SISTEMA BASEADO EM REGRAS PARA
POSICIONAMENTO DE COMPONENTES ELETRÔNICOS

Júlio César Gomes Pimentel

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Aprovada por:

Prof. Éber Assis Schmitz, D.Sc.
(Presidente)

Pesq. Reynaldo S. da Costa, M.Sc.

Prof. Júlio Salek Aude, D.Sc.

Prof.Antonio C. Mesquita Filho, D.État.

Prof. Edil Severiano T. Fernandes, D.Sc.

RIO DE JANEIRO, RJ - BRASIL
ABRIL DE 1990

PIMENTEL, JÚLIO CÉSAR GOMES

Sistema Baseado em Regras para Posicionamento de Componentes Eletrônicos [Rio de Janeiro] 1990. X, 133 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 1990)

Tese - Universidade Federal do Rio de Janeiro, COPPE.

1. Posicionamento Automático. I. COPPE/UFRJ II. Título (Série)

Agradecimentos

À direção do Centro de Pesquisas de Energia Elétrica pela oportunidade de desenvolvimento da tese.

Ao professor D.Sc. Éber Assis Schmitz pela orientação da tese e pelas sugestões valiosas que me pouparam várias horas de trabalho infrutífero.

Gostaria de agradecer ao pesquisador M.Sc. Reynaldo S. da Costa, chefe da área de Medição e Controle de Energia do Dpet-CEPEL e responsável pelo Programa para Infraestrutura em Microeletrônica, dentro da qual esse trabalho foi desenvolvido, pela co-orientação da tese, e pela compreensão e colaboração, colocando a infraestrutura do laboratório a minha disposição foram relevantes para o desenvolvimento desta tese.

Agradeço a Armando A. P. Ferreira e Aline D. Robson pela dedicação nas diversas reuniões que fizemos para análise e escolha dos algorítmos e métodos utilizados no SAP. Desejo agradecer também a Guilherme Nelson que me ajudou a esclarecer alguns conceitos relacionados a IA.

Aos especialistas em leiaute Mário A. de Campos e Rosemberg G. Ferreira de quem eu obtive a experiência e as regras de leiaute necessárias ao projeto, e a todos os membros do Dpet-CEPEL e em particular aos amigos dos laboratório C11 e C12 pelo apoio dedicado durante a tese. Agradeço especialmente a César J. Bandim e Mario A. de Campos pela colaboração prestada na implantação do sistema CADD, e a Rosalva P. de Oliveira que me ajudou na prepararação da dissertação da tese.

Meus agradecimentos a meus pais e irmãos pelo incentivo que me têm dispensado durante todos os momentos importantes da minha vida. Finalmente, eu quero agradecer também à minha esposa Sandra, não só por ter me ajudado no trabalho de revisão dos manuscritos, mas principalmente pelo encorajamento e pela inspiração que ela não me deixou faltar.

Resumo da Tese apresentada a COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

SISTEMA BASEADO EM REGRAS PARA POSICIONAMENTO DE COMPONENTES ELETRÔNICOS

Júlio César Gomes Pimentel

Abril de 1990

Orientadores: Éber Assis Schmitz
Reynaldo S. da Costa

Programa: Engenharia de Sistemas e Computação

Este trabalho mostra o desenvolvimento de um alocador automático de componentes eletrônicos que utiliza técnicas de inteligência artificial junto com algorítmos clássicos de otimização.

Até então, as técnicas clássicas existentes têm tratado a alocação automática de componentes como um problema de otimização, e em geral não levam em consideração as características funcionais do circuito. Essas técnicas não têm sido adequadas para obter o posicionamento de alguns tipos de placas, e em especial para o caso de placas digitais estruturadas em barramentos. Esta tese investiga a utilização de conhecimento empírico (extraído de um técnico especialista em leiaute), junto com algorítmos clássicos, visando direcionar a busca da solução e melhorar a qualidade final do posicionamento.

A partir das listas de redes e de componentes do circuito, o alocador inicialmente identifica e classifica subconjuntos desses objetos que sejam relevantes para a solução do problema. Em seguida, o sistema utiliza um conjunto de regras empíricas para determinar o posicionamento relativo dos componentes e utiliza algorítmos para os casos onde não foi possível identificar essas regras. Por último, é feita a otimização do

posicionamento para minimizar o comprimento total das ligações e equilibrar a sua distribuição sobre a placa.

A parte inteligente do sistema foi implementada em Turbo PROLOG. enquanto que os algorítmos estão escritos em linguagem C. O protótipo do sistema está atualmente instalado num microcomputador da linha IBM-PC.

Abstract of Thesis presented to COPPE/UFRJ as partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A RULE-BASED SYSTEM FOR ELETRONIC COMPONENTS PLACEMENT

Júlio César Gomes Pimentel

April, 1990

Thesis Supervisors: Éber Assis Schmitz
Reynaldo S. da Costa

Department: System Engineering and Computation

This thesis describes a new approach to the problem of component placement on printed circuit boards. It aims at developing an automatic placement system (SAP) that uses both artificial inttelligence techniques and classical optimizing algorithms.

The previous appraochs model the placement problem as a classical optimization problem, and they do not take into account the intrinsic features of the circuit. So, they have not been suitable to deal with some kind of printed circuit boards, mainly in the case of bus-structured ones. This dissertation researches the use of empirical knowledge (got from a layout human expert), with classical techniques, to enhance the placement performace and final result.

Starting from the component and net lists, SAP identifies and classifies groups of these objects which are important to the problem domain. Afterwards, it uses a rule base (acquired from a human expert) to find a relative placement to the components. SAP uses algorithms wherever there are no rules because it is difficult to get them or

they do not exist. Finally, the relative placement is optimized to minimize the total wire length and to equalize the distribuition of wires on the board.

