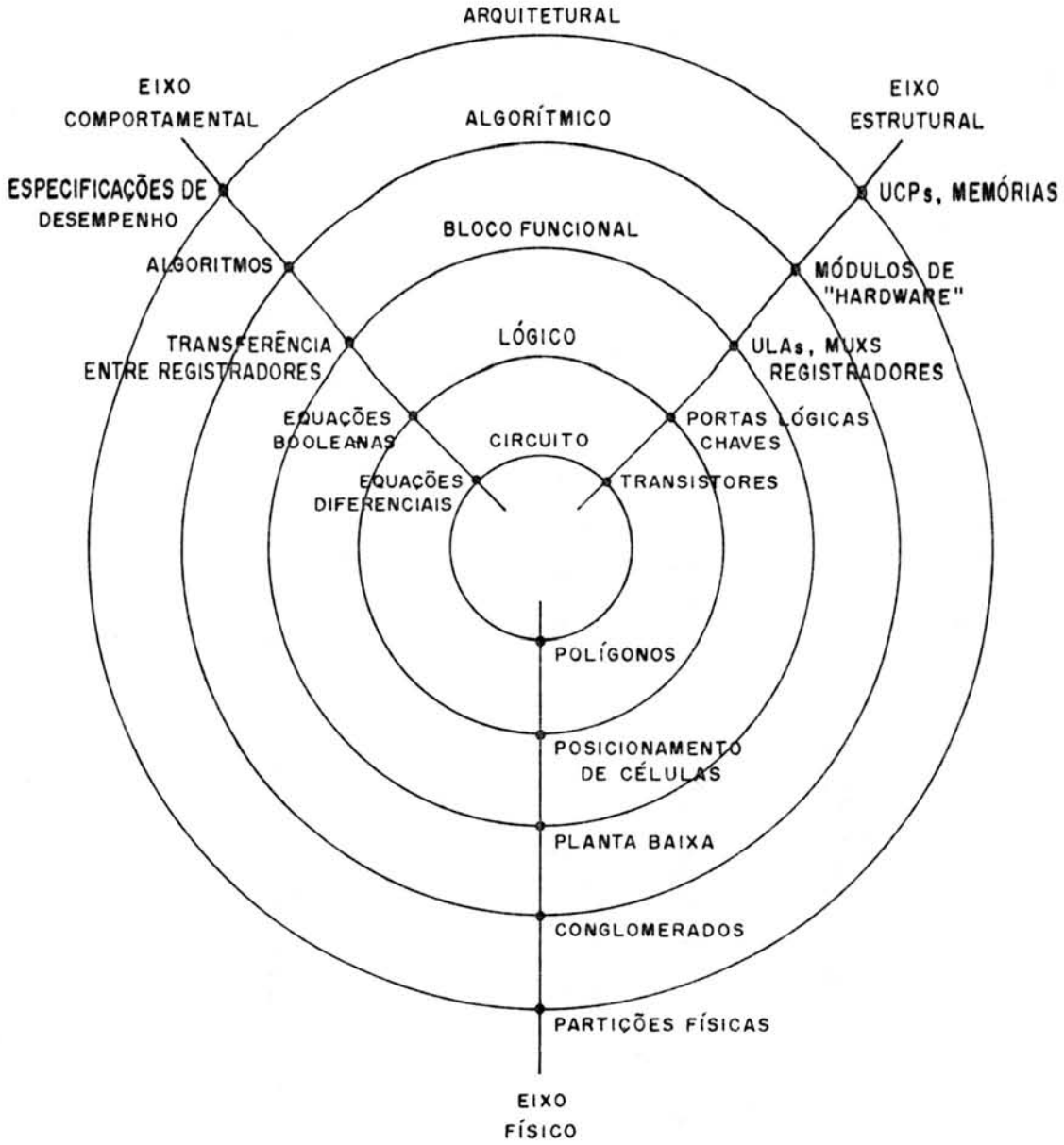


Figura 3.4 - Diagrama de Gajski ou Diagrama Y

Ferramentas de "software" são programas responsáveis pela transformação de representações de projeto. Pode-se, então, mostrar a função das ferramentas usando arcos dirigidos conectando formas de representação no diagrama Y. Para ilustrar o uso do diagrama, apresenta-se o sistema de PAC usado no CPqD - Telebrás para o projeto de "standard cells".

O CPqD conta com as seguintes ferramentas de PAC para o projeto no estilo pós-difundido:

. Editor de esquemáticos - Eddie;

. Simulador lógico, de falhas e gerador de padrões de teste - HILO;

. Traçador de Rotas e Posicionador - Pincel;

. Simulador elétrico - SPICE;

. Verificador de regras de projeto geométricas - XDRC;

. Extrator de circuitos - PAS;

. Ferramentas de geração interativa de "layout" - Estação de trabalho Applicon.

O processo de projeto é ilustrado pelo diagrama Y da figura 3.5.

Inicialmente, a partir de especificações informais de alto nível, é obtida uma representação estrutural mista de portas lógicas e blocos funcionais, usando o editor Eddie. Esta representação é validada pelo uso do simulador HILO. A representação resultante serve de entrada para o sistema PINCEL que gera uma versão do "layout" do circuito. Caso o PINCEL não tenha obtido um traçado completo do circuito, usa-se a estação de trabalho Applicon para finalizar o "layout". A verificação geométrica é feita pelo programa XDRC. A verificação elétrica é feita pelo SPICE, a partir da descrição do circuito gerada pelo extrator, que trabalha diretamente sobre o "layout" do CI.

Com este exemplo, conclui-se a apresentação da ótica pragmática para a especificação e implementação de métodos de projeto para circuitos pré-difundidos. A ótica em questão será usada, no capítulo 4, para a apresentação do método CIPREDI, objeto deste trabalho. Ela pode ser adaptada, sem esforço significativo, para a especificação

de métodos de projeto de circuitos em outro estilos de concepção. O diagrama Y será usado para ilustrar a estrutura de "software" dos sistemas de PAC citados ao longo do restante do trabalho. A figura 3.6 apresenta um esquema para ilustrar a organização da ótica apresentada.

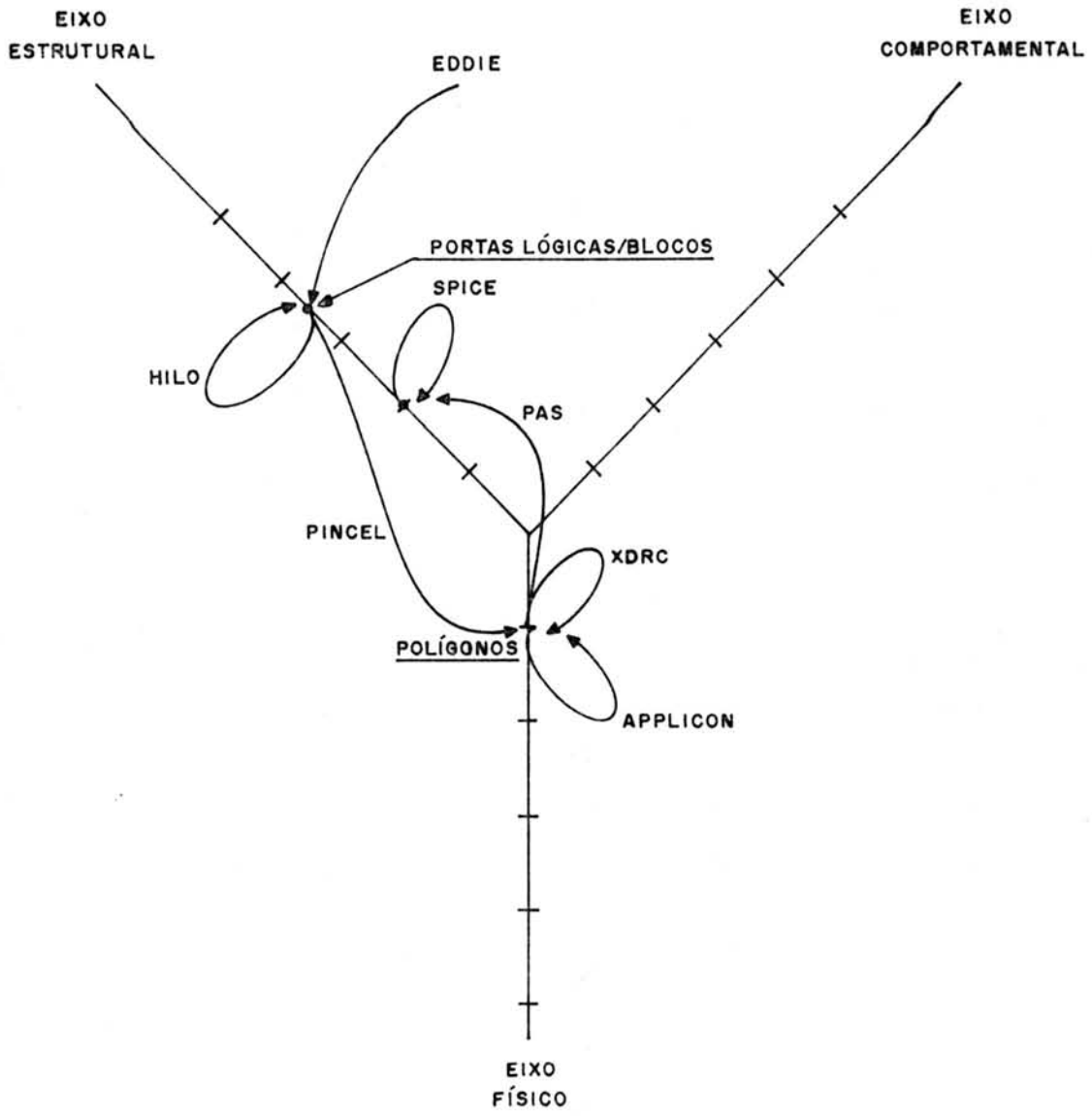


Figura 3.5 - Processo de projeto aplicado no CPqD da Telebrás

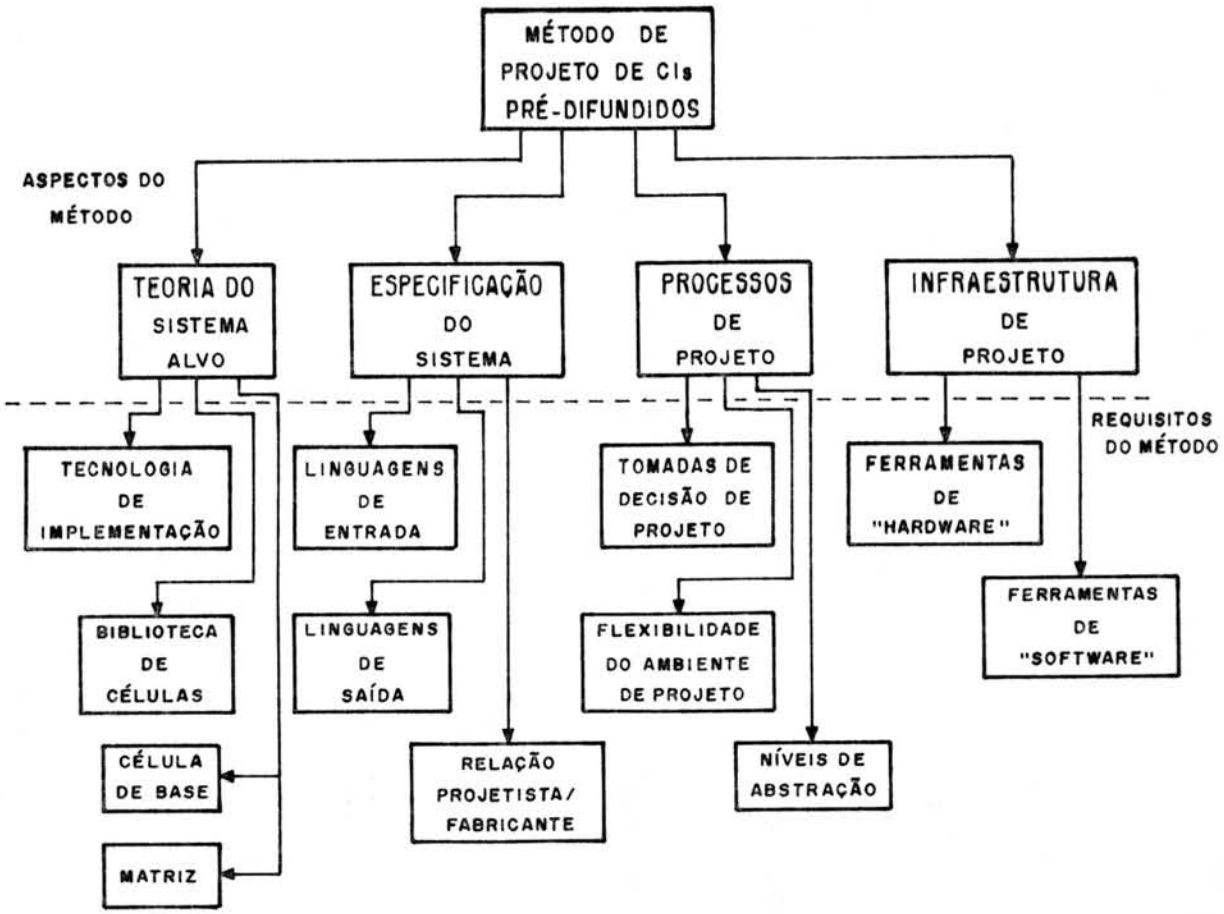


Figura 3.6 - Estrutura geral de métodos de projeto para CIs pré-difundidos

3.1.2 Ferramentas de "hardware" no projeto de pré-difundidos

Computadores são equipamentos imprescindíveis no projeto de circuitos integrados VLSI de qualquer natureza. A primeira geração de sistemas PAC caracterizou-se por aplicar programas de geração das geometrias das máscaras em computadores de grande porte ou "mainframes" /EGL 85/. Estes equipamentos foram gradativamente acrescidos de periféricos, que facilitavam o trabalho do projetista e aliviavam a carga de trabalho da UCP. À medida que poder computacional era acrescido aos periféricos, estes se tornavam mais e mais independentes dos computadores

centrais, usando os serviços destes apenas para tarefas de alta carga de processamento e armazenamento dos dados globais do projeto.

Os periféricos usados compreendiam estações gráficas, traçadores gráficos ("plotters"), mesa digitalizadora, "tablets", etc. A evolução do "hardware" determinou o surgimento do conceito de estações de trabalho para projeto ("workstation"). Duas formas de utilização das estações de trabalho são hoje correntes. Primeiro, empresas que mantêm a forma tradicional de projeto, baseada em máquinas genéricas de médio e grande porte. Estas máquinas encontram-se associadas a estações de trabalho destinadas a tarefas computacionalmente menos intensas, como captura de esquemas e simulação lógica de módulos. Segundo, empresas que defendem o uso de estações de trabalho de forma autônoma para a execução de todas as fases do projeto de CIs.

Qualquer que seja a forma utilizada, "workstations" são uma ferramenta de uso universal. Sua aparição revolucionou as técnicas e a economia do projeto de CIs. Esta revolução deveu-se principalmente aos aspectos econômicos de sua concepção:

. "workstations" são equipamentos do tipo monousuário, sendo então de desempenho previsível e não dependentes de aspectos como carregamento do sistema em ambientes multi-usuário;

. sua construção é baseada em dispositivos VLSI modernos, de baixo custo e alto desempenho, como microprocessadores e memórias de alta escala de integração;

. com estações de trabalho, obtém-se alto poder computacional e alta capacidade de processamento interativo com "hardware" de baixo custo;

. estações apresentam grande capacidade de manipulação de informações gráficas, devido aos periféricos

específicos, como vídeo gráfico de alta resolução, "mouse", mesa digitalizadora, impressoras gráficas e traçadores.

Como exemplo da primeira forma de utilização, apresentam-se dois casos: a LSI Corporation /BOY 83/ e a Hughes Aircraft Company /TIN 83/. No primeiro caso, Boyle descreve o papel das "workstations" no sistema de projeto LDS-II, voltado para pré-difundidos, da LSI, discorrendo sobre o conjunto de vantagens e desvantagens das estações de trabalho. Como principal motivo da escolha por esta forma de utilização, Boyle aponta a relação fabricante/projetista proposta pela LSI, ou seja, uma divisão rígida de tarefas de projeto:

. ao projetista cabem a descrição e a simulação do circuito até o nível lógico;

. a LSI cabem a geração do "layout", a fabricação e o teste do CI.

Esta separação corresponde à mesma separação de tarefas entre os equipamentos: estações de trabalho para as tarefas do usuário e "mainframes" para as tarefas da LSI. A escolha, segundo Boyle, facilita as atividades de troca de informação de projeto e minimiza as interações entre o projetista e a fundição.

No segundo caso, Ting procede a uma análise quantitativa de recursos de "hardware" requeridos pelas diversas ferramentas de "software". Os resultados desta análise são reproduzidos na tabela 3.2.

Com base neste levantamento, Ting propõe uma hierarquia de "hardware" a três níveis para implementação do sistema de PAC LSI da empresa. No nível inferior da hierarquia, estão as estações de trabalho, usadas em tarefas de processamento gráfico, como captura do esquema lógico e posicionamento/traçado de rotas interativos de

projeto físico. No nível intermediário, tem-se super-minicomputadores (VAX 11/780, por exemplo), usados para tarefas interativas de média complexidade, como simulação elétrica de pequenos módulos e simulação lógica de blocos do projeto. Finalmente, o nível superior da hierarquia conta com computadores de grande porte (IBM 4341-2 e AMDAHL 470/V8), voltados para tarefas não-interativas de grande atividade computacional: simulação de falhas, verificação de temporização e geração automática de "layout" são exemplos destas tarefas.

Tabela 3.2 - Análise de necessidades de recursos pelas ferramentas de projeto

Requisitos Computacionais Relativos de Ferramentas de Projeto de Circuitos Integrados.

TAREFAS DE PROJETO PROJETO ESTRUTURAL/COMPORTAMENTAL	MEMÓRIA PRINCIPAL	TEMPO EM UCP	ESPAÇO EM DISCO	RESOLUÇÃO GRÁFICA /RESPOSTA GRÁFICA
- CAPTURA DE ESQUEMÁTICO	A	M	M	M / A
- SIMULAÇÃO LÓGICA	M	A	M	M / M
- SIMULAÇÃO DE FALHAS	A	MA	M	NA / NA
- ANÁLISE DE TESTABILIDADE	M	M	M	NA / NA
- GERAÇÃO DE TESTES	A	MA	M	NA / NA
- ANÁLISE DE TEMPORIZAÇÃO	M	M	M	NA / NA
- SIMULAÇÃO DE CIRCUITO	M	A	M	M / M
PROJETO FÍSICO				
POSICIONAMENTO AUTOMÁTICO	B	A	B	M / M
TRAÇADO DE CONEXÕES AUTOMÁTICO	A	M	A	B / B
POSICIONAMENTO INTERATIVO	B	B	B	M / A
EDIÇÃO INTERATIVA DE CONEXÕES	A	B	A	A / MA
PROJETO DE CÉLULAS E MACROS	M	B	M	A / MA
GERAÇÃO DOS PADRÕES DE MÁSCARAS	A	MA	MA	NA / NA
VERIFICAÇÃO DAS MÁSCARAS				
VERIFICAÇÃO LÓGICA X MÁSCARAS	A	A	MA	A / M
VERIFICAÇÃO DE REGRAS ELÉTRICAS	A	A	MA	A / M
VERIFICAÇÃO DE REGRAS GEOMÉTRICAS	A	A	MA	A / M
EXTRAÇÃO DE PARÂMETROS	A	A	MA	NA / NA

B - BAIXO M - MÉDIO A - ALTO MA - MUITO ALTO NA - NÃO APLICÁVEL

A segunda forma de aplicação de estações de trabalho está, normalmente, associada a métodos de projeto para circuitos pré-caracterizados. Isto se dá devido às

exigências computacionais reduzidas presentes neste estilo de projeto. A maioria dos defensores desta forma de aplicação justifica sua escolha mencionando a maior flexibilidade e a velocidade de evolução das estações de trabalho em relação às máquinas de propósito geral, de grande e médio porte /DAH 85/ /JOH 86/ /WEI 86/.

Mesmo entre os defensores do uso de estações de trabalho autônomas no projeto de CIs pré-caracterizados, entretanto, existem dissensões. Podemos identificar, basicamente, três tipos de estações de trabalho, quanto ao porte e implementação:

. Estações baseadas em microcomputadores de 16 bits. Por exemplo, Applicon (PDP 11/34), CALMA-GDSII (Eclipse S/230), CADD 2/VLSI (Computer Vision CGP100) /WER 82/;

. Estações baseadas em microprocessadores de 32 bits. Como exemplo, Daisy (Apollo - Motorola 68000), Mentor Graphics, Lattice Logic MG1 (National 32016) /EGL 85/ /WER 82/;

Estações baseadas em microcomputadores pessoais de 16/32 bits. Por exemplo, Matra CAE/CAD (PC-AT - INTEL 80286), Futuneret Division, DATA I/O Corp. DASH GA Compiler (PC-AT INTEL 80286) /JOH 86/ /WEI 86/.

As primeiras são remanescentes do tempo em que estações de trabalho estavam vinculadas a computadores de grande porte. As estações baseadas em microprocessadores de 32 bits são as de mais alto poder computacional e mais rápida evolução. Elas apresentam grande quantidade de recursos gráficos e configurabilidade razoável. Aceleradores de "hardware", como placas para geração de "layout" /DAH 85/ e simulação lógica /GOE 88/, criam um ambiente próximo dos computadores de grande porte, em termos de desempenho.

As estações baseadas em computadores pessoais destinam-se à porção do mercado envolvida com projetos de baixa complexidade. Devido às severas limitações de memória e desempenho, estas estações trabalham sobre pré-difundidos não excedendo 2000 portas equivalentes CMOS.

Por exemplo, Weil /WEI 86/ e Johnston /JOH 86/ descrevem estações baseadas em computadores pessoais cujos resultados são bastante razoáveis no auxílio ao projeto de "gate arrays". Johnston cita que o sistema CAE/CAD da Matra Design Systems, implementado em um PC-AT IBM, foi capaz de simular um circuito descrito no nível lógico com 800 portas equivalentes em 47,5 minutos, enquanto que o mesmo programa executando a mesma simulação sobre um VAX 8600 levou 9,75 minutos.

A tendência atual, em termos de "hardware", é a aceleração no desenvolvimento das potencialidades das estações de trabalho para o projeto VLSI. A integração das ferramentas é facilitada e o compartilhamento de recursos e dados de projeto pode ser obtido, mediante o uso de redes locais ou remotas de "workstations".

Estações de trabalho proporcionam, na atualidade, um meio viável de se implementar projetos de CIs pré-difundidos. Em muitos casos, como no contexto nacional, apresentam-se como a solução economicamente mais atraente. Para CIs de baixa complexidade, as estações baseadas em computadores pessoais constituem uma solução de custo reduzido e eficiência aceitável /JOH 86/.

3.1.3 Ferramentas de "software" no projeto de pré-difundidos

Os sistemas de PAC atuais para o projeto de circuitos pré-difundidos ainda enfatizam primordialmente os

aspectos físicos de projeto. Ferramentas automáticas para o posicionamento de células e o traçado de interconexões são a base de qualquer um dos sistemas aplicados na produção deste tipo de CIs /GRA 82/. Nestes sistemas, o ramo estrutural do diagrama Y de Gajski é usado para a entrada de dados de projeto e validação lógica. O ramo funcional do diagrama Y encontra-se, ainda hoje, pouco utilizado.

Aos poucos, entretanto, a necessidade de lidar com CIs de alto grau de complexidade no projeto de "gate arrays" força os sistemas de PAC a se envolver com o tratamento de descrições funcionais dotadas de variados graus de abstração. Werner /WER 81/ elaborou, em 1981, um esquema mostrando um típico processo de projeto de pré-difundidos, reproduzido na figura 3.7, abaixo.

Sistema Tradicional Típico de Automação de Projeto de CIs Pré-Difundidos

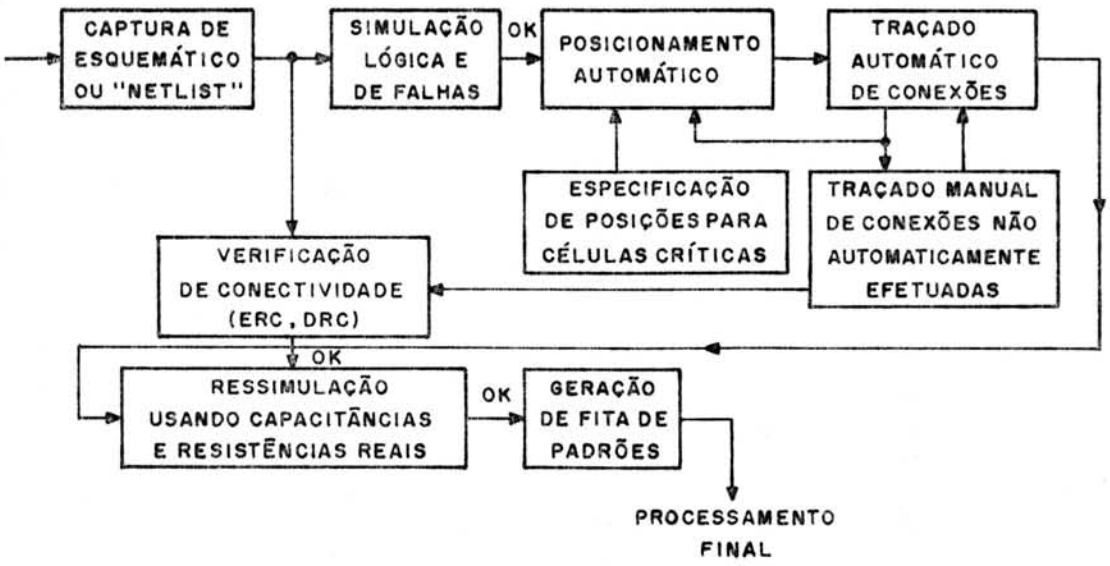


Figura 3.7 - Processo de projeto típico para pré-difundidos

Este processo de projeto pode ser ilustrado pelo diagrama Y mostrado na figura 3.8.

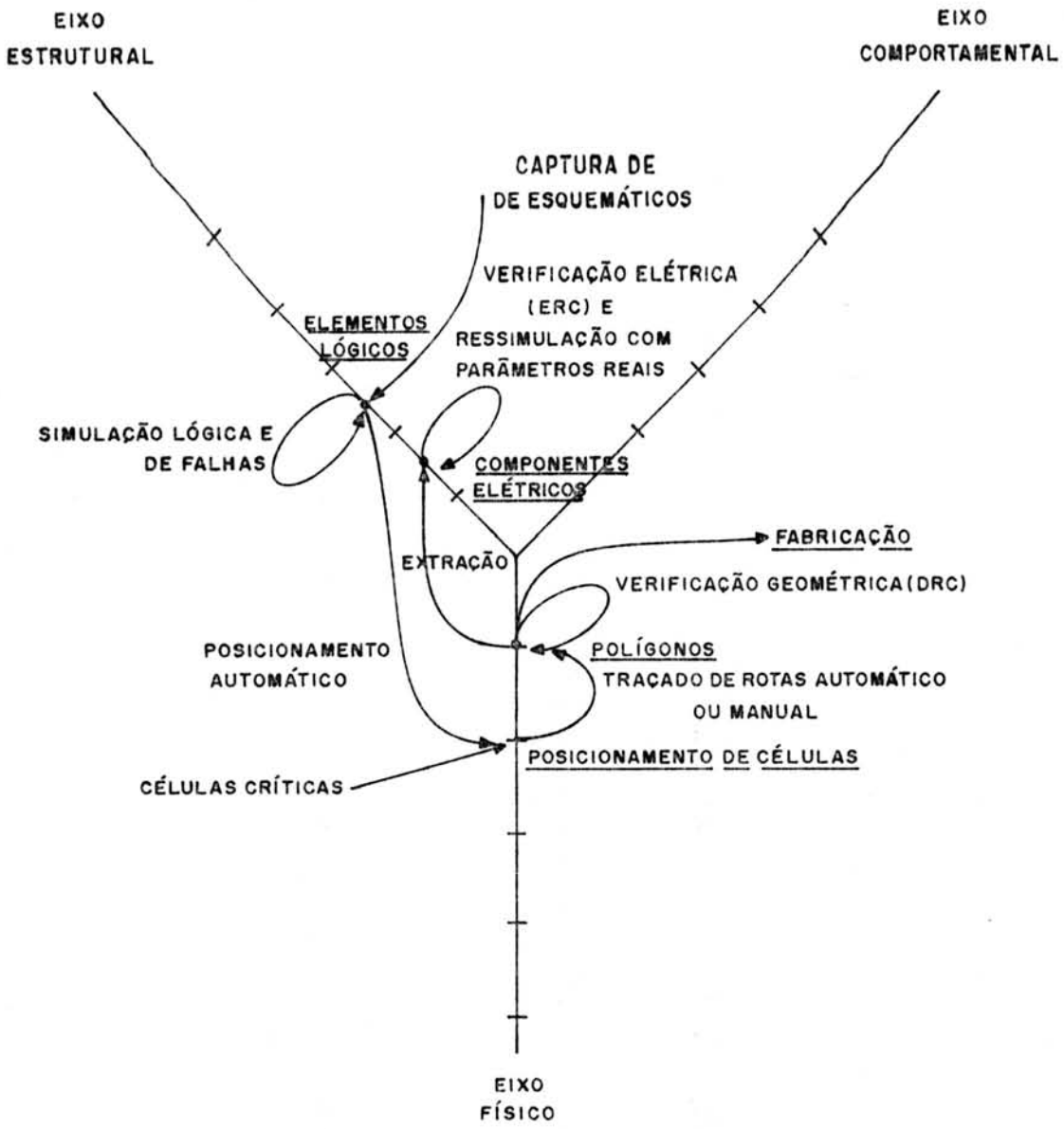


Figura 3.8 - Diagrama Y do processo de projeto típico

Werner citou os simuladores lógicos como a primeira ferramenta de "software" usada neste tipo de projeto. Isto ocorria porque a descrição estrutural da qual normalmente partia o sistema de PAC era obtida a partir da linguagem textual de entrada do simulador. Esta linguagem definia elementos primitivos e formas de conectar estes

elementos entre si. Os elementos primitivos compunham-se de portas lógicas e elementos especiais de níveis de abstração superiores, como o nível bloco funcional do diagrama Y. Werner revisou quatro dos mais empregados simuladores lógicos:

. o TEGAS 5, da COMSAT General Integrated Systems - CGIS;

. o LOGCAP, da Phoenix Data Systems;

. o LOGIS, da Information Systems Design- ISD;

. o LASAR, da Teradyne.

Quanto à geração de "layout", Werner citou, na época, dois sistemas de PAC com condições para executar geração automática de máscaras especificamente para "gate arrays", ambos incapazes, até então, de rotear pré-difundidos com apenas uma camada de metalização:

. MERLYN-G, da V-R;

. EDA-GA, da Silvar-Lisco.

Atualmente, mesmo após seis anos de efervescência na área de automação de projeto, podemos ainda usar a figura 3.7 para representar o processo de projeto de CIs pré-difundidos nos ambientes de produção. É claro que muito mudou neste meio tempo. A captura de projeto não se faz mais com descrições textuais de entrada vinculadas a simuladores, mas através de uso de ferramentas autônomas, baseadas em linguagens gráficas para edição de esquemas lógicos e/ou elétricos. Estas ferramentas acrescentam muito maior interatividade e capacidade documentacional ao projeto, além de sua saída servir de entrada para várias ferramentas componentes do ciclo de concepção.

Ao mesmo tempo, uma miríade de novos algoritmos, programas e sistemas de PAC povoam as cerca de trezentas fundições de silício e empresas de produção de ferramentas

envolvidas, em todo o mundo, com CIs pré-difundidos. A integração das ferramentas de projeto, antes fracamente acopladas, já pode ser obtida mediante o uso de bancos de dados de projeto /KAT 83/.

Modernamente, contudo, advoga-se o uso de ferramentas de PAC que ataquem os problemas de projeto em níveis superiores de abstração. Este ponto de vista visa ampliar as chances de aumento do desempenho dos sistemas em termos de velocidade e confiabilidade na implementação de pré-difundidos.

Newton e Sangiovanni-Vincentelli /NEW 86/ mencionam três tipos de ferramentas cujo emprego em sistemas de PAC futuros para circuitos personalizáveis deverá ser rotineiro:

. ferramentas de síntese automatizada;

. ferramentas avançadas de verificação de projeto;

. ferramentas de gerência do sistema de projeto.

As ferramentas de síntese automatizada têm sido aplicadas aos circuitos pré-difundidos com sucesso, seja em ambientes acadêmicos, seja no meio industrial /WEI 86/ /SAS 86/ /GRA 82/ /SAI 86/ /JOH 86/ /OLI 87/. Este sucesso deve-se, em parte, às restrições presentes neste estilo de projeto, que implicam a existência de limites no espaço de possibilidades de implementação, facilitando assim o uso de ferramentas de automação de projeto.

Costuma-se separar o problema de projeto de um sistema digital a ser implementado sobre pré-difundidos em dois subproblemas, o que constitui, segundo Davio /DAV 83/, uma abordagem natural:

. projeto de parte operativa;

. projeto de parte de controle.

Contudo, em qualquer dos subproblemas, existe um aspecto crítico a ser acessado, o da otimização de projeto. Este problema é subjacente a toda ferramenta de síntese automatizada. Ele está normalmente implícito nos procedimentos das ferramentas de geração de máscaras. Nos níveis superiores de abstração, entretanto, a complexidade envolvida dita a existência de ferramentas específicas para a otimização /BRA 86/.

O nível lógico é, em geral, o primeiro nível tratado pelas ferramentas de otimização. Esta preferência é ditada pelo amadurecimento atual da álgebra booleana, o ramo da matemática associado a este nível.

Uma última observação sobre ferramentas de síntese automatizada é necessária, ao considerar seu uso na implementação de CIs pré-difundidos. A topologia rígida associada aos pré-difundidos dificulta a geração de estruturas regulares, como PLAs e ROMs. Assim, procura-se empregar técnicas de síntese de lógica aleatória em conjunção com pré-difundidos. A automação dos processos de síntese deste tipo de lógica é uma disciplina relativamente recente, como demonstra Brayton /BRA 86/, constituindo, contudo, um meio capaz de gerar soluções eficientes no projeto de "gate arrays" /SOC 86/ /SAS 86/.

As ferramentas de verificação de projeto, por seu lado, desempenham um papel fundamental no projeto de pré-difundidos. Elas são responsáveis pela garantia de funcionamento dos protótipos e pela qualidade do projeto. Mesmo que ferramentas de síntese automatizada reduzam a necessidade de sua aplicação, dificilmente se atingirá o ideal de circuitos corretos por construção, pelo menos a

projeto têm lugar garantido em sistemas de PAC.

As ferramentas de verificação estão, normalmente, associadas ao problema do teste de circuitos. No ambiente industrial, uma parte significativa dos recursos de PAC é voltada para os aspectos de teste de projetos e de protótipos de sistemas digitais. Ferramentas de análise de projeto do ponto de vista do teste são tradicionalmente empregadas após a fase de implementação no nível lógico. Estas ferramentas realizam as seguintes tarefas:

. simulação de falhas;
. geração de vetores de teste para os protótipos;
. análise de cobertura de falhas dos vetores de teste.

A primeira e mais simples forma de simular falhas é mediante o uso do modelo "grudado-em" ("stuck-at"). Em circuitos construídos com tecnologia MOS, contudo, existem dúvidas sobre a efetividade deste modelo de falhas /TON 85/ para prever uma boa cobertura dos defeitos reais, possivelmente presentes no circuito fabricado. Os simuladores voltados para descrições a nível de chaves surgem, neste caso, como uma alternativa mais atraente para esta análise /CAI 87/.

Dentro do aspecto teste de circuitos pré-difundidos, um conceito vem crescendo de importância nos ambientes de projeto, o de projeto visando a testabilidade, ou PVT. Este conceito consiste no uso de técnicas que agregam características de observabilidade e controlabilidade a um projeto de pré-difundido. Tonge /TON 85/ divide as técnicas de PVT em dois grandes grupos:

. técnicas "ad hoc";
. técnicas estruturadas.

As primeiras dizem respeito ao acréscimo de estruturas que melhorem a testabilidade de um circuito pronto. As últimas envolvem restrições impostas sobre a forma de projetar o circuito, visando também o aumento da testabilidade. Em ambos casos existe um acréscimo na lógica do circuito final, com a conseqüente degradação do desempenho. As técnicas estruturadas podem causar, segundo Meyer /MEY 86/, uma degradação de até 20% no desempenho do circuito final, o que faz com que sejam preteridas em relação às técnicas "ad hoc". Alternativamente, combinações das duas técnicas são usadas para melhorar as figuras de mérito de desempenho.

A inexistência de um formalismo maduro, capaz de garantir a validade das técnicas de PVT atualmente disponíveis, aponta hoje para a aplicação de sistemas especialistas baseados em regras. Estes sistemas contituem uma forma eficiente de verificar a testabilidade de circuitos implementados segundo uma ou um conjunto de técnicas de PVT /CAB 86/.

Como último comentário, deve-se esclarecer que o problema do teste de pré-difundidos é específico de um fabricante e específico para cada projeto. Este fato normalmente implica uma alta taxa de interação entre projetista e fabricante durante a especificação do programa de teste dos protótipos. Além disto, o fabricante elabora as matrizes de pré-difundidos baseado no método de teste que julga mais conveniente. As matrizes de pré-difundidos possuem, não raro, estruturas de "hardware" embutidas, visando a aplicação de determinadas técnicas de PVT durante o projeto /RES 83/ /NAK 86/ /CHE 86/ /LAK 86/. Outra alternativa usada pelos fabricantes é obrigar o projetista ao uso de ferramentas capazes de inserir estruturas de teste de forma automática em um circuito digital, garantindo sua testabilidade /SHI 85/.

Finalmente, as ferramentas de gerência do sistema de projeto são vistas de forma macroscópica, como um banco de dados capaz de incorporar capacidades de manutenção de versões e garantir consistência, ainda que parcial, das diferentes descrições que compõem um projeto de CI pré-difundido /KAT 83/.

Serão vistos, a seguir, dois estudos de caso de sistemas de PAC, um dirigido aos níveis de abstração mais baixos e a descrições físicas, e outro orientado para descrições com alto nível de abstração. Os sistemas são, respectivamente:

. o MEC CAD, da General Electric Corporation /HIG 86/;

. o DASH GA Compiler, da Futurenet Division, Data I/O Corporation /WEI 86/;

O sistema MEC CAD da GE usa uma abordagem conservadora de projeto. O diagrama Y do sistema é mostrado na figura 3.9.

O MEC CAD recebe como entrada diagramas de esquemático a nível de símbolos de biblioteca, supridas por uma das estações de trabalho que contenha os símbolos da biblioteca de macros da GE Semiconductor. As estações que suportam esta biblioteca são:

. CALMA TEGAStation;
. DAISY Logician;
. Mentor Graphics;
. P-CAD IBM-PC;
. Futurenet IBM-PC;
. VALID.