The intelligent part of SAP was implemented in Turbo PROLOG and the algorithmic part was written in C. The prototype of the automatic placement system is installed in a IBM-PC like microcomputer.

ÍNDICE

CAPÍTULO I
Introdução

O projeto de placas de circuito impresso é um componente chave na produção industrial de equipamentos eletrônicos. Portanto, obter o domínio total dessa etapa do processo de produção é fundamental para garantir nossa independência tecnológica. Atualmente, devido à complexidade cada vez maior dessas placas, tornou-se necessária a utilização de ferramentas computacionais que auxiliem o seu desenvolvimento. Embora existam vários desses sistemas disponíveis comercialmente no exterior, eles são em geral pouco utilizados no Brasil por várias razões, entre as quais o seu custo elevado, dificuldade de aquisição, e a falta de suporte adequado à sua correta implantação e utilização.

O trabalho descrito a seguir é fruto da experiência adquirida com o projeto e implementação de um sistema para auxílio à confecção de leiautes de placas de circuito impresso (PIMENTEL e BANDIM [1989]), no Departamento de Eletrônica do Centro de Pesquisas de Energia Elétrica (DPET-CEPEL). O sistema compreende um editor gráfico interativo, uma biblioteca que contém a maioria dos encapsulamentos necessários à confecção de leiautes de PCIs, e um programa que recebe o desenho feito no editor e prepara as artes finais que serão enviadas para fabricação.

O sistema CADD está atualmente em operação no Laboratório de Medição e Controle de Energia do Dpet-CEPEL, onde vem sendo utilizado na confecção de várias placas de circuito impresso analógicas e digitais. Foram confeccionadas até o momento mais de vinte placas.

A utilização do sistema durante esses dois anos mostrou que realizar o posicionamento dos componentes na PCI é uma das etapas mais demoradas, e diretamente responsável pela qualidade final do leiaute. Esta tese apresenta o desenvolvimento de um alocador automático de componentes que será posteriormente integrado ao sistema CADD, visando reduzir o tempo total gasto na confecção das

PCIs. Detalhes sobre o projeto e implementação do sistema CADD não fazem parte do escopo desta tese, podendo ser encontrados em (COSTA e PIMENTEL [1986], e PIMENTEL e BANDIM [1989]).

I.1 - Escopo do Trabalho.

Até então, as técnicas de posicionamento existentes tratavam o posicionamento de componentes em PCIs como um caso clássico de programação matemática ou otimização em grafos. Essas técnicas impõem um conjunto de restrições ao problema original para torná-lo computacionalmente solúvel. Entretanto, pode ocorrer que a solução obtida não seja a solução ótima do problema original.

Atualmente, as placas de circuito impresso têm incorporado um número cada vez maior de circuitos integrados LSI (integração em larga escala), tais como: microprocessadores, dispositivos de entrada/saída, e memórias. Essas placas possuem como característica marcante a existência de um grande número de redes organizadas em barramentos. As técnicas clássicas têm se mostrado inadequadas a esse tipo de PCIs, principalmente pela impossibilidade de modelar a estrutura dos barramentos. Assim, a qualidade do posicionamento obtido é normalmente inferior ao resultado obtido pelos técnicos especialistas em leiaute.

A interação com os especialistas mostrou que além de trabalharem com um número menor de restrições, eles desenvolveram uma metodologia que leva em consideração às características funcionais da placa, bem como a experiência adquirida nos leiautes anteriores. Dessa forma eles descartam as soluções que são consideradas inadequadas.

Esta tese investiga a utilização de uma nova técnica que incorpora conhecimento e algorítmos clássicos para determinar o posicionamento ótimo de componentes em PCIs. Foi adotada uma estratégia semelhante à utilizada pelos especialistas humanos e que pode ser resumida em dois princípios básicos:

. utilizar regras empíricas para determinar o posicionamento relativo dos componentes na placa.
. utilizar algorítmos para os casos em que não for possível identificar regras ou por motivo de eficiência.

Com este objetivo foi desenvolvido um alocador automático para placas de circuito impresso digitais estruturadas em barramentos que utiliza conhecimento armazenado na forma de cláusulas PROLOG (fatos e regras) e técnicas clássicas de posicionamento, dentro dos princípios descritos acima. A partir das listas de redes e componentes, o alocador inicialmente identifica subconjuntos desses objetos que sejam relevantes para a solução do problema. Em seguida, ele utiliza um conjunto de regras empíricas e algorítmos para determinar o posicionamento relativo ótimo dos componentes. Por último é feita a otimização do posicionamento para minimizar o comprimento total dos fios e equilibrar sua distribuição sobre a placa.

Finalizando, cabe dizer que embora este trabalho trate essencialmente de placas digitais estruturadas em barramentos, a técnica apresentada pode ser estendida para lidar com outros tipos de placas. Nesse caso, basta ampliar a base de regras para identificar novas estruturas, o que diminuirá o tamanho da parte do circuito processada por algorítmos. É importante notar que ela possui um potencial muito grande e poderá ser útil inclusive na solução de outros problemas de leiaute, até mesmo em outras áreas da engenharia.

I.2 - Trabalhos Relacionados.

Um dos trabalhos mais antigos relacionados com esta área foi realizado por (ODAWARA et alii [1983]]) que propõe a combinação de várias técnicas de posicionamento. Embora o sistema proposto considere algumas características da placa, tais como a estrutura dos barramentos e a posição dos conectores, essa informação está armazenada na forma algorítmica, dificultando a implementação de várias

heurísticas. Posteriormente, (ODAWARA et alii [1985] e ODAWARA et alii [1987]) descrevem duas tentativas de utilizar conhecimento, armazenado na forma de fatos e regras, na solução do problema. Segundo ODAWARA et alii [1987]), os resultados obtidos são superiores aos produzidos pelas técnicas clássicas, e em alguns casos até melhor que o leiaute confeccionado pelos especialistas.

(DEJESUS et alii [1986]) descreve um sistema baseado em conhecimento para auxiliar à confecção de leiautes de fontes de alimentação. O sistema PEARL, desenvolvido pela Digital Equipment Corporation (DEC), tem sido utilizado como ferramenta de produção em muitos de seus centros de desenvolvimento.

Combinação de conhecimento e algorítmos estão sendo utilizados também para a implementação de ferramentas de CAD aplicadas na confecção de leiautes de circuitos integrados (CI) tanto na etapa de posicionamento (LIN e GAJSKI [1987]) como no roteamento (MORI et alii [1984] e JOOBANI [1987]).

Esta técnica tem sido utilizada também em outras áreas, tais como os trabalhos publicados por (MARUYAMA et alii [1984] e STEELE [1987] em síntese de circuitos digitais. (BHAWMIK e PALCHAUDHURI [1989]) descrevem um sistema especialista para configurar estruturas de teste ("design for testability") em circuito integrado VLSI.

I.3 - Plano da Tese

O capítulo II descreve o problema de posicionamento, os fatores que afetam a sua qualidade, e algumas funções que podem ser utilizadas como estimativa do leiaute final. Neste capítulo estão descritas algumas das principais técnicas clássicas, e suas características principais.

O capítulo III apresenta um resumo de sistemas baseados em conhecimento, sua arquitetura, e algumas formas de representação do conhecimento. O final do capítulo contém um breve resumo da linguagem PROLOG utilizada para o desenvolvimento do protótipo.

No capítulo IV é apresentada a estratégia do Sistema de Posicionamento (SAP), além de algumas definições

necessárias à sua implementação. Este capítulo mostra as etapas do processo onde são utilizadas as regras e os algorítmos, e tece algumas considerações sobre quando usar um ou outro.

O capítulo V descreve a arquitetura do SAP, os diversos módulos que o compõem, e a organização desses módulos dentro do programa. Este capítulo descreve os detalhes de implementação do sistema entre os quais as modificações realizadas nas técnicas clássicas, visando atender às características próprias do SAP.

O capítulo VI descreve a metodologia utilizada para validar cada módulo de programa que compõe o sistema SAP, bem como a análise dos resultados obtidos.

CAPÍTULO II
Posicionamento de Componentes em PCI

O leiaute automático de componentes eletrônicos em PCIs consiste basicamente de duas tarefas: determinar a posição dos componentes eletrônicos na PCI, chamada posicionamento; e interconectar os pinos desses componentes de acordo com um conjunto de regras pré-definidas (regras de projeto), chamada roteamento. Muito embora posicionamento e roteamento sejam tarefas interdependentes, devido à complexidade do problema de leiaute, essas duas tarefas têm sido historicamente tratadas separadamente. (LOO [1979]) descreve uma das poucas tentativas de tratar o problema de forma combinada. Entretanto, a obtenção de um bom posicionamento tem influência decisiva na qualidade final do leiaute. Isto é, um posicionamento mal feito pode tornar muito difícil, ou até mesmo impossível rotear a PCI, enquanto um posicionamento de boa qualidade facilita bastante a tarefa do roteador. Além disso, como o posicionamento determina também o comprimento mínimo dos fios de interconexão, ele pode limitar o desempenho do circuito. Dessa forma, determinar um bom posicionamento para os componentes, é a chave para se obter um bom leiaute final.

II.1 - Apresentação do Problema

As placas de circuito impresso compõem-se de um conjunto de componentes eletrônicos analógicos (ex.: resistores, capacitores, transistores, amplificadores operacionais) e digitais (ex.: memórias, microprocessadores, e registradores), e de um conjunto de fios que interconectam os componentes.

Os componentes têm pinos de conexão localizados normalmente na periferia. O conjunto dos pinos conectados

eletricamente entre si é denominado uma rede de sinal, ou simplesmente rede. Como cada componente possui vários pinos de conexão, geralmente um componente pertence a mais de uma rede.

O problema de posicionamento consiste em mapear o conjunto de componentes para posições dentro da PCI de forma a minimizar alguma função de custo definida sobre as interconexões. Esta função deve ser definida de forma que o leiaute final satisfaça às seguintes restrições:

. o casamento entre os sinais ("cross-talk") deve ser eliminado ou pelo menos minimizado.
. a dissipação de calor na PCI deve ser mantida em níveis aceitáveis.
. não deve haver concentrações de calor num mesmo ponto da placa.
. a roteabilidade da PCI deve ser maximizada.
. em muitos casos o leiaute deve ocupar a menor área possível para diminuir o custo da placa.

Entretanto, na maioria dos casos práticos, é impossível atender a todas essas restrições. Portanto, deve-se escolher uma função de custo que permita minimizar alguns critérios, mantendo os demais dentro de limites aceitáveis. (WILSON e SMITH [1976], BOXLEITNER [1989], QUINN e BREUER [1979]) afirmam que minimizar o comprimento total dos fios é um critério que pode ser usado para satisfazer parcialmente a estes objetivos.