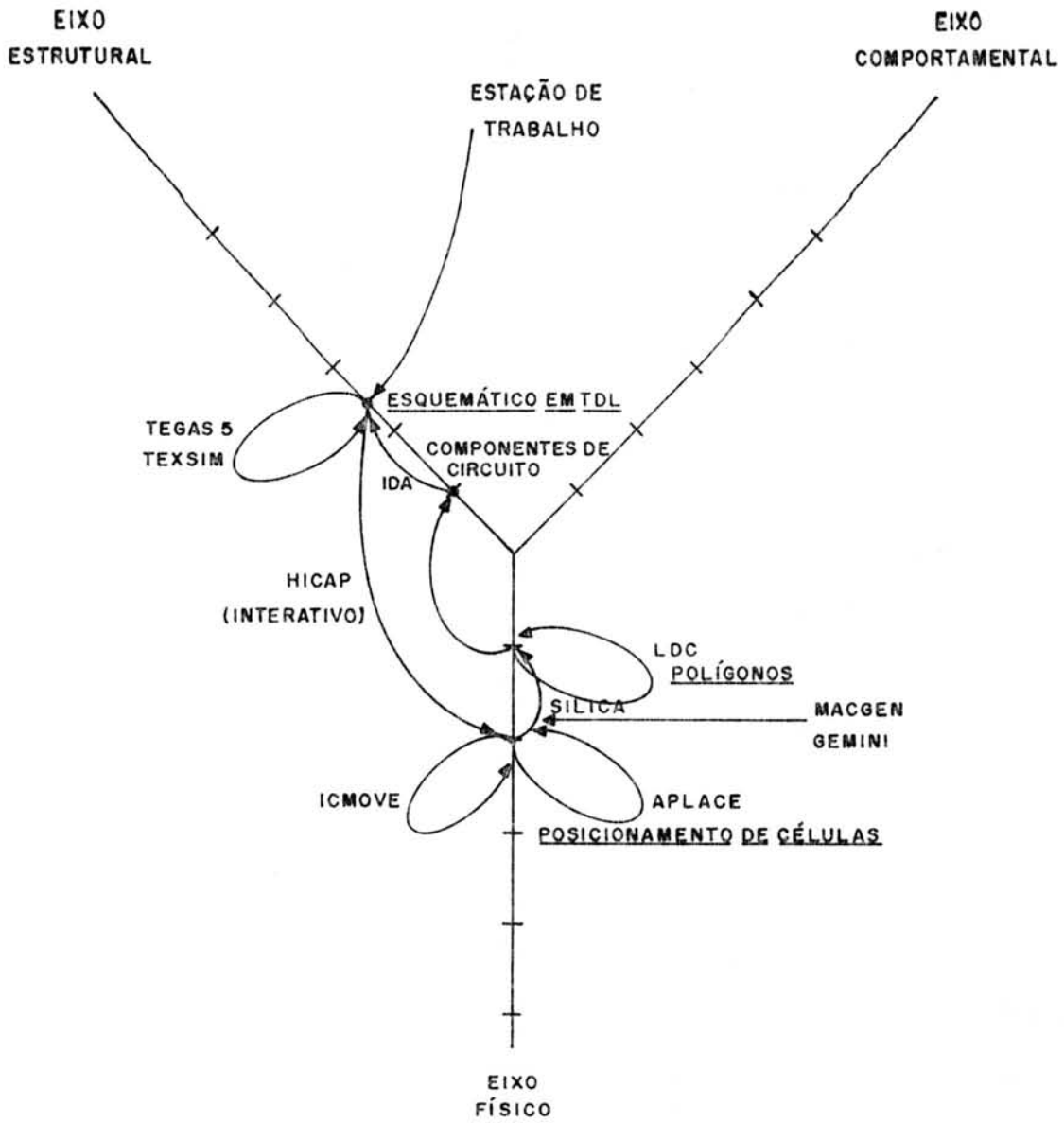


Figura 3.9 - Processo de projeto empregado pelo sistema MEC-CAD da GE

As saídas das estações de trabalho são convertidas para a linguagem TDL (TEGAS Design Language), a entrada primária do sistema. Um programa não mostrado no diagrama Y é responsável pela tarefa de gerência do

ambiente, o CADEXEC. A descrição mostrada no diagrama Y corresponde à do subsistema SILCA, voltado para CIs pré-difundidos. O MEC CAD possui um segundo subsistema (PLINT), dedicado ao projeto de CIs pós-difundidos.

Os simuladores TEGAS 5 e TEXSIM podem ser usados, após a geração da entrada do sistema, para os vários passos de verificação:

. simulação do comportamento;
. simulação de temporização;
. simulação de falhas;
. simulação de falhas detectáveis;
. geração de vetores de teste.

Para pré-difundidos, a GE desenvolveu GAMDL ("Gate Array Master Design Language"), uma linguagem de descrição de matrizes de pré-difundidos. GAMDL permite a especificação de parâmetros topológicos da matriz, tais como posições de contatos e canais de roteamento. Estes parâmetros dirigem os processos responsáveis pelo posicionamento de células e pelo traçado de interconexões.

A partir da descrição TDL, das informações constantes na biblioteca de células e da descrição da matriz em GAMDL, o programa HICAP guia o usuário na geração do posicionamento inicial, baseado na hierarquia TDL. ICMOVE é um programa interativo que trabalha sobre o resultado de HICAP, o posicionamento inicial, buscando gerar um melhor aproveitamento do espaço ocupado. Finalmente, o programa APLACE gera o posicionamento final de forma automática, através do uso da técnica de recozimento simulado ("simulated annealing").

Após obtido o posicionamento final, SILICA, um traçador de interconexões construído com base no algoritmo de Lee, é usado para a geração automática das rotas. SILICA

usa o conceito de transparência das células de base para obter um melhor aproveitamento de área.

O "layout" final é verificado usando-se LDC, um verificador de regras/extrator de circuitos. O resultado da extração elétrica é uma descrição estrutural do nível de abstração de circuitos. Esta descrição é usada pelo programa IDA, de avaliação de atrasos das interconexões. A saída gerada por IDA é utilizada na ressimulação lógica do diagrama de esquemáticos, após corrigidas as estimativas de atrasos iniciais ("back annotation").

Uma última observação sobre o sistema MEC CAD diz respeito à geração da biblioteca de macros. Dois programas são aplicados nesta tarefa, MACGEN e GEMINI. O primeiro é um editor gráfico interativo capaz de gerar macros para o núcleo da matriz do pré-difundido. O segundo é dedicado à criação de macros de E/S e circuitos analógicos especiais, usados principalmente na periferia do "gate array".

O sistema está instalado em computadores da linha DEC-VAX (VAX 11/780 e VAX 8600).

A figura 3.10 mostra o diagrama Y associado ao DASH GA Compiler, da Futurenet Division, Data I/O Corp.

O primeiro comentário possível sobre este sistema é que ele foi desenvolvido não no âmbito de uma fundição de silício, mas por um fornecedor de ferramentas de projeto genéricas ("third party vendor"). O sistema DASH GA, como se pode notar a partir do diagrama Y, trabalha sobre descrições de natureza funcional e estrutural, abordando o nível físico de forma bastante restrita. A intenção deste posicionamento é obter independência do sistema de PAC em relação à fundição onde será implementado o circuito, uma característica desejável a sistemas de PAC genéricos.

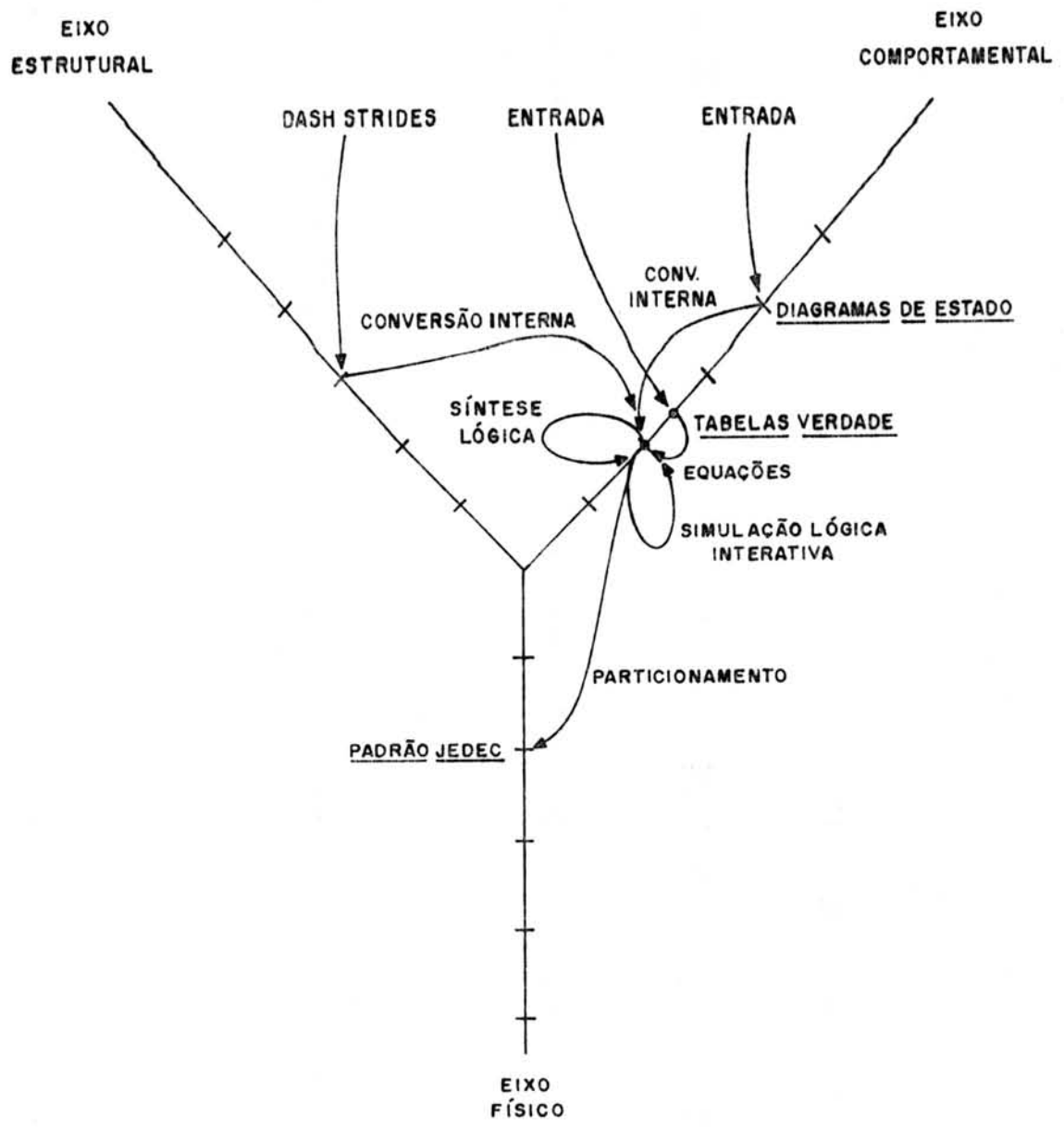


Figura 3.10 - Processo de projeto empregado pelo sistema DASH-GA Compiler da Futurenet

Segundo Weil et alii /WEI 86/, circuitos lógicos digitais podem ser classificados em três grandes grupos:

. partes operativas ("data paths");
. blocos funcionais;
. lógica de controle.

Ainda segundo os mesmos autores, apenas partes

operativas se prestam para a descrição gráfica a nível de portas lógicas, ou seja, a abordagem adotada na maioria dos sistemas de captura de projeto (captura de esquemáticos). Os autores defendem uma descrição em termos funcionais como a mais adequada para os outros dois grupos.

O DASH GA permite, desta maneira, a descrição dos blocos de um projeto em um de quatro formatos possíveis:

. mediante uso de uma ferramenta de captura de esquemáticos tradicional para a entrada de partes operativas, o editor hierárquico DASH/STRIDES;

. mediante uso de equações ou de tabelas verdade para a captura de blocos funcionais;

. mediante o uso de diagramas de estado para a captura de lógica de controle.

No diagrama Y, percebe-se a natureza estrutural da descrição gerada como saída do DASH/STRIDES. Equações, tabelas verdade e diagramas de estado são especificados através da linguagem de projeto de ASICs ("ASIC Design Language"), fornecida junto com o sistema DASH GA.

A captura de projeto inicial se faz com o DASH/STRIDES, obtendo-se um diagrama de blocos. Cada bloco é a seguir descrito usando o formato de entrada considerado mais conveniente, dentre os quatro citados. Caso o diagrama de blocos seja simples, pode-se evitar usá-lo, descrevendo cada módulo diretamente no formato mais adequado. O diagrama Y ilustra as duas possibilidades.

Todas as descrições funcionais são internamente convertidas para equações e tratadas de forma idêntica.

A verificação de projeto é feita por simulação lógica interativa da descrição global do sistema ou de módulos estanques, estejam eles descritos funcional ou

estruturalmente. O passo posterior à simulação é denominado de síntese lógica, estando dividido em três passos básicos:

. conversão de equações em somas de produtos booleanos;

. eliminação de redundâncias na lógica;

. fatoração dirigida pelo usuário, para prover um balanço adequado entre área ocupada e desempenho.

Após a etapa de síntese, é feito o particionamento do projeto em módulos físicos. Protótipos podem ser implementados usando PLDs ("Programmable Logic Devices"). O particionamento só exige do usuário a especificação das saídas dos módulos, as entradas necesárias sendo detectadas e separadas automaticamente pelo sistema.

Finalmente, é gerada uma descrição padronizada de cada módulo em termos de portas lógicas universais, usando para tanto as macrocélulas do padrão número 12 JEDEC. Este padrão pode ser fornecido a uma fundição para a confecção do CI pré-difundido, mediante posicionamento e traçado de interconexões automatizados. O sistema funciona em equipamentos compatíveis com a linha PC-AT da IBM.

Muitos outros sistemas de PAC foram encontrados na bibliografia, cada qual com um enfoque particular em função de decisões de implementação. Uma lista não exaustiva dos sistemas analisados é mostrada abaixo:

. o sistema ULA CAD da Ferranti Electronics Limited, dedicado a circuitos pré-difundidos bipolares e construído sobre um minicomputador PDP 11/23 /RAM 81/;

. o subsistema LORES-2 da Mitsubishi Electric Corporation, associado com o subsistema MARS-M3, o primeiro voltado para a síntese lógica e otimização, e o segundo voltado para a geração automatizada de "layout" de pré-

difundidos a partir da lógica gerada por LORES-2 /TAN 81a/ /TAN 81b/;

. o sistema integrado DA, da Hitachi Limited, dedicado ao tratamento de circuitos pré-caracterizados, capaz de realizar conversões de CIs pré-difundidos em CIs pós-difundidos e vice-e-versa /OHN 82/;

. o sistema Silicon Compiler da Lattice Logic Limited, voltado para os níveis superiores de abstração de projeto, com subsistemas de depuração estrutural, de depuração funcional e implementação física distintos /GRA 82/;

. o sistema UK5000, resultante de um convênio entre sete entidades, estatais e privadas, do Reino Unido, capaz de construir pré-difundidos 100% testáveis, com geração de "layout" completamente automatizada e portável entre pelo menos 4 máquinas alvo (IBM 370, ICL 2900, Prime 750 e VAX 11/780) /GRI 83/;

. o sistema CAE/CAD da Matra Design Systems, funcionando em PC-AT da IBM ou na linha VAX e realizando operações comumente só executadas em computadores de porte, como simulação de falhas e geração de "layout" /JOH 86/;

. o sistema DDL/SX da Fujitsu Laboratories Limited, baseado na síntese de circuitos pré-difundidos com uso de sistemas especialistas em vários pontos do processo de concepção /SAI 86/;

. o MACDAS, desenvolvido na Universidade de Osaka, sobre computadores compatíveis com IBM-PC e realizando a implementação de "gate arrays" com aplicação de procedimentos de síntese algorítmica automatizada de lógica multinível /SAS 86/.

3.2 Tecnologias, Células de base, Matrizes e Biblioteca de Células

Nesta seção, serão discutidos os requisitos do aspecto teoria do sistema-alvo, constituintes da ótica

pragmática elaborada para métodos de projeto de CIs pré-difundidos. Na realidade, estes requisitos são responsáveis por criar uma diferenciação no tratamento dado a circuitos pré-difundidos. Ferramentas para a manipulação de "gate-arrays" devem levar em consideração estes requisitos para extrair deste estilo de projeto o máximo desempenho.

3.2.1 Tecnologias

Os muitos fornecedores de pré-difundidos oferecem um largo espectro de tecnologias /HAR 83/. Podem ser escolhidas desde as tecnologias de alto desempenho como ECL ("Emitter-Coupled Logic") e GaAs, passando pelas lógicas bipolares de média velocidade como CML ("Current Mode Logic"), TTL ("Transistor-Transistor Logic"), STL ("Schottky Transistor Logic"), ISL ("Integrated Schottky Logic") e IIL ("Integrated Injection Logic"), chegando às de baixa freqüência de operação e alta densidade, como DTL ("Diode Transistor Logic"), NMOS e CMOS.

Além do compromisso <u>freqüência de operação/potência dissipada</u> usado na escolha da tecnologia, existe a consideração a ser feita quanto à complexidade lógica do circuito. Pré-difundidos digitais possuem complexidades variando entre 50 e 50000 portas equivalentes /HAR 83/ /CAR 86/, com pinagens indo de 8 a 256 pinos /HAR 83/. Pré-difundidos que integram funções analógicas e digitais já são aplicados comercialmente /SCH 85/.

Poucos fornecedores oferecem uma grande variedade de tecnologias, mas isto não é nem muito viável nem muito necessário.

3.2.2 Células de base

Um método de projeto, como discutido anteriormente, deve suportar várias células de base. Para tanto, o domínio de características destas células é requerido. Estas características são enumeradas abaixo, de forma não exaustiva:

. número de canais de interconexão;
. tipo de componentes;
. número de componentes na célula;
. conectividade intra e extra-celular;
. camadas personalizáveis;
. transparência ao roteamento global.

O método deve prover mecanismos de adaptação a células de base com características distintas no ambiente de projeto. Tais mecanismos devem permitir:

. especificação de células;
. designação do número e posição dos canais de interconexão;
. descrição das regras de projeto para uso da célula.

Faz-se agora uma breve revisão das características e problemas do estado da arte em células de base para pré-difundidos, baseado no exposto em /CAL 87a/.

Com a proliferação do uso de CIs pré-difundidos, muitas propostas de células de base têm surgido, visando reduzir as deficiências abaixo citadas, inerentes ao estilo de projeto com pré-difundidos:

. utilização ineficiente da área de substrato semicondutor;
. dificuldade de implementar blocos regulares de

alta densidade (RAM/ROM/Multiplicadores);

. dificuldade de implementar circuitos de alto desempenho;

. dificuldade para utilizar um grande percentual dos elementos de circuito disponíveis.

O primeiro problema é atacado através do uso de células de base com o máximo de área ativa e o mínimo de canais de interconexão. Três abordagens existem hoje para acessar a relação área-ativa versus área de canais de interconexão:

. células isoladas ("block cell"), separadas umas das outras por canais verticais e horizontais de interconexão;

. filas de células ("row cell") percorrendo o CI de lado a lado, com canais de interconexão na horizontal;

. mar de células ("sea of gates" ou "channeless"), onde os canais de interconexão explícitos não existem.

As figuras 3.11, 3.12 e 3.13 mostram uma planta baixa para cada uma das abordagens.

As duas primeiras abordagens são tradicionais. A abordagem moderna ("sea of gates") obtém o máximo de área útil no núcleo do CI pré-difundido, com as rotas sendo traçadas mediante duas técnicas possíveis:

. criação dinâmica de canais, impedindo o posicionamento de células de biblioteca em determinadas regiões da matriz;

. canais de interconexão atravessando células de biblioteca em posições estratégicas (transparências), onde não interferem com a função lógica desta, técnica derivada da abordagem aplicada no estilo pós-difundido /REI 88/.

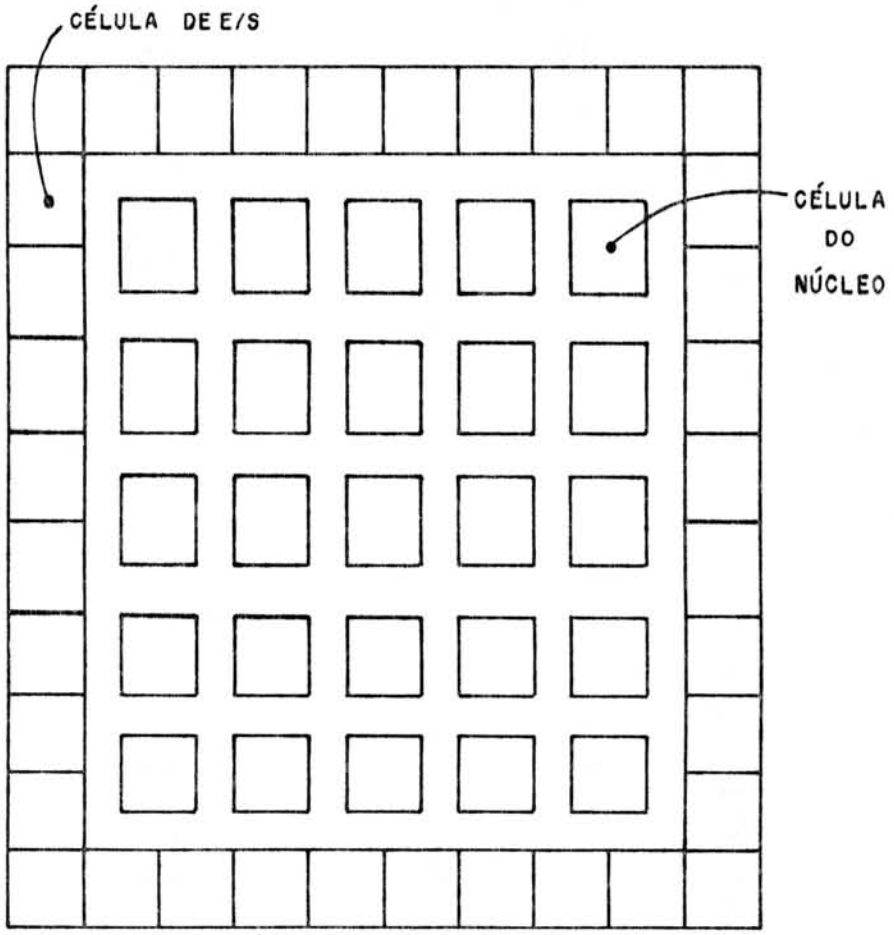


Figura 3.11 - Planta baixa de matriz com células isoladas

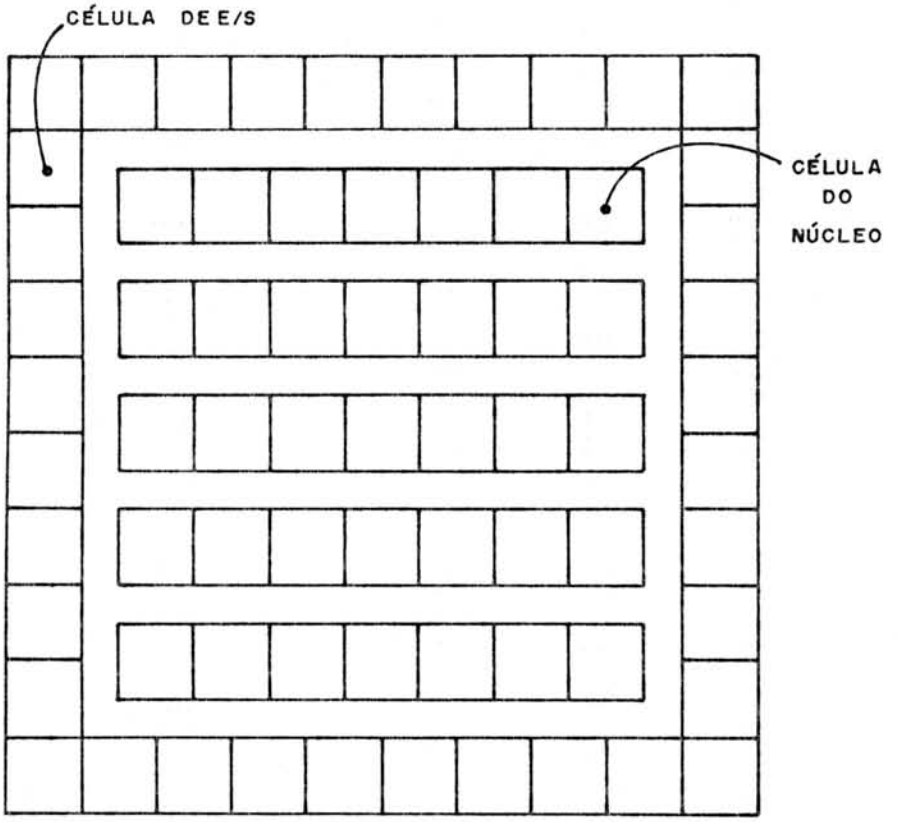


Figura 3.12 - Planta baixa de matriz com filas de células

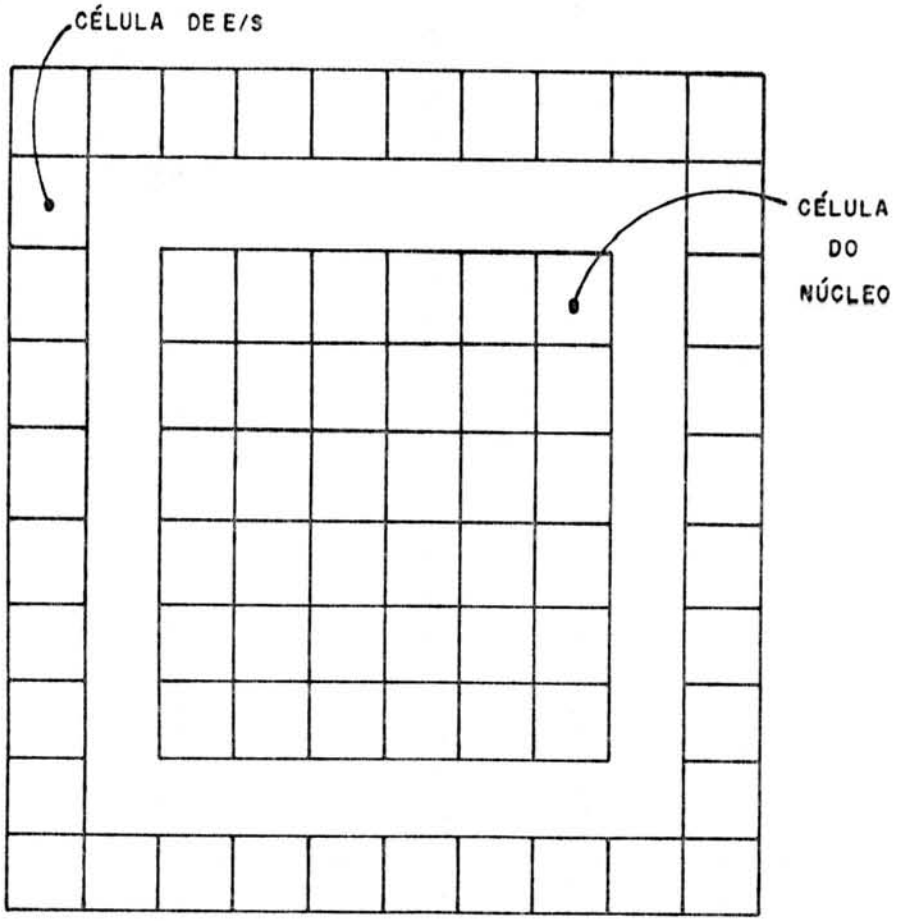


Figura 3.13 - Planta baixa de matriz com mar de células

As figuras 3.14 e 3.15 mostram exemplos de traçado de rotas com as duas técnicas.

A segunda técnica exige o uso de ferramentas poderosas na geração automática de interconexões, pois o uso eficiente das transparências é uma tarefa um tanto complexa.

Outra abordagem capaz de economizar área de semicondutor consiste em usar células de base com isolação por porta de polissilício ao invés da tradicional isolação por óxido espesso. Nesta técnica, as zonas de difusão formam linhas contínuas, interrompidas a intervalos regulares por regiões de óxido fino onde está presente uma porta de transistor. Cada linha constitui uma conexão série de transistores com canal comum. Caso um destes transistores seja colocado permanentemente no modo de

funcionamento de corte, os dois lados da conexão série ficam eletricamente isolados . Esta abordagem facilita a implementação de portas lógicas de muitas entradas, além de economizar canais de interconexão entre células de base separadas por óxido espesso /NOI 85/. Sakashita et alii /SAK 85/ relatam economias de área de 10 a 26%, obtidas com esta abordagem em relação àquela com isolação por óxido espesso.

As figuras 3.16 e 3.17 ilustram as duas abordagens.

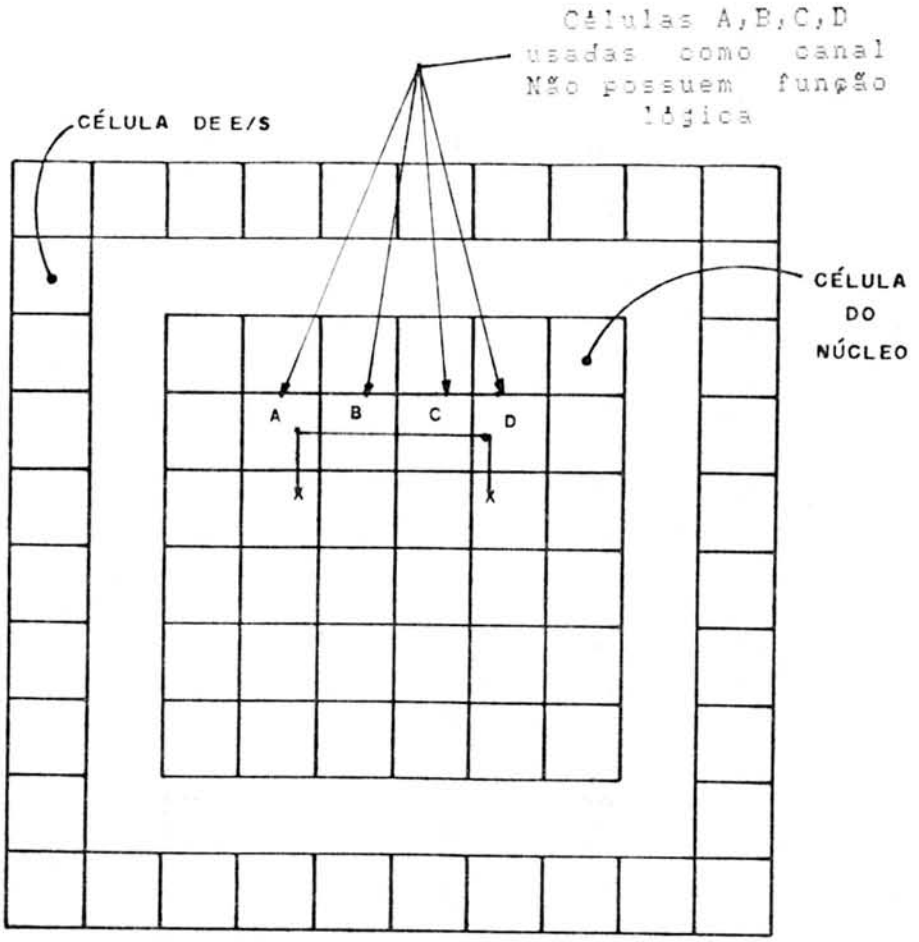


Figura 3.14 - Alocação dinâmica de canais

O problema dos blocos de alta densidade é atacado de duas formas distintas:

. empregando estruturas especiais em posições estratégicas da matriz, com formatos e células diferentes da célula de base do núcleo do pré-difundido;

. empregando uma célula de base adequada para a geração dos blocos de alta densidade necessários.

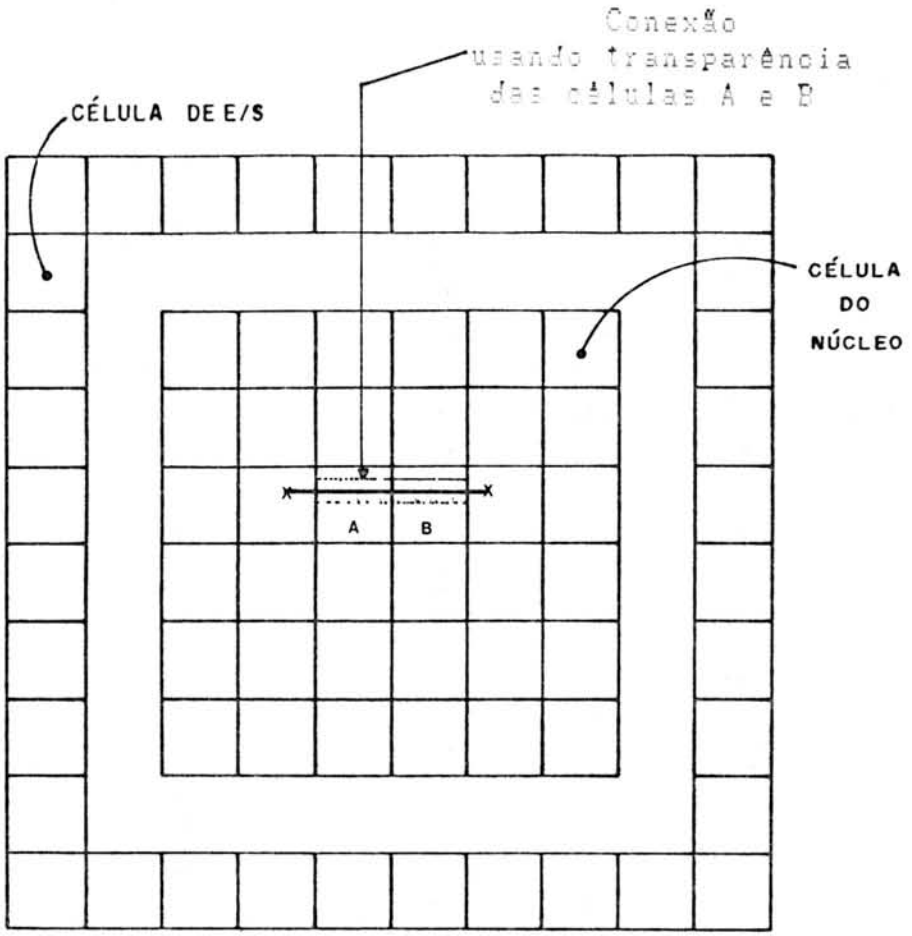


Figura 3.15 - Conexão com uso de transparências

A primeira forma implica a presença de regiões funcionalmente pré-definidas do CI, otimizadas em termos de área para a aplicação em vista. Os blocos de alta densidade são, em geral, personalizáveis (por exemplo, relação número de palavras versus largura da palavra em RAM ou ROM especificáveis pelo usuário) e ocupam a periferia do CI. Exemplos são desenvolvidos por Miyahara et alii /MIY 86/ e Nakasato et alii /NAK 86/. A figura 3.18 ilustra a proposta

de Miyahara.

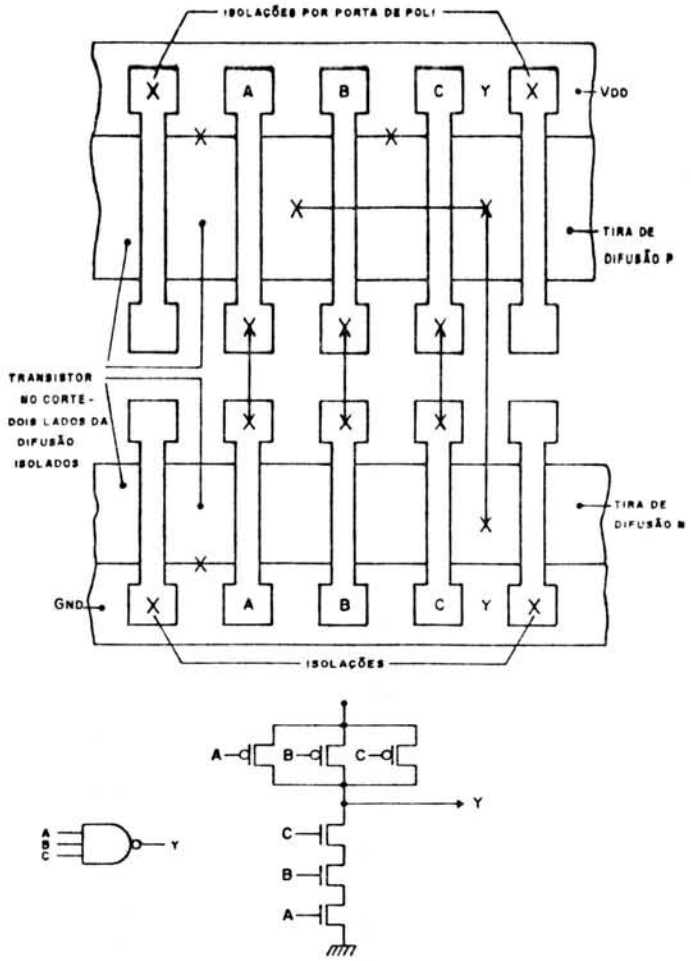


Figura 3.16 - Isolação por porta de poli

A segunda forma implica um estudo detalhado das características de conectividade e flexibilidade das células de base, de maneira a garantir eficiência para um conjunto amplo de aplicações. Exemplo desta forma é a proposta de Takahashi et alli /TAK 85/, e a célula de base proposta no método CIPREDI descrita em /CAL 87a/ e no capítulo 5 deste trabalho.

O baixo desempenho é um fator de difícil correção nos pré-difundidos. As abordagens deste item restringem-se ao uso de tecnologias avançadas de implementação ou de tecnologias que permitem por características construtivas o

uso de altas freqüências de operação, como ECL e CML.