As técnicas de posicionamento existentes atualmente são normalmente divididas em dois grupos: o posicionamento contínuo e o não contínuo. No primeiro caso (QUINN e BREUER [1979], e FISK [1967]) a PCI é tratada como um plano contínuo sobre o qual os componentes são posicionados, e podem ocupar portanto qualquer região da placa. No caso não-contínuo a área da placa é particionada numa matriz de células ("slots") dentro das quais os componentes são posicionados de tal forma que cada componente pode ocupar somente uma célula, conforme mostra a figura (II.1). Uma discussão bastante completa do caso contínuo pode ser

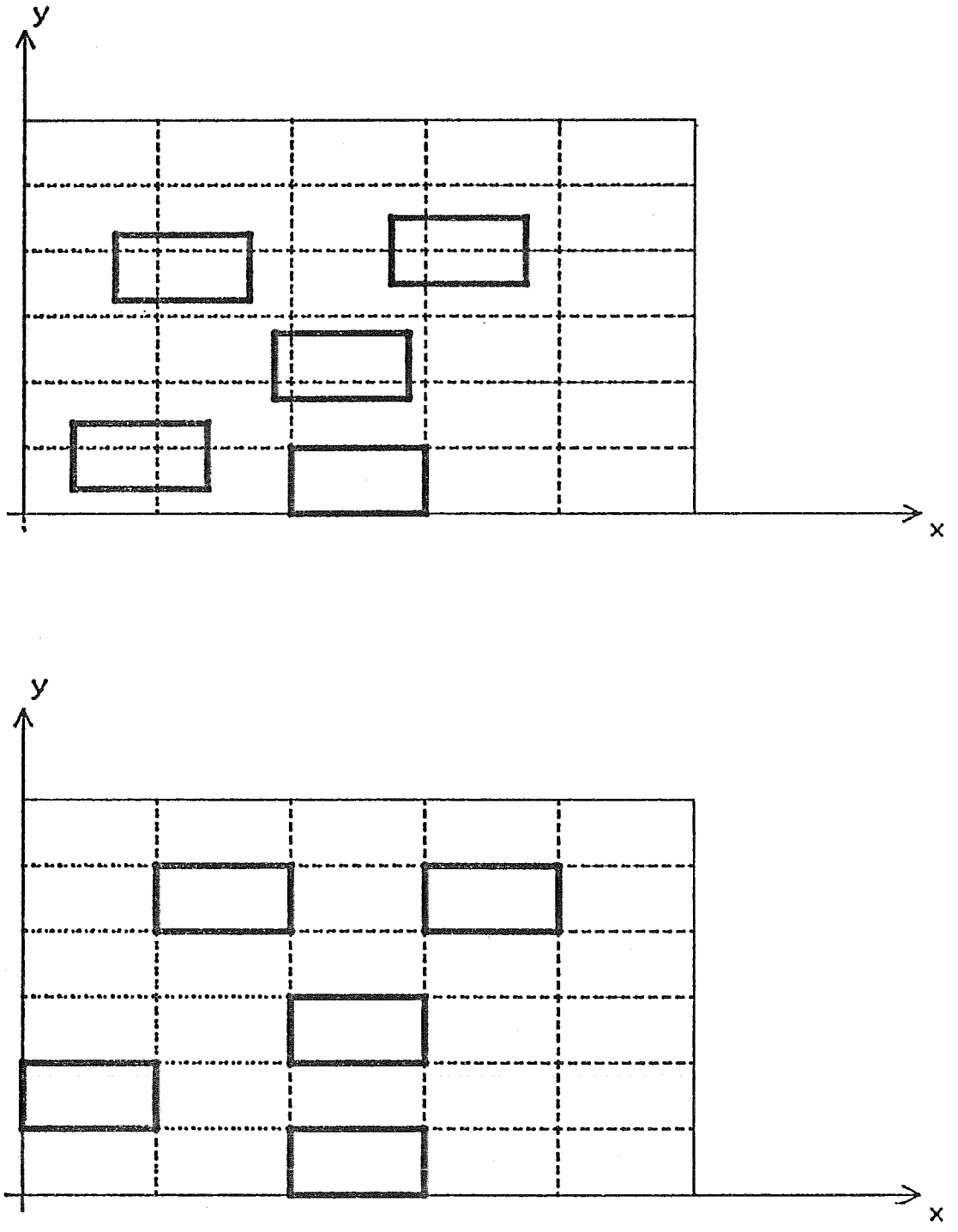


Figura II.1 - Posicionamento Contínuo e Não-Contínuo.

encontrada em (HANAN et alii [1976], e HANAN e KURTZBERG [1972]). Os itens II.3.1 a II.3.6 apresentam resumos de algumas técnicas relativas ao caso não-contínuo. Os itens II.3.7 e II.3.8 apresentam duas técnicas clássicas de como tratar o caso contínuo. Cabe notar que a técnica de MIN-CUT pode ser usada nos dois casos.

II.1.1 - Topologia das Interconexões

Após o posicionamento dos componentes na PCI, a próxima etapa é realizar a interconexão dos pinos que formam as redes (roteamento). Considerando-se os k-pinos de uma rede como sendo os vértices de um grafo não orientado (HANAN et alii [1976]), então a topologia de interconexão é uma árvore definida nesses k-vértices. Existem tipicamente três tipos de árvores:

i) Árvore de Espalhamento Mínimo ("Minimum Spanning Tree - MST") é definida como a árvore de comprimento mínimo que conecta os k-pinos de uma rede no plano, e cujos vértices estão localizados sobre os k-pinos (JOOBANI [1987]).

ii) Em muitos casos de roteamento é possível adicionar um ou mais pontos extras ao conjunto dos K-pinos originais para construir uma árvore mais curta. Os pontos extras são chamados pontos de Steiner, e a árvore de comprimento mínimo formada pela inclusão desses pontos é chamada de Árvore de Steiner Mínima ("Minimum Steiner Tree") (CHANG [1972] e JOOBANI [1987]). Em geral é mais fácil obter uma MST do que uma árvore de Steiner Mínima.

iii) árvores especiais, geralmente de comprimento mínimo, e que satisfazem a certas restrições tais como número de arcos incidentes em um vértice, ou outras restrições topológicas quaisquer (HANAH et alii [1976]).