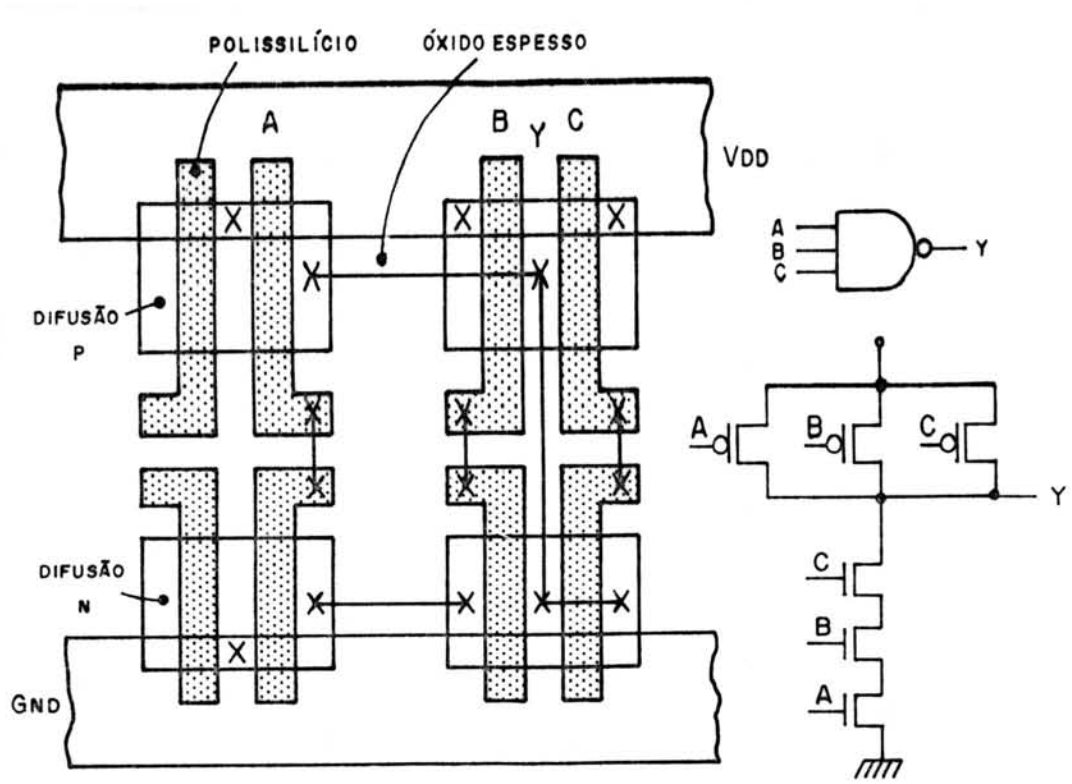


Figura 3.17 - Isolação por óxido espesso

Para alcançar uma alta taxa de utilização dos elementos de circuito da matriz, usa-se múltiplos níveis de conexão, ao invés do único nível da abordagem tradicional. Vários fabricantes oferecem, hoje, opções de pré-difundidos com duas, três, quatro ou mais camadas personalizáveis. Esta flexibilidade, contudo, contribui para o aumento substancial nos custos de fabricação e portanto no custo final de projeto /SAI 85/, além de exigir ferramentas de tratamento de "layout" mais sofisticadas.

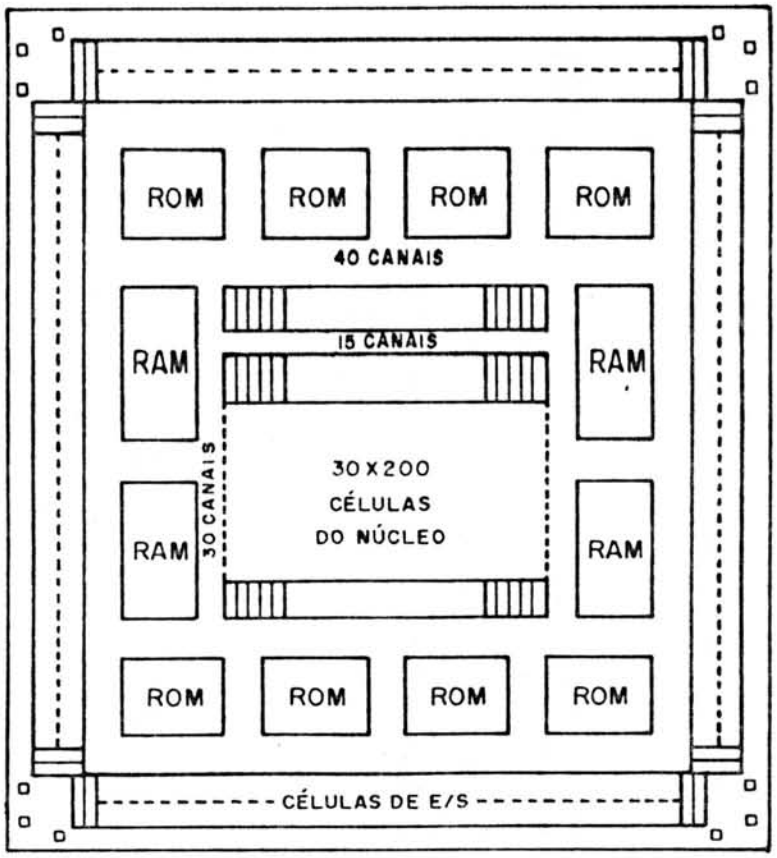


Figura 3.18 - Matriz com blocos especializados

3.2.3 Matrizes

As principais considerações de projeto para um sistema integrado são, dependendo da aplicação:

. custo;

. desempenho;

. outros critérios, como por exemplo, tipo e dimensões do encapsulamento.

Cada fundição organiza os pré-difundidos segundo famílias ou séries, que consideram estes pontos. Em geral os pré-difundidos das diversas famílias possuem características que provêem um espectro quase contínuo de escolhas para o usuário. Os critérios tecnológicos

envolvidos na especificação de uma família são:

. dimensões mínimas do componente, em geral, comprimento de canal de um transistor;
. encapsulamentos usados;
. número máximo de pinos de E/S;
. número de camadas de personalização;
. tipo de técnica de isolação;
. utilização percentual máxima das células da matriz para geração de lógica /LSI 84/.

Cada fabricante oferece um conjunto de passos que devem ser seguidos pelo usuário na escolha da tecnologia, matriz e encapsulamento mais adequados para um dado projeto, dentro do espectro de matrizes fornecidas pela fundição.

Como exemplo, citamos a LSI /LSI 84/, com 4 séries de "gate arrays", cada uma contendo entre 6 e 10 matrizes diferentes, com largura de canal entre 3,5 e 2 microns e pinos de E/S variando entre 32 e 216. A GE/Intersil, por sua vez, possui uma única família, MEC4U /GEN 83/, com 3 matrizes distintas e largura de canal de 4 microns.

3.2.4 Biblioteca de células

Biblioteca de células são, do ponto de vista do usuário, um conjunto de blocos construtivos de CIs. Sua utilidade maior é permitir o uso de tecnologia de concepção de CIs aos projetistas de circuitos discretos, ou ainda, a qualquer pessoa familiarizada com os métodos de projeto lógico /GOU 85/.

O fabricante fornece uma descrição de sua biblioteca de células de forma tabular contendo, dentre

outras, as seguintes informações, para cada célula:

. nome;

. símbolo lógico;

. tabela verdade do comportamento lógico;

. tabela de capacitâncias de entrada, contendo o número de cargas capacitivas unitárias associadas a cada ponto de entrada da célula;

. tamanho, em número de portas equivalentes;

. sintaxe de uso na linguagem de descrição de "hardware" usada pela fundição;

. características de chaveamento, onde se explicita o atraso de propagação da célula para uma carga capacitiva unitária na saída;

. equação de cálculo do atraso de propagação, para cargas capacitivas não unitárias conectadas na saída;

. esquema lógico, para células complexas;

. curvas de desvio, para o cálculo de variações no atraso de propagação devido a mudanças na temperatura, tensão de alimentação e parâmetros de processo /GOU 85/.

Muitos fabricantes fornecem bibliotecas de células compatíveis com circuitos de catálogo tradicionais, como a série TTL 7400 e a série CMOS 4000. Isto torna mais fácil a adaptação dos projetistas de sistemas à concepção com pré-difundidos. Os projetistas devem contudo atentar para diferenças existentes entre "gate arrays" e SSI/MSI. Lipp e Krejcik apontam um conjunto de regras a serem seguidas para levar em conta estas diferenças /LIP 83/ /KRE 83/.

Existem ainda bibliotecas contendo células de mais de um tipo. A AMI /GOU 85/ emprega bibliotecas com células de dois tipos: macros pré-roteadas e macros de "software". As primeiras são usadas para implementar as portas lógicas básicas, "latches" e "flip-flops"; estas macros possuem "layout" pré-definido e são caracterizadas

eletricamente com precisão. As macros de "software" incorporam várias macros pré-roteadas para gerar blocos funcionais complexos como contadores e registradores de deslocamento. Seu "layout" não é pré-definido, sendo gerado pelas ferramentas automáticas de posicionamento e traçado de rotas. Isto aumenta a flexibilidade na geração das máscaras. A caracterização elétrica das macros de "software" é obtida somando os atrasos das macros pré-roteadas que a compõem.

Finalmente, bibliotecas de células podem ser generalizadas para uso com diferentes células de base e com diferentes estilos de projeto.

No capítulo 5, uma implementação de biblioteca de células para pré-difundidos é proposta, dentro do desenvolvimento do método CIPREDI.

3.3 A Relação Cliente/Fornecedor

O aspecto mais crítico no projeto de circuitos pré-difundidos é, sem dúvida, a interação entre o cliente interessado em pré-difundidos e o fornecedor, conforme ressalta Hardage /HAR 83/.

O sucesso de um projeto de "gate array" exige uma estreita cooperação entre o projetista e o fornecedor. É importante que tanto cliente quanto fornecedor entendam os objetivos do projeto, bem como as conseqüências da escolha do estilo de implementação com "gate arrays".

Em primeiro lugar, a alteração do projeto após a prototipação é uma tarefa muito complicada, muito mais que a alteração de placas de circuito impresso. Portanto, em projeto com pré-difundidos, é essencial que o primeiro CI fabricado funcione dentro de todas as especificações e

expectativas do cliente. Falhas na primeira fabricação podem facilmente inviabilizar o produto final, seja devido aos custos adicionais envolvidos no reprojeto, seja devido a atrasos de cronograma que acarretem a obsolescência do produto /BRU 83/.

Para evitar problemas de comunicação e interação entre cliente/fornecedor, as fundições de silício envolvidas com a manutenção de pré-difundidos aplicam duas técnicas:

. o estabelecimento de formas pré-definidas de interação, criando responsabilidades específicas para cada uma das partes envolvidas;

. o uso de documentos formais para especificação do circuito e modelos padronizados para efetuar troca de informações de projeto.

Antes de apresentar as técnicas acima, uma observação deve ser feita quanto aos profissionais envolvidos no projeto e fabricação de pré-difundidos. Os usuários de "gate arrays" constituem um público especialista em projeto de sistemas utilizando CIs de catálogo. Contudo, a capacidade destes projetistas para a concepção de CIs pré-caracterizados varia muito. Hardage /HAR 83/ classifica-os em três grupos quanto a esta capacidade:

. sofisticados - contam com amplos recursos humanos e computacionais, além de experiência comprovada em projeto de CIs;

. experientes - possuem alguma experiência em projeto de CIs e poucos recursos;

. inocentes - não possuem experiência em projeto de CIs.

Quanto aos fornecedores, Jack /JAC 85/ identifica

três categorias principais:

. fornecedor de semicondutores nativo - as próprias fundições de silício;

. as "systems houses" - com toda a infra-estrutura de projeto montada, subcontratam fundições para a execução de serviços de manufatura e teste do componente;

. as "design houses" independentes - podem agir como consultores, pois possuem experiência de projeto com várias fundições e com sistemas de PAC atuais.

Cosley /COS 83/ apresenta uma discussão detalhada deste último tipo de fornecedor. Os serviços prestados por estas "design houses" vêm sendo bastante procurados. No Brasil, a forma encontrada é a segunda, "systems houses", vinculada a uma única fundição no exterior.

A subseção 3.3.1 trata da primeira técnica, denominada de formas de interação. A subseção 3.3.2 comenta os documentos e modelos usados, apresentando um estudo de caso.

3.3.1 Formas de interação

Existe um certo estágio no ciclo de projeto e fabricação de um componente pré-difundido onde a responsabilidade pelos passos posteriores da implementação muda da esfera de ação do cliente para a esfera de atuação do fornecedor. Rupp /RUP 87/ identifica três estágios diferentes onde esta troca de esfera de atuação do projeto pode se dar. A figura 3.19 ilustra os possíveis caminhos percorridos ao longo da implementação.

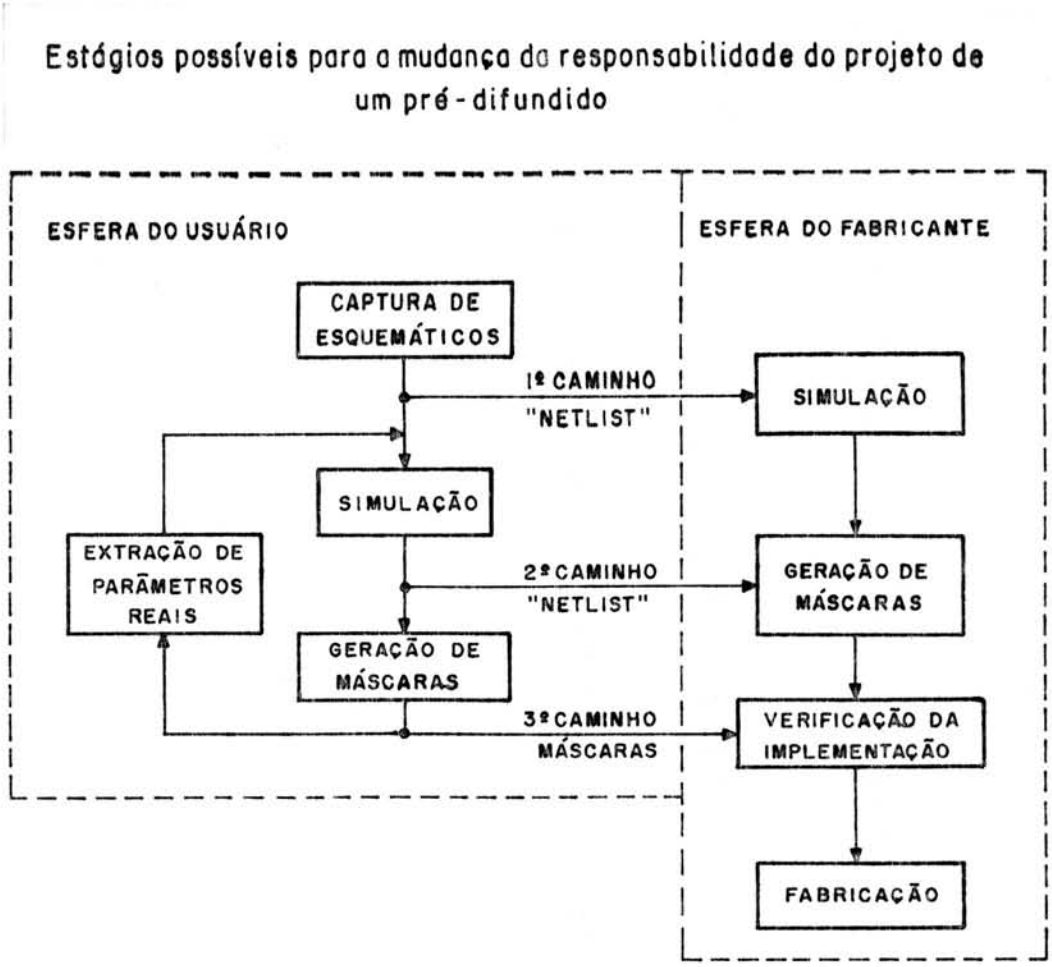


Figura 3.19 - Pontos possíveis de mudança da responsabilidade de projeto

A observação da figura leva ao estabelecimento dos pontos de troca dos dados de projeto entre cliente e fornecedor:

. após a captura do esquemático;

. após a simulação do esquema fornecer resultados satisfatórios (validação do esquema);

. após a ressimulação e a geração das máscaras.

No primeiro caso, o cliente precisa suprir, para o fornecedor, uma representação lógica do circuito. Em troca, ele receberá um protótipo validado por este. O custo desta abordagem pode atingir alto preço, tipicamente acima

de 50000 dólares. Além disso, a equipe de projeto do cliente não tem qualquer controle sobre desempenho, compromissos de projeto ou tempo de implementação.

O segundo caminho garante maior controle do projeto pelo cliente, pois este valida o esquemático através de iterações de simulação. Ao mesmo tempo ocorre a geração dos vetores de teste a serem usados pelo fornecedor na verificação do componente manufaturado. O fornecedor, entretanto, ressimula o projeto com seu próprio simulador sintonizado para o processo de fabricação específico. Esta ressimulação valida a integridade da simulação executada pelo cliente, além de salvaguardar o fornecedor contra o não cumprimento de especificações no projeto original. A ressimulação é necessária, uma vez que dados de fabricação e projeto não estão disponíveis para o usuário. Questões de sigilo do ambiente do fornecedor limitam a validade dos esforços de simulação do cliente.

O último caso implica o envolvimento do cliente com a captura de esquemáticos, simulação, posicionamento e traçado de rotas e ressimulação para a personalização do pré-difundido. Para esta forma de interação, é indispensável a utilização das bases de dados lógicas e físicas do fabricante, de forma a permitir ressimulações confiáveis e geração do "layout". Uma especificação lógica e física é fornecida para o fabricante, cabendo a este a verificação final do projeto e manufatura do CI. A grande vantagem deste caminho está na drástica redução das despesas de engenharia não recorrentes e no total controle do projeto pelo cliente. Entretanto, toda a responsabilidade de cumprimento das especificações pelo componente fabricado recai também sobre o cliente.

Um exemplo prático de forma de interação durante o projeto e a fabricação de pré-difundidos é apresentado no

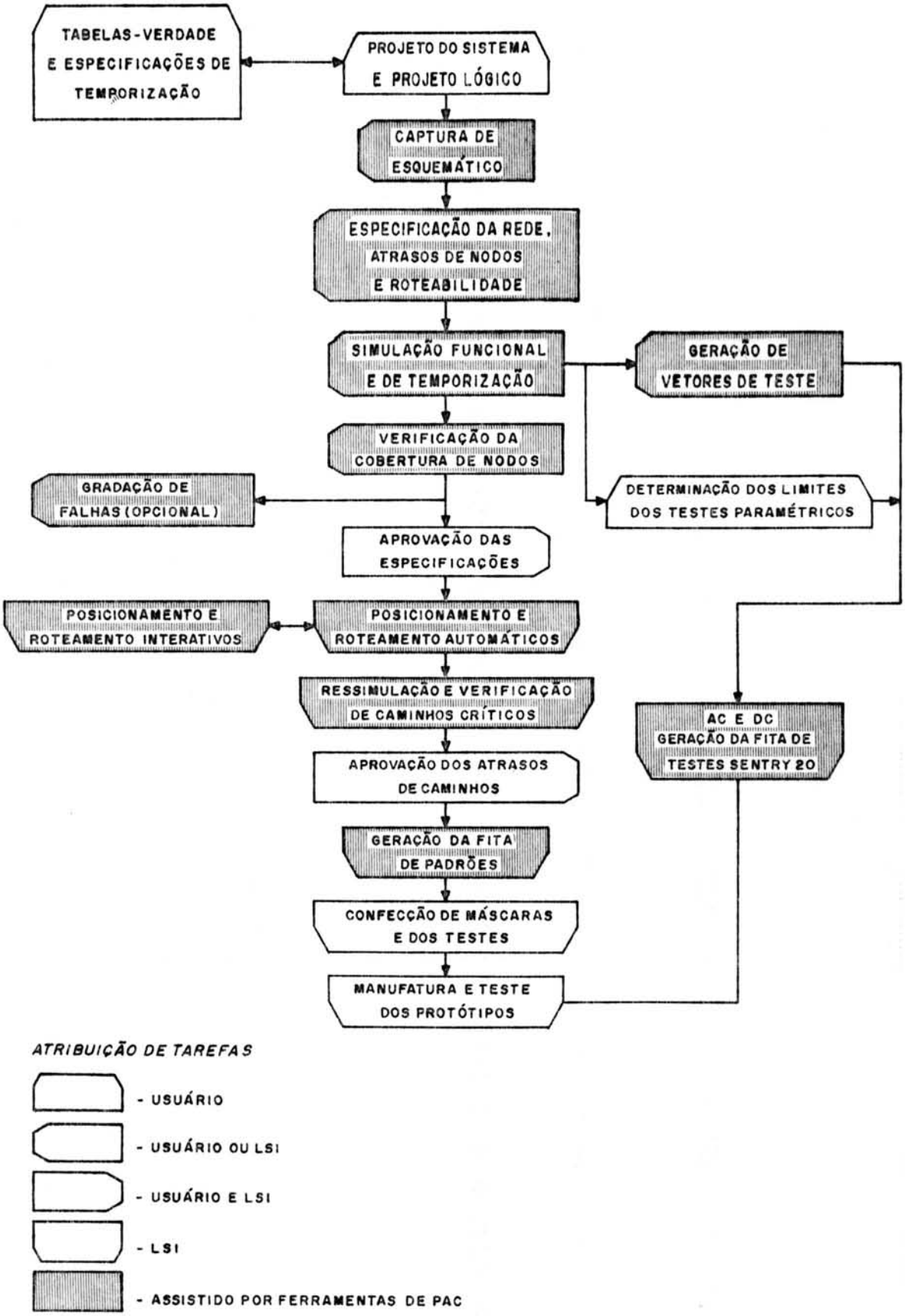


Figura 3.20 - Projeto de pré-difundidos no sistema LDS da

A figura mostra os diferentes passos de implementação de um "gate array" usando o método proposto pela LSI Corporation /LOB 83/. Note-se, ainda na figura 3.20, a distribuição sugerida de encargos entre o usuário e a fundição.

Tradicionalmente, o projeto físico tem sido domínio exclusivo do fornecedor de pré-difundidos, ou seja, a última forma de interação não tem estado disponível para o público usuário de ASICs. A evolução da tecnologia para processos submicron e complexidade de dezenas de milhares de portas equivalentes, contudo, obriga a consideração de aspectos físicos da concepção pelo projetista lógico. O "layout" de um CI pré-difundido pode atender ou não às especificações de funcionamento e de desempenho iniciais, independente da qualidade do projeto lógico final.

Para que projetos utilizando tecnologia do estado da arte em pré-difundidos sejam eficientemente implementados, torna-se necessário o estabelecimento de uma nova relação entre o fabricante de "gate arrays" e o fornecedor de ferramentas de PAC. Tal relacionamento deve permitir a criação de um ambiente de projeto físico de pré-difundidos que conte com com o aval do fabricante.

3.3.2 Documentos formais

Para que a relação cliente/fornecedor alcance resultados satisfatórios, uma interface formal deve ser estabelecida. A elaboração desta interface é feita pelo fabricante de CIs. Um conjunto de documentos padronizados é concebido e distribuído para os usuários dos serviços da fundição. Estes documentos contêm as informações de processo necessárias ao cliente e formulários a serem

especificação formal do projeto.

Hardage /HAR 83/ classifica as informações geradas pelo cliente em dois grupos:

. especificações funcionais;
. especificações elétricas.

As especificações funcionais obedecem a um modelo normalmente associado a uma linguagem formal de descrição de circuitos ("hardware description language"-HDL), aceita pelas ferramentas do fabricante. As primitivas desta linguagem são do nível lógico, na maioria das vezes, e podem ser de três tipos:

. compatíveis com circuitos MSI/SSI de catálogo, como as famílias TTL 54/7400, 100K ECL e 4000 CMOS;
. compatíveis com as células de biblioteca do fabricante;
. padrões binários de teste (descrição puramente funcional sem estrutura associada).

As especificações elétricas estão, antes de mais nada, incompletas ao serem recebidas pelo fabricante. Algumas especificações típicas para circuitos pré-difundidos são:

. descrição em alto nível do circuito, contendo um diagrama de blocos funcionais, fluxogramas e formas de onda de entrada e saída;
. características gerais de operação, como faixa de tensões de alimentação, freqüências máximas, atrasos de propagação, limitações na alimentação, dissipação de potência, tipo de encapsulamento e requisitos de temporização;

das entradas e saídas, como níveis lógicos, capacidades de fornecimento de corrente, capacitância de entrada e circuitos especiais (osciladores, "schmitt triggers", etc);

. lista de pinos de entrada, saída e alimentação, com a atribuição de pinos no encapsulamento escolhido, caso seja necessário.

Embora todos os fornecedores de CIs empreguem alguma padronização no formato das informações de processo e projeto distribuídas para os usuários, não existe ainda uma padronização de formatos aceita amplamente por um número considerável de fabricantes.

Este fato dificulta a apreensão das capacidades de cada fornecedor no momento da escolha da fundição a implementar determinado projeto. Especificações técnicas variam grandemente de fabricante para fabricante. Especificações de "pior caso" são obscuras, uma vez que raramente se define o que é pior caso. Alguns aspectos que merecem atenção do usuário nesta documentação são:

. número de portas equivalentes de cada configuração de CI pré-difundido em relação às taxas esperadas de utilização das portas, que podem variar de 40% /CAR 86/ a 95% /HAR 83/;

. atrasos de propagação das portas, raramente avaliados sob as mesmas condições para fabricantes diferentes;

. efeito do carregamento capacitivo das saídas, que pode acrescentar de 25 a 50% no atraso das portas;

. características de E/S podendo não satisfazer os requisitos de projeto do cliente;

. dissipação de potência em relação à freqüência de operação, uma consideração muitas vezes negligenciada pelo fabricante e esquecida pelo cliente.

Como exemplo de documentação fornecida pelos

fabricante e cliente durante um projeto de pré-difundidos, tem-se o caso da família MEC4U de "gate arrays" comercializados pela General Electric/Intersil /GEN 83/. A GE/Intersil usa dois documentos formais para intercâmbio de informações com o cliente. Estes documentos são denominados Especificação de Objetivos Preliminar e Especificação Final de Objetivos. Os documentos formais são fornecidos pelo cliente em dois pontos do ciclo de projeto, ditando marcos de verificação de andamento:

. antes do início do projeto lógico;

. antes do início dos trabalhos de simulação lógica.

No Anexo I deste trabalho existem exemplos de formatos para alguns do documentos constantes das Especificações de Objetivos. Para o preenchimento dos documentos, o fabricante fornece um conjunto de regras, tais como:

. o cliente deve especificar todos os requisitos dinâmicos (AC) do CI, inclusive caminhos críticos;

. a escolha do encapsulamento adequado deve seguir as especificações do cliente e respeitar os tipos de matrizes suportadas pela família MEC4U.

A tabela 3.3 abaixo mostra as características das matrizes disponíveis na família MEC4U.

Tabela 3.3 - Matrizes disponíveis na família MEC4U da GE/Intersil

MATRIZ	CÉLULAS DA BASE NA MATRIZ	CÉLULAS DE E/S NA MATRIZ	COLUNAS	LINHAS	DIMENSÕES (em mils)
MEC4U408	272	34	8	34	133X178
MEC4U756	504	44	14	36	188X186
MEC4U1500	1000	62	20	50	243X236

4 O MÉTODO CIPREDI

Este capítulo introduz o método CIPREDI. Cada um dos aspectos de métodos de projeto descritos no capítulo anterior corresponde a uma seção do presente capítulo.

Um conjunto de objetivos foi definido para o método CIPREDI:

a) criar um ambiente integrado para a concepção de CIs pré-difundidos, dotado de alto grau de interatividade com o usuário;

b) proporcionar um máximo de portabilidade e flexibilidade ao sistema de projeto, através do uso de técnicas e ferramentas com capacidade de reconfiguração;

c) abordar os principais níveis de abstração relacionados com o projeto de pré-difundidos;

d) aproveitar ao máximo conceitos desenvolvidos em sistemas de projeto já existentes, buscando a incorporação das virtudes de cada ambiente considerado;

e) permitir a implantação do sistema de projeto em equipamentos de custo reduzido, próprios para a atual realidade econômica do País;

f) garantir a automação extensiva dos processos de projeto de complexidade elevada.

Obviamente, os objetivos acima refletem um ideal difícil de ser alcançado em curto prazo. Contudo, eles fornecem diretrizes úteis para guiar a elaboração do método. Os objetivos são, em alguns casos, conflitantes, principalmente ao se considerar a implantação em equipamento de baixo custo aliada a um ambiente integrado e flexível.

De início, o cumprimento do objetivo c) deve ser descartado do escopo desta dissertação, devido a sua demasiada abrangência. Em seu lugar, o trabalho prevê a

abordagem inicial dos dois aspectos considerados mais críticos no projeto de pré-difundidos:

. a validação de uma descrição estrutural/funcional no nível lógico do diagrama Y;

. a conversão da descrição estrutural/funcional do nível lógico em um "layout" de máscaras de CI pré-difundido, de forma interativa.

A integração (objetivo a)) destes dois aspectos é sugerida na seção 4.2, e uma proposta geral de integração no método CIPREDI é apresentada na seção 6.1.

O último objetivo, ou seja, a automação de processos de projeto, está reservado para tratamento em passos futuros da implementação do método , como delineado no capítulo 7 e na última seção do presente capítulo.

Os objetivos restantes constituem metas a serem atingidas em cada passo do método CIPREDI, inclusive neste trabalho.

O método CIPREDI trabalha sobre conceitos desenvolvidos em dois projetos em andamento no CPGCC/UFRGS:

. o projeto AMPLO;
. o projeto LCCI.

O projeto AMPLO /WAG 87d/ é voltado para a concepção de sistemas digitais nos níveis superiores de abstração. Os níveis de trabalho do AMPLO denominam-se lógico, RT, e sistema, correspondendo, no diagrama de Gajski, aos níveis lógico, bloco funcional e algorítmico, respectivamente. O AMPLO usa os ramos estrutural e comportamental do diagrama, sem se envolver, até o momento, com representações físicas. Isto o torna independente do estilo de concepção adotado, seja lógica discreta, LSI de

catálogo ou circuitos personalizáveis.

O projeto LCCI (Laboratório de Concepção de Circuitos Integrados) /CUR 86/, aborda principalmente aspectos físicos e estruturais do projeto de CIs. Os níveis de abstração do diagrama Y acessados hoje pelo LCCI são o nível circuito e o nível lógico.

O método CIPREDI procura integrar ferramentas e linguagens de descrição dos dois projetos, com o intuito de criar um ambiente adequado para a concepção de pré-difundidos.

4.1 Teoria do Sistema Alvo

Os requisitos relacionados a este aspecto do método são de responsabilidade do fabricante de pré-difundidos. Contudo, a proposição de um método de projeto sem considerar estes requisitos é impensável /HAR 83/. Dentro da elaboração do CIPREDI foram realizadas escolhas e implementações, visando dar suporte ao desenvolvimento posterior do método. As implementações são descritas no capítulo 5 deste trabalho, e envolvem os requisitos células de base e biblioteca de células. Nesta seção, aborda-se os outros dois requisitos: tecnologia de implementação e matrizes, com as escolhas realizadas.

4.1.1 Tecnologias e matrizes

As escolhas que guiam, atualmente, a elaboração do método CIPREDI têm sido feitas com base muito mais na restrição de meios disponíveis para a implementação de CIs no ambiente acadêmico, do que em uma criteriosa seleção de recursos.

A tecnologia escolhida para a implementação do método foi CMOS digital, por uma série de motivos:

. CMOS é hoje a tecnologia com maior capacidade de integração de componentes em um único CI;

. a alta densidade permite a implementação de funções complexas com pequena ocupação de substrato de silício;

. a exploração de tecnologias baseadas em silício, dentre estas a tecnologia CMOS, é o campo mais desenvolvido e dominado da eletrônica do estado sólido;

. CMOS é a tecnologia com menor consumo de potência em uma ampla faixa de freqüências de operação;

. a disponibilidade de fabricantes de pré-difundidos na tecnologia CMOS é maior do que em qualquer outra;

. o projeto de circuitos digitais CMOS é simples, pois, sendo CMOS uma lógica não relacionada, permite uma tradução direta de descrições abstratas para implementações físicas com um mínimo de cuidados, o que facilita a automação dos processos de projeto;

. a tecnologia CMOS é hoje um padrão industrial;

. CMOS é uma das tecnologias mais difundidas em todo o mundo, tanto nos meios produtivos como nos ambientes acadêmicos e de pesquisa /ALE 85/.

Além dos motivos genéricos citados acima, um outro conjunto de razões determinou esta escolha, baseado nas condições de contorno do método CIPREDI:

. a intenção de se implantar linhas de produção CMOS no País em um prazo curto, como mencionado no capítulo 2 deste trabalho;

. a disponibilidade de meios de projeto e fabricação de CIs, ainda que precários, no CPGCC/UFRGS, através de convênios com instituições nacionais e estrangeiras /REI 87/.

. a experiência prévia do grupo de microeletrônica do CPGCC/UFRGS em projetos com tecnologia MOS;

. a existência de empresas nacionais envolvidas com o projeto de pré-difundidos CMOS, criando possibilidade de interação, a nível de documentos e "know-how", entre empresas e o grupo interessado na proposta de um método de projeto de CIs pré-difundidos.

Os convênios existentes permitem a implementação de CIs projetados no estilo dedicado. Dois destes convênios foram responsáveis pela confecção de mais de duas dezenas de projetos elaborados pelo grupo de microeletrônica do CPGCC/UFRGS /REI 87/:

. o convênio com o CMP ("Circuit Multi-Projet") francês, através do Institut National Polytechnique de Grenoble, INPG;

. o convênio com o CMP (Circuito Multi-Projeto) brasileiro, através do Instituto de Microeletrônica do Centro Tecnológico para Informática, CTI.

O convênio com o CMP francês foi escolhido para as implementações de suporte no método CIPREDI. Esta escolha foi efetuada a partir da comparação dos convênios:

. o convênio com o CMP francês é bem mais antigo, além de oferecer "rodadas" de fabricação com maior freqüência (hoje, quatro vezes ao ano);

. a tecnologia de implementação do CMP francês é mais evoluída, permitindo o projeto em CMOS com porta de polissilício com dois níveis de interconexão em metal;

. a tecnologia do CMP francês está mais próxima dos objetivos estabelecidos para as linhas de produção CMOS nacionais;

. o CMP francês oferece, atualmente, uma facilidade para a implementação de circuitos pré-difundidos

/INP 86/, estando hoje em estudo uma maneira de explorar esta facilidade no desenvolvimento posterior do método CIPREDI.

Um último comentário sobre tecnologia é necessário. Como mencionado nesta seção, os convênios que permitem a fabricação de CIs projetados no CPGCC/UFRGS suportam o estilo dedicado. As implementações descritas no capítulo 5 foram elaboradas para permitir a simulação de um ambiente de projeto de pré-difundidos. Os circuitos experimentais foram todos concebidos dentro deste ambiente.

Quanto às matrizes, considera-se difícil, no estágio atual do método CIPREDI, estabelecer parâmetros quantitativos capazes de permitir a criação de famílias eficientes de CIs pré-difundidos. Além disso, considera-se desnecessária a existência de uma proposta de configurações de matrizes no método, dada a generalidade desejada. Assim, este requisito não será abordado neste trabalho, malgrado sua importância em um ambiente de produção seja crucial.

4.2 Modos de Especificação

O uso de um conjunto de linguagens está previsto pelo método CIPREDI. As linguagens do projeto AMPLO constituem uma maneira de especificar o comportamento e a estrutura de um pré-difundido em vários níveis de abstração. Linguagens específicas para o tratamento de descrições físicas estão previstas no método. As subseções a seguir discutem as linguagens de entrada, de saída, e a relação cliente/fornecedor concebidas para aplicação no método CIPREDI.

4.2.1 Linguagens de entrada

Dois tipos de linguagem de entrada são usadas, no momento, pelo método CIPREDI:

. as linguagens de especificação do projeto AMPLO /WAG 87c/;

. as linguagens de tratamento de descrições físicas.

O AMPLO emprega quatro linguagens para a descrição de sistemas digitais:

. NILO, para o nível lógico;
. KAPA, para o nível RT;
. LAÇO, para o nível sistema;
. REDES, para descrições puramente estruturais em qualquer dos três níveis anteriores.

Todas as linguagens possuem formas gráfica e textual, usadas alternativamente. As formas textuais são entrada de compiladores capazes de gerar estruturas manipuláveis pelo banco de dados AMPLO. As duas formas são completamente análogas, exceto pela informação gráfica em si.

NILO e KAPA são linguagens estruturais, mas as primitivas que estas empregam possuem um comportamento embutido, como por exemplo as portas lógicas em NILO e multiplexadores em KAPA. Estas linguagens são, portanto, mistas do ponto de vista do diagrama de Gajski, representando estrutura e comportamento em uma mesma descrição /WAG 87b/ /WAG 87a/.

LAÇO é uma linguagem puramente comportamental /SIL 88/, baseada em redes de Petri /PET 77/ /PET 81/ /REI

abstratas no nível algorítmico do diagrama Y.

REDES é uma linguagem puramente estrutural, que trabalha com descrições de interfaces entre módulos de projeto. Cada um destes módulos é detalhado usando uma das outras três linguagens ou usando a própria linguagem REDES, gerando assim uma hierarquia de descrições estruturais. As folhas desta hierarquia correspondem, em um projeto completo, a uma descrição em NILO, KAPA ou LAÇO. Nodos não-terminais da hierarquia são descritos apenas em REDES /WAG 87d/.

Simuladores são as formas escolhidas pelo projeto AMPLO para o exercício e a validação das descrições de um sistema digital. A descrição e implementação parcial de simuladores para o nível lógico (NILO) é uma das tarefas do presente trabalho. A seção 6.3 apresenta as atividades realizadas neste sentido.

Na figura 4.1 é mostrado o diagrama Y do projeto AMPLO, quando em funcionamento pleno.

As linguagens de descrição físicas são problemas característicos dos pré-difundidos, sendo portanto propostas dentro do método CIPREDI.

Conforme descrito no início deste capítulo, um dos objetivos do método em proposição é obter um ambiente flexível de projeto, uma forma de permitir sua adaptação a diferentes organizações.

Uma primeira maneira de criar um ambiente flexível é permitir a configuração de tecnologias e matrizes manipuladas pelas ferramentas. No método CIPREDI, as ferramentas trabalham sobre uma biblioteca de tecnologias e matrizes, que guarda parâmetros topológicos e

pelo sistema. Para a criação desta biblioteca, deverá ser usada uma linguagem de especificação de matrizes conforme será delineado na seção 7.1.

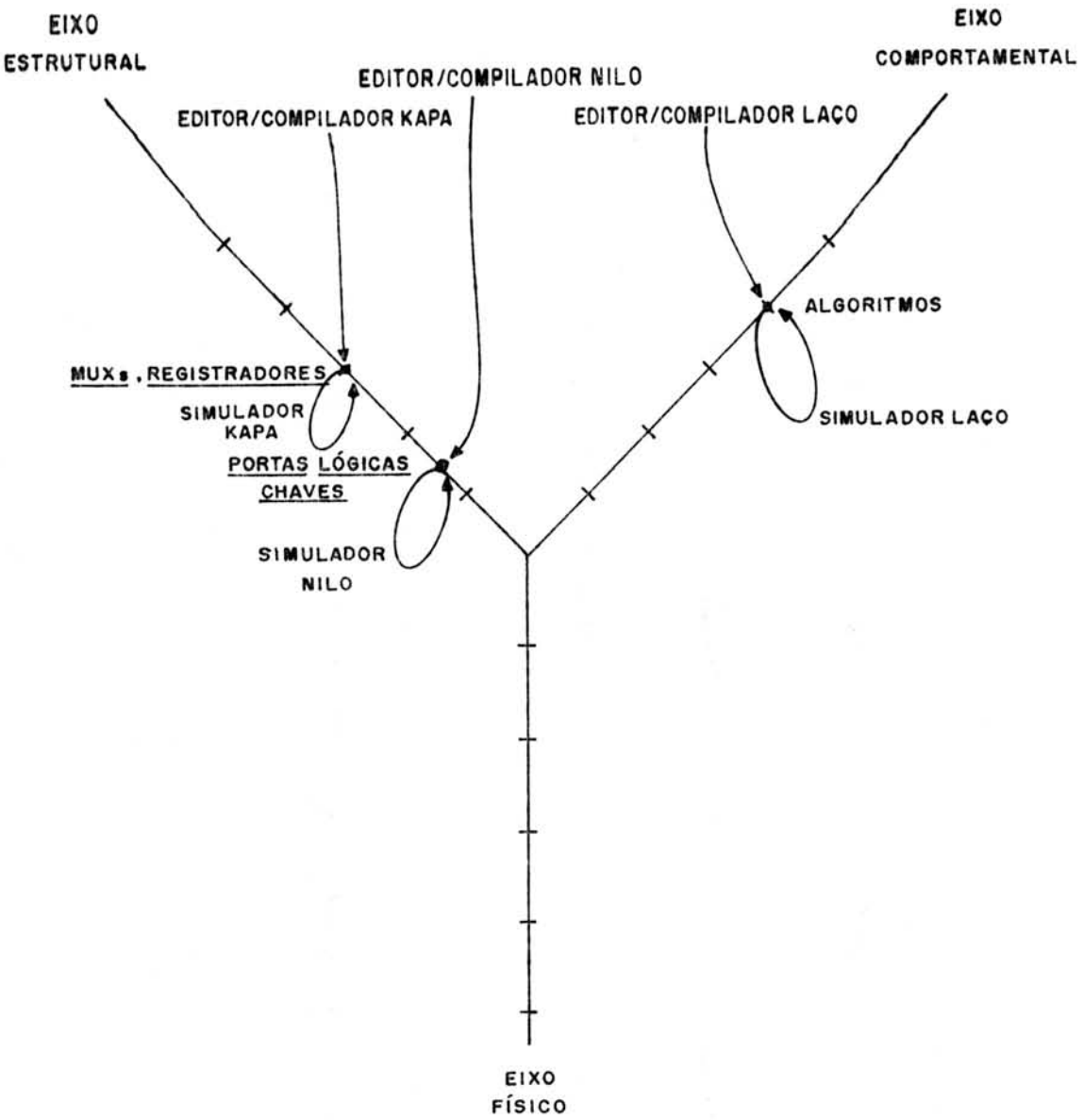


Figura 4.1 - Processo de projeto empregado pelo sistema AMPLO

Atualmente, o posicionador e traçador interativo de conexões, EDGAR, a ser descrito na seção 6.2, trabalha sobre matriz e tecnologia fixas, sendo usado para o

matrizes usadas no projeto dos circuitos de teste foram elaboradas usando a linguagem hierárquica de descrição geométrica RS /TOD 84/, um subconjunto da linguagem CIF.

Uma segunda maneira de garantir flexibilidade é permitir a criação e alteração das bibliotecas de células do sistema. Hoje, isto é feito através de uma versão alterada do programa Microeditor /JAC 86/. Esta ferramenta é um editor gráfico interativo de máscaras de CI, que gera como saída uma descrição geométrica não-hierárquica de cada célula de biblioteca, armazenada em um arquivo DOS, no formato RS. Na seção 7.1 apresenta-se uma descrição sumária das linguagens de descrição de bibliotecas previstas para implementação no método proposto.

A figura 4.2 apresenta um esquema do estado inicial das linguagens de entrada no método CIPREDI.

Figura 4.2 - Linguagens de entrada atuais do CIPREDI

4.2.2 Linguagens de saída

As linguagens de saída no método CIPREDI estão

da forma gráfica de representação, a maneira mais clara e natural de interação para o projetista de sistemas.

Em um caso, entretanto, é interessante o uso de linguagens de saída textuais. Quando é necessária a troca de informações de projeto entre equipamentos não-compatíveis, como no caso da troca de informações entre cliente e fornecedor, o meio mais usado é a interação através de códigos textuais de uso universal, como o código ASCII. A forma gráfica, apesar de mais natural, carece de uma padronização para adequar-se a estes casos.

Atualmente, no método CIPREDI, toda a troca de informações geométricas é feita através de arquivos textuais no formato RS. Na seção 7.1 discute-se o uso de outros formatos adequados para o intercâmbio de informações do nível geométrico.

4.2.3 A relação cliente/fornecedor

Dentre as formas de interação possíveis, citadas na subseção 3.3.1 deste trabalho, o método CIPREDI opta pela última, ou seja, o fornecimento de descrições de máscaras para o fabricante.

Vários fatores colaboraram para esta decisão:

. dificuldade de interação com fundições estrangeiras no cenário nacional, devido à localização geográfica do País;

. desejo de prover um máximo de auto-suficiência em projeto de pré-difundidos para os usuários do método;

. intenção de garantir o máximo de controle sobre os parâmetros de concepção por parte do projetista;

. desejo de minimizar o tempo do ciclo de

presentes quando a interação cliente/fornecedor é freqüente.

As próprias ferramentas de "software" do método, apresentadas no capítulo 6, refletem esta escolha.

O ponto documentos formais, embora importante na definição da interface entre cliente e fornecedor, não será abordado neste trabalho, visto ser sempre estabelecido pelas próprias fundições.

4.3 Processos de Projeto

Relembrando o exposto na subseção 3.1.1, três requisitos compõem este aspecto de um método de concepção de CIs pré-difundidos:

. tomadas de decisão de projeto;
. flexibilidade do ambiente de projeto;
. níveis de abstração.

Com respeito ao primeiro item, decidiu-se, no método CIPREDI, pela abordagem CAD, pelo menos nos passos iniciais de sua elaboração. Esta opção deveu-se à imaturidade relativa da comunidade envolvida com projeto de CIs no âmbito nacional. É necessária uma investigação da problemática de projeto para identificar onde e como aplicar técnicas avançadas como sistemas especialistas e síntese automática.

A flexibilidade do ambiente é a máxima possível, devido à natureza do meio acadêmico onde CIPREDI está em elaboração. A adaptação a ambientes com características diversas será considerada em passos posteriores do método (ver capítulo 7), tendo em vista o eventual repasse de idéias e ferramentas para centros interessados.

O requisito níveis de abstração está subordinado, no método CIPREDI, à escolha de um ambiente flexível como alvo. As abordagens ascendente e descendente são permitidas para a concepção. Nos níveis abstratos inferiores, a abordagem aconselhada é a ascendente, pois esta proporciona a manipulação de pequenas porções de um projeto de forma independente. Células de biblioteca são construídas segundo esta abordagem. Níveis superiores são tratados de maneira estruturada. Neste caso, a abordagem descendente é a mais aconselhada. Esta última permite garantir boas características globais a um projeto, como por exemplo a testabilidade e a modularidade /WAG 86/.

4.4 Infraestrutura de Projeto

Nesta seção será delineada a estrutura das ferramentas de projeto do método CIPREDI. A subseção 4.4.1 trata dos meios físicos de implementação do método, e a subseção 4.4.2 discute linguagens de implementação, sistemas operacionais, ferramentas de PAC e o algoritmo de aplicação destas.

4.4.1 Ferramentas de "hardware"

O método CIPREDI visa o desenvolvimento de um ambiente de concepção adequado à realidade brasileira. Esta premissa dita as escolhas realizadas para o "hardware" de imlementação. Dentre as opções citadas na subseção 3.1.2, o método sugere:

. uso de estações de trabalho de forma autônoma para a execução de todas as fases de projeto de pré-difundidos;

nacional;

. uso de configurações padronizadas de "hardware", disponíveis nos ambientes acadêmico e de produção, visando incrementar a capacidade de adaptação do sistema de PAC do CIPREDI.

Considerando os pontos citados, aliado às necessidades computacionais mínimas de um sistema de projeto para "gate arrays", decidiu-se iniciar a implementação do método sobre um microcomputador pessoal compatível com o PC-AT da IBM. A configuração inicial desta estação de trabalho é:

. 1 Mbyte de memória principal;
. vídeo gráfico colorido, de alta resolução;
. placa gráfica compatível com o padrão EGA;
. impressora gráfica;
. dispositivo de apontamento do tipo "mouse";
. mesa digitalizadora.

A memória é dimensionada para permitir a realização de tarefas como simulação lógica global e simulação elétrica de módulos da biblioteca. O vídeo gráfico é colorido e de alta resolução, para permitir ergonomia na geração interativa de máscaras do CI. Dispositivos de entrada gráfica acrescentam velocidade e facilidade de interação com o sistema de PAC.

Finalmente, o método prevê a integração desta estação de trabalho monousuário com os ambientes de projeto AMPLO e LCCI /CUR 86/. Ambos, AMPLO e LCCI, visam o uso de uma estação de trabalho nacional, multiusuário, multitarefa, usando um processador com palavra de 32 bits. Ambos projetos objetivam o acoplamento de estações gráficas monousuários a esta configuração. Uma descrição da estrutura de "hardware" típica destes projetos pode ser encontrada em /JAC 88a/. A estação multiusuário está

baseada em um sistema operacional compatível com o UNIX.

Já estão disponíveis, no mercado nacional, supermicrocomputadores com as características exigidas pelos projetos citados, sendo estes candidatos naturais para a implementação dos sistemas de PAC referidos nesta subseção.

Inicialmente o objetivo do método é a integração com um computador hospedeiro modelo ED680, um supermicro da EDISA, conforme descrito em /CUR 86/.

4.4.2 Ferramentas de "software"

Tendo como base a escolha das ferramentas de "hardware" descritas na subseção anterior, o método CIPREDI propõe um ambiente de desenvolvimento de "software" com as seguintes características:

. sistema operacional compatível com o MS-DOS /MIC 86a/, o mais difundido para uso em microcomputadores compatíveis com o PC IBM;

. linguagem de programação C /MIC 86b/, para garantir máxima portabilidade ao sistema;

Estas escolhas procuram atender ao objetivo b) do método CIPREDI, conforme descrito no início deste capítulo. Quanto à linguagem de programação, a escolha é óbvia, sendo C a mais usada e a mais indicada em qualquer aplicação que procure uma maximização de eficiência e portabilidade. Atualmente, as ferramentas CIPREDI são implementadas com uso do compilador C da Microsoft Inc, versão 4.0.

Quanto ao sistema operacional (SO), a escolha não foi tão simples. Os sistemas compatíveis com o DOS possuem alguns inconvenientes sérios para a implantação de

sistemas com porte. O sistema MS-DOS usa, no máximo, 640 Kbytes de memória principal com endereçamento direto. Qualquer aplicação que necessite uma quantidade maior de memória deve optar por uma técnica alternativa, como acesso a meios magnéticos de armazenamento ou emprego de "discos virtuais" em memória principal inacessível de forma direta.

A primeira técnica provoca uma redução significativa do desempenho de programas. A segunda exige uma memória mais extensa, o que não é incomum em microcomputadores compatíveis com o PC-AT da IBM. Ambas técnicas, contudo, geram perdas de desempenho, devido ao necessário envolvimento com a gerência explícita de memória por parte dos programas.

Existem SOs disponíveis para estas máquinas, alguns compatíveis com o padrão UNIX, o sistema mais usado para aplicações de PAC em Microeletrônica. Contudo, o uso destes sistemas em PCs têm sido bastante limitado, principalmente por motivos históricos. A comunidade de pesquisa inicialmente não julgava que a linha PC-IBM fosse capaz de suportar sistemas de PAC para o projeto de CIs, o que atrasou a chegada ao mercado de SOs com as características desejáveis à implementação deste tipo de sistemas. A disponibilidade destes SOs se deu em um momento onde já se encontrava, no mercado, um conjunto extenso de ferramentas de PAC construídas sobre o MS-DOS ou sobre soluções particulares, empregando "hardware" não padrão.

O método CIPREDI optou pelo MS-DOS para atender aos objetivos estabelecidos dentro das condições de contorno de implementação, e como forma de atender à padronização do sistema de PAC resultante.

Cuidados de implementação têm sido tomados para facilitar uma possível migração de todo ou parte do sistema para um SO UNIX, como por exemplo o uso de rotinas de

biblioteca da linguagem C disponíveis em ambos sistemas, MS-DOS e UNIX, e estruturação de programas para permitir seu uso sob a forma de processos concorrentes.

O mapeamento de descrições abstratas (comportamentais/estruturais) para descrições físicas é o passo mais crítico do projeto de CIs pré-difundidos. É necessário um forte acoplamento entre estas descrições, para garantir a precisão dos modelos abstratos em relação ao comportamento do circuito final. O método CIPREDI, neste passo inicial, dedica-se às ferramentas responsáveis por este mapeamento. O fato do método procurar integrar ambientes desenvolvidos de forma independente dificulta esta tarefa.

Neste trabalho, optou-se pela implementação de duas ferramentas de "software":

. NILO, um simulador lógico, interativo, orientado a eventos e dirigido por tabelas, definido dentro do escopo do projeto AMPLO;

. EDGAR, um editor gráfico interativo de posicionamento e traçado de conexões entre células da biblioteca de pré-difundidos, definido no escopo do projeto LCCI.

A primeira ferramenta, dentro do modelo tradicional de projeto de pré-difundidos, é usada para validar esquemas lógicos. Este é o último passo antes do início das tarefas de geração de máscaras. A segunda ferramenta é usada para a própria geração das máscaras de personalização do pré-difundido, a partir do esquema lógico validado, das matrizes e das células das bibliotecas de projeto físico.

Pretende-se, desta maneira, estabelecer o núcleo do método. Este núcleo deve expandir-se iterativamente, a

partir da experiência obtida com as ferramentas iniciais e trabalhos posteriores de implementação.

No capítulo 6, as ferramentas citadas são apresentadas em detalhe, junto com uma proposta de integração entre elas.

É claro que um ambiente de produção exige ferramentas adicionais às deste núcleo. A automação das tarefas de projeto físico e a preocupação com a testabilidade são imprescindíveis para que os CIs pré-difundidos mostrem-se economicamente viáveis. Estes aspectos devem ser os próximos abordados no desenvolvimento do método CIPREDI.

No CIPREDI, uma abordagem inicial do problema do teste é obtida com uso da ferramenta de simulação lógica, descrita na seção 6.3 deste trabalho. O simulador NILO incorpora características que permitem a simulação do tipo "grudado-em" ("stuck-at"). Estas são as falhas mais facilmente modeláveis em simuladores que trabalham com descrições a nível de portas lógicas.

Algumas idéias para a evolução posterior do método são apresentadas no capítulo 7 deste trabalho. Os passos seguintes do CIPREDI devem discutir e guiar o trabalho de implementação dentro das diretrizes básicas especificadas aqui.

Na figura 4.3. mostra-se o diagrama Y do estado atual do método CIPREDI.

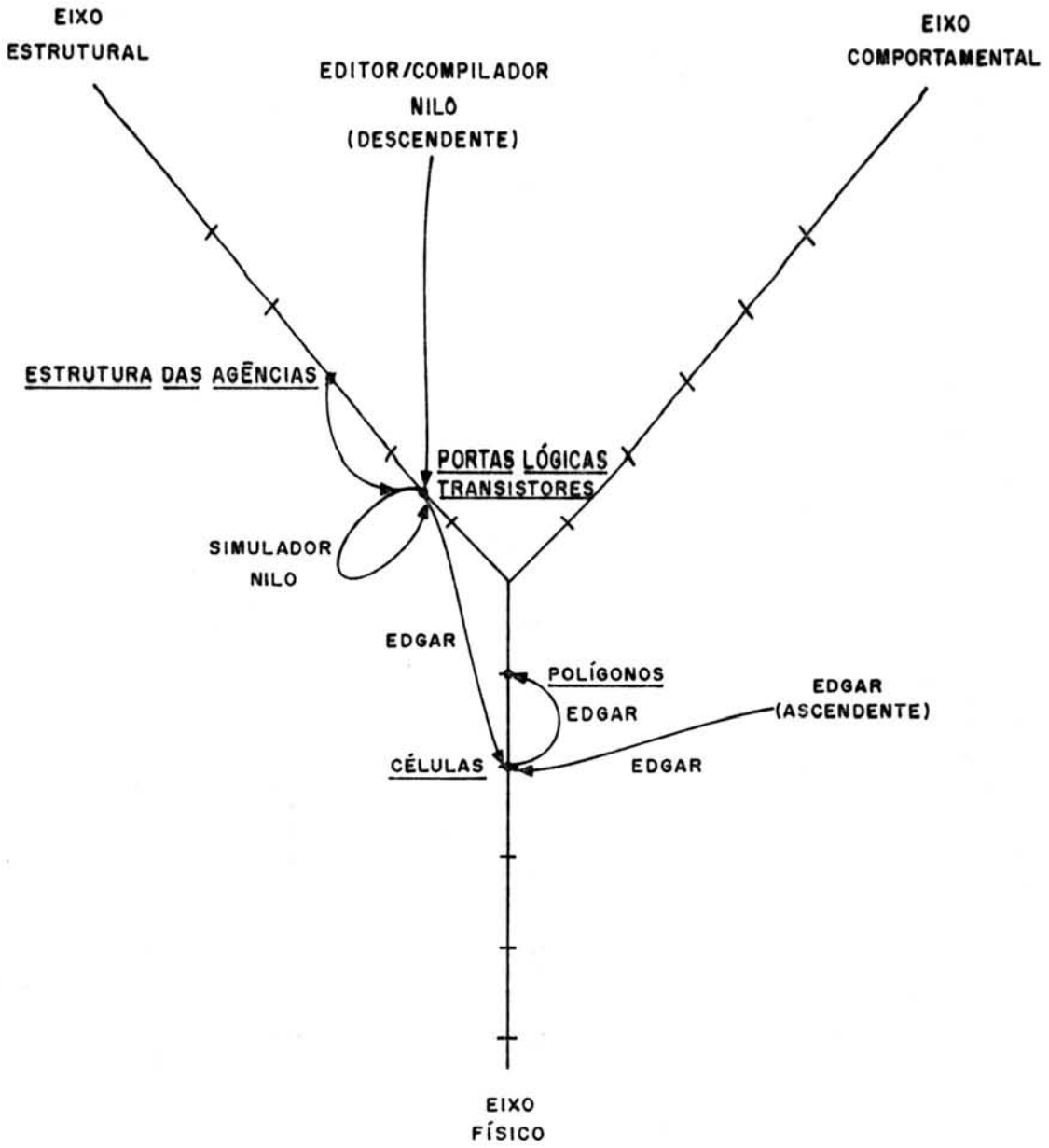


Figura 4.3 - Processo de projeto atual do método CIPREDI

5 IMPLEMENTAÇÃO DE SUPORTE À CONCEPÇÃO

Dentre os meios empregados para dar suporte à concepção de CIs pré-difundidos, dois são manipulados intensamente pelo projetista:

. a(s) célula(s) de base usada(s) no núcleo do CI;

. a biblioteca de células de personalização.

O primeiro meio está relacionado de forma direta com a facilidade de implementação das diferentes funções sobre a matriz do pré-difundido. O segundo meio é responsável pelo mapeamento entre os níveis de abstração estruturais e comportamentais e o nível de máscaras. A flexibilidade e a extensão da biblioteca de células ditam a unicidade e a confiabilidade de tal mapeamento.

Devido ao papel crítico desempenhado por estes meios, o método CIPREDI optou pela sua implementação e estudo. O objetivo desta decisão foi o de adquirir subsídios para a implementação das ferramentas do método. Não houve preocupação em obter uma célula de base otimizada ou uma biblioteca de células completa, visto serem estas preocupações do fabricante e não do projetista de pré-difundidos.

As seções 5.1 e 5.2 apresentam esta implementação. A seção 5.3 mostra os circuitos de teste, confeccionados com a célula de base descrita na seção 5.1 e com a biblioteca descrita na seção 5.2.

5.1 GME - Uma Nova Célula de Base

Antes de apresentar a célula de base desenvolvida dentro do método CIPREDI, apresenta-se algumas razões para

esta atividade de implementação:

. não existiu, durante o desenvolvimento inicial do método, uma biblioteca comercial disponível para uso;

. calcar todo um método de projeto de pré-difundidos sobre uma biblioteca comercial não é adequado para o meio acadêmico;

. no contexto nacional, a inexperiência com projeto de CIs torna necessário o envolvimento do método com meios de suporte ao projeto, como células de base, matrizes e biblioteca de células;

Após o estudo qualitativo e quantitativo de células de base descritas na literatura, conforme discutido em /CAL 87a/ e na seção 3.2.2 desta dissertação, foi proposta uma nova célula de base para pré-difundidos.

Nesta célula, foi colocada ênfase na obtenção de uma técnica de projeto que refletisse o estado da arte em pré-difundidos. Esta escolha implica altos custos de projeto e fabricação, não adequados à realidade do País. Contudo, o enfoque deste trabalho é o ciclo de projeto. As células de base do estado da arte apresentam um grau de dificuldade maior para a geração das máscaras. Este grau de dificuldade conduz a um acréscimo de flexibilidade nas ferramentas, tornando-as mais genéricas. Como um dos objetivos do método CIPREDI é a flexibilidade e a adaptabilidade do ambiente, considera-se justificada a opção por uma célula de base com as características mencionadas.

A tecnologia de implementação da célula de base foi CMOS com dois níveis de metalização, canal de polissilício de largura mínima de 2 microns. Esta é a tecnologia do CMP francês mencionada na subseção 4.1.1.

Esta tecnologia é suportada por 13 fabricantes

que participam das rodadas de fabricação deste CMP. As regras de projeto são escaláveis, com o parâmetro lambda valendo atualmente 1 micron /INP 86/. As regras de projeto do CMP são uma espécie de mínimo múltiplo comum entre as regras dos 13 fabricantes, sendo, portanto, bastante desotimizadas e bastante genéricas.

Para a simulação de um ambiente de implementação de CIs pré-difundidos foram feitas as seguintes escolhas:

. camadas fixas - imutáveis na matriz do pré-difundido - poço-p, poço-n, difusão p, difusão n, polissilício;

. camadas programáveis - projetista não escolhe posição, escolhe a existência ou não de determinada geometria - contato metal 1/difusão, contato metal 1/polissilício, via metal 1/metal 2;

. camadas personalizáveis - aquelas que o projetista tem liberdade de especificar a geometria - metal 1 e metal 2.

A personalização da matriz se faz através de quatro níveis de máscaras. Esta escolha se deve à maior flexibilidade de projeto associada a um número maior de máscaras, em relação às abordagens tradicionais de um nível de personalização. A escolha da tecnologia conduziu igualmente a esta abordagem, visto que as regras de projeto usadas implicam um desperdício significativo de área, caso apenas a última camada de metalização seja personalizável (linhas de metal 2 são 66% maiores que linhas de metal 1).

O "layout" das possíveis células de base GME é mostrado na figura 5.1.

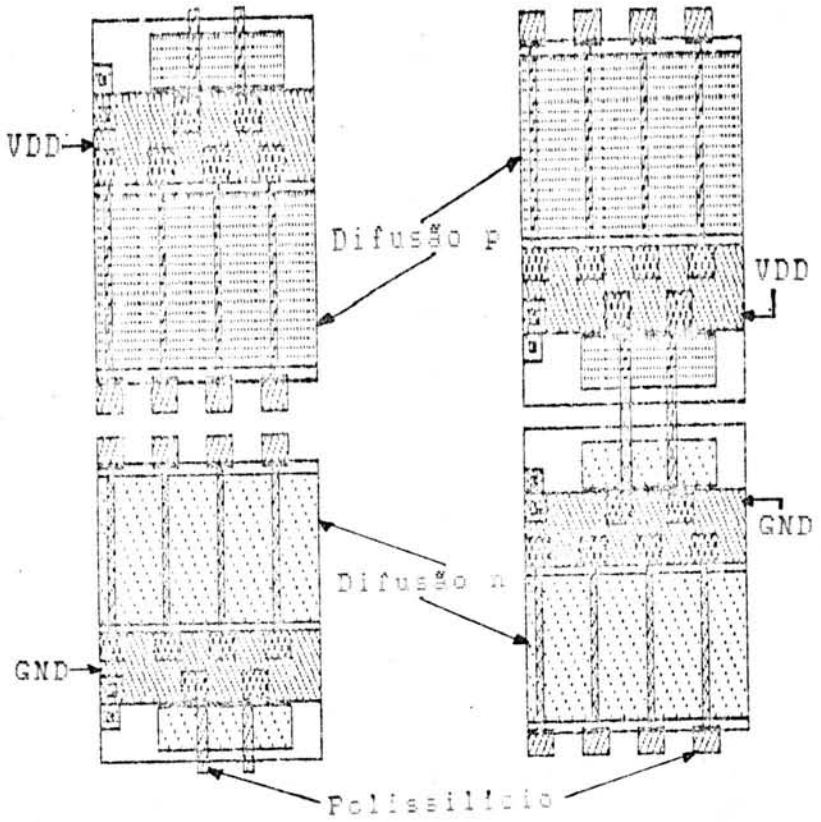


Figura 5.1 - Formatos utilizáveis para a célula de base GME

A célula GME possui as seguintes características topológicas:

. 8 canais de transparência verticais usados preferencialmente pelo primeiro nível de metal;

. 11 canais de transparência horizontais, usados preferencialmente pelo segundo nível de metal;

. 8 transistores grandes, 4 tipo n, 4 tipo p, com portas independentes;

. 2 pares p-n de transistores pequenos com porta compartilhada;

. alimentação em metal 2, na horizontal;

. os transistores grandes usam a técnica de isolamento por porta de polissilício ("gate isolation") /OHK 82/ ;

(abordagem "sea of gates").

A escolha de níveis de metalização percorrendo direções perpendiculares foi extraída das técnicas usadas em geração de conexões para placas de circuito impresso dupla face. A escolha das direções preferenciais para o primeiro e o segundo nível de metal foi devida às regras de projeto distintas para estas camadas na tecnologia usada. A espessura das linhas de metal 1 é menor, e sua capacidade de conexão é muito maior que as linhas de metal 2. A alimentação é feita em metal 2, na horizontal.

Na matriz de pré-difundido as células de base são posicionadas em faixas horizontais sucessivas, sem espelhamento ou rotação da célula original. As células de base (a) e (b) são na realidade a mesma. A diferença de representação geométrica serve para ilustrar as possibilidades da célula, em termos de geração de lógica.

A técnica de projeto mais aconselhada para a célula de base GME é a seguinte:

. usar a parte central das células do tipo (b) (transistores grandes) para a implementação de lógica combinacional ou lógica seqüencial esparsa;

. a parte central das células do tipo (a) (transistores pequenos com porta compartilhada) para a implementação de pontos de memória do tipo par de inversores realimentados, usando os transistores grandes para controle de endereçamento, escrita e leitura.

A figura 5.2 mostra um exemplo de lógica combinacional gerada sobre os transistores grandes.

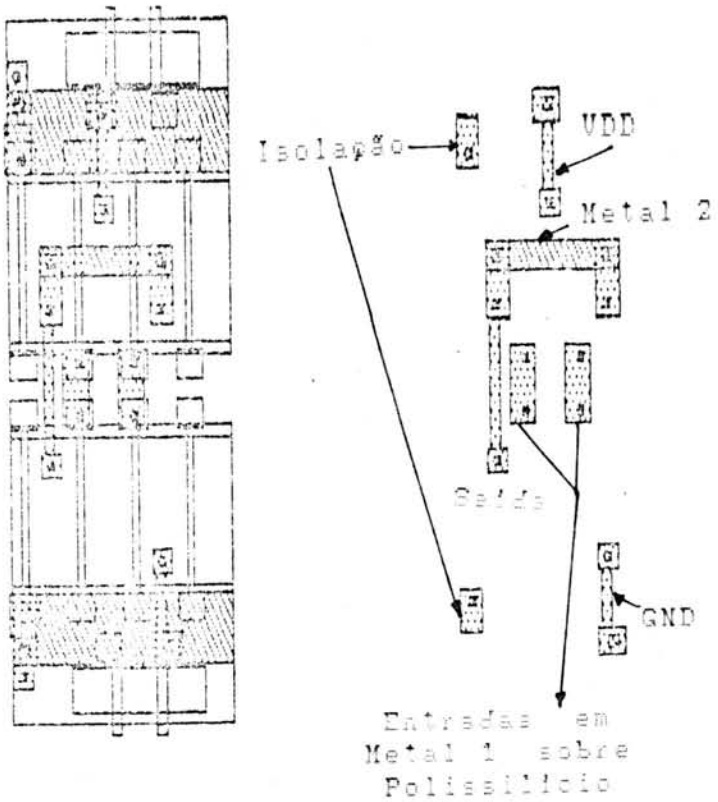


Figura 5.2 - Porta NAO-E implementada sobre transistores grandes da célula GME

A figura 5.3 mostra a implementação de uma célula de memória de 8 transistores.

A elaboração de circuitos com a célula de base GME levou à densidade de 78 bits/mm2 para circuitos de memória. Estes valores são considerados razoáveis, levando-se em conta as regras de projeto empregadas.

A conjugação de dois tamanhos distintos de transistores mostra-se um bom compromisso entre fexibilidade de interconexão e ocupação otimizada de área. A técnica de isolamento por porta de polissilício incrementa a área útil do CI de forma perceptível /NOI 85/.

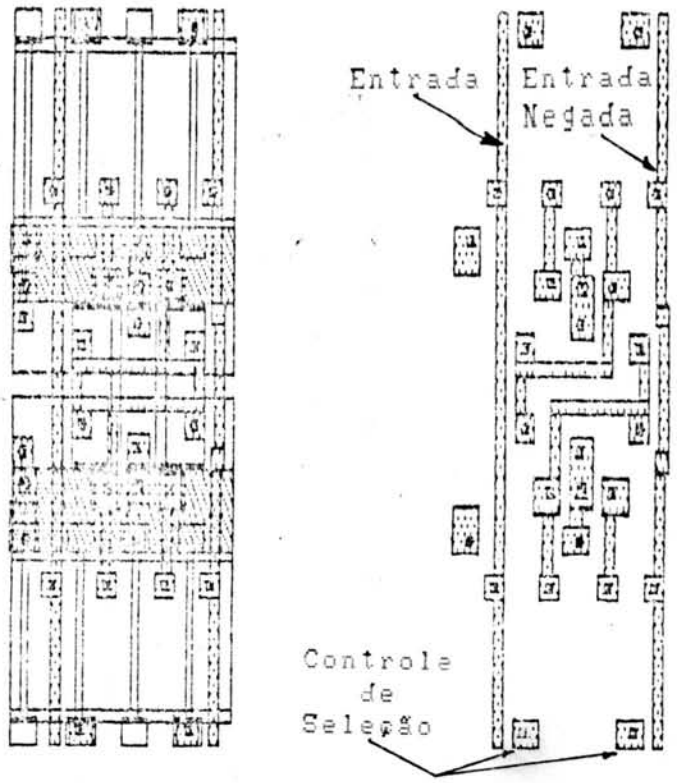


Figura 5.3 - Bit de memória RAM construído com oito transistores

5.2 Implementação e Caracterização de uma Biblioteca de Células Mínima

A biblioteca de células proposta no método CIPREDI possui atualmente 30 elementos. Esta biblioteca foi criada com a finalidade de permitir a prática de projeto de CIs pré-difundidos usando uma célula de base GME. Os elementos da biblioteca implementam primitivas do nível lógico e do nível bloco funcional do diagrama Y. Uma lista das células de biblioteca atualmente disponíveis é mostrada na tabela 5.1.

A utilização de uma célula base com isolamento por porta de polissilício implica algumas técnicas na geração de máscaras com a biblioteca de células. Internamente, cada célula de biblioteca tem a implementação facilitada pela continuidade das regiões de difusão na

horizontal. Portas lógicas com número arbitrário de entradas podem ser criadas, sem necessidade de pontes em metal conectando células de base discretas.

Tabela 5.1 - Células de biblioteca disponíveis no CIPREDI

NOME	FUNÇÃO
A2OI.GM6	DUPLO E DE 2 ENTRADAS - OU NEGADO
A3OI.GM6	TRIPLO E DE 2 ENTRADAS - OU NEGADO
AND2.GM6	PORTA E DE 2 ENTRADAS
AND3.GM6	PORTA E DE 3 ENTRADAS
BITMEM.GM5	"BIT" DE MEMÓRIA - CÉLULA DE 8 TRANSISTORES
BUF. GM6	"BUFFER" NÃO INVERSOR
CLA. GM6	SOMADOR DO TIPO "CARRY LOOK AHEAD" DE 1 BIT
CLG1.GM6	PRIMEIRO ESTÁGIO DE 1 GERADOR DE "CARRY LOOK AHEAD"
CLG2. GM6	SEGUNDO ESTÁGIO DE 1 GERADOR DE "CARRY LOOK AHEAD"
CLG3. GM6	TERCEIRO ESTÁGIO DE 1 GERADOR DE "CARRY LOOK AHEAD"
DECH1-2. GM6	DECODIFICADOR COM HABILITAÇÃO / DEMUX 1:2
FAD.GM6	SOMADOR COMPLETO DE 1 BIT
FFD.GM6	"FLIP-FLOP" D SENSÍVEL À BORDA, MESTRE ESCRAVO
FFDSR.GM6	"FLIP-FLOP" D SENSÍVEL À BORDA COM SET E RESET ASSÍNCRONOS
HAD.GM6	MEIO SOMADOR DE UM BIT
INV.GM6	INVERSOR MÍNIMO
LTDE.GM6	"LATCH" D DE 1 BIT
MUX2-1.GM6	MULTIPLEXADOR 2:1
NAN 2.GM6	PORTA NÃO-E DE 2 ENTRADAS
NAN 3.GM6	PORTA NÃO-E DE 3 ENTRADAS
NAN 4.GM6	PORTA NÃO-E DE 4 ENTRADAS
NOR2.GM6	PORTA NÃO-OU DE 2 ENTRADAS
NOR3.GM6	PORTA NÃO-OU DE 3 ENTRADAS
NOR4.GM6	PORTA NÃO-OU DE 4 ENTRADAS
NXO2.GM6	PORTA EQUIVALÊNCIA DE 2 ENTRADAS
O2AI.GM6	DUPLO OU DE 2 ENTRADAS - E NEGADO
OR2.GM6	PORTA OU DE 2 ENTRADAS
OR3.GM6	PORTA OU DE 3 ENTRADAS
TG.GM6	PORTA DE TRANSMISSÃO
XOR2.GM6	PORTA OU-EXCLUSIVO DE 2 ENTRADAS

Por outro lado, cada célula, ou cada parte de uma célula sem relação com partes adjacentes, deve ser devidamente isolada. Esta isolação é obtida através de corte de transistores. Cada célula termina em uma região de difusão limitada por uma porta de transistor. Este transistor é cortado conectando-se sua porta ao terminal de alimentação adequado: VDD para transistores tipo p e VSS

para transistores tipo n. Cada uma das células da biblioteca proposta possui isolamento à direita, mas não no limite esquerdo. Esta técnica facilita o posicionamento automático de células numa mesma linha de difusão. Os terminais mais externos à esquerda podem estar sem isolamento sem que isto cause problemas. Caso haja necessidade explícita de isolamento, células especiais, não mencionadas na tabela 1, são usadas. Estas células não têm função lógica, apenas função topológica.

O posicionamento de células é feito por faixas horizontais, colocando células da direita para a esquerda, com os isolamentos à esquerda garantido automaticamente a separação. Caso haja necessidade de se deixar espaços vazios entre células, isto é feito isolando-se explicitamente o lado direito antes do espaço. Estes espaços surgem, tanto no roteamento manual como no automatizado, a partir da necessidade de efetuar conexões entre faixas horizontais.

A célula de base é composta por canais de transparência horizontais e verticais pré-definidos. Cada célula de biblioteca usa um subconjunto destes canais para a implantação do seu comportamento. Os canais restantes são reservados para o traçado global de conexões.

No Anexo 2 é apresentada uma proposta de documentação para as células da biblioteca. Está em confecção um manual de utilização contendo informações organizadas segundo esta proposta para cada uma das células. A proposta é uma extensão daquela sugerida por Carro /CAR 87/ para a documentação de uma biblioteca de células de circuitos do tipo "standard cell" (pós-difundidos). Os itens adicionais compreendem características específicas encontradas nos pré-difundidos e deficiências encontradas na sugestão inicial.

As células da biblioteca foram caraterizadas do ponto de vista elétrico. Esta caracterização permite o cálculo simplificado dos parâmetros relevantes, de cada célula, para a simulação lógica. Cada célula possui valores especificados para os seguintes parâmetos:

. atraso intrínseco;
. atraso devido à carga capacitiva na saída;
. capacitância de entrada.

O primeiro parâmetro é especificado para uma carga capacitiva unitária na saída, equivalente a um "fan-out" de um inversor mínimo na tecnologia empregada. O segundo fornece uma maneira de estimar o atraso de propagação de uma célula dentro de um circuito , levando em consideração o "fan-out" total na saída e o atraso intrínseco. O último parâmetro é o que pemite o cálculo da capacitância associada a um nodo do circuito implementado. Esta capacitância serve para o cálculo de atraso total associado a cada componente com saída conectada neste modo.

Os dois primeiros parâmetros são especificados para cada saída e o último é especificado para cada uma das entradas.

5.3 Circuitos de Teste

Nesta seção são apresentadas os dois circuitos de teste implementados com o uso da célula de base GME. Os objetivos da confecção de tais circuitos foram:

a) caracterização elétrica da célula GME;
b) avaliação da utilidade da biblioteca de células implementadas;
c) avaliação das capacidades de transparência e facilidade de conexão entre células de biblioteca;

d) levantamentos de necessidades para ferramentas de geração automática de máscaras;

e) obtenção de dados para a especificação de matrizes de pré-difundidos para as próximas implementações.

O primeiro circuito, um oscilador em anel de 339 estágios, serviu para cumprir os dois primeiros objetivos citados acima. Sendo um circuito topologicamente muito simples e de reduzido número de sinais externos, não é possível considerá-lo em relação aos demais objetivos.

A caracterização elétrica forneceu uma estimativa de freqüência máxima de chaveamento de 50 MHZ para inversores da biblioteca de células, o que se considera razoável, dada a desotimização de espaço e das regras de projeto usadas.

Quanto ao segundo objetivo, detectou-se algumas deficiências da biblioteca. A topologia da célula de base é complexa. Esta complexidade faz com que seja difícil trabalhar apenas com uma topologia para cada célula lógica da biblioteca, sem introduzir um desperdício de área significativo.

Como exemplo deste inconveniente cita-se o oscilador em anel. Este foi projetado com dois tipos de inversores. Esta medida ocasionou uma economia de 50% em área em relação a uma abordagem onde se utilizasse apenas o inversor disponível na biblioteca de células. A característica topológica que possibilitou esta economia foi o fato de se poder implementar dois inversores em uma única célula de base. Caso apenas o inversor disponível na biblioteca fosse usado, seria necessário uma célula de base por inversor.