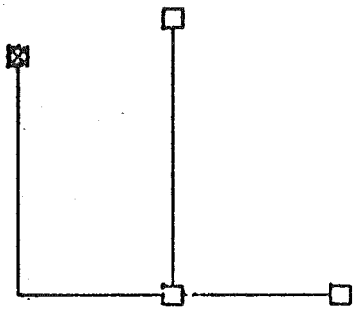
A figura (II.2) mostra uma rede formada por quatro vértices e quatro topologias diferentes para interconectá-la. O vértice cheio representa o pino fonte (pino de saída de um componente), e os vértices vazios representam os pinos destinos (pinos de entrada dos componentes). A figura (II.2a) é uma MST, enquanto a figura (II.2b) é uma árvore mínima de Steiner, na qual foram inseridos dois vértices extras. As árvores mostradas nas figuras (II.2c) e (II.2d) são árvores especiais. A primeira é uma MST que tem no máximo dois arcos incidentes em cadas vértice (esta árvore é denominada cadeia). A segunda é uma árvore que satisfaz à restrição de haver uma conexão do pino fonte para cada pino destino.

A escolha da topologia que será utilizada para interconectar as redes depende principalmente da tecnologia utilizada para implementar o circuito (ex.: ECL, TTL, NMOS, CMOS, etc.). Em alguns casos, cada pino de conexão pode ter no máximo dois ou três pinos ligados a ele. Em outros, todas as conexões são fios do pino fonte para todos os pinos destinos. Portanto, a topologia das interconexões deve ser levada em consideração na escolha da função de custo do algorítmo de posicionamento.

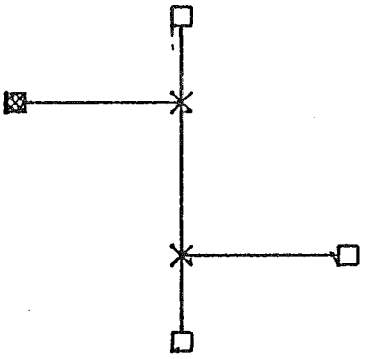
II.1.2 - Estratégia Clássica para Solução do Problema de Posicionamento

O problema de posicionamento pode ser definido como (HANAN et alii [1976]):

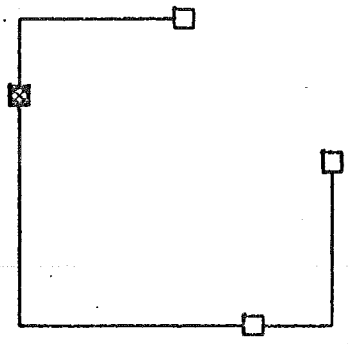
Definição I: Dado um conjunto de componentes, um conjunto de redes interligando subconjuntos desses componentes, e um conjunto de células pertencentes a uma placa, associar os componentes às células de forma a



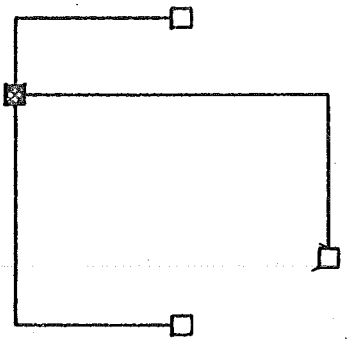
a) ARVORE DE ESPALHAMENTO MÍNIMA



b) ARVORE DE STEINER MÍNIMA



c)



d)

Figura II.2 - Topologia das Interconexões.

minimizar a soma do comprimento das MSTs associadas às redes.

Da forma como está definido, encontrar a solução para o problema de posicionamento é computacionalmente difícil. A estratégia utilizada na maioria dos algorítmos clássicos é transformar o problema de posicionamento num problema de associação quadrática, e resolvê-lo. Sendo este computacionalmente mais simples, sua solução é obtida num tempo muito menor, possibilitando a solução de casos inicialmente considerados intratáveis. O problema de associação quadrática pode ser enunciado como:

Definição II: Dados n componentes e n posições possíveis para esses componentes, e uma matriz $[c_{ij}]$, onde c_{ij} é uma medida da afinidade entre os componentes i e j, além da matriz $[d_{kl}]$, onde d_{kl} é a distância da posição k à posição l, e sabendo que p(i) e p(j) são as posições dos componentes i e j respectivamente, achar a associação de um e um só componente i a cada posição p(i) de forma que a função $G = \Sigma\, c_{ij} d_{p(i)p(j)}$ seja mínima.

Como podemos notar, o problema de associação quadrática é um caso particular do problema de posicionamento. A diferença básica é que no segundo caso a distância é calculada sobre um conjunto de pontos (pinos das redes) enquanto que na associação quadrática a mesma é calculada sobre pares de pontos. Consequentemente, o tempo necessário para resolver a associação quadrática é consideravelmente menor que o tempo necessário para o mesmo algorítmo resolver o problema de posicionamento.

O modo mais simples para transformar um problema de posicionamento num problema de associação quadrática é escolher c_{ij} igual à soma das redes comuns aos componentes i e j para i diferente de j, e $c_{ii} = 0$, e escolher a matriz distância D igual à matriz distância do problema de posicionamento original. Entretanto, a solução obtida com essa transformação não é necessáriamente a solução ótima

do problema de posicionamento original (HANAN e KURTZBERG [1972], e HANAN et alii [1976]).

II.2 - Medidas da Qualidade do Posicionamento

O critério utilizado normalmente pelos algorítmos de posicionamento é minimizar o comprimento total dos fios das interconexões (QUINN e BREUER [1979], e HANAN et alii [1976]). Contudo, como o posicionamento é feito antes do roteamento, não é possível calcular o valor real do comprimento total dos fios, e deve ser, portanto, estimado. A qualidade final do posicionamento (e consequentemente do leiaute) dependerá de quanto mais próximo essa estimativa ficar do resultado final do roteamento. A seguir apresentamos alguns critérios usados para medir a qualidade do posicionamento e usados normalmente como função de custo destes algorítmos (WEIS e MYLNSKI [1988], GOTO [1981], ODAWARA et alii [1982], e BREUER [1972]).

II.2.1 - Árvore Retilínea Mínima de Steiner.