A figura 5.4 ilustra as duas abordagens. Nota-se nela figura, que os dois inversores da figura 5.4 (b)

possuem topologias de metalização diferentes, devido ao fato de estarem localizados em posições relativas distintas na célula de base. Os problemas surgem devido às diferentes maneiras de alimentar e isolar cada transistor presente na célula de base.

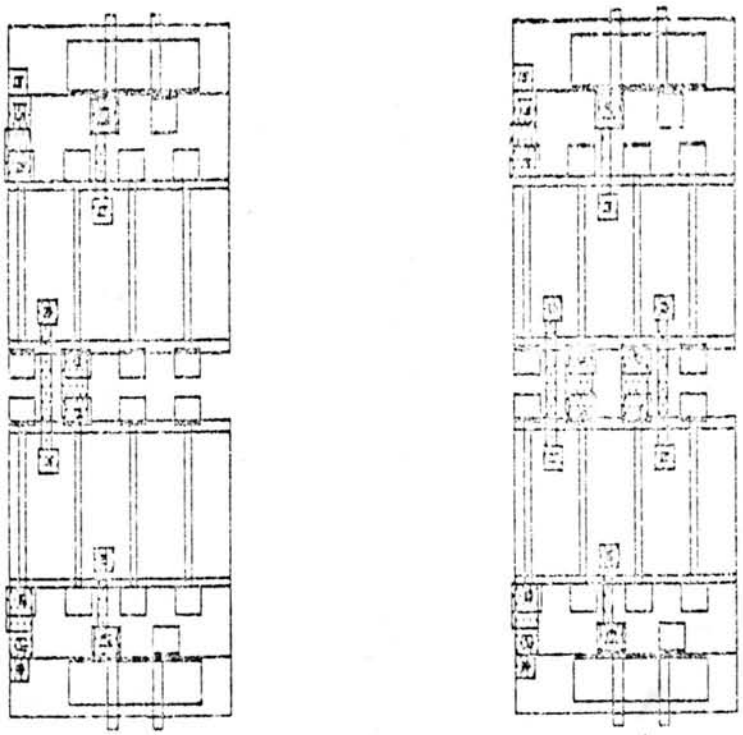


Figura 5.4 - Duas formas de implementar inversores sobre a célula GME

Para que a automatização dos passos de geração de máscaras obtenha resultados satisfatórios usando a célula GME, alguns cuidados devem ser tomados. Estes cuidados foram inferidos a partir dos problemas citados acima, sendo sintetizados em duas alternativas:

. oferecimento de várias versões de cada célula da biblioteca, levando em conta os aspectos topológicos envolvidos (para a célula de base GME, 4 versões de cada célula seriam necessárias);

. acréscimo de inteligência nas ferramentas de

geração de máscaras, para que as diferenças de posições relativas da célula sobre a matriz sejam automaticamente contornadas.

A primeira alternativa é a mais viável a curto prazo. Obviamente ela implica em redundância na biblioteca de células, mas o custo desta redundância é considerado inferior ao custo da segunda alternativa. Esta última resulta, contudo, numa solução mais elegante para a geração de máscaras com baixo desperdício de área.

Este circuito foi fabricado e testado, mas os protótipos apresentaram problemas de curto circuito, não chegando a funcionar. Devido à mudança ocorrida nas regras de projeto do CMP francês, ele deve ter o "layout" alterado para adaptar-se às novas regras e ser novamente enviado para a fabricação.

Atualmente, entretanto, o envio de circuitos está impossibilitado pela inexistência de um meio confiável de obtenção de uma fita geradora de padrões.

O segundo circuito foi um multiplicador combinacional de números inteiros binários positivos de 4 bits. Multiplicadores combinacionais /LER 84/ /DHU 84/ são circuitos de alto desempenho usados em várias aplicações modernas, tais como processamento digital de sinais em tempo real /WAS 78/. O fato do circuito ser totalmente combinacional reduz o aspecto de problemas de implementação a uma classe única, além de tornar a geração de máscars uma tarefa uniforme. A complexidade do circuito é reduzida, se comparado aos citados na bibliografia, como por exemplo em /LER 84/. Este dimensionamento foi escolhido devido à precariedade dos recursos de PAC disponíveis, e por considerar a simetria do problema independente, até certo ponto, do número de bits. A geração manual das máscaras para o multiplicador 4X4 demonstrou ser uma tarefa de

complexidade razoável.

O porte do circuito permitiu a consideração dos objetivos c), d) e e) citados no início desta seção. Os dados do circuito são mostrados na tabela 5.2.

Tabela 5.2 - Características do multiplicador

PARÂMETRO	VALOR
ÁREA	2.360 X 2.520 = 59.472 MICRONS^2
PRODUTOS PARCIAIS	16 PORTAS E DE 2 ENTRADAS
CÉLULAS DE BASE	6 LINHAS DE 15 CÉLULAS CADA = 90 CÉLULAS
ENCAPSULAMENTO	18 PINOS
ENTRADAS	2 VETORES DE 4 BITS ; 2 DE ALIMENTAÇÃO
SAÍDA	VETOR DE 8 BITS
TÉCNICA DE IMPLEMENTAÇÃO	SOMADORES DO TIPO "CARRY-SAVE"
TECNOLOGIA	CMOS, 2 MICRONS, 2 NÍVEIS DE METALIZAÇÃO, PORTA DE POLISSILÍCIO
TIPO DE CÉLULA DE BASE	GME

A figura 5.5 mostra a organização topológica do multiplicador, com os produtos parciais envolvidos em cada estágio representados por um par letra-número.

A figura 5.6 mostra o "layout" final do multiplicador, como enviado para a primeira fabricação.

Com respeito ao objetivo c), avaliação das capacidades de transparência e facilidade de conexão entre células de biblioteca, o multiplicador forneceu resultados interessantes. A célula de base GME permite um bom aproveitamento das transparências com geração manual de conexões. A automação do "layout" deve ser baseada no tratamento de transparências a cada célula de base, para permitir o traçado eficiente de conexões, e não a cada célula de biblioteca. Isto se deve à grande ocupação de

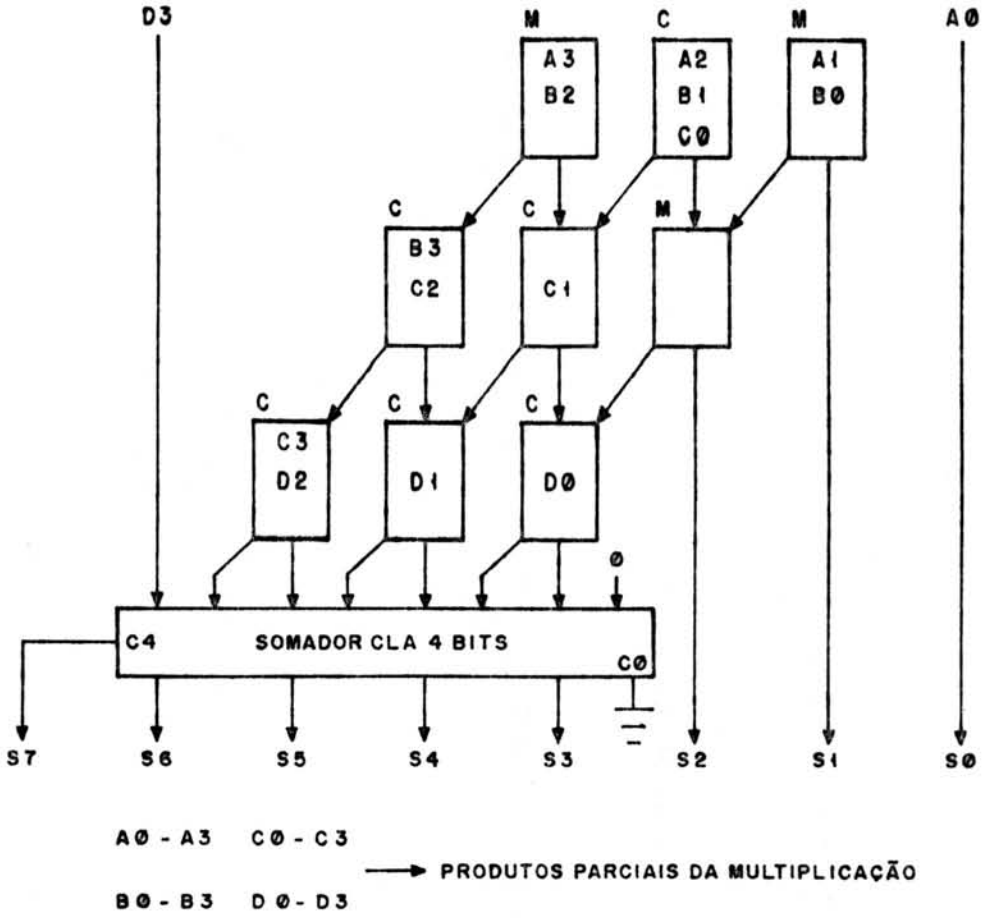
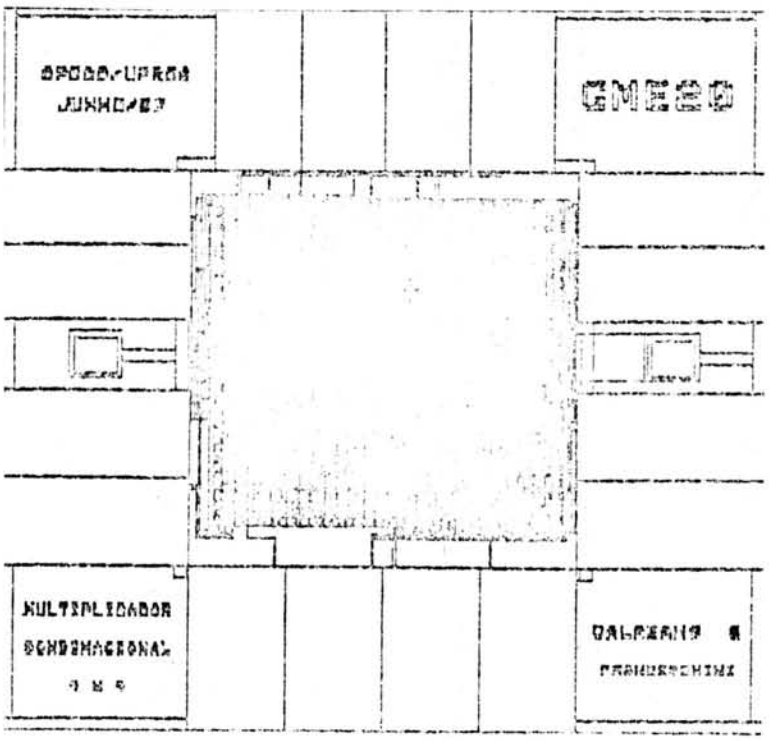


Figura 5.5 - Topologia do multiplicador



O objetivo d), levantamento de necessidades para ferramentas de geração automática de máscaras, também foi alcançado. A célula GME permite uma taxa de ocupação alta do núcleo da matriz (acima de 80%, tipicamente). Contudo, a tarefa de posicionamento de células não deve ser subestimada. No projeto do multiplicador, um posicionamento inicial pouco elaborado ocasionou o gasto de 8 homens-hora para o traçado das últimas cinco conexões do "layout". O posicionamento inicial não foi refeito, devido à urgência do prazo de envio à fabricação.

O último objetivo, relacionado à matriz do pré-difundido, pôde também ser alcançado. Deve existir um espaço razoável, acima de 12 canais de interconexão para o caso do multiplicador, entre o núcleo da matriz e os "pads" de E/S, para evitar impedimentos no traçado global de rotas. Além disso, o posicionamento automático deve lidar com ocupação do núcleo não muito superior a 60%, para evitar problemas na geração automática de conexões.

Finalmente, é possível tecer alguns comentários gerais com respeito aos circuitos de teste. Um problema crítico é a interação com os CMPs para fabricação. O melhor projeto não resiste a uma forma precária de interação. Este problema afetou ambos circuitos, impossibilitando a fabricação de um (multiplicador) e dificultando a alteração do outro (oscilador).

Para obter tal interação é necessária uma forma automática confiável de geração de máscaras, o que ainda não está disponível para a implementação de circuitos. Para suplantar o problema constante das modanças de regras, um sistema de "layout" simbólico seria muito útil Encontra-se em andamento no CPGCC/UFRGS um trabalho neste sentido /CAR 88/. Nestes sistemas a geração das máscaras finais é obtida

sistema de implementação de pré-difundidos este passo intermediário não ocasiona desperdício de área, o que sempre existe se o mesmo sistema é empregado na geração de máscaras de circuitos dedicados.

6 FERRAMENTAS DO MÉTODO CIPREDI

Neste capítulo, a implementação inicial de ferramentas do método CIPREDI é descrita, junto com o conceito geral de integração do ambiente. A seção 6.1 mostra como deve proceder a interação e a integração dos componentes de "software" do método, enquanto que as seções 6.2 e 6.3 discutem o simulador lógico e o editor de máscaras implementados neste trabalho, respectivamente.

6.1 Integração das Ferramentas

Nesta seção será discutido um esquema geral de interação entre as ferramentas do método CIPREDI. Será dada ênfase na descrição do acoplamento entre informações contidas nas representações estrutural e/ou funcional de um circuito e informações associadas à representação geométrica do mesmo, seguindo a nomenclatura introduzida por /GAJ 83/ e apresentada no capítulo 3. Estas informações são obtidas, as primeiras do banco de dados do sistema AMPLO, descrito em /GOL 88/ e as últimas do banco de dados GERME, definido por Morschel em /MOR 87/. Esta ênfase é justificada pelo fato do escopo de implementação deste trabalho envolver de maneira específica o acoplamento mencionado.

A conversão de informações estruturais e/ou funcionais em informações físicas é um dos aspectos críticos do projeto de circuitos pré-difundidos, como já foi delineado no capítulo 4.

6.1.1 Relação entre ambiente de concepção e ferramentas

Define-se ambiente de concepção como sendo todo o conjunto de representações de um projeto, completas ou não,

Ao usuário do método CIPREDI é permitido o acesso ao projeto através das ferramentas, definidas como meios de visualizar, transformar, eliminar ou associar elementos das diversas representações. Na figura 6.1, mostra-se o esquema geral de interação para uma ferramenta genérica i do método. A figura pretende salientar, graficamente, como o ambiente provê a integração das ferramentas.

INTERAÇÃO ENTRE FERRAMENTAS E O AMBIENTE DE CONCEPÇÃO CIPREDI

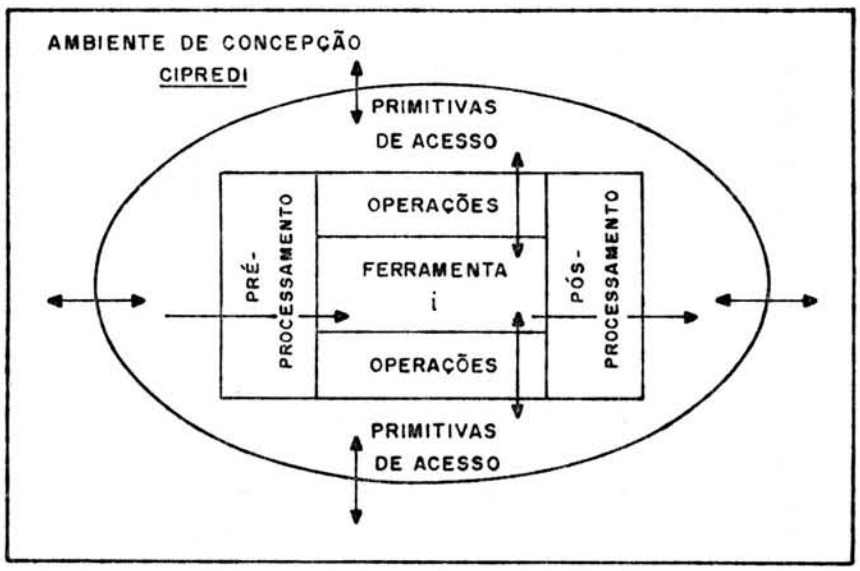


Figura 6.1 - Proposta de interação entre ferramentas e o ambiente de concepção CIPREDI

Cada ferramenta coloca à disposição do usuário um conjunto de operações, o que define uma janela de atuação, ou seja, uma visão direcionada da parte do ambiente de concepção relevante para os trabalhos de projeto. Para a consecução destas operações, a ferramenta dispõe de uma estrutura de dados otimizada para seus objetivos. Esta estrutura é inacessível aos usuários, pois estes lidam sempre com as operações colocadas a sua disposição, sem se preocupar em como estas são realizadas. As ferramentas

interagem com o ambiente através de atividades de pré e pós-processamento, e com os usuários através das operações.

O pré-processamento é necessário para que seja estabelecida a estrutura de dados de trabalho da ferramenta. Exemplos desta atividade são a leitura da topologia de um circuito para fins de simulação lógica e a aquisição da representação geométrica de um circuito para a edição de máscaras.

O pós-processamento possibilita a atualização do ambiente após a tarefa de uma ferramenta ser interrompida ou concluída. Exemplos típicos seriam o salvamento de resultados de uma simulação lógica e a atualização das geometrias das máscaras após uma edição. Além disto, o pós-processamento possibilita a criação de formas alternativas de representação, como, por exemplo, geração de máscaras no formato CIF - Caltech Intermediate Form, para a criação de fitas geradoras de padrões.

A separação entre operações, estruturas de dados, pré e pós-processamento garante a independência entre os algoritmos que implementam as tarefas básicas da ferramenta e os formatos de dados para entrada e saída, uma característica desejável em sistemas concebidos de forma incremental. Em particular, no método CIPREDI, esta independência é fundamental, devido à mutabilidade de meios de implementação de CIs no âmbito acadêmico.

Finalmente, a figura 6.1 mostra uma camada intermediária entre as ferramentas e o ambiente de concepção: as primitivas de acesso. Estas primitivas provêem uma maneira organizada de obter ou fornecer informações de ou para o ambiente de concepção. O funcionamento das primitivas é conhecido pelas ferramentas, e é somente através delas que cada aplicação enxerga as estruturas do ambiente.

A organização proposta permite independência e simultaneidade na implementação das ferramentas e dos bancos de dados que compõem o ambiente. Outra característica resultante é a possibilidade de verificação de obediência às restrições de integridade exigidas pela(s) base(s) de dados.

6.1.2 Integração das ferramentas do CIPREDI

O método CIPREDI suporta os estilos de projeto ascendente, descendente e misto. Nesta seção serão vistas as características de integração das ferramentas implementadas e o ambiente de concepção resultante, os quais possibilitam o exercício do método.

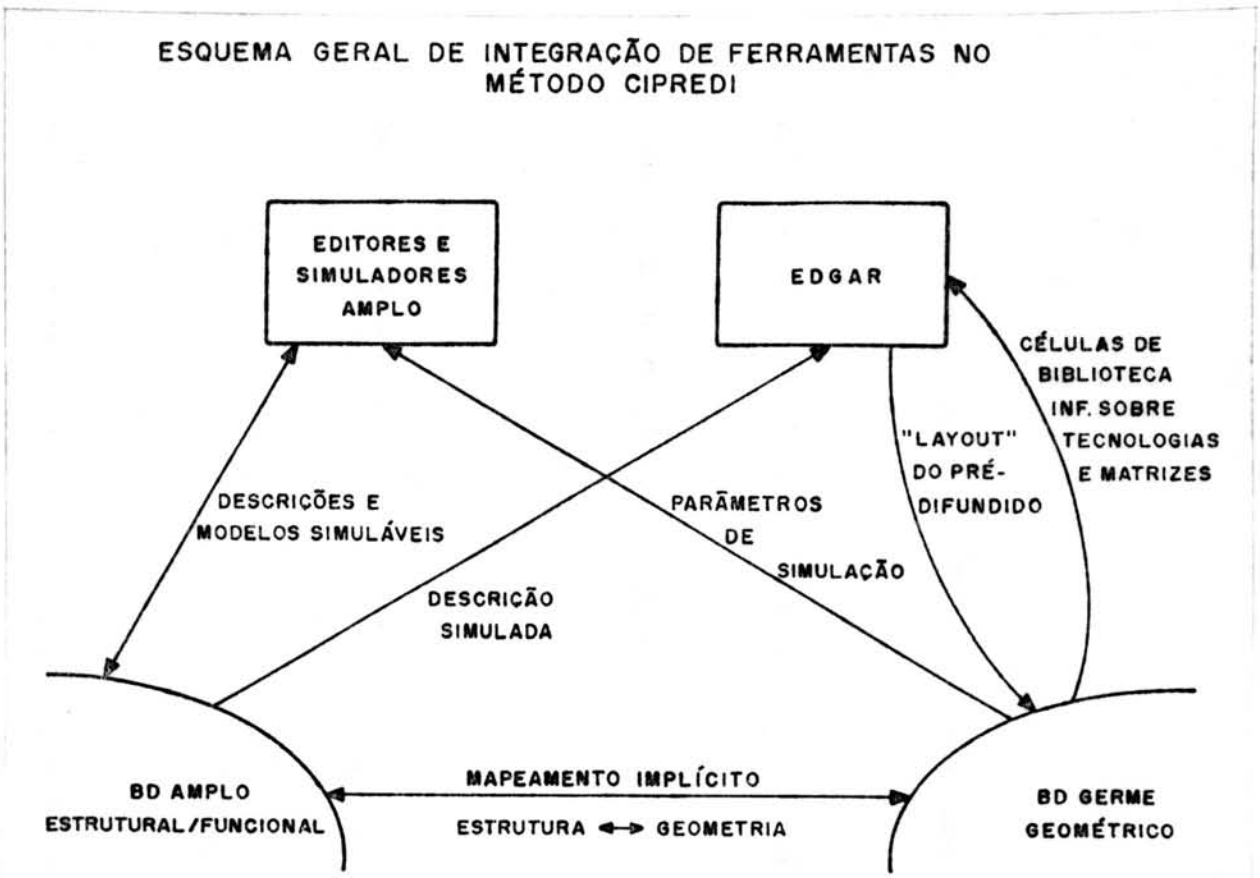


Figura 6.2 - Proposta de integração entre ferramentas no

Na figura 6.2, mostra-se o estado inicial do ambiente. A figura ressalta a conexão entre as ferramentas que lidam com as representações funcional e estrutural (à esquerda) e a que lida com a representação geométrica (à direita). O método objeto deste trabalho usa conceitos criados para o sistema AMPLO, apresentado em /WAG 87d/, um ambiente integrado de apoio ao projeto de sistemas digitais, e procura associá-los a um ambiente criado para suportar a concepção de circuitos integrados descrito em /MOR 87/. O objetivo é efetuar a conexão entre os dois conjuntos de níveis de abstração, possibilitando ao projetista partir de uma descrição abstrata de um sistema, navegar ao longo de um conjunto de representações com variados graus de detalhamento, e chegar a uma especificação geométrica das máscaras de um CI.

O sistema AMPLO lida com representações estruturais e comportamentais dos níveis sistema, transferência entre registradores - RT - e lógico, com algum envolvimento com o nível de chaves conforme descrito em /WAG 87b/, /CAL 88/ e /REY 87/. O sistema GERME, por outro lado, envolve a manipulação de informações do nível de chaves, do nível elétrico e do nível de máscaras. Deve-se notar a presença de redundância nas representações de um mesmo projeto nos dois sistemas de gerência de dados.

A separação de informações de projeto em um banco de dados de descrições abstratas (BD lógico) e um banco de dados de descrições físicas (BD geométrico) facilita o gerenciamento de informações com modelos, representações e formatos singulares.

O método CIPREDI advoga o uso do sistema AMPLO para a manipulação de informações até o nível lógico, ou seja, toda a gama de representações oferecida, lançando mão

representação ausentes no AMPLO. Isto garante uma estruturação máxima no uso de estilos de projeto predominantemente descendentes, o que se considera aconselhável. Também, devido ao fato da concepção de circuitos pré-difundidos destinar-se a pessoal técnico sem envolvimento maior com a área de microeletrônica, é possível garantir para estes projetistas o acesso apenas às informações pertinentes a sua área de atuação de maneira mais eficaz.

O engenheiro de sistemas usa, normalmente, abstrações até o nível de chaves, no máximo, deixando os níveis restantes sob a responsabilidade de ferramentas de síntese automática e/ou do fornecedor de circuitos integrados.

A visão que o usuário do método tem dos níveis inferiores é aquela provida pelas ferramentas que ele manipula. Por exemplo, caso este use um posicionador e traçador de rotas automatizado, o usuário verá o seu projeto como um conjunto de caixas pretas com entradas e saídas, que implementam funções primitivas do nível lógico, do nível RT ou do nível sistema. Estas caixas pretas serão interligadas segundo a lista de conexões obtida após a validação do projeto em níveis mais altos de abstração. A figura 6.3 ilustra esta abordagem com uso do diagrama Y.

Em uma atividade de concepção típica, o usuário utiliza as ferramentas do sistema AMPLO (figura 6.2, acima à esquerda) para especificar e validar o circuito até o grau de detalhamento necessário. Este detalhamento depende diretamente dos conteúdos das bibliotecas de células disponíveis para o projeto de pré-difundidos, pois cada folha da hierarquia validada pelo usuário deve corresponder a uma célula de biblioteca.

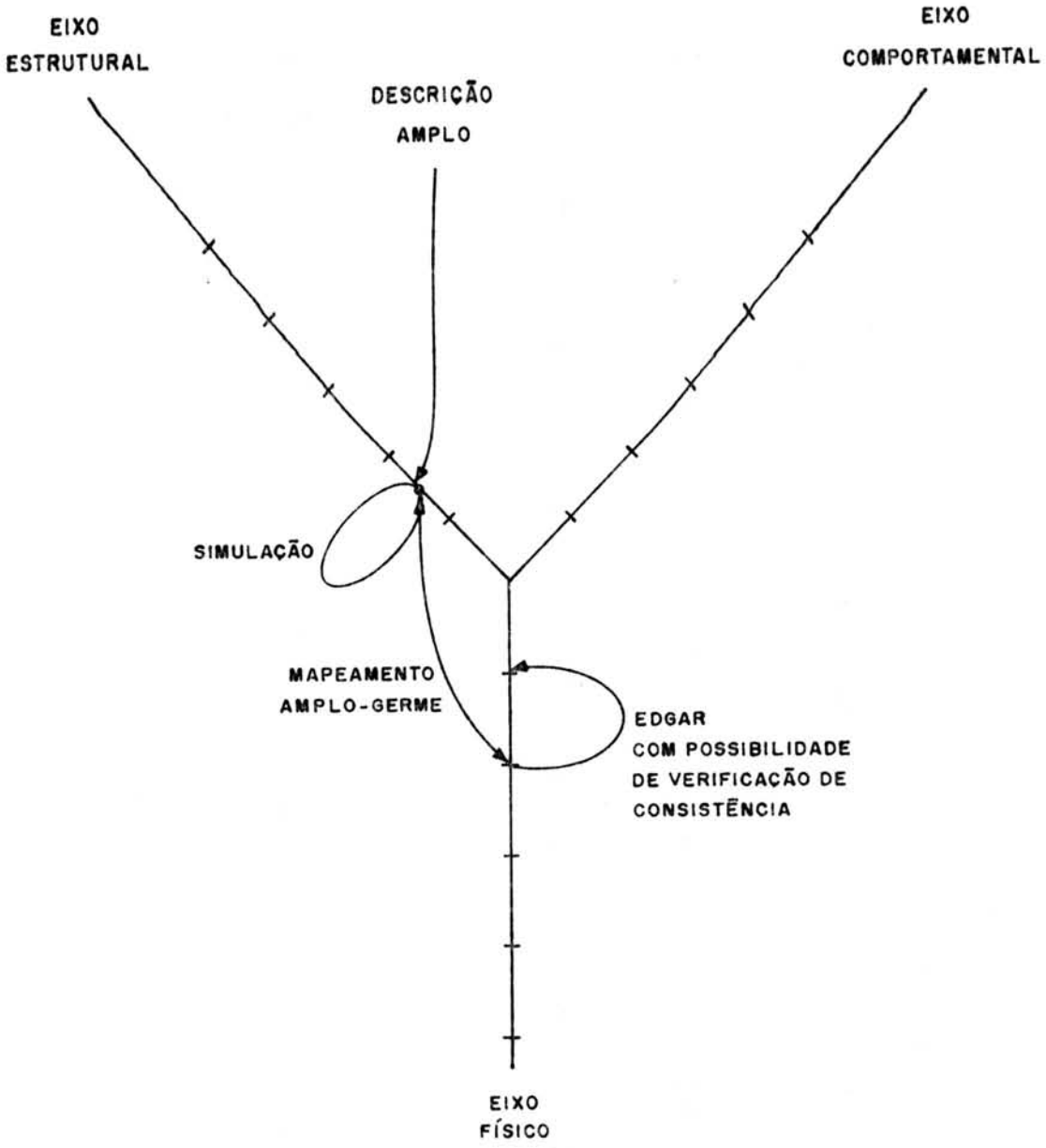


Figura 6.3 - Processo de projeto descendente automatizado

As células de biblioteca são gerenciadas pelo banco de dados GERME e detalham as folhas da hierarquia nos níveis inferiores de abstração.

Em tempo de geração de máscaras, deverá haver uma conexão implícita entre folhas da hierarquia AMPLO e a descrição de máscaras destas folhas nas bibliotecas do GERME. Esta conexão guiará o usuário na tarefa de geração interativa do "layout" do CI através do uso do EDGAR. O

EDGAR, tendo conhecimento da hierarquia e da topologia da descrição AMPLO, pode executar verificação de consistência entre o "lay-out" gerado e o projeto validado nos níveis de abstração superiores.

Um segundo modo de utilizar o EDGAR é através da aplicação de uma forma ascendente de projeto, onde o "lay-out" é criado interativamente, mediante instanciamento de células da biblioteca de pré-difundidos do GERME. Esta forma impede, contudo, a verificação de consistência descrita anteriormente, sendo mais adequada para a criação das bibliotecas de células em si, visto que estas se compõem de primitivas descritas com relativa facilidade sem uso de hierarquia. O diagrama Y da figura 6.4 ilustra esta abordagem.

Uma última observação sobre a integração de ferramentas se faz necessária: na figura 6.2, mostra-se a conexão entre ferramentas do AMPLO e o GERME para fornecimento de parâmetros de simulação. Devido à atual inexistência de uma interface consistente entre estas entidades, obtém-se hoje estes parâmetros a partir de versões de alternativas de agências do sistema AMPLO. Estas agências são criadas como imagens da biblioteca de células do GERME. Isto acarreta redundância adicional, e eventual inconsistência entre as duas bases de dados, mas garante a possibilidade de uso do método CIPREDI neste meio tempo.

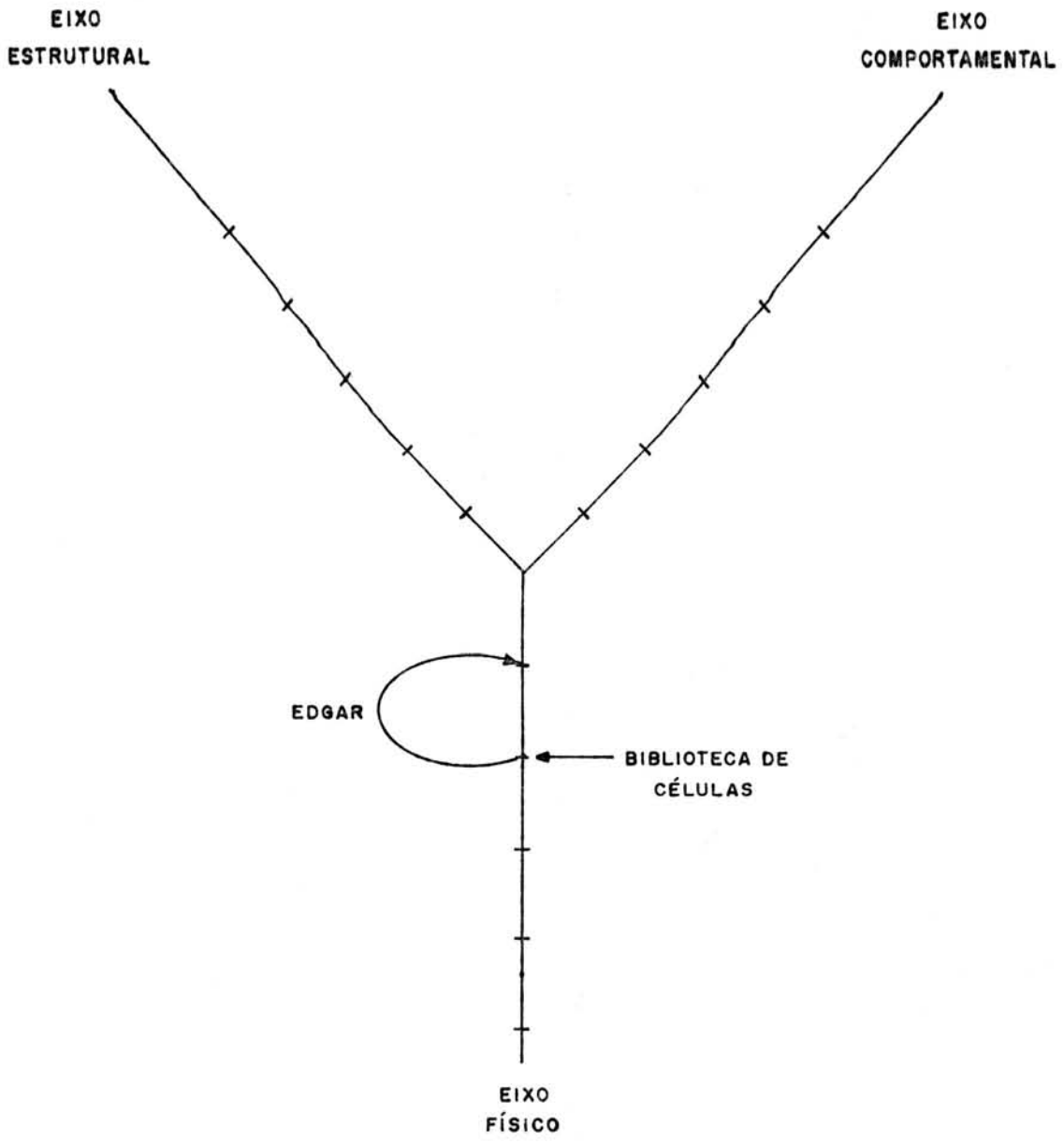


Figura 6.4 - Processo de projeto manual ascendente

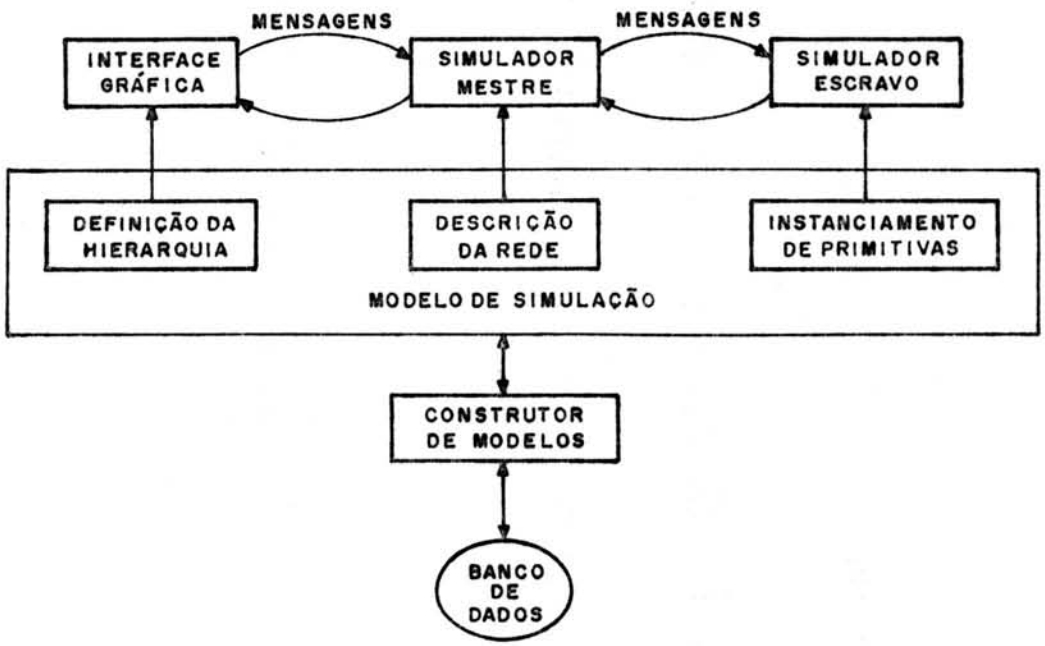
6.2 O Simulador NILO

Esta ferramenta foi definida dentro do escopo do projeto AMPLO, conforme mencionado na seção 4.2 deste trabalho. Levou-se em consideração as necessidades tanto de projeto de sistemas digitais com lógica discreta, como de projeto de circuitos integrados, na fronteira de cujos

O sistema AMPLO emprega o conceito de redes de agências /WEN 80/ para organizar a construção de sistemas digitais complexos. Agências são descritas segundo uma das linguagens mencionadas na seção 4.2 e são validadas através de um processo de simulação. O simulador NILO é uma das ferramentas responsáveis por este processo.

Dentro do sistema AMPLO, a separação explícita de estrutura e comportamento é reforçada pelo método de projeto escolhido. Esta dicotomia reflete-se na organização dos simuladores, que possuem uma estrutura do tipo mestre-escravo explicada mais adiante nesta seção. Nesta organização, o Mestre é responsável pela gerência das interfaces dos módulos, enquanto que o Escravo gerencia o interior dos módulos.

O ambiente de simulação do sistema AMPLO é mostrado na figura 6.5.



Cada nível abordado pelo sistema possui esta estrutura.

O modelo de simulação é uma rede de agências primitivas, ou seja, cada agência que compõe este modelo é descrita em uma das linguagens LAÇO, KAPA ou NILO. Este modelo é construído a partir de uma hierarquia de profundidade arbitrária presente no banco de dados, transformada, pelo construtor de modelos, em uma hierarquia a dois níveis usada na simulação. O processo de escolha da hierarquia inicial, bem como o processo interativo de simulação, são guiados pelo usuário através da interface.

A simulação é controlada pelo usuário via a interface gráfica unificada do sistema AMPLO. A partir dela, o usuário pode gerar as formas de onda de entrada do circuito, gerenciar as respostas em dispositivos de exibição gráfica e controlar a interrupção ou prosseguimento do processo de simulação. Várias opções para controle da simulação são empregadas:

. simulação passo a passo;
. simulação por tempo relativo;
. simulação por tempo absoluto;
. pontos de quebra ("break points").

Toda a interação entre a interface gráfica, simulador mestre e simulador escravo é obtida através de um protocolo unificado de troca de mensagens.

O nível NILO de descrição do sistema AMPLO lida com descrições mistas de portas lógicas e chaves bidirecionais (transistores). A figura 6.6 mostra um exemplo de agência descrita em NILO, nas versões textual e gráfica.

```
cbench RAM_CELL ) 1
level = NILO
interface
     in  ADDR, DATA_IN, WRITE : terminal;
     out  DATA_OUT : terminal ;
behavior
     gate
       G1 ( in = DI; contr = WR, out = DO; int = TRI ) : tristate;
          G3 ( in = 1; del = D1, int = TTL ),
          G4 ( in = 1; del = D1; int = TTL ) : not;
          G5 ( in = 1; del = D2; int = TTL ) : buffer;
     instate G2 ( cont1 = AD; data = DA, DB ) : nmos;
     delay D1 = 6, 10;
           D2 = 5, 9;
           tristate = 6, 8, 10, 10;
           nmos = 6, 2;
     identify TTL = strong, strong;
              TRI = strong, strong;
     node N1 = G1.DO, G2.DA, G5.1;
          N2 = G2.DB, G3.1, G4.2;
          N3 = G3.2, G4.1;
     equivalence DATA_IN = G1.DI; WRITE = G1.WR;
                 ADDR = G2.AD; DATA_OUT = G5.2;
end;
```

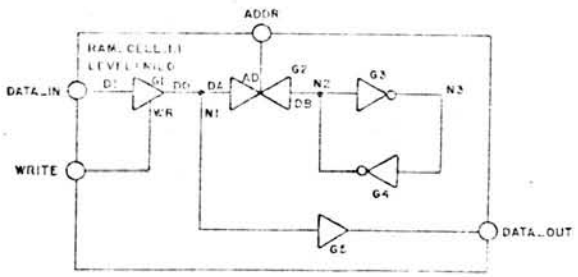


Figura 6.6 - Uma célula de memória RAM MOS modelada em NILO

O objetivo estabelecido para o presente trabalho foi a implementação do simulador escravo NILO, definido por Rey em /REY 87/, e a definição geral do simulador mestre /CAL 88/. As subseções a seguir discutem a interação entre os componentes do processo de simulação no nível NILO, além de apresentar o objetivo definido acima em detalhe.

6.2.1 Mensagens

Dentro do sistema AMPLO, foi estabelecido um protocolo unificado de troca de mensagens entre os elementos componentes do ambiente de simulação:

. interface;

. simulador mestre;

. simulador escravo.

Este protocolo é respeitado em todos os níveis suportados (NILO, KAPA e LAÇO) e também pelo simulador multinível, definido em /WAG 86/. O objetivo de tal formalização é permitir o compartilhamento de código entre os ambientes de simulação e garantir uma interface unificada para os simuladores. Atualmente, estas mensagens são implementadas sob a forma de subrotinas que trabalham sobre uma estrutura de dados específica sendo a uniformidade mantida apenas a nível de nome, número e tipo dos parâmetros envolvidos.

Define-se assim, quatro conjuntos de mensagens para um ambiente de simulação em AMPLO:

. da interface para o mestre;
. do mestre para o escravo;
. do escravo para o mestre;
. do mestre para a interface.

A hierarquia a três níveis definida dentro deste ambiente é explicitada pelo próprio nome das mensagens. Mensagens descendentes (dois primeiros tipos) são ordens, expressas como sentenças abreviadas, contendo um verbo na terceira pessoa do singular do imperativo afirmativo. Mensagens ascendentes (dois últimos tipos) são avisos, expressos como sentenças abreviadas, contendo um verbo na primeira pessoa do singular do pretérito perfeito do indicativo.

Na implementação atual, estas mensagens são parte do código do simulador. Os próprios simuladores, mestre e escravo, constituem um único programa, sem identidade distinta do ponto de vista de sistema operacional. As mensagens constituem desta forma, a única maneira através

interferir na estrutura de dados ou no fluxo de controle dos módulos restantes. Por exemplo, as mensagens enviadas do mestre para o escravo são rotinas do escravo, pois alteram a estrutura de dados do último. O mestre conhece apenas o nome da mensagem e os parâmetros para sua chamada.

Com a eventual evolução do sistema AMPLO para um ambiente multitarefa, conforme descrito no capítulo 4, a estrutura física deste sistema de mensagens deve ser alterada, para permitir o melhor aproveitamento da concorrência embutida nos conceitos dos simuladores. Mestre, Escravo e Interface podem atuar, sem grandes modificações, como processos concorrentes, trocando mensagens segundo mecanismos previstos no sistema UNIX. Watkins /WAT 87/ oferece um estudo detalhado destes mecanismos e sua aplicabilidade.

6.2.2 A implementação do simulador escravo NILO

Como mencionado na seção 6.2 acima, o simulador escravo exercita descrições mistas de circuitos. Ao lado das portas lógicas tradicionais, elementos "tristate" e resistores de "pull-up" e "pull-down" podem ser usados. Além destes, três tipos de portas bidirecionais estão disponíveis (NMOS, PMOS e CMOS). Esta representação mista permite o uso de uma notação mais compacta que uma descrição pura a nível de chaves /HAY 87/, bem como uma modelagem mais fiel dos fenômenos associados ao comportamento de dispositivos bipolares e unipolares.

O caráter bidirecional dos transistores MOS é simulado de maneira eficiente pela aplicação de três princípios básicos:

. particionamento da agência em subredes

. aplicação de um algoritmo de relaxação local às subredes;

. uso de uma técnica heurística para o cálculo dos atrasos associados a eventos, eventos estes resultantes da avaliação das subredes.

Estes princípios são detalhados em /REY 87/, sendo o algoritmo de relaxação local uma proposta de Dumlugöl /DUM 83/ adaptada às primitivas e a descrição do nível NILO do sistema AMPLO.

Para que a simulação pudesse modelar de forma mais precisa o comportamento de dispositivos em várias tecnologias, foi elaborada uma álgebra de 12 elementos. Esta álgebra representa o valor de um sinal, em qualquer nodo do circuito descrito em NILO, através de um par de elementos: um estado e uma intensidade. Um estado pode ser: 0 ou 1, representando os estados lógicos definidos naturalmente na álgebra de chaveamento, ou X, representando um estado desconhecido ou indefinido. A intensidade pode ser Z, W, D ou E. Z indica uma situação de alta impedância ou armazenamento capacitivo de carga. W significa um sinal fraco, como, por exemplo, aquele obtido pelo uso de transistores de depleção, resistores de "pull-up" ou "pull-down". D identifica um sinal forte, obtido via transistores de enriquecimento saturados conectados à alimentação. E é usado para a representação de sinais "grampeados" ou de alimentação (VDD ou GND). A modelagem de falhas do tipo "grudado em" ("stuck-at") é permitida pelo uso da intensidade E, que não pode ser alterada pelo processo de simulação. O diagrama de Hasse /HAY 86/ da figura 6.7 mostra a hierarquia de valores do simulador NILO.

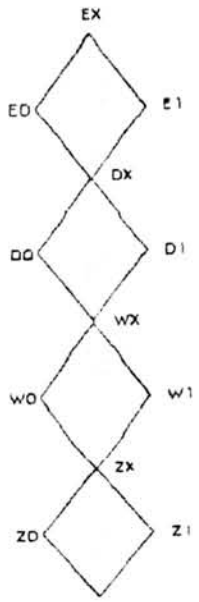


Figura 6.7 - Diagrama de Hasse (Reticulado Hierárquico) da álgebra do simulador NILO.

Para alcançar uma modelagem razoável do ponto de vista de atrasos, foi escolhido o modelo de atraso de tempo de propagação, onde a influência de uma variação nas entradas surge na saída após um tempo t, dependente da porta lógica em questão e do estado final da saída. Transições na saída de uma porta lógica possuem 3 atrasos associados:

. tpd 0->1 - atraso de propagação quando a saída varia do estado 0 para o estado 1;

. tpd 1->0 - atraso de propagação quando a saída varia do estado 1 para o estado 0;

. atraso unitário - para qualquer outra transição de estado ou para transições apenas de intensidade.

Dispositivos "tristate" possuem, além destes, o tempo da ativação/desativação, considerado unitário, dada a complexidade de prever a temporização de sinais em barramentos, onde este tipo de dispositivos é normalmente

Chaves bidirecionais, por seu lado, possuem também três atrasos associados. Quando ativas, existe o atraso necessário para equilibrar ambos os lados da chave em 0 e o atraso para equilibrar ambos os lados da chave em 1. De novo, o tempo de ativação/desativação foi assumido como unitário.

Como os atrasos fixos estão associados aos elementos de circuito, o cálculo do atraso real para um determinado nodo pode depender de vários componentes, em função de qual elemento domina o estado e a intensidade final do nodo. Este modelo obriga a existência de um mecanismo de cancelamento de eventos /REY 87/. Não são permitidas especificações de atraso nulo ("zero delay"). Atrasos são números naturais diferentes de 0.

Para a implementação do simulador foi usada a linguagem C, conforme mencionado no capítulo 4. O compilador C da Microsoft Versão 4.0 foi escolhido para implementação.

Com base no estudo dos algoritmos propostos por Rey, foi feito um particionamento do código do simulador em seis módulos logicamente distintos:

. mensagens do Mestre para o Escravo;
. avaliação de Subredes;
. algoritmo de relaxação para subredes bidirecionais;
. avaliação de portas lógicas e transistores;
. avaliação de atrasos;
. manipulação de listas;

A seguir são detalhadas a função e características de código de cada um dos módulos.

6.2.2.1 Mensagens do mestre para o escravo

Na implementação atual as mensagens enviadas do Mestre para o Ecravo são:

- informe_valor_sinal;
- informe_tipo_sinal;
- inicialize_agência;
- finaliza_inicialização;
- monitore_sinal;
- cancele_monitoração_sinal;
- ative_agência;
- varie_ambiente_de_entrada;
- mude_sinal;
- grampeie_sinal;
- desgrampeie_sinal;

Uma discussão detalhada destas mensagens está contida em /REY 87/, exceto das duas primeiras rotinas, surgidas em função de necessidades da definição do simulador mestre NILO.

Dentre as mensagens citadas acima, as seis primeiras não implicam alteração do estado da agência, as restantes o fazem. Esta característica faz com que as últimas, ao lado de executarem a função da mensagem, provoquem o disparo do algoritmo do simulador escravo. A rotina principal é mostrada na figura 6.8.

Cada linha da rotina simula_agência corresponde a um dos passos do algoritmo. A rotina efetiva_eventos atribui novos valores aos nodos da agência, baseada em informações da lista de eventos local (LEN) para o tempo corrente.

```
void simula_agência (void)
     {
       efetiva_eventos ( );
       envia_msg_variações ( );
       avalia_subredes ( );
       envia_LPRV ( );
       verifica_eventos ( );
     }
```

Figura 6.8 - Rotina principal do simulador escravo

A rotina envia_msg_variações coleta todos os valores que devem ser informados ao mestre e gera a mensagem adequada. Deve-se notar a posição deste passo no algoritmo do simulador escravo. Ele ocorre antes da avaliação do novo estado da agência. Esta característica permite uma maior eficiência em um ambiente multi-tarefa, aumentando o paralelismo entre os processos mestre e escravo. Ela é possibilitada pela restrição de inexistência de eventos com atraso nulo, o que não deixa de ser uma modelagem adequada de fenômenos elétricos em dispositivos reais /WAG 84/. Isto significa que, durante a avaliação da agência, nunca serão gerados eventos para o tempo atual, garantindo-se a estabilidade de qualquer configuração de elementos de circuito, a cada instante de tempo.

A rotina avalia_subredes é o corpo do algoritmo e será vista mais adiante. A rotina envia_LPRV existe por compatibilidade com o nível KAPA de descrição. Finalmente, verifica_eventos é o passo final do escravo, onde a LEN é varrida, descobrindo o evento com o tempo de efetivação mais próximo, caso exista. Esta informação é passada para o mestre, garantindo a nova ativação do escravo no máximo neste tempo.

6.2.2.2 Avaliação de subredes

A estrutura de uma agência NILO é fornecida para o simulador sob a forma de uma partição de subredes. O conceito de subredes é baseado na localidade do comportamento bidirecional em um circuito digital /DUM 83/. Subredes são uma forma de encapsular este comportamento em módulos unidirecionais, do ponto de vista de entradas e saídas. O algoritmo de relaxação local é responsável pela avaliação de subredes que apresentam algum comportamento bidirecional.

Este comportamento está presente apenas quando se usam chaves bidirecionais. O algoritmo de particionamento de uma agência NILO em subredes é descrito por Rey e implementado pelo editor/compilador NILO. Três tipos de subredes são possíveis de existir em uma descrição NILO:

- subredes complexas bidirecionais;
- subredes complexas unidirecionais;
- subredes simples.

O algoritmo de relaxação só se aplica às primeiras. As restantes são avaliadas mediante algoritmos tradicionais de simulação a nível de portas lógicas.

Durante a execução da rotina efetiva_eventos são identificadas as subredes que devem ser avaliadas, sendo estas armazenadas em uma Fila de Propagação (FP) de valores. A rotina avalia_subredes extrai os identificadores de subrede desta fila e dispara o algoritmo de avaliação adequado, dependendo do tipo de subrede em questão.

6.2.2.3 Algoritmo de relaxação para subredes bidirecionais

Este módulo é responsável pela avaliação de subredes contendo pelo menos um elemento do tipo chave bidirecional. Ele é descrito em detalhe por Dumlugöl. Este algoritmo surgiu como uma alternativa ao algoritmo de avaliação empregado em simuladores de chaves tradicionais /BRY 81/ /HAY 81/, que empregam uma abordagem global do circuito, descrito como um grafo. A baixa complexidade algorítmica alcançada com a relaxação local é o maior mérito do trabalho de Dumlugöl, ao lado da aplicação de heurísticas que introduzem um baixo grau de pessimismo na modelagem.

6.2.2.4 Avaliação de portas lógicas e transistores

Neste módulo do simulador escravo são calculados os novos estados de saídas de portas lógicas e o estado atual de atividade de chaves bidirecionais. A intensidade do novo valor é calculada com base na porta lógica em questão, uma vez que esta é característica do elemento, não da intensidade das entradas.

A avaliação é feita mediante uma pesquisa "hash" em tabelas de elementos. Três tipos de portas lógicas (AND, OR, XOR) possuem uma tabela associada. Internamente, estas tabelas são arranjos unidimensionais, onde o índice de acesso ao arranjo é montado a partir do valor atual das entradas. Para portas com múltiplas entradas, a avaliação é feita duas a duas (para XOR e NXOR) ou três a três (para AND, OR, NAND e NOR). Para portas com uma entrada, a avaliação é feita diretamente por rotinas especializadas. A rotina avalia_porta controla toda a execução neste módulo.

6.2.2.5 Avaliação de atrasos

A avaliação de atrasos é obtida através de uma heurística proposta por Rey /REY 87/, onde um conjunto de casos está previsto. Cada variação de valor é pesquisada, tentando encaixar o conjunto de condições que a provocaram em um destes casos. Quando nenhum caso se aplica, dois fatos podem se verificar:

. programação com atraso unitário;
. identificação de erro de modelagem.

O primeiro se aplica na maioria dos casos. Devido à dificuldade de previsão do atraso real, uma abordagem de pior caso é assumida. O segundo caso é dedicado a revelar as limitações do modelo utilizado, sendo o usuário avisado da impossibilidade de previsão do atraso real do circuito, e aplicado atraso unitário.

Durante a inicialização, forçada pelo usuário através da interface, todos os atrasos são considerados unitários. Esta técnica constitui uma forma de acelerar a estabilização da agência, levando-a mais rapidamente para o estado inicial esperado.

6.2.2.6 Manipulação de listas

Dentro deste módulo, estão contidas todas as rotinas de gerenciamento das estruturas dinâmicas do simulador escravo. Ao todo, são dez estruturas: duas filas, uma pilha, uma fila circular e seis listas encadeadas. Além disto, toda a estrutura do circuito está armazenada em nove listas seqüenciais com a mesma estrutura. O acesso a estas listas é obtido usando como índice um identificador único.

Como exemplo, tome-se a lista de nodos. Cada um

dos nodos de uma agência é numerado de 0 a n-1, sendo n o número de nodos da agência. Existe, na interface gráfica apenas, a denominação dos nodos segundo identificadores textuais, conforme especificado nas linguagens do sistema AMPLO. A interface mantém uma tabela de conversão de identificadores textuais para identificadores internos. A forma de recuperar as informações de um nodo é pelo acréscimo do identificador interno ao ponteiro que especifica o início da lista seqüencial de nodos. A estrutura detalhada de cada uma das listas e a informação contida em cada elemento estão descritas em /REY 87/. Esta organização foi concebida para satisfazer os requisitos de desempenho do simulador.

A estrutura dinâmica mais complexa e importante é a lista de eventos local (LEN), usada para a programação de eventos. A estrutura desta lista é uma versão modificada do laço delta-T proposto por Ulrich /ULR 69/. Ela permite a total eliminação do manuseio de listas de "overflow" no tratamento de eventos.

A LEN é uma fila circular, composta por um número fixo (por modelo de simulação) de elementos denominados escaninhos, cada um representando um instante unitário de tempo. A fila circular possui o tamanho igual ao maior atraso de componente primitivo presente na agência. Cada escaninho é um ponteiro para uma lista encadeada de eventos, cada evento contendo informações sobre uma futura mudança de valor em um nodo da agência. Junto com o evento está armazenado o tempo para o qual ele está programado. Esta técnica permite a programação de eventos com um atraso maior que o tamanho do laço delta-T a partir do instante atual, sem a necessidade de emprego de listas de "overflow". Atrasos com estas características podem ocorrer devido a dois fatores:

. armazenamento da programação de variações de

entradas primárias;

. avaliação de atrasos como soma de atrasos de elementos primitivos, em particular se existirem cadeias de chaves bidirecionais em série com saídas de portas lógicas.

A figura 6.9 ilustra a descrição acima com uma LEN de 8 escaninhos com 9 eventos programados.

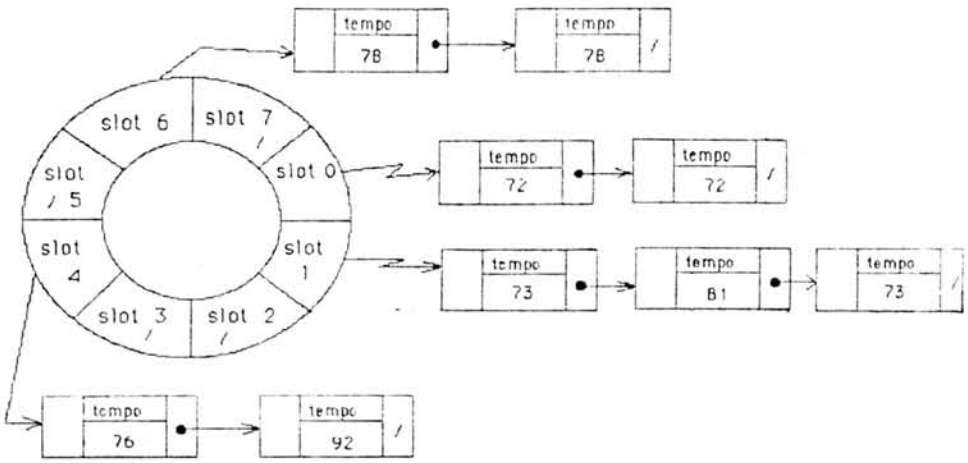


Figura 6.9 - Estrutura da LEN através de um exemplo

Para cada instante absoluto de tempo existe um escaninho contendo os eventos programados para aquele instante. A identificação deste escaninho é feita pela fórmula:

Escaninho_do_tempo_corrente = Tempo_atual MOD Total_de_escaninhos

6.2.3 A definição do simulador mestre NILO

As funções básicas dos simuladores mestre do

. avançar incrementalmente o tempo de simulação;

. coordenar o disparo do(s) simulador(es) escravo(s) quando isto se fizer necessário;

. calcular o consenso de sinais gerados pela contribuição de saídas de mais de uma agência simultaneamente;

. interagir com a interface gráfica a nível de mensagens;

. tratar parte dos erros gerados pelo simulador escravo.

Dois conjuntos de mensagens compõem o simulador mestre:

. mensagens da interface gráfica para o mestre;
. mensagens do escravo para o mestre.

A seguir são listadas estas mensagens, com uma breve descrição de sua função. Após, as principais rotinas do simulador mestre são definidas, junto com algumas estruturas de dados fundamentais.

6.2.3.1 Mensagens da interface gráfica para o mestre

. monitore_breakpoint (condição, &nro_break) - estabelece uma condição para interromper os simuladores, caso esta se verifique em algum estado estável. A condição é uma expressão booleana, envolvendo valores da álgebra de 12 elementos mostrada na subseção 6.2.2. O nro_break é um parâmetro de retorno, um identificador interno que serve de referência para a condição de "break point";

. cancele_breakpoint (nro_break) - descarta a condição de interrupção do simulador devida ao "breakpoint"

. monitore_sinal_interface (id_sinal) - solicita o aviso, à interface gráfica, cada vez que uma transição de estado e/ou intensidade ocorrer no sinal identificado por id_sinal;

. cancele_mon_sinal_interface (id_sinal) - descarta a necessidade de aviso de transição no sinal identificado por id_sinal;

. mude_sinal_interface (id_sinal, novo_valor) - altera instantaneamente o valor do sinal id_sinal na interface de uma agência para novo_valor;

. grampeie_sinal_interface (id_sinal, valor) - força valor permanentemente sobre o sinal de interface id_sinal. Serve para a modelagem de falhas de tipo "stuck-at";

.desgrampeie_sinal_interface (id_sinal) - libera o sinal de interface id_sinal para ser alterado pelo processo de simulação;

. programe_entrada_primária (id_sinal, lista de (tempo, valor)) - insere na lista de eventos geral (ver item 6.2.3.3) eventos futuros para sinal id_sinal;

. ative_mestre ( ) - rotina principal do simulador mestre, explicada em detalhes na subseção 6.2.3.3;

. monitore_sinal_interno, cancele_mon_sinal_interno, mude_sinal_interno, grampeie_sinal_interno, desgrampeie_sinal_interno - idênticas às cinco primeiras mensagens descritas acima, apenas lidando com sinais no interior de uma agência, e não em sinais de sua interface, possuindo um

em questão;

. inicialize_circuito ( ) - coloca todas as agências do modelo simulável no modo de inicialização, onde todos os atrasos são unitários;

. finalize_inicialização_circuito ( ) - descarta o modo de inicialização, fazendo com que o algoritmo de simulação considere os atrasos de propagação de cada elemento em particular.

6.2.3.2 Mensagens do escravo para o mestre

. variei_ambiente_de_saída (id_agência, lista de saídas que variaram) - informa ao mestre todas as mudanças de valor ocorridas na interface da agência id_agência durante uma execução do simulador escravo;

. modifiquei_sinal (id_agência, lista de monitorados que variaram) - informa ao mestre todas as variações de valor nos sinais atualmente monitorados;

. programei_pulso_de_relógio (id_agência, tempo, valor_anterior, próximo_valor, id_sinal) - existe por compatibilidade com o nível KAPA. Informa a ocorrência de um evento em um sinal do tipo CLOCK, de acordo com os tipos definidos em /WAG 87c/;

. cancelei_pulso_de_relógio (id_agência, tempo, id_sinal) - similar à anterior, só que informa o cancelamento de um evento antes programado, ou a inexistência de eventos, para o sinal identificado por id_sinal, após a atividade do simulador escravo haver se extinguido;

informa ao mestre que a agência deve ser simulada novamente no instante tempo, o mais tardar, pois existe ao menos um evento interno programado para este instante;

. cancelei_evento (id_agência) - indica para o mestre ou a inexistência de eventos programados para a agência identificada por id_agência, ou o cancelamento do respectivo evento atualmente na Lista de Eventos Geral;

. constatei_erro (id_agência, tipo, id_nodo) - aponta a ocorrência de erros dentro do simulador escravo. Os erros podem ser de modelagem, de limites (falta de memória, etc), do usuário (parâmetros de mensagens fora da faixa, etc), entre outros.

6.2.3.3 Algoritmo geral do simulador mestre

A figura 6.10 descreve, em alto nível de abstração, a principal rotina do simulador mestre, usando uma linguagem similar à implementação em C.

Uma descrição também em alto nível das listas empregadas no processo de execução do simulador mestre, a definição de tipos em linguagem C e o detalhamento das duas rotinas mais importantes do simulador podem ser encontrados no anexo 3 deste trabalho.

Na figura 6.10 nota-se que o laço de simulação é executado mediante a verificação de 3 condições:

. não ultrapassagem do tempo final;
. não atingimento do "break point";
. inexistência de interrupção pela interface gráfica.

Estas condições são testadas no início da

execução do mestre e ao fim do processamento de todas as atividades para um determinado instante de tempo, antes do incremento da variável global Tempo_Atual. A variável Tempo_Final é estabelecida pelo usuário de forma explícita ou pela interface gráfica de forma implícita, dependendo da condição de parada assumida.

```
ative_mestre ()
  {
    enquanto ( Tempo_Atual <= Tempo_Final &&
               Nao_ha_breakpoint () &&
               Nao_ha_interrupcao_do_usuario () )
      {
        Tempo_Atual++ ;
        if ( existem eventos no slot atual da Lista
             de Eventos Geral )
            {
              for ( cada evento no slot atual com
                    tempo_de_evento == Tempo_Atual )
                  if ( evento for ativacao-de-Agencia )
                      {
                        - inserir indice da Agencia na
                          Lista de Ativacao ;
                        - eliminar evento do Laco delta-T ;
                      }
                  else  /* entao eh
                           variacao-de-entrada-primaria */
                      {
                        - trata_variacao_sinal_interface
                              ( sinal_do_evento,
                                novo_valor_evento ) ;
                        - eliminar evento do Laco delta-T ;
                      }
              for ( cada Agencia na Lista de Ativacao )
                    {
                      if ( Lista de Variacoes da Agencia
                           - LVA - nao esta vazia )
                         varie_ambiente_de_entrada ( indice
                              da Agencia, Lista de Variacoes
                              da Agencia - LVA ) ;
                      else
                         ative_agencia ( indice da Agencia ) ;
                    }
            }
      }
    simulacao_parou ( tipo de motivo, motivo ) ;
  }
```

Figura 6.10 - Rotina ative_mestre

A ocorrência ou não de "breakpoints" é verificada de forma exaustiva em cada estado estável da rede. A

interrupção da simulação é permitida a qualquer momento, através da interface gráfica. A rotina não_há_interrupção_do_usuário verifica o estado do pedido de interrupção, também apenas a cada estado estável.

Todos os eventos gerenciados pelo simulador mestre podem ser colocados em uma de duas classes:

. variação de entrada primária;
. nota de tempo.

Variações de entradas primárias são geradas por mensagens da interface gráfica e armazenadas diretamente na Lista de Eventos Geral, uma lista com a mesma estrutura da Lista de Eventos Local (LEN) descrita anteriormente, um laço delta_T modificado. Notas de tempo são resultado das mensagens programei_evento enviadas pelas agências para o mestre, através do simulador escravo. Elas indicam o tempo máximo a ser esperado para a ativação de uma agência, onde ativação significa o disparo do simulador escravo para avaliar a agência.

Normalmente, agências podem ser ativadas por dois motivos:

. existência de uma nota de tempo com identificador da agência em questão no escaninho do tempo atual da LEG;
. variação de um sinal de entrada ou bidirecional nos limites da agência em questão.

Este procedimento corresponde à aplicação da técnica tradicional conhecida como "selective trace", aplicada em simuladores lógicos /WAG 84/.

Durante a avaliação do modelo simulável, em um instante particular de tempo, dois procedimentos são

executados em seqüência:

. identificação dos tipos de evento presentes no escaninho da LEG correspondente ao tempo atual, inserção do índice da agência presente no evento em uma lista de ativação e eliminação do evento da LEG;

. processamento da lista de ativação para propagar os efeitos dos eventos presentes na LEG para o interior das agências, usando a mensagem adequada, que ocasionará o disparo do simulador escravo.

Este formato aparece na figura 6.10 como dois laços "for", sucedendo o incremento do tempo atual. Ao término da atividade do simulador mestre, uma mensagem é enviada à interface gráfica, informando o motivo da interrupção.

Dois comentários finais são necessários sobre o simulador mestre. Primeiro, a linguagem REDES permite a manipulação, assim como a linguagem NILO, de vetores de bits agrupados logicamente. Estas linguagens permitem ainda a composição e decomposição destes vetores.

Dentro do simulador, a identidade lógica destes vetores é descartada, evitando perdas de eficiência na estrutura de dados do simulador e aumentando a flexibilidade e a simplicidade do algoritmo. Assim, todos os sinais manipulados pelos simuladores mestre e escravo NILO são bits. A identidade dos vetores é mantida ao nível de interface gráfica, como forma de manter a ergonomia da mesma.

Segundo, outra rotina importante para a compreensão do simulador mestre é a trata_variação_sinal_interface, responsável, entre outras atividades, pelo cálculo do consenso em barramentos conectando sinais de interface do tipo BUS.

Esta rotina é executada em duas situações possíveis:

. quando ocorre variação de entrada primária;

. quando da geração de uma mensagem variei_ambiente_de_saída pelo escravo.

Em ambos casos, a mudança de um valor de sinal de interface é propagada através da Lista de Ativação e da Lista de Variações de cada agência para todas as agências com algum sinal do tipo IN ou BUS conectados ao sinal de interface em questão.

O processamento posterior da Lista de Ativação garante o correto ordenamento de eventos na fronteira entre agências. O algoritmo da rotina trata_variação_sinal_interface está mostrado no Anexo 3.

6.3 O editor EDGAR

Dentro do método CIPREDI, o EDGAR é a primeira ferramenta a prover uma maneira de manipular o eixo físico de representação do diagrama Y.

Esta ferramenta é um editor gráfico, interativo, dirigido por cardápios e orientado à geração de posicionamento de células e traçado de conexões em um circuito pré-difundido.

Nas subseções a seguir, é apresentada a estrutura geral da ferramenta. Na subseção 6.3.1 as informações manipuladas pelo EDGAR são separadas em compartimentos logicamente distintos, sendo esta separação justificada brevemente. A subseção 6.3.2 discute a relação do EDGAR com o usuário, por um lado, e com os BDs AMPLO e GERME, por

outro. A seguir, a conexão entre as tarefas de geração de máscaras com o EDGAR e a "netlist" validada obtida por simulação no ambiente AMPLO é abordada. A subseção 6.3.4 menciona a forma de criação de células de biblioteca usadas no EDGAR. Finalmente, são descritas as estruturas de dados internas mais importantes do editor.

O detalhamento maior do EDGAR foge aos limites deste trabalho, sendo abordado em outras publicações.

6.3.1 Diagrama de fluxo de dados do editor EDGAR

A figura 6.11 mostra os diferentes tipos de informações externas manipuladas pelo EDGAR. O editor interage com cada um dos módulos através de um conjunto de primitivas de acesso associadas ao BD GERME. Atualmente, as primitivas de acesso usam um conjunto de arquivos contendo as diversas informações necessárias ao programa.

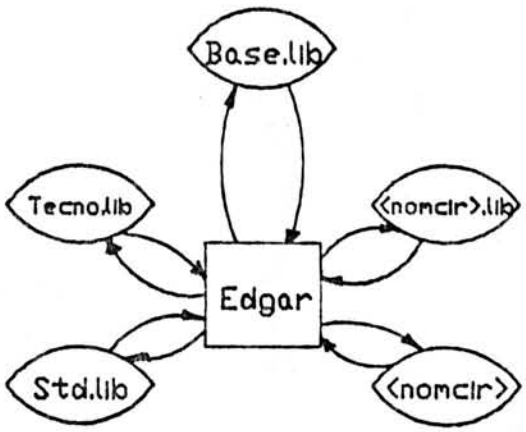


Figura 6.11 - Diagrama de fluxo de dados do EDGAR

A função de cada um dos blocos do DFD é apresentada a seguir /CAL 87b/, à exceção do próprio editor, discutido mais adiante. A separação dos blocos é

lógica, não possuindo relação direta com o armazenamento de informações em disco.

TECNO.LIB é uma biblioteca de tecnologias, criada e mantida pelo fabricante. Contém informações sobre os níveis de máscara de cada tecnologia, sobre a capacidade de personalização de cada uma destas máscaras pelo projetista de pré-difundidos e sobre as configurações de matrizes de pré-difundidos disponíveis para o projeto. Para a especificação de matrizes, este bloco guarda referências à biblioteca BASE.LIB.

BASE.LIB armazena todas as descrições de células de base e células de E/S pré-difundidas, em cada configuração de matriz suportada. Células pré-difundidas de blocos especializados também são armazenados nesta biblioteca.

STD.LIB possui as células de biblioteca propriamente ditas, ou seja, os padrões de metalização que implementam funções com nível de abstração mais alto. Este módulo é, normalmente, criado pelo fabricante e utilizado pelo projetista para a geração das máscaras. Esta tarefa é realizada sobrepondo descrições desta biblioteca a uma planta baixa descrita em TECNO.LIB. Eventualmente, a ausência de uma célula ou a inadequação de qualquer das células existentes em STD.LIB pode ditar a criação de novas células de biblioteca. Esta operação é efetuada seja pelo fabricante, seja pelo projetista, dependendo da complexidade do problema e da competência das partes envolvidas.

<nomcir>.LIB é uma biblioteca constituída pelas células empregadas pelo projetista ao longo da síntese do circuito de nome <nomcir>. Estas células podem ser simples referências a células de STD.LIB ou células compostas hierarquicamente, onde as folhas da hierarquia são

referências a células de STD.LIB.

Finalmente, o módulo denominado <nomcir> contém instanciamentos das células pertencentes ao topo da hierarquia de projeto do circuito, se caracterizando como um diagrama de blocos geral, onde cada bloco é descrito por referências a células de <nomcir>.LIB.

6.3.2 Ambiente de implementação do EDGAR

De acordo com a discussão apresentada na primeira seção deste capítulo, o EDGAR interage com duas fontes de informação para executar a geração das máscaras de personalização de um CI pré-difundido:

. os bancos de dados AMPLO e GERME;
. o projetista de pré-difundidos.

A primeira fonte, os bancos de dados utilizados no método CIPREDI, provêem uma organização das informações estáticas, como matrizes, células de biblioteca e tecnologias, e das informações dinâmicas, geradas ao longo de um projeto, tais como conectividade das células e máscaras de personalização, completas ou não. A interação entre o EDGAR e esta fonte se faz através de primitivas de acesso, definidas /JAC 88b/ /BEC 88/ pelos bancos de dados envolvidos, conforme visto na seção 6.1.

A segunda fonte de informação, o projetista de pré-difundidos, especifica as atividades a serem executadas pelo editor, conhecendo o estado atual das informações contidas nos bancos de dados. O projetista controla, além disso, o processo de construção de máscaras, seja ele realizado segundo o estilo ascendente ou descendente. O projeto ascendente, como visto no capítulo 4, parte de uma especificação não armazenada em qualquer dos bancos de

dados, apenas conhecida, possivelmente, pelo projetista. O projeto descendente, por outro lado, usa como guia, na geração de máscaras, a "netlist" obtida após o exercício das descrições lógicas com o simulador NILO. Esta "netlist" é usada pelo EDGAR como forma de garantir uma consistência parcial entre as descrições lógica e física de um projeto de pré-difundido.

A interação projetista-EDGAR se faz mediante uma interface gráfica especialmente desenvolvida. Esta interface é interativa, orientada a cardápios e independente da configuração de "hardware" usada no sistema. Esta interface foi implementada com base no trabalho de Osório /OSO 87/, e fornece um conjunto de funções que implementam características desejáveis para o EDGAR:

. interface em alto nível com o conjunto de dispositivos previstos na configuração de "hardware" proposta no capítulo 4, ou seja, "mouse", mesa digitalizadora, vídeo gráfico colorido e traçador gráfico;

. especificação e gerência automática de cardápios hierárquicos interativos;

. traçado de primitivas geométricas, pontos, retas e retângulos, em até duas janelas simultâneas, de forma independente de dispositivo;

. apontamento interativo de coordenadas;

. mapeamento independente de coordenadas nas duas janelas gráficas disponíveis.

6.3.3 Relação com descrições AMPLO

Esta subseção detalha alguns aspectos do estilo de projeto descendente usando o EDGAR. A escolha deste estilo é realizada pelo projetista através da interface

a existência de uma "netlist" validada por simulação no BD AMPLO, a descrição lógica correspondente ao circuito em questão é carregada pelo editor em uma tabela interna de elementos e conexões entre estes elementos. Para tanto, é realizada, pelo EDGAR, uma operação de pós-processamento da saída do Construtor de Modelos (vide seção 6.2) do projeto AMPLO, criando um formato interno adequado ao tratamento das operações de posicionamento de células e traçado de conexões. Esta tabela associa, a cada elemento primitivo do modelo simulável, uma lista de pontos de entrada e/ou saída. Cada um dos pontos de E/S deve, em tempo de posicionamento da célula física correspondente ao elemento lógico em questão, ter associada uma coordenada física na planta baixa do pré-difundido. Este relacionamento entre pontos de E/S de uma descrição NILO e coordenadas físicas em matrizes e em células de biblioteca é usado como guia na construção das conexões. O usuário pode assim, durante o passo de traçado de conexões, verificar a coerência entre as conexões da descrição lógica e as conexões físicas do presentes no "layout" de pré-difundidos.

O processo de posicionamento, traçado de conexões e verificação de coerência entre descrições físicas e lógicas é interativo. O estado de um "layout", do ponto de vista destes três aspectos, pode ser congelado ou recuperado a qualquer momento durante o processo de edição.

6.3.4 Criação de células de biblioteca

A confecção de novas células de biblioteca, uma atividade sempre necessária durante o estabelecimento de novas tecnologias e novos tipos de matrizes de pré-difundidos associadas, é obtida hoje com uma ferramenta derivada do programa Microeditor /JAC 86/ conforme mencionado anteriormente neste trabalho. Esta versão

alterada permite a visualização de camadas não personalizáveis da matriz, formando um "mosaico" estático, sobre o qual o projetista ou o fabricante cria um padrão de personalização, que constituirá a nova célula de biblioteca.

Atualmente, a inexistência, no âmbito do CPGCC/UFRGS, de uma forma de descrever de maneira mais completa uma célula de biblioteca, limita o formato de saída do editor de células a uma descrição puramente geométrica das mesmas. Eventualmente, com a definição de uma linguagem e compilador associados para descrever os demais aspectos relevantes de células (vide capítulo 7), estas informações serão agregadas ao formato de saída da versão alterada do Microeditor.

6.3.5 Estruturas de dados internas

Em /CAL 87b/ é apresentada uma breve revisão e um conjunto de referências a publicações onde estruturas de dados para armazenamento de informações de máscaras de circuitos integrados são discutidas.