O roteamento das redes de uma PCI é feito em geral sobre uma grade retilínea (Manhattan). Assim, procurar o caminho mínimo de interconexão dos pinos da rede equivale a construir uma árvore retilínea minima de Steiner (MRST) sobre a grade de (HANAN et alii [1976] e BREUER [1972]). (EVEN [1979] e COCKAYNE [1970]) afirmam que construir uma MRST é em geral problema NP-completo. Embora existam algorítmos para construir a MRST de alguns casos especiais (AHO et alii [1977] e BARATZ [1981]), essa tarefa é em geral bastante dispendiosa, o que impede utilizá-las na prática como função de custo nos algorítmos de posicionamento.

II.2.2 - Comprimento de Roteamento

Define-se comprimento de roteamento de uma rede (Figura II.3) como o semi-perímetro (HPCR) do menor retângulo que envolve todos os pinos dos componentes ligados à rede (também chamado de retângulo circunscritor CR). Embora essa definição seja somente uma aproximação da MRST associada à rede (GOTO E KUH [1978] e WEIS [1988]), utilizar o HPCR como estimativa do comprimento total é bastante conveniente pelas seguintes razões:

i) da forma como foi definido, o comprimento de roteamento é muito fácil de ser calculado.
ii) o comprimento de roteamento representa um limite inferior para o comprimento da MRST.
iii) para redes de dois ou três pinos, ela é igual ao comprimento da MRST.
iv) na maioria dos circuitos práticos, 70 a 80 porcento das redes estão associadas a no máximo três pinos.

Seja a rede N_m definida pelo conjunto $N_m = \{p_{m1}, p_{m2}, \ldots, p_{mk}\}$ dos seus k-pinos, e cada pino p_{mj} é definido pelas suas coordenadas (x_{mj}, y_{mj}). O semi-perímetro da rede N_m é calculado da seguinte forma:

$$HPCR(m) = (v_m - u_m) + (v'_m - u'_m)$$

onde:

$u_m = \min (x_{mj})$ $\quad$ $v_m = \max (x_{mj})$

$u'_m = \min (y_{mj})$ $\quad$ $v'_m = \max (y_{mj})$

Para redes com um número muito grande de pinos tais como linhas de clock, o HPCR não é uma boa aproximação. Entretanto, esse tipo de rede percorre normalmente uma região muito grande da placa, de forma que o seu comprimento estimado não varia muito com o posicionamento (GOTO [1981]).

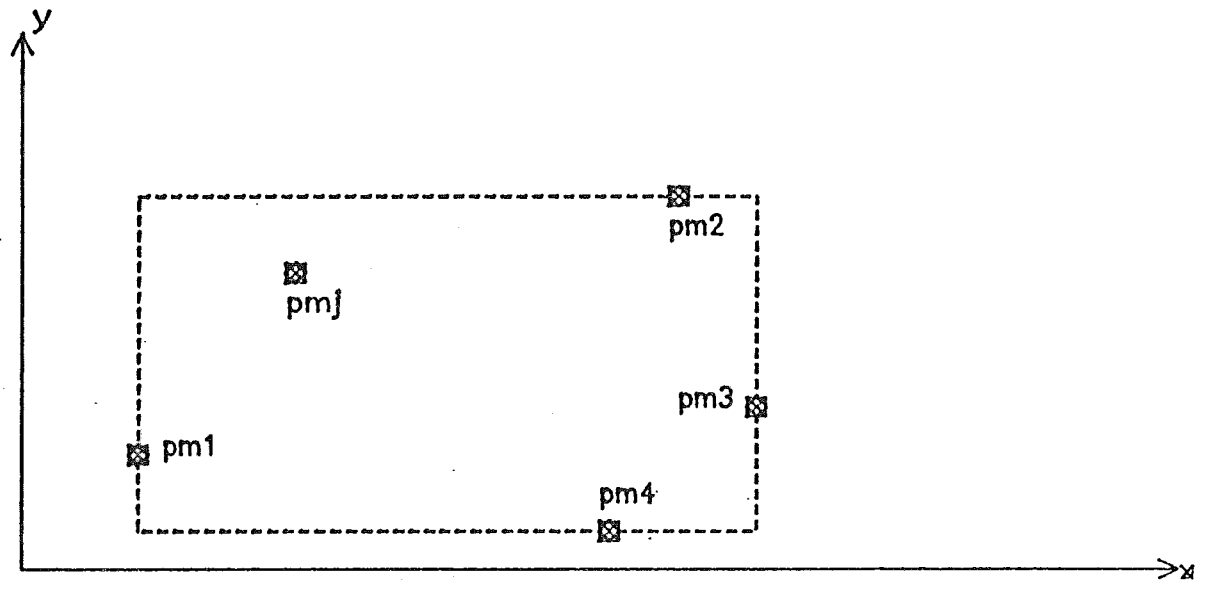


Figura II.3 - Semi-perímetro de uma Rede de Cinco Pinos.

II.2.3 - Comprimento de Fio Acima do Limiar (OTWL)

Além do comprimento total dos fios, um outro critério importante que normalmente aumenta a probabilidade de roteamento da placa é realizar um posicionamento que leve a uma densidade uniforme dos fios sobre a placa. Entretanto, minimizar esses dois critérios ao mesmo tempo é normalmente difícil, e na maioria dos casos impossível. Em outras palavras, quando o comprimento total é minimizado, normalmente ocorrerão concentrações de fios em algumas regiões da placa. E quando a densidade dos fios é mantida uniforme, o comprimento total tende a aumentar. (ODAWARA et alii [1982]) propõe uma função de custo chamada comprimento de Fio Acima do Limiar ("Over Threshold Wire Length"), definida como:

$$F = \Sigma (L_i - S * Th / 100)$$

para todo i tal que $L_i > S * Th / 100$.