Uma característica peculiar do EDGAR é a unidade mínima de armazenamento, como será descrito a seguir. Na maioria das ferramentas disponíveis no meio acadêmico ou industrial, a unidade básica é uma figura geométrica, tipicamente retângulos associados a um dos níveis de máscaras. Mais raramente, as ferramentas empregam polígonos de forma arbitrária como unidade. Esta escolha deriva do fato das ferramentas serem, em geral, originárias da adaptação de ferramentas voltadas para a geração de máscaras de circuitos dedicados.

Levando em consideração o maior nível de abstração associado a CIs pré-difundidos, optou-se por uma

unidade básica mais especializada, a conexão. O EDGAR opta pelo armazenamento do "layout" em termos de conexões entre pontos de uma matriz de pré-difundido.

Ao invés de listas de retângulos, o EDGAR guarda conexões, sendo estas conexões definidas, internamente, por conjuntos de retângulos que as implementam, apenas por simplicidade de mapeamento para descrições geométricas textuais. O número de retângulos e camadas envolvidas não possui qualquer relação com a conexão em si, dependendo apenas da tecnologia, da matriz e da técnica de traçado empregada. Na figura 6.12 aparece um exemplo de conexão e a estrutura de dados usada para representá-la internamente no EDGAR.

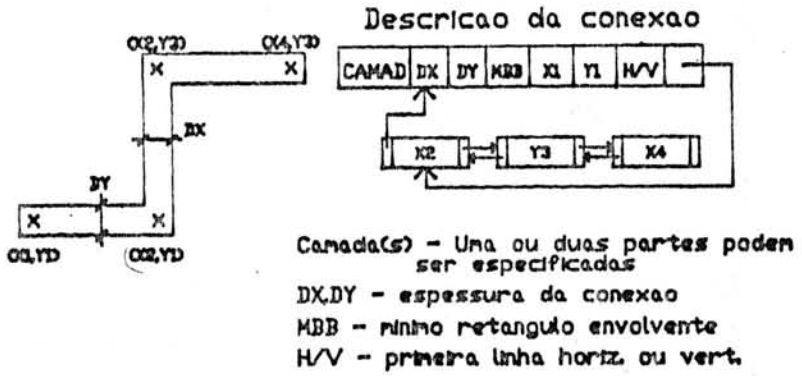


Figura 6.12 - Conexão - a unidade básica de "layout" no EDGAR

Foi mencionado que o EDGAR lida com descrições hierárquicas de "layout". Estas descrições são apresentadas a seguir.

A estrutura geral do pré-difundido constitui-se de duas porções distintas: a porção pré-difundida e a porção personalizável. A primeira é imutável ao longo do processo de edição, sendo então montada internamente como uma estrutura hierárquica estática de rápido acesso para

operações de consulta. A porção personalizável é dinâmica. A estrutura de dados interna escolhida para organizar ambas porções foi a árvore de pesquisa binária /BEN 79a/ /BEN 79b/. A escolha desta estrutura derivou de vários fatores, entre eles:

. a simplicidade de mapeamento de uma hierarquia de profundidade e ramificação lateral arbitrárias para uma estrutura do tipo árvore binária cria a possibilidade de uso de algoritmos eficientes para as operações mais comuns de inserção, remoção e caminhamento presentes nas operações de edição interativa de máscaras;

. a estabilidade dos algoritmos constantes na literatura para esta estrutura de dados revela sua maturidade para aplicações onde a gerência de hierarquias é fator preponderante;

. a semelhança de operações de pesquisa em bancos de dados com operações de pesquisa em representações geométricas bidimensionais /BEN 79a/ garante a validade do uso desta estrutura em um editor de máscaras.

Voltando ao assunto conexões, editores tradicionais procuram compensar a falta de informação semântica associada às listas de retângulos mediante tratamento diferenciado para retângulos constituintes de conexões, usando conceitos auxiliares tais como nós e fronteiras /JAC 88b/. O EDGAR, por sua vez, emprega listas de retângulos para a porção estática da estrutura e o conceito de conexões para a porção dinâmica.

Pode-se assim afirmar que o EDGAR é um editor de conexões, antes de ser um editor de máscaras. Isto decorre do fato de se estar lidando com circuitos pré-difundidos, onde a personalização nada mais é que a realização de conexões entre elementos previamente dispostos sobre um substrato semicondutor.

7 DIREÇÕES FUTURAS DO MÉTODO CIPREDI E CONCLUSÕES

Neste capítulo, apresenta-se um conjunto de diretivas para implementações futuras de conceitos e ferramentas no método CIPREDI. Estas diretivas foram elaboradas a partir de tópicos discutidos nos capítulos 4, 5 e 6 deste trabalho, em sistemas de PAC recentes voltados para CIs pré-difundidos e nas condições de contorno encontradas no ambiente proposto para a instalação do método.

A seção 7.1 apresenta o ambiente mínimo previsto pelo CIPREDI para a obtenção de um ambiente produtivo. A seção 7.2 discute a evolução do método segundo vários aspectos. A seção 7.3 mostra as conclusões do presente trabalho.

7.1 Ambiente Mínimo

A implementação descrita nos capítulos 4, 5 e 6 está longe de constituir um ambiente completo de projeto de pré-difundidos. Nesta seção é apresentada uma estratégia de prioridades para a obtenção de um ambiente produtivo mínimo, baseado no estado e nas necessidades atuais do método CIPREDI. Cada aspecto do método é discutido em separado nas subseções a seguir.

7.1.1 Teoria do sistema alvo

A principal necessidade neste aspecto é a confecção de uma biblioteca de células mais poderosa, de forma a reduzir o envolvimento dos projetistas no processo de geração de máscaras.

A evolução da biblioteca de células é prevista em

dois aspectos. Primeiro, a confecção de circuitos leva à necessidade de criação de células não disponíveis na biblioteca atual ou a alteração de células para adequá-las às condições de um projeto. Esta tarefa termina por expandir a biblioteca com novas células e com novas versões de uma célula.

Segundo, a implementação de ferramentas de geração automática de máscaras (particionador, posicionador e roteador) deverá formalizar a topologia e a interface das células da biblioteca, impondo restrições no tamanho e na especificação destas células. Esta segunda evolução deve conduzir a uma limitação de tamanho máximo para as células com "layout" pré-estabelecido, dependendo dos requisitos encontrados na implementação das ferramentas citadas. Uma abordagem similar à da AMI INC /GOU 85/, citada no capítulo 3, deverá resultar desta evolução, ou seja, uma hierarquia de células onde as folhas correspondem a células simples, com "layout" pré-definido, e células complexas, cujo "layout" é gerado automaticamente, mediante uso de células simples.

7.1.2 Modos de especificação

Dentro deste aspecto, divisa-se três metas a serem atingidas, visando satisfazer as necessidades de um ambiente mínimo:

. especificação de uma linguagem de descrição de matrizes e tecnologias;

. especificação de uma linguagem de descrição de células de biblioteca;

. especificação de formatos de saída textuais padronizados para intercâmbio de informações com o fabricante.

A escolha das duas primeiras linguagens deriva do fato do método CIPREDI prever o uso de duas bibliotecas de projeto físico:

. a biblioteca de matrizes e tecnologias;
. a biblioteca de células.

A primeira linguagem, de especificação de matrizes e tecnologias, é exclusiva para pré-difundidos. Será, a princípio, textual, e deverá permitir a especificação de informações tais como:

. tecnologia usada;
. nome da família;
. camadas personalizáveis (para cada família);
. distribuição geométrica dos pinos (para cada matriz);
. distribuição geométrica da(s) célula(s) de base (para cada matriz);
. canais de interconexão (para cada célula de base);
. estrutura geométrica dos pinos e roteamento das linhas de alimentação;
. topologia da(s) célula(s) de base;
. topologia e geometria de blocos especiais, bem como função destes;
. planta baixa (para cada matriz);
. etc.

Eventualmente, a especificação da matriz poderá ser obtida mediante uma ferramenta de edição gráfica interativa, com possibilidade de especificação textual de atributos não-gráficos para uma descrição completa de matrizes e tecnologias de uma dada instalação.

Na prática, descrições separadas para tecnologias e matrizes devem ser armazenadas, uma vez que uma

tecnologia é empregada em diversas famílias, e cada família suporta diversas matrizes distintas.

A segunda linguagem é uma necessidade do projeto de CIs em geral, não só dos pré-difundidos. Devido a esta característica, ela deve ser definida levando em conta as necessidades do projeto LCCI /CUR 86/ como um todo. Em princípio, esta linguagem será parte textual, parte gráfica. Isto torna-se necessário, pois uma biblioteca de células deve comportar informação adicional às descrições das geometrias de máscaras das células. No capítulo 3, seção 3.1, foi apresentado um exemplo típico de informações comumente associadas às células de biblioteca.

Um ponto importante na biblioteca de células é o estabelecimento de uma relação entre descrições físicas (máscaras) e descrições mais abstratas. Isto deve ser obtido mediante a inclusão, na biblioteca de células, de parâmetros que referenciem as descrições abstratas do BD AMPLO, conforme descrito na proposta de documentação contida no anexo 2 (itens 3.8 a 3.10).

Pelos motivos expostos na subseção 4.2.2, o método CIPREDI deve suportar tradutores de informações para formatos textuais de emprego corrente. Descrições de máscaras em CIF, LUCIE /PAI 85/ e RS são um conjunto inicial necessário, dentro do ambiente do CPGCC/UFRGS. Linguagens de descrição mais genéricas, como EDIF /EDI 84/, devem ter a viabilidade de seu uso avaliada em passos posteriores de implementação.

Deve-se perceber que os itens listados acima introduzem graus de liberdade no sistema, aumentando sua adaptabilidade a diferentes ambientes. A subseção 7.1.4 a seguir sumariza as ferramentas necessariamente associadas às linguagens de descrição apresentadas aqui.

7.1.3 Processos de projeto

A consideração deste aspecto do método para se alcançar um ambiente mínimo não é necessária, pois os requisitos envolvidos estão completamente especificados. Contudo, deve-se salientar que o cuidado em manter as características de adaptabilidade do sistema deve permear a implementação de novas ferramentas de projeto.

7.1.4 Infraestrutura de projeto

Este é o aspecto mais importante na obtenção de um ambiente mínimo, e deve ser restrito à implementação de ferramentas de "software", já que o "hardware" está definido, conforme visto no capítulo 4.

Em primeiro lugar, a especificação de linguagens de descrição proposta na subseção 7.1.2 implica a existência de tradutores das linguagens para o formato interno dos bancos de dados usados no método CIPREDI. A prioridade deve ser máxima para as ferramentas que tratam das bibliotecas de projeto físico, ou seja, a biblioteca de matrizes e tecnologias e a biblioteca de células.

Em conformidade com os objetivos de automação previstos no método CIPREDI, ao estabelecimento definitivo dos formatos das bibliotecas de projeto físico, deve seguir-se a implementação de uma ferramenta de traçado automático de interconexões a partir de um "netlist" gerado pelo editor e validado pelo simulador do nível lógico do AMPLO. Após, ou de forma concomitante, deve ser desenvolvida uma ferramenta de particionamento e posicionamento de módulos do eixo físico do diagrama Y.

O desenvolvimento de ferramentas de validação de

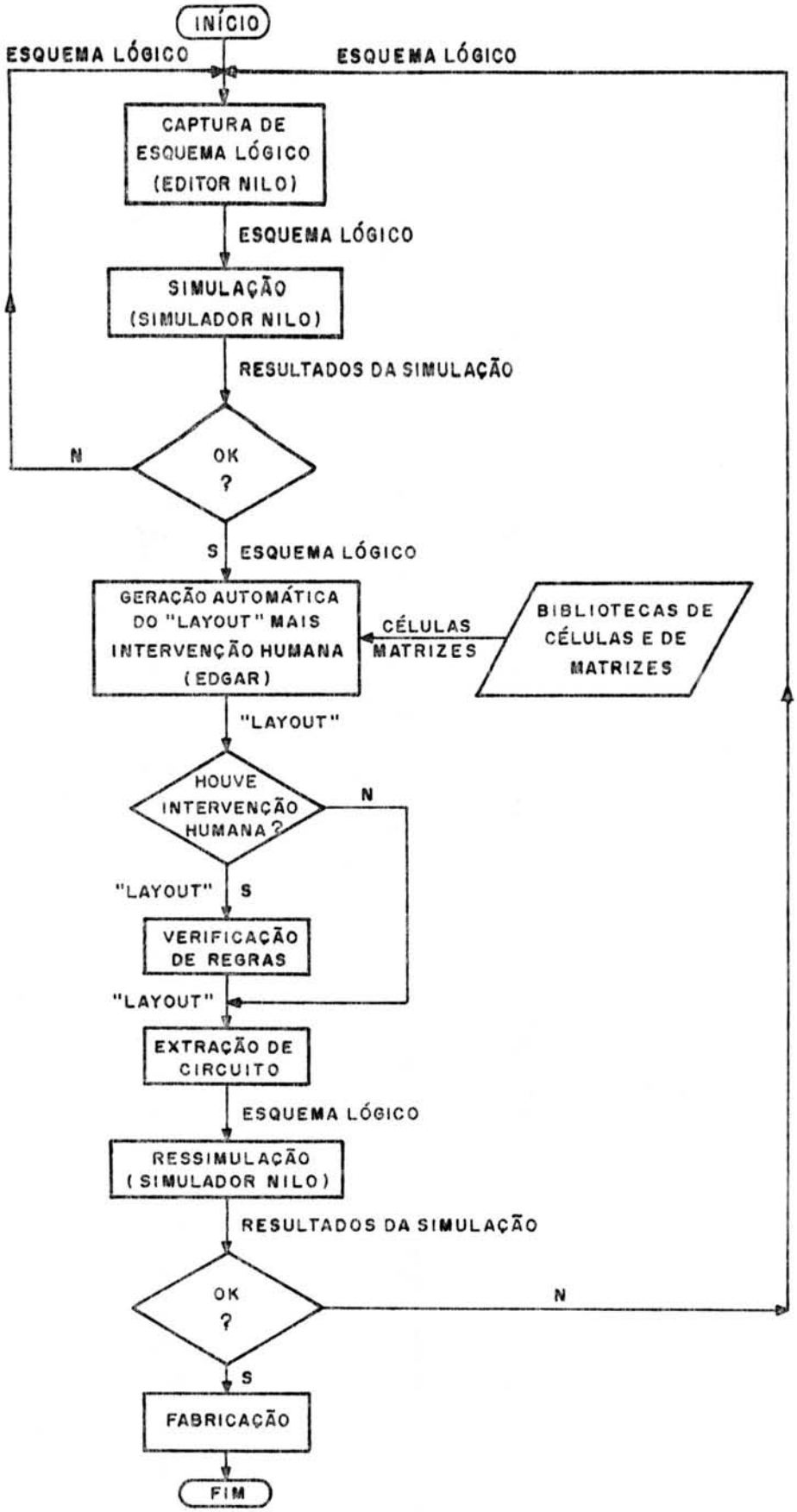
descrições do eixo físico pode ser feito em paralelo com as tarefas descritas no parágrafo anterior, vista a sua simplicidade no projeto de pré-difundidos.

Ferramentas de verificação de regras geométricas e extração de circuitos devem ser elaboradas. A primeira destina-se a validar as máscaras geradas do ponto de vista de violação das regras de projeto do fabricante, tais como espaçamento mínimo entre linhas de metal, distância mínima entre contatos, separação máxima entre duas polarizações de substrato, etc. Devido à automação proposta nas ferramentas de geração de máscaras, esta ferramenta tem baixa prioridade de implementação, uma vez que o "layout" gerado sem intervenção humana é correto por construção. A possibilidade de intervenção manual, contudo, não permite descartar de todo esta ferramenta.

A segunda ferramenta é responsável por fechar o laço iterativo de projeto, daí a alta prioridade de sua implementação. Sua função é obter uma descrição a nível lógico do circuito, a partir das máscaras, além de extrair parâmetros de atraso devidos às interconexões longas. O diagrama lógico assim obtido pode ser ressimulado e o seu comportamento comparado com os resultados da simulação inicial. O ciclo de projeto previsto pelo método CIPREDI aparece detalhado no fluxograma da figura 7.1.

A estrutura do método, após os passos de implementação delineados no presente capítulo, deverá ser aquela mostrada no diagrama de Gajski da figura 7.2.

Note-se a mudança de papel do EDGAR no esquema da figura 7.2. Após estarem disponíveis as ferramentas de geração automática das máscaras, o EDGAR existirá apenas para permitir a intervenção humana no processo de criação do "layout".


INÍCIO
ESQUEMA LÓGICO
ESQUEMA LÓGICO
CAPTURA DE ESQUEMA LÓGICO (EDITOR NILO)
ESQUEMA LÓGICO
SIMULAÇÃO (SIMULADOR NILO)
RESULTADOS DA SIMULAÇÃO
OK ?
N
S
ESQUEMA LÓGICO
GERAÇÃO AUTOMÁTICA DO "LAYOUT" MAIS INTERVENÇÃO HUMANA (EDGAR)
CÉLULAS MATRIZES
BIBLIOTECAS DE CÉLULAS E DE MATRIZES
"LAYOUT"
HOUVE INTERVENÇÃO HUMANA ?
N
"LAYOUT"
S
VERIFICAÇÃO DE REGRAS
"LAYOUT"
EXTRAÇÃO DE CIRCUITO
ESQUEMA LÓGICO
RESSIMULAÇÃO (SIMULADOR NILO)
RESULTADOS DA SIMULAÇÃO
OK ?
N
S
FABRICAÇÃO
FIM

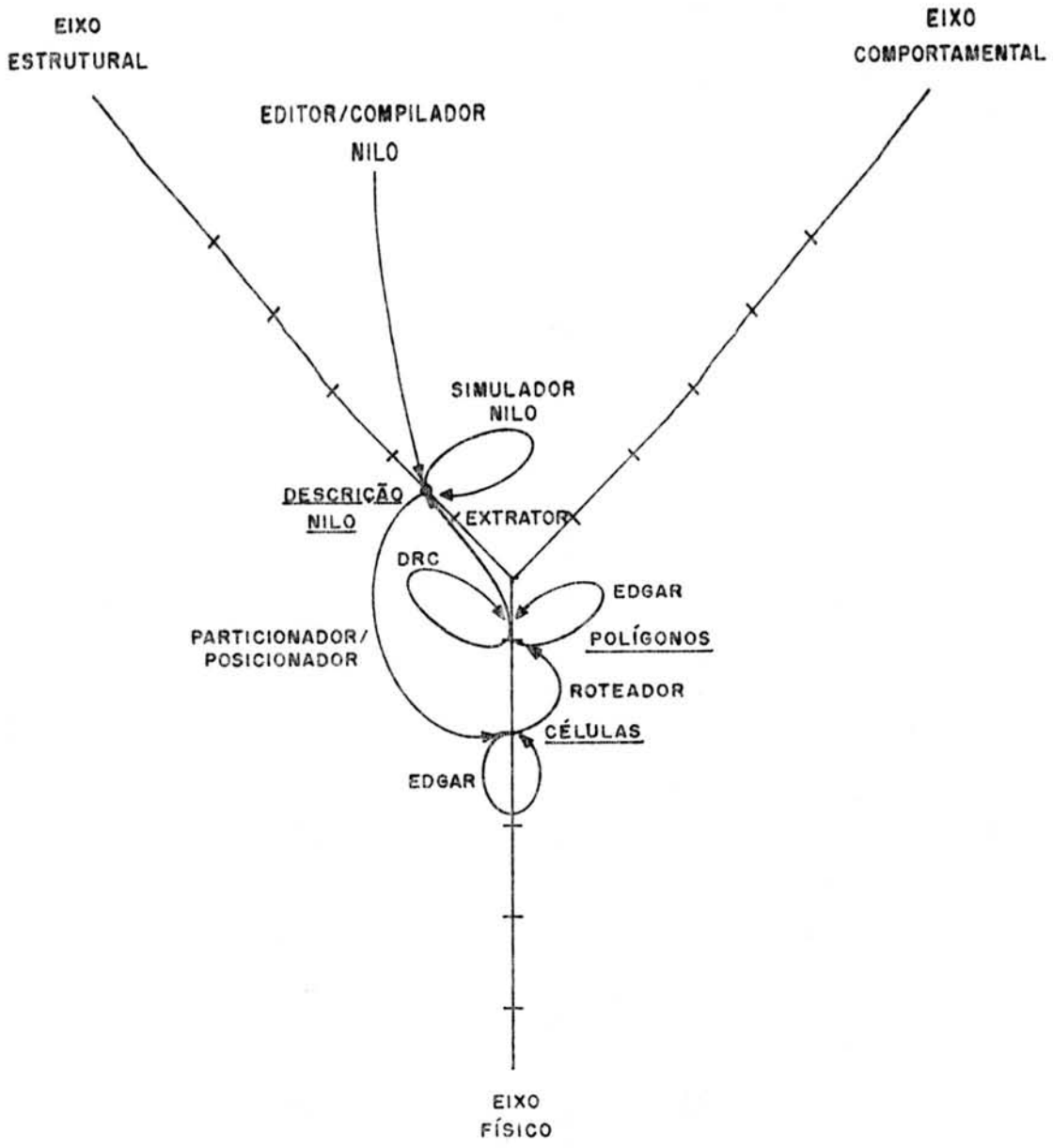


Figura 7.2 - Processo de projeto no ambiente mínimo

A tabela 7.1 mostra as tarefas destinadas a completar o ambiente mínimo para projeto de pré-difundidos. A tabela está organizada por prioridade de execução.

Tabela 7.1 - Tarefas e prioridades de implementação para o ambiente mínimo

TAREFA	PRIORIDADE
- MELHORAMENTO DA BIBLIOTECA DE CÉLULAS E RELACIONAMENTO COM O BD AMPLO	1
- ESPECIFICAÇÃO DE LINGUAGEM E TRADUTOR PARA DESCRIÇÕES DE MATRIZES E TECNOLOGIAS	1
- ESPECIFICAÇÃO DE LINGUAGEM E TRADUTOR PARA CÉLULAS DE BIBLIOTECA	1
- TRAÇADOR AUTOMÁTICO DE INTERCONEXÕES	2
- EXTRATOR DE CIRCUITOS	2
- PARTICIONADOR E POSICIONADOR DE MÓDULOS	3
- VERIFICADOR DE REGRAS DE PROJETO	4

7.2 Evolução

O sistema descrito na seção anterior deve suprir as principais necessidades de um ambiente de projeto pouco exigente. Usuários sofisticados, interessados em tecnologia de ponta e projetos com alto grau de complexidade algorítmica necessitam de ferramentas mais elaboradas.

Nesta seção são apresentadas diretivas para a implementação de um ambiente mais poderoso de auxílio ao projeto.

7.2.1 Níveis superiores de abstração

Para que ferramentas avançadas de auxílio ao projeto sejam efetivas no tratamento do problema dos pré-difundidos, elas devem lidar com descrições dotadas de alto grau de abstração, como delineado no capítulo 4. O eixo comportamental do diagrama Y é o mais indicado para situar estas descrições.

O método deve então estudar o uso de linguagens de descrição que sirvam de entrada para estas ferramentas. Neste estudo, os níveis de abstração abordados devem ser:

. nível lógico - equações booleanas;

. nível bloco funcional - transferência entre registradores;

. nível algorítmico - algoritmos.

Descrições comportamentais nestes níveis são a entrada para as ferramentas de síntese de descrições estruturais correspondentes, ou descrições com nível de abstração mais baixo.

Uma questão de interesse é a aplicação, em pré-difundidos, de ferramentas que trabalhem com as linguagens KAPA e LAÇO do projeto AMPLO. Estas linguagens estão definidas e adequam-se à necessidade de abstração apontada no primeiro parágrafo desta subseção.

7.2.2 Síntese automatizada

Conforme visto no capítulo 3, qualquer sistema produtivo de projeto de pré-difundidos conta com ferramentas automatizadas para a geração de máscaras (particionamento, posicionamento e traçado de conexões). Ao lado destas, são cada vez mais comuns as ferramentas de síntese que partem de descrições estruturais ou comportamentais de níveis superiores de abstração e geram a entrada das ferramentas de geração de máscaras /GRE 82/ /NEW 86/.

Dentro do escopo estabelecido no capítulo 4, devem ser estudadas e implementadas três ferramentas de síntese automatizada, inicialmente:

. sintetizador de parte operativa;
. sintetizador de parte de controle;
. otimizador lógico.

Estas ferramentas constituem um ambiente mínimo adequado para o tratamento de descrições de projeto com alto nível de abstração. As linguagens de entrada para estas ferramentas devem ser comportamentais, lidando com primitivas do nível lógico e/ou do nível bloco funcional, pelo menos.

7.2.3 Verificação de projeto e projeto visando o teste

No capítulo 3 foi mostrado que as ferramentas de verificação de projeto desempenham papel fundamental na concepção de CIs pré-difundidos.

No capítulo 4 o simulador NILO foi apontado como uma primeira forma de abordar o problema do teste de pré-difundidos.

O método CIPREDI deve avaliar a viabilidade de uso do simulador apresentado neste trabalho do ponto de vista de teste de projetos, propondo eventualmente o refinamento do modelo de falhas disponível.

Adicionalmente, a estrutura do método CIPREDI determina que pelo menos três ferramentas devem ser analisadas para possível implementação:

. simulador de falhas específico;
. extrator de padrões de teste;
. analisador de cobertura de falhas.

7.2.4 Integração do ambiente

No capítulo 4 foi apresentada a estrutura de integração das ferramentas do método CIPREDI através de dois bancos de dados, um responsável pela manipulação de informações estruturais e comportamentais e o outro encarregado das descrições físicas.

A experiência adquirida com esta forma dual tem revelado, contudo, um problema crítico: o do mapeamento de descrições de um banco de dados para o outro. A garantia de consistência entre informações correspondentes contidas em bancos de dados distintos, revela-se uma tarefa complexa e demorada. Além disso, a separação dos bancos de dados exige o uso de redundância em ambos, o que não é uma característica desejável.

O método CIPREDI deve acessar a questão de uso de um banco de dados integrado, capaz de lidar simultaneamente com descrições físicas, estruturais e comportamentais.

7.2.5 Interação com fundições de silício

Até o momento, o método proposto neste trabalho tem sido desenvolvido internamente ao CPGCC/UFRGS. Não existiu, neste primeiro passo, o envolvimento seja de fundições de silício, seja de instituições de projeto de pré-difundidos, exceto a nível de fornecimento de informações verbais e material bibliográfico.

A decisão de assumir esta postura foi proposital, baseada nos objetivos iniciais do método. A vinculação direta com meios produtivos dificilmente permitiria a proposta de um ambiente com o grau de flexibilidade e adaptabilidade sugeridos.

Por outro lado, os passos seguintes do método não podem privar-se de uma interação estreita com ambientes de projeto e fabricação de pré-difundidos, sob pena de acarretar a desvinculação do método da realidade industrial. Afinal, é aos meios de produção, em última análise, que a proposta deste trabalho se destina.

7.2.6 Novos circuitos

O capítulo 5 apresentou dois circuitos implementados ao longo da execução deste trabalho. Este é um conjunto reduzido para validar um método de projeto. Logo, ao longo da implementação dos passos posteriores do método CIPREDI, deve ser efetuada a confecção de novos circuitos, visando avaliar as virtudes da célula de base proposta, levantar necessidades de ferramentas adicionais e validar as ferramentas implementadas.

Os circuitos a serem implementados devem conter elementos capazes de fornecer parâmetros de medida para itens como:

. capacidade de geração de circuitos seqüenciais;
. capacidade de geração de blocos de memória;
. capacidade de geração de lógica combinacional regular;
. capacidade de geração de circuitos de alta complexidade.

7.3 Conclusões Finais

Este trabalho apresentou uma proposta de método de projeto para circuitos integrados no estilo pré-difundido. O método encontra-se nos passos iniciais de desenvolvimento, mas sua estrutura geral está estabelecida.

bem como as diretrizes para sua evolução.

A ótica pragmática apresentada no capítulo 3 revelou-se bastante útil na formalização dos conceitos fundamentais para a proposta do método.

A tarefa de especificação de um método de projeto para circuitos integrados revelou-se extensa, requerendo a colaboração de muitos trabalhos adicionais para alcançar uma estrutura estável e uma implementação madura de ferramentas de auxílio ao projeto.

A maneira mais simples e mais segura de avaliar os acertos e desvios do método proposto é confeccionar circuitos, usando as técnicas e ferramentas desenvolvidas. Atualmente, dentro do CPGCC/UFRGS, isto é feito mediante trabalhos de alunos em disciplinas de projeto de CIs e trabalhos de pesquisa conduzidos pelo grupo de Microeletrônica.

Integrar as ferramentas desenvolvidas é tarefa das mais árduas dentro da elaboração do método. Durante a implementação destas, deve ser considerada a compatibilidade dos formatos e descrições usados com os formatos dos bancos de dados, bem como a eventual evolução dos bancos de dados para uma solução integrada única.

Mesmo o sistema mínimo proposto neste capítulo não aborda, ainda, o problema do teste de circuitos pré-difundidos. Em passos futuros, a testabilidade de "gate arrays" e ferramentas que facilitem a abordagem deste aspecto devem ser, necessariamente, levados em conta.

Considerações sobre fundições de sílicio estão, necessariamente, dentro da perspectiva do método. O contato mais estreito com estas instituições deve ser buscado

O contato com empresas nacionais envolvidas no projeto de pré-difundidos é um primeiro passo nesta direção. A interação com a, até agora, única fundição de silício do País deve ser buscada para verificar a viabilidade das técnicas e ferramentas propostas dentro do ambiente CIPREDI, tendo em vista a situação nacional.

ANEXO 1

EXEMPLO DE DOCUMENTOS FORMAIS USADOS NO INTERCÂMBIO DE INFORMAÇÕES ENTRE O CLIENTE E O FORNECEDOR DE PRÉ-DIFUNDIDOS (Extraído de /GEN 83/)

PIN DESCRIPTION FORM

Package size: _____pins

Name	Type	Function	Pin # (Only if necessary)

ELECTRICAL SPECIFICATION FORM

Customers should fill in this form within the limitations listed on the Intersil Data Sheet (Section 2.3).

1. Operating Characteristics

Absolute Maximum Rating (Referenced to V_{SS})

Parameter	Symbol	Limits	Unit
DC Supply Voltage	V_{DD}		V
Input Voltage	V_I		V
DC Input Currents	I_I		mA
Storage Temperature Range (Ceramic)	T_{STG}		°C
Storage Temperature (Plastic)	T_{STG}		°C

2. Operating Conditions

Parameter	Symbol	Limits	Unit
DC Supply Voltage	V_{DD}		V
Operating Ambient Temperature Range	T_A		°C

ELECTRICAL SPECIFICATION FORM
DC Characteristics

Use this form to specify a gate array with a TTL interface

Operating Conditions: Voltage_____V ± ____;Temperature_____°C to____°C

Symbol	Parameter	Condition	Limit	Units	Min/Max
I_{DD}	Standby current	All inputs at V_{DD} or V_{SS}		mA	Max
V_{OL}	Low Level Output Voltage	I_{OL} = ___mA		V	Max
V_{OH}	High Level Output Voltage	I_{OH} = -___uA		V	Min
V_{IL}	Low Level Input Voltage	---		V	Max
V_{IH}	High Level Input Voltage	---		V	Min
I_{IN}	Input Leakage Current (any pin)	V_{IN} = 0 or V_{DD}		uA	Max
I_{OZ}	Three State Output Leakage Current (any pin)	V_O = 0 or V_{DD}		uA	Max
I_{OS}	Short Circuit Output Current (any pin)	V_O = 0 or V_{DD}		mA	Max

ELECTRICAL SPECIFICATION FORM
DC Characteristics

Use this form to specify a gate array with a CMOS interface

Operating Conditions: Voltage_____V ± ____;Temperature_____°C to____°C

Symbol	Parameter	Condition	Limit	Units	Min/Max
I_{DD}	Standby current	All inputs at V_{DD} or V_{SS}		mA	Max
V_{OL}	Low Level Output Voltage	I_{OL} = 1uA		V	Max
V_{OH}	High Level Output Voltage	I_{OH} = -1uA		V	Min
V_{IL}	Input Low Voltage	---		V	Max
V_{IH}	Input High Voltage	---		V	Min
I_{OL}	Output Low (Sink) Current	V_{OL} = ___V		mA	Min
I_{OH}	Output High (Source) Current	V_{OH} = ___V	-	mA	Max
I_{IN}	Input Leakage Current (any pin)	V_{IN} = 0 or V_{DD}		uA	Max
I_{OZ}	Three State Output Leakage Current (any pin)	V_O = 0 or V_{DD}		uA	Max
I_{OS}	Short Circuit Output Current (any pin)	V_O = 0 or V_{DD}		mA	Max

ELECTRICAL SPECIFICATION FORM

MISCELLANEOUS PARAMETERS

Input Capacitance, CIN _____

Powerup Reset Rise Time _____

Input Rise Time _____

DC Standby Current _____

Power Dissipation _____

CELL COUNT FORM

FORM #_____

MODULE_____

CHIP INPUTS TO MODULE: _____
CHIP OUTPUTS FROM MODULE: _____
INTERNAL INPUTS TO MODULE: _____
INTERNAL OUTPUTS FROM MODULE:' _____

Type	Description	Worst Case Fanout	Count	Cells per Macro	Total Array Cells	Total I/O Cells

PAGE TOTAL: ARRAY CELLS _____

PAGE TOTAL: I/O CELLS _____

ARRAY SELECTION FORM

Sheet #	Worst Case Fanout	Array Cells	I/O Cells	Sheet #	Worst Case Fanout	Array Cells	I/O Cells
SUBTOTALS							

TOTAL ARRAY CELL COUNT _____

TOTAL I/O CELL COUNT _____

ARRAY SELECTION BASED ON ARRAY CELLS _____

ARRAY SELECTION BASED ON I/O CELLS _____

FINAL ARRAY SELECTION _____

PACKAGE SELECTION FORM

Select a package by checking the corresponding box.

PACKAGE TYPE

Number of Pins	Plastic DIP	CerDIP	Side Braze DIP	Leadless Chip Carrier	Pin Grid Array
8	408 ☐	408 ☐	408 ☐		
14	408 ☐	408 ☐	408 ☐		
16	408 ☐	408 ☐	408 ☐		
18	408 ☐	408 ☐	408 ☐		
20		408 ☐	408 ☐		
24	408 ☐	408 ☐	408 ☐		
	756 ☐	756 ☐	756 ☐		
	1500 ☐	1500 ☐	1500 ☐		
28	408 ☐	408 ☐	408 ☐		
	756 ☐	756 ☐	756 ☐		
	1500 ☐	1500 ☐	1500 ☐		
40	408 ☐	408 ☐	408 ☐		
	756 ☐	756 ☐	756 ☐		
	1500 ☐	1500 ☐	1500 ☐		
44				756 ☐	
				1500 ☐	
48			756 ☐		
			1500 ☐		
52				1500 ☐	
68				1500 ☐	1500 ☐

Alternate Package Selection _____
(confirm selection with General Electric/Intersil representative)

ANEXO 2

PROPOSTA DE DOCUMENTAÇÃO PARA A BIBLIOTECA DE CÉLULAS DE PRÉ-DIFUNDIDOS

1. Nome - nome através do qual a célula é referenciada e nome do arquivo onde sua descrição se encontra.

2. Função - descreve, em poucas palavras o comportamento da célula.

3. Características - conjunto de atributos topológicos e elétricos da célula.

3.1. Tecnologia - tipo de processo usado para a implementação da célula.

3.2. Regras - conjunto detalhado de regras geométricas fornecidas pelo fabricante e fabricante que as forneceu.

3.3. Célula de base - tipo de célula de base usada na implementação.

3.4. Tamanho e área - número de células de base gastas e quantidade de portas equivalentes consumidas de matriz, além do mínimo retângulo envolvente da célula.

3.5. Parâmetros elétricos - conjunto de características obtidas por teste, simulação ou estimativas de valores para a célula, indicando a fonte da informação.

3.5.1. Consumo - extraído da simulação elétrica, normalmente dado em termos de corrente.

3.5.2. Atraso intrínseco - extraído da simulação elétrica, conforme caracterização. Fornecido com base em uma carga com capacitância unitária na saída. Um valor para cada saída.

3.5.3. Equação do atraso de propagação - equação de cálculo estimado do atraso de propagação para cada saída, usada quando cargas capacitivas não unitárias estão conectadas na saída.

3.5.4. Capacitância de entrada - estimada

conforme caracterização. Fornecida como um multiplicador inteiro de uma carga capacitiva unitária, sendo um valor para cada entrada da célula.

3.5.5. Resistência de saída - resistência de "pull-up" ou "pull-down" estimadas.

3.5.6. "Fan-out" - capacidade de corrente em ns/pF. Extraído da simulação elétrica conforme caracterização.

3.5.7. Largura de pulso mínimo - estimada a partir do atraso intrínseco.

3.6. Topologia de E/S - nomes das entradas e saídas com a coordenada "lambda" do canto inferior da linha de metal correspondente a cada ponto de E/S, relativa ao canto inferior esquerdo do mínimo retângulo envolvente da célula.

3.7. Canais de transparência - informa os canais disponíveis para o cruzamento de conexões do roteamento global através da região ocupada pela célula. As transparências podem ser horizontais ou verticais.

3.8. Agência AMPLO correspondente - informa qual o nome (texto com até 12 caracteres) da agência correspondente a esta célula de biblioteca no BD AMPLO.

3.9. Alternativa da Agência AMPLO - número da alternativa da agência no BD AMPLO.

3.10. Versão da Alternativa de Agência AMPLO - número da versão da agência no BD AMPLO.

4. Descrição - explicação do funcionamento da célula, tipo de lógica, exigências de condições especiais para funcionamento, etc.

5. Diagrama lógico - apresenta a descrição do funcionamento lógico através de figuras, diagramas de tempo, tabelas verdade, etc.

6. Diagrama elétrico - diagrama de transistores, bem como nome das entradas e saídas e também resistências e

capacitâncias significativas de linhas de conexão, se for o caso. Opcionalmente, podem ser colocados números dos nós correspondentes na simulação elétrica.

7. "Status" - estado da célula, segundo a seguinte convenção:

. checada - verificada por DRC e extração;

. testada - implementada e funcionando;

. não checada - não verificada nem por DRC, nem por extração;

. simulada - simulada eletricamente;

. uso geral - usada em mais de 3 projetos, após estar testada.

8. Resultado da simulação - listagem Spice que valida funcionamento elétrico e permite a noção das formas de onda, conforme caracterização. Se necessário, acrescentar o diagrama de tempos usado na simulação.

9. "Layout" - "hardcopy" da célula com nomes indicativos das linhas de alimentação, massa, sinais de entrada, saída e relógios.

9.1. Listagem em linguagem de descrição de máscaras - listagem com cabeçalho. No cabeçalho deve constar, pelo menos, o nome da célula e o autor. Pode ser em RS, LUCIE, CIF, EDIF ou outra linguagem utilizável.

10. Observações - todas as que o projetista julgar importante para os usuários da célula (por exemplo, tempo de "set-up", tempo de "hold", etc).

11. Meio de armazenamento - identificação do meio físico onde está a descrição das células, com as seguintes convenções:

. <nome>.CEL - arquivo gerado pelo Microeditor;

. <nome>.RS - arquivo com a descrição RS;

. <nome>.LUC - arquivo com descrição LUCIE;

. <nome>.CIR - arquivo com descrição SPICE;

. <nome>.DOC - arquivo contendo documentação.

12. Autor - nome do principal responsável pelo que está descrito pelos itens anteriores.

13. Data de criação.

14. Data da versão final.

15. Apêndice - breve histórico do desenvolvimento da célula, com as alternativas de topologia elétrica e o motivo da escolha implementada. Deve relatar também a data do projeto para o qual foi concebida a célula e sua motivação (projeto de circuito, projeto de disciplina, disciplina ou orientador, conforme o caso).

16. Alterações - data, motivo e implicações de cada alteração realizada na célula após o estabelecimento da documentação final.

ANEXO 3

PRINCIPAIS ROTINAS E ESTRUTURAS DE DADOS DO SIMULADOR MESTRE NILO

```
/*-------------------------- NILOAML.DOC -----------------
*
* ALGORITMO DO MESTRE LOGICO - ROTINAS
*
* POS-GRADUACAO EM CIENCIA DA COMPUTACAO - CPGCC/UFRGS
*
* GRUPO DE CAD PARA SISTEMAS DIGITAIS  -  SISTEMA AMPLO
*
* Autor : Ney Calazans
*
* Data de Inicio : 03 de Maio de 1988
*
* Ultima Modificacao : 19/05/88
*
*-------------------------------------------------------*/
```

```
/*-------------------------------------------------------
*
* NOME        : ative_mestre
*
* FUNCAO      : Implementar algoritmo principal do mestre,
*               varrendo a LEG desde o Tempo_Atual ate o
*               Tempo_Final ou ate que ocorra um breakpoint
*               ou uma interrupcao do usuario. Para cada
*               evento encontrado, dispara uma execucao do
*               escravo para trata-lo
*
* PARAMETROS  : NENHUM
*
*
* CHAMA       : nao_ha_breakpoint, nao_ha_interrupcao_do_
*               usuario, trata_variacao_sinal_interface,
*               ative_agencia, varie_ambiente_de_entrada,
*               simulacao_parou
*
* GLOBAIS     : Tempo_Atual, Tempo_Final
*
* RETORNA     : NADA
*
*-------------------------------------------------------*/
ative_mestre ()

  {
    enquanto ( Tempo_Atual <= Tempo_Final &&
               Nao_ha_breakpoint () &&
               Nao_ha_interrupcao_do_usuario () )
      {
        Tempo_Atual++ ;
        if ( existem eventos no slot atual da Lista
             de Eventos Geral )
            {
              for ( cada evento no slot atual com
                    tempo_de_evento == Tempo_Atual )
                  if ( evento for ativacao-de-Agencia )
                      {
                        - inserir indice da Agencia na
                          Lista de Ativacao ;
                        - eliminar evento do Laco delta-T ;

                      }
                  else  /* entao eh
                           variacao-de-entrada-primaria */
                      {
                        - trata_variacao_sinal_interface
                              ( sinal_do_evento,
                                novo_valor_evento ) ;
                        - eliminar evento do Laco delta-T ;

                      }

              for ( cada Agencia na Lista de Ativacao )
                  {
                    if ( Lista de Variacoes da Agencia -
                         LVA - nao esta vazia )
                        varie_ambiente_de_entrada
                              ( indice da Agencia,
                                Lista de Variacoes da Agencia
                                - LVA ) ;
                    else
                        ative_agencia ( indice da Agencia ) ;

                  }

            }

      }

    simulacao_parou ( tipo de motivo, motivo ) ;
```

```
/*-------------------------------------------------------
*
* NOME        : trata_variacao_sinal_interface
*
* FUNCAO      : Calcular novos valores para sinais de
*               interface, a partir de uma possivel
*               variacao causada seja por uma variacao
*               de entrada primaria, seja por uma variacao
*               de ambiente de saida de agencia. Caso sinal
*               com variacao possivel seja do tipo BUS,
*               realiza consenso instantaneo; caso
*               contrario, apenas propaga efeitos da
*               variacao para agencias com entrada no sinal.
*
* PARAMETROS  : sinal - identifica o sinal a tratar variacao
*               novo_valor - indica qual o suposto novo
*               valor de sinal
*
* CHAMA       : calcule_novo_valor_de_consenso,
*               constatei_erro
*
* GLOBAIS     : Tempo_Atual, Tempo_Final
*
* RETORNA     : NADA
*
*-------------------------------------------------------*/
```

```
trata_variacao_sinal_interface ( sinal, novo_valor )

  {
    if ( tipo do sinal for BUS )
       [
         - calcule_novo_valor_de_consenso ( sinal ) ;
         if ( novo_consenso == novo_valor )
            if ( novo_valor == valor_atual do sinal )
               - nao houve mudanca ;
            else
               for ( cada Agencia da Lista de Agencia/Nodo
                     de Destino do Sinal - LAND )
                   {
                     - inserir novo_valor na Lista de
                       Variacoes da Agencia - LVA ;
                     - inserir indice da Agencia na Lista
                       de Ativacao ;

                   }
         else
            if ( valor_atual do sinal == novo_valor )
               - constatei_erro ( absurdo: novo_consenso
                                  distinto do valor_atual
                                  e da variacao ) ;
            else /* variacao nao sera efetivada */
                {
                  - avisa responsavel pela tentativa
                    de mudanca ;
                  if ( novo_consenso == valor_atual
                       do sinal )
                     - nao houve mudanca ;
                  else
                     for ( cada Agencia da Lista de
                           Agencia/Nodo de Destino do
                           Sinal - LAND )
                         {
                           - inserir novo_consenso na Lista
                             de Variacoes da Agencia - LVA ;
                           - inserir indice da Agencia na
                             Lista de Ativacao ;
                         }

                }

       }
    else
       if ( novo_valor != valor_atual do sinal )
          for ( cada Agencia da Lista de Agencia/Nodo de
                Destino do Sinal - LAND )
              {
                - inserir novo_valor na Lista de Variacoes
                  da Agencia - LVA ;
                - inserir indice da Agencia na Lista de
                  Ativacao ;

              }
       else
          - nao ha mudanca ;

  }
```

```
/*-------------------------- NILOEDM.DOC -----------------
*
* ESTRUTURA DE DADOS DO MESTRE LOGICO
*
* POS-GRADUACAO EM CIENCIA DA COMPUTACAO - CPGCC/UFRGS
*
* GRUPO DE CAD PARA SISTEMAS DIGITAIS  -  SISTEMA AMPLO
*
* Autor : Ney Calazans
*
* Data de Inicio : 03 de Maio de 1988
*
* Ultima Modificacao : 19/05/88
*
*-------------------------------------------------------*/
/*-------------------------------------------------------
* Formato das Listas e Estruturas
*-------------------------------------------------------*/

Lista de Sinais de Subvetores - LSSV - Estrutura do Nodo

   - id interno
   - tipo
   - *Lista de ids de Agencia/Nodo de Origem   - LANO
   - *Lista de ids de Agencia/Nodo de Destino  - LAND
   - Flag de Monitoracao
   - Valor do Sinal


Lista de Variacoes da Agencia - LVA - Estrutura do Nodo

   - id interno
   - novo_valor


Lista de Eventos Geral - LEG - Estrutura do Nodo

   - tipo de evento
   - validade
   - tempo
   - uniao
     - id agencia              /* caso ativacao de agencia */
     - id sinal de subvetor    /* caso variacao de entrada
                                  primaria */
       novo_valor

Obs: - Caso o evento seja uma ativacao de agencia, usa-se
       o campo id agencia e o nodo nunca e desalocado,
       pois faz parte  do descritor da agencia, sendo
       estatico; caso contrario, trata-se de uma variacao
       de entrada primaria. Neste caso,usa-se a estrutura
       id sinal subvetor / novo valor.


Lista de Agencias - LA - Estrutura do Nodo

   - id interno
   - *Lista de Sinais de Subvetor associados
      a Nodos internos da agencia          - LSSN
   - *Lista de Variacoes da Agencia        - LVA
   - Nota de evento estatica para ativacao_de_Agencia
   - *Estrutura de Dados interna da Agencia

Obs: - A agencia com indice 0 corresponde a englobante,
       descrita em REDES.
     - A LSSN e uma lista de unsigned. O indice de cada
       elemento corresponde ao identificador do nodo
       associado ao sinal de subvetor cujo identificador e
       o elemento em questao.
```

```
Condicao - Lista de operadores e operandos

   - tipo_elemento  /* Operador/Operando */
   - tipo_operando  /* Sinal Interface/Nodo Interno */
   - uniao
     - operador     /* +, .,~, ( ou )  -- > corresponde a
       uma forma fatorada */
     - uniao        /* Operando */
        - id sinal de subvetor   /* Sinal de Interface */
        - id de agencia
          id de nodo          /* Sinal Interno */


Lista de Breakpoints - LBK - Estrutura do Nodo

   - id break
   - estado_break
   - *Condicao


Circuito - Estrutura

   - *Lista de Sinais de Subvetores - LSSV
   - *Lista de Agencias - LA
   - *Lista de Eventos Geral - LEG
   - *Lista de Breakpoints - LBK
   - *Lista de Ativacao - LATV

Obs: - A LATV e uma lista de unsigned. Cada elemento
       corresponde a um identificador de agencia a ser
       ativada no tempo atual.
```

BIBLIOGRAFIA

/ALE 85/ ALEXANDER, Bill. MOS and CMOS arrays. In: GATE Arrays : Design Techniques and Applications. New York. McGraw-Hill Book Company, 1985. Cap 3, p.80-285.

/AMI 82a/ AMERICAN MICROSYSTEMS. A training course: designing with gate arrays. Part 1. technology and circuit elements. VLSI Design, Palo Alto, 3(3):31-7, May/Jun. 1982.

/AMI 82b/ AMERICAN MICROSYSTEMS. A training course: designing with gate arrays. Part 2. structures and layout techniques. VLSI Design, Palo Alto, 3(4):31-7, Jul/Aug. 1982.

/AMI 82c/ AMERICAN MICROSYSTEMS. A training course: designing with gate arrays. Part 3. design considerations, testing packaging and interfacing with suppliers. VLSI Design, Palo Alto, 3(5):31-41, Sept/Oct. 1982.

/BEC 88/ BECKER, Karin. Manual de referência para uso das primitivas de acesso à base de dados AMPLO - Versão 2. a ser publicado.

/BEN 79a/ BENTLEY, Jon Louis & FRIEDMAN, Jerome H. Data structures for range searching. Computing Surveys, New York, 11(4):397-409, Dec. 1979.

/BEN 79b/ BENTLEY, Jon Louis. Multidimensional binary search trees in database applications. IEEE Transactions on Software Engineering, New York, SE-5(4):333-40, July. 1979.

/BER 81/ BERKELEY UNIVERSITY. SPICE2G users guide. CAD/CAM Dept. Jan. 1981.

/BER 83/ BERESFORD, Roderic. Comparing gate arrays and standard-cell ICs. VLSI Design, Palo Alto, 4(8):30-6, Dec. 1983.

/BIE 84/ BIER, Paulo J. & CALAZANS, N. L. V. Análise microfotográfica da arquitetura interna do controlador de acesso direto à memória AMD 9517. In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE, 11., Viçosa, 21-27 Julho, 1984. Anais. Viçosa, Sociedade Brasileira de Computação, 1984. p.149-60.

/BOY 83/ BOYLE, Douglas B.; CONNEL, J. Michael; ROHRS, Patricia. The role of work stations in a computer-aided-design (CAD) environement. In: IEEE INTERNATIONAL CONFERENCE on COMPUTER DESIGN, New York, Oct 31-Nov 3, 1983. Proceedings. Silver Spring, IEEE, 1983. p.196-8.

/BRA 86/ BRAYTON, R. K. Algorithms for multi-level logic synthesis and organization. In: NATO ADVANCED INSTITUTE ON LOGIC SYNTHESIS AND SILICON COMPILATION FOR VLSI DESIGN, L'Aquila, July 7-18, 1986. Proceedings. s.l., SSGRR, 1986.

/BRU 83/ BRUEDERLE, Stan. The future of gate arrays - a general perspective. In: WESCON/83, San Francisco, Nov 8-11, 1983. Professional Program Session Record no. 30 : Gate arrays - The user/vendor relationship. Electronic Conventions, 1983. 30/1 p.1-5.

/BRY 81/ BRYANT, R. E. A switch-level model of VLSI logic circuits. Cambridge, MIT, Mar. 1981. (PhD dissertation).

/BUR 85/ BURSKY, Dave. Distinctions blur between gate arrays and cells as digital technology evolves. Electronic Design, Hasbrouck Heights, 33(14):81-6, June 13, 1985.

/CAB 86/ CABODI, Gianpiero; CAMURATI, Paolo; PRINETTO, Paolo. The use of PROLOG for executable specification and verification of easily testable circuits. In: ANNUAL INTERNATIONAL SYMPOSIUM ON FAULT-TOLERANT COMPUTING SYSTEMS, 16., Vienna, July 1-4, 1986. Digest of Papers. Washington, IEEE, 1986. p.390-95.

/CAI 87/ CAISSO, Jean-Paul. Contribution à la verification des circuits intégrés dans un environnement multivaluée. Grenoble, Institut National Polytechnique de Grenoble, 1987.

/CAL 87a/ CALAZANS, Ney Laert Vilar & BARONE, Dante Augusto Couto. Proposta de uma nova célula de base para circuitos pré-difundidos na metodologia CIPREDI. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE MICROELETRONICA, 2., São Paulo, 14-16 Jul., 1987. Anais. São Paulo, SBMICRO, 1986. p.212-22.

/CAL 87b/ CALAZANS, Ney Laert Vilar. Especificação do EDGAR - um editor de máscaras para circuitos integrados do tipo "gate array". In: SEMINÁRIO INTEGRADO DE SOFTWARE E HARDWARE, 14., Salvador, 11-19 Julho, 1987. Anais. Salvador, Sociedade Brasileira de Computação, 1984. p.117-30.

/CAL 88/ CALAZANS, Ney Laert Vilar; REY, Leandro Fortes; WAGNER, Flávio Rech. A logic simulator for an integrated environment of digital hardware design. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE MICROELETRONICA, 3., São Paulo, 12-14 Jul., 1988. Anais. São Paulo, SBMICRO, 1988. p.385-95.

/CAR 86/ CARNEY, William & CURTIN, Isabelle. 50K gate array meets tomorrow's design challenges. Digital Design, Boston, 16(3):84-8, Mar., 1986.

/CAR 87/ CARRO, Luigi; NETTO, João Cesar; BARONE, Dante Augusto Couto. Desenvolvimento e caracterização de uma biblioteca de standard cells. Porto Alegre, PGCC da UFRGS, 1987. (Relatório de pesquisa nro 081).

/CAR 88/ CARRO, Luigi. Gerador parametrizável de partes operativas CMOS. Porto Alegre (trabalho em andamento).

/CHE 86/ CHEN, Kunnau. On-chip testability circuit for CMOS gate arrays. VLSI Systems Design, Palo Alto, 7(1):48-50, Jan. 1986.

/COS 82/ COSTA, Antônio Carlos da Rocha. Um guia para elaboração de métodos de projeto de sistemas. Porto Alegre CPGCC-UFRGS, 1982. (Relatório de pesquisa nro. 003).

/COS 83/ COSLEY, John. The role of the third party design in the development of semi-custom devices. In: WESCON/83, San Francisco, November 8-11, 1983. Professional Program Session Record no. 30 : Gate arrays - The user/vendor relationship. Electronic Conventions, 1983. 30/2 p.1-4.

/CUN 87/ CUNHA, Gilberto J. da et al. Computação Gráfica: o padrão GKS. São Paulo, Atlas, 1987.

/CUR 86/ CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DA UFRGS. Laboratório de concepção de circuitos integrados. In: -... Laboratório de Concepção e Desenvolvimento de Sistemas - LCD. Porto Alegre, 1986. (Projeto encaminhado à Financiadora Especial de Projetos - FINEP).

/DAH 85/ DAHLBERG, Bob. Hardware acceleration applied to gate array layout. VLSI Systems Design, Palo Alto, 6(10):72 Oct. 1985.

/DAV 83/ DAVIO, M.; DESCHAMPS, J.-P.; THAYSE, A. Digital systems with algorithm implementation. Chichester, John Wiley, 1983.

/DHU 84/ DHURKADAS, A. Faster parallel multiplier. Proceedings of the IEEE, New York, 72(1):134-6, Jan. 1984.

/DIC 83/ DICKEN, Howard K. Calculating the manufacturing costs of gate arrays. VLSI Design, Palo Alto, 4(8):51-5, Dec. 1983.

/DUM 83/ DUMLUGÖL, Öndar et al. Local relaxation algorithms for event-driven simulation of MOS networks including assignable delay modeling. IEEE Transactions on Computer-aided Design, New York, CAD-2(3):193-202, July. 1983.

/EDI 84/ EDIF Steering Committee. Preliminary EDIF specification. 1984.

/EGL 85/ EGLIN, David. A supermicro workstation for third generation CAE. Computer Aided Design, London, 17(3):130-1, Apr., 1985.

/FEY 86/ FEY, Curt F. & PARASKEVOPOULOS, Demetris E. Studies in LSI technology economics II: a comparison of product costs using MSI, gate arrays, standard cells and full custom VLSI. IEEE Journal of Solid-state Circuits, New York, SC-21(2):297-303, Apr. 1986.

/GAG 83/ GAGLIARDI, Michael P. The effect of cell design on CMOS gate arrays. In: IEEE INTERNATIONAL CONFERENCE on COMPUTER DESIGN, New York, Oct. 31-Nov. 3, 1983. Proceedings. Silver Spring, IEEE, 1983. p.383-5.

/GAJ 83/ GAJSKI, D. D. & KUHN, R. H. New VLSI tools. Computer, New York, 16(12):11-7, Dec. 1983.

/GEN 83/ GENERAL ELECTRIC Company. Gate Array Designer's Reference Manual. 1983.

/GOE 88/ GOERING, Richard. Simulation accelerators address throughput issues. Computer Design, Littleton, 27(6):42-51, Mar. 15, 1988.

/GOL 88/ GOLENDZINER, L. G. & BECKER, K. Interface orientada a objetos para um ambiente de CAD de sistemas digitais. In: SIMPÓSIO BRASILEIRO DE BANCO DE DADOS, 3., Recife, 23-25 Mar. 1988. Anais. Recife, SBC, 1988. p. 213-25.

/GOU 85/ GOULD AMI Semiconductors. 1985 3 microns HCMOS Gate Array Databook. 1985.

/GRA 82/ GRAY, John P.; BUCHANAN, Irene; ROBERTSON, Peter S. Designing gate arrays using a silicon compiler. In: DESIGN AUTOMATION CONFERENCE, 19., Las Vegas, June 14-16, 1982. Proceedings. New York, IEEE, 1982. p.377-83.

/GRE 86/ GREGORY, David; BARTLETT, Karen; DE GEUS, Aart; HACHTEL, Gary. SOCRATES: a system for automatically synthesizing and optimizing combinational logic. In: DESIGN AUTOMATION CONFERENCE, 23., Las Vegas, June 29- July 2, 1986. Proceedings. New York, IEEE, 1986. p.79-85.

/GRI 83/ GRIERSON, J. R. et al. The UK5000 - successful collaborative development of an integrated design system for a 5000 gate CMOS gate array with built-in test. In: DESIGN AUTOMATION CONFERENCE, 20., Miami, June 27-29, 1983. Proceedings. New York, IEEE, 1983. p.629-36.

/HAR 83/ HARDAGE, Charlie. Technology aspects of the user/vendor interface insemicustom. In: WESCON/83, San Francisco, November 8-11, 1983. Professional Program Session Record no. 30 : Gate arrays - The user/vendor relationship. Electronic Conventions, 1983. 30/3 p.1-7.

/HAY 81/ HAYES, John P. A logic design theory for VLSI. In: CALTECH CONFERENCE ON VLSI, Jan 1981. Proceedings. p.455-76.

/HAY 86/ HAYES, John P. Digital simulation with multiple logic values. IEEE Transactions on Computer-aided Design, New York, CAD-5(2):274-83, Apr. 1986.

/HAY 87/ HAYES, John P. An introduction to switch-level modeling. IEEE Design & Test, New York, 4(4):18-25, Aug. 1987.

/HIG 86/ HIGHTOWER, David et al. Computer-aided-design research at the GE microelectronics center. IEEE Design & Test, New York, 3(5):49-56, Oct. 1986.

/INP 86/ INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE. Informations CMP nro 15. Nov., 1986.

/INT 79/ INTEL CORPORATION. MCS 80/85 family user's manual. Oct. 1979.

/JAC 85/ JACK, Mervyn A. Using and Designing with gate arrays. Part 1 - Application considerations. In: Gate Arrays : Design Techniques and Applications. McGraw-Hill Book Company, 1985. Cap 8, p.286-310.

/JAC 86/ JACOBI, R. P. Microeditor: editor gráfico para microeletrônica em PC-IBM. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE MICROELETRONICA, 1., Campinas, 15-17 Jul., 1986. Anais. São Paulo, SBMICRO, 1986. p.408-18.

/JAC 88a/ JACOBI, Ricardo Pezzuol et al. Sistema de PAC de circuitos integrados. In: SIMPÓSIO BRASILEIRO DE CONCEPÇÃO DE CIRCUITOS INTEGRADOS, 3., Gramado, 13-15 Abr., 1988. Anais. Porto Alegre, SBC, 1988. p.107-18.

/JAC 88b/ JACOBI, Ricardo Pezzuol et al. Editor de máscaras hierárquico para layout VLSI. In: SIMPÓSIO BRASILEIRO DE CONCEPÇÃO DE CIRCUITOS INTEGRADOS, 3., Gramado, 13-15 Abr., 1988. Anais. Porto Alegre, SBC, 1988. p.1-11.

/JOH 86/ JOHNSTON, Bill & CASSIDY, Mike. PC supports end to end gate array design. Digital Design, Boston, 16(3):84-8, Mar., 1986.

/KAT 83/ KATZ, Randy H. Managing the chip design database. Computer, Los Alamitos, 16(12):26-36, Dec 1983.

/KRE 83/ KREJCIK, Milos & LIPP, Robert. Logic design with CMOS gate arrays. Part 2: complex functions. VLSI Design, Palo Alto, 4(6):86-8, Oct. 1983.

/LAK 86/ LAKE, Ron. A fast 20K gate array with on-chip test system. VLSI Systems Design, Palo Alto, 7(6):46-55, June. 1986.

/LEE 83/ LEE, Gary. Gate arrays from a users viewpoint. In: WESCON/83, San Francisco, Nov 8-11, 1983. Professional Program Session Record no. 30 : Gate arrays - The user/vendor relationship. Electronic Conventions, Inc, 1983. 30/5 p.1-3.

/LER 84/ LEROUGE, Claude P.; GIRARD, Pierre; COLARDELLE, Joël S. A fast 16 bit NMOS parallel multiplier. IEEE Journal of Solid-state Circuits, New York, SC-19(3):338-42, Apr, 1986.

/LIP 83/ LIPP, Robert & KREJCIK, Milos. Logic design with CMOS gate arrays. Part 1: flip-flops. VLSI Design, Palo Alto, 4(5):70-3, Sept. 1983.

/LOB 83/ LOBO, Keith R. Gate arrays: the vendor perspective. In: WESCON/83, San Francisco, November 8-11, 1983. Professional Program Session Record no. 30 : Gate arrays - The user/vendor relationship. Electronic Conventions, Inc, 1983. 30/4 p.1-5.

/LSI 84/ LSI LOGIC CORPORATION. CMOS macrocell manual. Milpitas, LSI Logic Corporation, Sept., 1984.

/MAD 83/ MADAN, Pradip. Logic array CAD alternatives for the system designer. In: WESCON/83, San Francisco, November 8-11, 1983. Professional Program Session Record no. 10 : CAD technology alternatives for semi-custom and custom VLSI design. Electronic Conventions, Inc, 1983. 10/3 p.1-7.

/MCM 85/ McMINN, Stephen E. Gate array manufacturing and packaging. In: Gate Arrays : Design Techniques and Applications. New York. McGraw-Hill, 1985. Cap 6, p.229-46.

/MEA 80/ MEAD, Carver & CONWAY, Lynn. Introduction to VLSI systems. Reading, Addison-Wesley, 1980.

/MEY 86/ MEYER, Ernest L. Gate array testability: a customer perspective. VLSI Systems Design, Palo Alto, 7(6):34-42, June. 1986.

/MIC 86a/ MICROSOFT CORPORATION. Microsoft MS-DOS user's reference. 1986.

/MIC 86b/ MICROSOFT CORPORATION. Microsoft C compiler for the MS-DOS operating system: users guide. 1986.

/MIY 86/ MIYAHARA, Norio; ISHIKAWA, Keiji; HAMAGUCHI, Shigetatsu; HORIGUCHI, Shoji; AOKI, Makoto. A composite CMOS gate array with 4K RAM and 128K ROM. IEEE Journal of Solid-state Circuits, New York, SC-21(2):228-33, Apr. 1986.

/MOR 87/ MORSCHEL, Ivan José. GERME - gerenciador de projetos para microeletrônica. Porto Alegre, DI-UFRGS, Dezembro 1987. (Trabalho de Conclusão do Bacharelado em Ciência da Computação).

/MOR 88/ MORSCHEL, Ivan & BARONE, Dante Augusto Couto. GERME - um gerenciador de projetos como suporte a estações de trabalho. In: SIMPÓSIO BRASILEIRO DE CONCEPÇÃO DE CIRCUITOS INTEGRADOS, 3., Gramado, 13-15 abr, 1988. Anais. Porto Alegre, SBC, 1988. p.119-28.

/MUR 82a/ MUROGA, Saburo. Fabrication and Cost Analysis. In: VLSI System Design. New York, John Wiley, 1982. Cap 2, p.24-60.

/MUR 82b/ MUROGA, Saburo. Full-Custom and Semi-Custom Design Approaches. In: VLSI System Design. New York, John Wiley, 1982. Cap 9, p.355-427.

/NAK 86/ NAKAZATO, H.; IKEDA, R.; YAMADA, S.; SANO, T.; OGAWA, T.; NISHIMURA, Y.; MARUI, Y. 1.4ns gate arrays with configurable RAM and high testability. In: ESSCIRC'86, 1986. p.53-5.

/NAT 80/ NATIONAL SEMICONDUCTOR CORPORATION. MOS Data book. 1980.

/NEW 86/ NEWTON, A. R. & SANGIOVANNI-VINCENTELLI, A. L. CAD tools for VLSI design. In: NATO ADVANCED INSTITUTE ON LOGIC SYNTHESIS AND SILICON COMPILATION FOR VLSI DESIGN, L'Aquila, July 7-18, 1986. Proceedings. s.l., SSGRR, 1986.

/NOI 85/ NOIJE, Wilhelmus A. M. van & DECLERCK, Gilbert J. Advanced CMOS gate array architecture combining "gate isolation" and programmable routing channels. IEEE Journal of Solid-state Circuits, New York, SC-20(2):469-80, Apr. 1985.

/OHK 82/ OHKURA, I. et al. Gate isolation - a novel basic cell configuration for CMOS gate arrays. In: IEEE 1982 CICC, May, 1982. Proceedings. p.307-10.
Apud SAKASHITA, Kazuhiro; UEDA, Masahiro; ARAKAWA, Takahiko; ASAI, Sotoju; FUJIMURA, Tatsuo; OHKURA, Isao. A 10K-gate CMOS gate array based on a gate isolation structure. IEEE Journal of Solid-state Circuits, New York, SC-20(1):413-7, Feb., 1985. p.413.

/OHN 82/ OHNO, Yasuhiro et alii. Integrated design automation system for custom and gate array VLSI design. In: ICCC82, London, Sept. 7-10, 1982. Proceedings. Amsterdam, North-Holland, 1982. p.512-5.

/OLI 87/ OLIVEIRA, Flávio Moreira de. Aplicação de sistemas especialistas no posicionamento e roteamento de circuitos integrados. Porto Alegre, PGCC-UFRGS, Junho 1987. (Dissertação de Mestrado).

/OSO 87/ OSÓRIO, Fernando Santos. Estudo e implementação de uma interface de entrada e saída gráfica para o EDGAR. Porto Alegre, DI-UFRGS Dezembro 1987. (Trabalho de Conclusão do Bacharelado em Ciência da Computação).

/PAI 85/ PAILLOTIN, J-F. Le système LUCIE. IMAG - Informatique et Mathématiques Appliquées de Grenoble. Jul. 1985.

/PET 77/ PETRI, C. A. General net theory. In: PROCEEDINGS OF THE JOINT IBM UNIVERSITY OF NEWCASTLE UPON TYNE. SEMINAR, Newcastle upon Tyne, Sept. 7-10, 1976. Proceedings. University of Newcastle upon Tyne, 1977. p.131-69.

/PET 81/ PETERSON, J. L. Petri net theory and the modeling of systems. Englewood Cliffs, Prentice-Hall, 1981.

/PIT 81/ PITTS, Ray C. Gate arrays - cost-slashing replacement for SSI, MSI. Electronic Design, Rochelle Park, 29(26):127-133, Dec. 24, 1981.

/RAM 81/ RAMSAY, Frank R. A remote design station for customer uncommitted logic array designs. In: DESIGN AUTOMATION CONFERENCE, 18., Nashville, June 29-July 1, 1981. Proceedings. New York, IEEE, 1981. p.498-504.

/REA 85/ READ, John. Introduction and basic Technology. In: Gate Arrays : Design Techniques and Applications. McGraw-Hill Book Company, 1985. Cap 1, p.1-14.

/REI 83/ REIS, Ricardo. Evaluateur topologique prédictif pour la génération automatique des plans de masse de circuits VLSI. Grenoble, Institut National Polytechnique de Grenoble, 1987.

/REI 85/ REISIG, W. Petri nets: an introduction. Berlin, Springer-Verlag, 1985.

/REI 87/ REIS, Ricardo et alii. A microeletrônica na UFRGS. In: CONGRESSO NACIONAL DE INFORMÁTICA, 20., São Paulo, ago 31-set 6 1987. Anais. São Paulo, SUCEU, 1987. p.1153-1162.

/REI 88/ REIS, Ricardo; GOMES, Rogério; LUBASZEWSKI, Marcelo. An efficient design methodology for standard cell circuits. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS, Helsinki, June 7-9, 1988. Proceedings . New York, IEEE, 1988. v.2 p.1213-6.

/RES 83/ RESNICK, David R. Testability and maintainability with a new 6K gate array. VLSI Design, Palo Alto, 4(2):34-8, Mar./Apr. 1983.

/REY 87/ REY, L. F. Simulador lógico para o sistema AMPLO. Porto Alegre, CPGCC-UFRGS, Julho 1987. (Trabalho Individual).

/RUP 87/ RUPP, W. Kelly & LaPLANTE, O. David. Gate array users tackle place and route tasks. Computer Design, Littleton, 26(10):85-90, May 15, 1987.

/SAI 85/ SAIGO, Takashi; NIWA, Kiyoshi; OHTO, Takeshi; KUROSAWA, Sachiko; TAKADA, Tomoji. A triple-level wired 24K-gate CMOS gate array. IEEE Journal of Solid-state Circuits, New York, SC-20(5):1005-11, October 1985.

/SAI 86/ SAITO, Takao; SUGIMOTO, Hiroyuki; YAMAZAKI, Masami; KAWATO, Nobuaki. A rule-based logic circuit synthesis system for CMOS gate arrays. In: DESIGN AUTOMATION CONFERENCE, 23., Las Vegas, June 29-July 2, 1986. Proceedings. New York, IEEE, 1986. p.594-600.

/SAK 85/ SAKASHITA, Kazuhiro; UEDA, Masahiro; ARAKAWA, Takahiko; ASAI, Sotoju; FUJIMURA, Tatsuo; OHKURA, Isao. A 10K-gate CMOS gate array based on a gate isolation structure. IEEE Journal of Solid-state Circuits, New York, SC-20(1):413-7, Feb. 1985.

/SAS 86/ SASAO, Tsutomu. MACDAS: multi-level AND-OR circuit synthesis using two-variable functions generators. In: DESIGN AUTOMATION CONFERENCE, 23., Las Vegas, June 29-July 2, 1986. Proceedings. New York, IEEE, 1986. p.86-93.

/SCH 85/ SCHINDLER, Max. Better CAE tools help marry digital and analog circuitry. Electronic Design, Rochelle Park, 33(14):81-6, June 13, 1985.

/SEG 86/ SEGAL, B. & GOMES, M. J. M. Metodologia de Particionamento de módulos lógicos para integração em gate array. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE MICROELETRONICA, Campinas, 15-17 Julho, 1986. Anais. São Paulo, SBMICRO, 1986. p.125.

/SIL 88/ SILVA FILHO, J. F.; WAGNER, F. R.; LE FAOU, C. LACO - uma linguagem para descrição de hardware no nível de sistema. Porto Alegre, CPGCC-UFRGS, Outubro, 1988. (Relatório Técnico nro 94).

/TAK 85/ TAKAHASHI, Hiromasa; SATO, Shinji; GOTO, Gensuke; NAKAMURA, Tetsuo; KIKUCHI, Hideo; SHIRATO, Takehide. A 240K transistor CMOS array with flexible allocation of memory and channels. IEEE Journal of Solid-state Circuits, New York, SC-20(5):1012-17, Oct. 1985.

/TAN 81a/ TANAKA, Chiyoji et alii. An integrated computer aided design system for gate array masterslices: Part 1. logic reorganization system LORES-2. In: DESIGN AUTOMATION CONFERENCE, 18., Nashville, June 29-July 1, 1981. Proceedings. New York, IEEE, 1981. p.59-65.

/TAN 81b/ TANAKA, Chiyoji et alii. An integrated computer aided design system for gate array masterslices: Part 2. the layout design system MARS-M3. In: DESIGN AUTOMATION CONFERENCE, 18., Nashville, June 29-July 1, 1981. Proceedings. New York, IEEE, 1981. p.812-9.

/TAV 85/ TÁVORA, Virgílio. Política nacional de informática. Tomo I. Brasília, Serviço Federal, Centro Gráfico, 1985.

/TEL 85/ TELEBRÁS. Projeto de Circuitos Integrados na Técnica de Gate-arrays. 1985.

/TEX 76/ TEXAS INSTRUMENTS INCORPORATED. The IIL Data Book for Design Engineers. 1976.

/TIN 83/ TING, B.S. et alii. Toward a work station for VLSI design. In: IEEE INTERNATIONAL CONFERENCE on COMPUTER DESIGN, New York, Oct 31-Nov 3, 1983. Proceedings. Silver Spring, IEEE, 1983. p.188-91.

/TOD 84/ TODESCO, A. R. W. Sistema RS. Manual do usuário. (não publicado).

/TON 85/ TONGE, J. D. Testing gate arrays. In: Gate Arrays : Design Techniques and Applications. McGraw-Hill, 1985. Cap 7, p.247-285.

/ULR 69/ ULRICH, E. G. Exclusive simulation of activity in digital networks. Communications of the ACM, 12(2):102-10, Feb. 1969.

/WAG 84/ WAGNER, F. R. Basic Techniques of gate level simulation - a tutorial. Porto Alegre, CPGCC-UFRGS, Outubro, 1984. (Relatório de pesquisa nro 012).

/WAG 86/ WAGNER, F. R. A multi-level digital systems simulator based on nets of agencies. In: JSST CONFERENCE ON RECENT ADVANCES IN SIMULATION OF COMPLEX SYSTEMS, Tokyo, July 15-17, 1986. Proceedings. p.125-30.

/WAG 87a/ WAGNER, F. R. KAPA - Uma linguagem para descrição de hardware no nível de transferência entre registradores. Porto Alegre, CPGCC-UFRGS, Abril, 1987. (Relatório de pesquisa nro 068).

/WAG 87b/ WAGNER, F. R. & FREITAS, C. M. D.-S. NILO - uma linguagem de descrição de hardware no nível de portas lógicas. Porto Alegre, CPGCC-UFRGS, Março 1987. (Relatório de pesquisa nro 066).

/WAG 87c/ WAGNER, F. R.; FREITAS, C. M. D.-S.; GOLENDZINER, L. G. Linguagens de descrição de hardware para suporte à integração do processo de projeto AMPLO. Porto Alegre, CPGCC-UFRGS, Março, 1987. (Relatório de pesquisa nro 065).

/WAG 87d/ WAGNER, F. R. et al. A digital systems design methodology based on nets of agencies. In: IFIP WG 10.2 INTERNATIONAL CONFERENCE ON COMPUTER HARDWARE DESCRIPTION LANGUAGES AND THEIR APPLICATIONS, 8.. Amsterdam, Apr. 27-29, 1987. Proceedings. Amsterdam, North-Holland, 1988. p. 213-24.

/WAG 88/ WAGNER, F. R. AMPLO - um ambiente integrado para projeto de circuitos VLSI. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE MICROELETRÔNICA, 3., São Paulo, 12-14 Julho, 1988. Anais. São Paulo, SBMICRO, 1988. p.437-47.

/WAS 78/ WASER, Shlomo. High-speed monolithic multipliers for real-time digital signal processing. Computer, Long Beach, 11(10):19-29, Oct. 1978.

/WAT 87/ WATKINS, Marvin L. Software architecture and the UNIX operating system: an introduction to interprocess communication. Hewlett-Packard Journal, Palo Alto, 38(6):26-36, June. 1987.

/WEI 86/ WEIL, Steven; SUI, Denny C.; PELLERIN, Dave B. Gate-array compiler cuts big designs down to size. Electronic Design, Hasbrouck Heights, 34(15):101-8, June 26, 1986.

/WEN 80/ WENDT, S. On the partitioning of computing agencies into communicating agencies. In: GI-NTG FACHTAGUNG - STRUKTUR UND BETRIEB von RECHENSYSTEMEN. Kiel, Mar. 1980. Proceedings. Berlin, Springer-Verlag, 1980. p.194-204.

/WER 81/ WERNER, Jerry. Software for gate array design: who is really aiding whom? VLSI Design, Palo Alto, 2(4):22-32, 4th Quarter 1981.

/WER 82/ WERNER, Jerry. IC CAD workstations: entering the engineers realm. VLSI Design, Palo Alto, 3(2):44-50, Mar./Apr. 1982.

/WOL 85/ WOLF, William J. Converting gate array designs to standard cells. Computer Design, Littleton, 24(3):125-128, Mar., 1982.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Pós-Graduação em Ciência da Computação

CIPREDI : Contribuição inicial para um método de concepção de circuitos integrados pré-difundidos

Dissertação apresentada ao Srs.

Dante Augusto Couto Barone

Flávio Rech Wagner

Altamiro Amadeu Suzim

Visto e permitida a impressão
Porto Alegre, 22/03/89

Dante Augusto Couto Barone
Orientador

Ingrid E. S. Jansch Pôrto
Coordenadora do Curso de Pós-Graduação
em Ciência da Computação