L_i - comprimento dos fios na i-ésima célula.
S - área de uma célula.
Th - valor do limiar expresso em porcentagem.

A constante Th é chamada valor de limiar, e define o limite de ocupação da célula. Se o comprimento dos fios L_i for menor que o valor de limiar, a célula será facilmente roteada. O excesso do comprimento dos fios acima do limiar (L_i - S * Th / 100) reflete a incerteza da célula ser roteada. Portanto, a soma de todas esses excessos é uma boa medida para verificar a eficiência do roteamento. Fazer o valor de limiar igual a zero equivale a minimizar somente o comprimento total dos fios, enquanto que Th = 40% prioriza a densidade uniforme dos fios.

II.3 - Algorítmos Clássicos de Posicionamento

Vários algorítmos de posicionamento têm sido propostos e implementados. Esses algorítmos dividem-se basicamente em dois grupos: os construtivos e os iterativos. Os algorítmos construtivos são usados para criar um posicionamento inicial dos componentes na placa, enquanto que os iterativos são usados normalmente para otimizar o posicionamento inicial. Os algorítmos iterativos possuem a mesma estrutura cuja a forma genérica está mostrada abaixo (PREAS [1986]).

```
CALCULE o custo do posicionamento atual;
REPITA
        SELECIONE o componente a ser movido;
        MOVA o componente para a posição objetivo;
        CALCULE o custo do posicionamento atual;
        SE troca reduziu o custo ENTÃO aceita a troca:
        SENÃO MOVA o componente para a posição inicial;
ATÉ QUE o critério de parada seja satisfeito;
```

O item II.3.1 descreve um algorítmo construtivo genérico. O resto da seção apresenta resumidamente alguns algorítmos interativos representativos. O final do capítulo apresenta uma discussão das características comuns desses algorítmos.

II.3.1 - Algorítmos Construtivos (CIP)

Nos algorítmos de posicionamento construtivos ("Construtive Initial Placement - CIP") os componentes são relacionados sequencialmente (um de cada vez), segundo uma função de seleção pré-definida, e depois posicionados na posição da placa que minimiza uma dada função de custo (HANAN et alii [1976] E GOTO [1981]). Após serem posicionados, eles não são mais removidos da placa.

A eficiência dos algorítmos CIP depende diretamente da função de seleção e da função de custo utilizadas. (BREUER [1972], SCHWEIKERT [1976] e ODAWARA et alii [1982]) apresentam algumas funções utilizadas nessa classe de algorítmos. Além disso, a escolha do primeiro componente a ser posicionado influencia tanto a sequência de escolha dos próximos componentes quanto na posição onde estes serão posicionados. Algumas estratégias para solucionar esses problemas estão descritas em (ODAWARA et alii [1982] e SCHWEIKERT [1976]).

II.3.2 - Algorítmo de Steinberg (SB)

O algorítmo de Steinberg (STEINBERG [1961]) lineariza o problema de posicionamento iteragindo sobre conjuntos independentes de componentes, isto é, conjuntos de componentes que não possuem redes em comum, e resolve um problema de associação linear para cada conjunto de componentes independentes.

O problema de Associação linear (KUHN [1955]) é um caso particular do problema de associação quadrática, e é usualmente enunciado em termos de associação de homens a tarefas como:

Definição III: Dados n homens e n tarefas, e o custo c_{ij} de associar o homem i à tarefa j, achar a associação de um e somente um homem a cada tarefa, de tal forma que o custo total seja mínimo (HANAN et alii [1976]).

Representando por p(n) a permutação dos primeiros n inteiros, então o problema pode ser escrito matematicamente como achar o mínimo da função $F = \Sigma c_i * p(i)$. Assim, o algorítmo SB pode ser descrito como:

```
SELECIONE todos os D grupos de componentes
          desconectados;
Termina:= FALSO;
REPITA
  I:= 1;
  ENQUANTO (I <= D) E NÃO(Termina) FAÇA
    REMOVA os componentes do grupo da placa;
    GERE a matriz de custo para os elementos do grupo;
    RESOLVA o problema de associação linear para obter
            as novas posições dos componentes;
    CALCULE o valor da função objetivo G(i);
    SE G(i) < G(i-1)
    ENTÃO adotar as novas posições:
          Num. tentativas:= 0;
    SENÃO Num. Tentativas:= Num. Tentativas + 1;
    SE Num. Tentativas = D ENTÃO Termina:= VERDADE;
    I:= I + 1;
ATÉ QUE Termina;
```

II.3.3 - Troca de Pares (PI)

O algorítmo de troca de Pares ("Pairwise Interchange") é talvez o mais simples de implementar. A partir do posicionamento inicial, são escolhidos dois componentes para serem trocados de posição. Se a troca resultar numa redução de custo do posicionamento, a troca é aceita caso contrário, os componentes retornam às suas posições originais. Todos os $n = n*(n-1)/2$ pares possíveis de componentes são testados a cada ciclo.

Um ciclo inicia com a escolha de um componente para iniciar as trocas (componente primário). É experimentada a troca do componente primário com todos os demais componentes (componentes secudários), sendo aceitas aquelas que resultarem na diminuição da função de custo. Após todos os componentes serem escolhidos como primário o ciclo é

terminado. (BREUER [1972]) apresenta algumas alternativas para sequênciar a escolha dos componentes primários. Uma forma é ordenar os componentes de acordo com sua posição na placa. Esta forma de ordenação faz com que a sequência de escolha dos componentes primários mude após cada ciclo em que pelo menos uma troca tenha sido aceita.

II.3.4 - Troca na Vizinhança (NI)

O algorítmo de troca na vizinhança ("Neighborhood Interchange") é semelhante ao algorítmo PI. A diferença está no número de tentativas de trocas realizadas, ou seja, somente os componentes na vizinhança do módulo primário são escolhidos como secundários.

A definição da dimensão da vizinhança é um parâmetro do algorítmo e está diretamente relacionada ao número de trocas que serão experimentadas em cada ciclo. Quanto maior a vizinhança, maior será o número de trocas por ciclo.

II.3.5 - Relaxação Dirigida por Força (FDR)

No algorítmo de relaxação dirigida na força (HANAN et alii [1976]) é calculado o vetor força sobre cada componente n definido por:

$$F_n = \Sigma \ (C_{ni} * S_{ni})$$

onde C_{ni} é o número de ligações entre os componentes n e i e S_{ni} é o vetor distância de n para i. Utilizando o vetor Fn, é calculado o ponto alvo da placa onde a soma de todas as forças sobre o componente n se anula. A célula mais próxima do ponto alvo é então escolhida como a posição objetivo do componente. Se existir mais de uma célula, então a posição objetivo é escolhida arbitrariamente.

O algorítmo de relaxação dirigida por força é uma técnica que permite relaxar, isto é, reduzir sequencialmente a força resultante sobre os componentes, um de cada vez, conforme descrito abaixo:

i) primeiro, o componente escolhido para relaxação é posicionado na posição objetivo, se esta estiver vazia. Então, um novo componente é escolhido para relaxação na ordem decrescente do seu "valor - Q" definido como Q (m) = Σ C_{ni} que é essencialmente o número de redes às quais o componente n pertence.

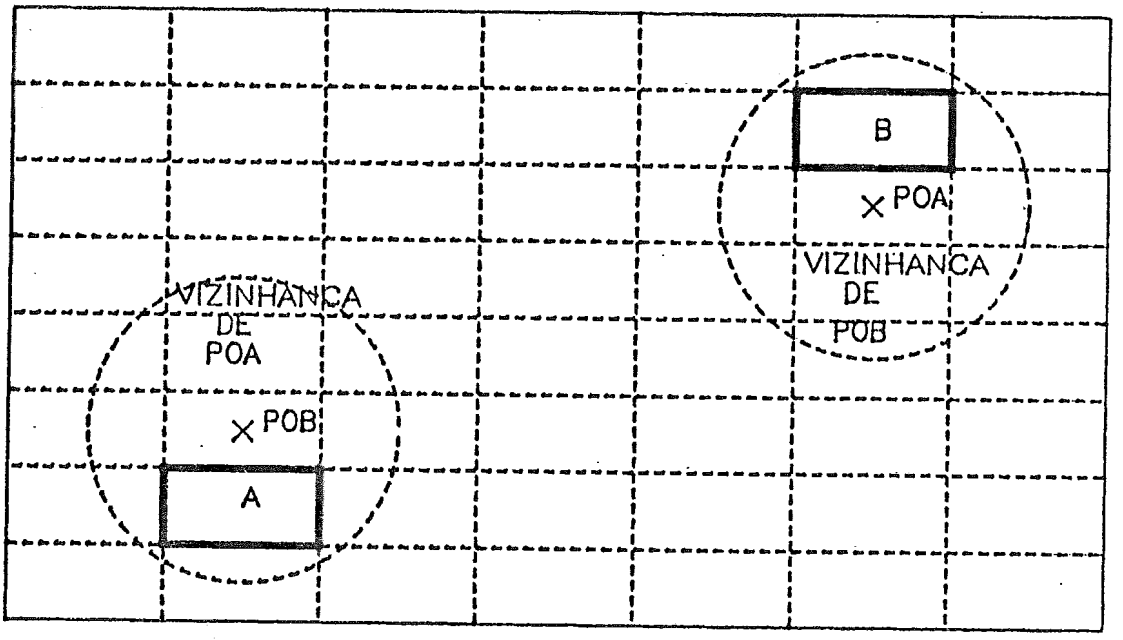
ii) se a posição objetivo estiver ocupada por outro componente, e ainda não estiver trancada, este componente é retirado e relaxado em seguida, iniciando uma lista.

iii) se a posição objetivo estiver trancada, o componente será colocado na posição não trancada de menor custo.

Uma lista termina quando um componente relaxado entra em uma posição vazia. Quando isso ocorre, é calculado o custo do novo leiaute, e se ele for menor que o custo original, a lista é aceita e todos os seus componentes são trancados em suas posições pelo restante do ciclo. Um ciclo termina quando todos os componentes iniciam uma lista, ou estão em uma lista aceitável.

II.3.6 - Relaxação Dirigida por Força com Troca de Pares (FDPR).

Da mesma forma que o FDR, o algorítmo de relaxação com troca de pares ("Force-Directed Pairwire Relaxation") baseia-se também no conceito de vetor força e ponto alvo onde a resultante das forças é zero. Um componente é escolhido para relaxação se existir algum componente M2 na vizinhança da posição alvo M1, tal que a posição alvo de M2 está, na vizinhança de M1, então é experimentada a troca de posição de M1 com M2, conforme ilustra a figura (II.4). A troca é aceita se resultar na diminuição do custo. Um ciclo termina quando todos os componentes tiverem sido escolhidos para relaxação.



POA - ponto alvo de A
POB - ponto alvo de B

Figura II.4 - Escolha de componentes para troca.

A E-vizinhança de uma célula pode ser definida em função da distância euclidiana ou da distância Manhattan. No primeiro caso ela é definida pela região circular que circunda a célula. No caso Manhattan, a E-vizinhança é definida pelo quadrado de aresta 2 e centro na célula em questão. Tanto o valor de E como o número de ciclos são parâmetros do algorítmo.

II.3.7 - Algorítmo de Corte Mínimo (MIN-CUT)

O problema da bi-partição em grafos, ou MIN-CUT é definido como:

Definição IV: dividir um grafo em dois conjuntos de tal forma que, os dois conjuntos tenham o mesmo tamanho (mesmo número de vértices), e o número de conexões entre os dois conjuntos seja minimizado (PREAS [1986] e SHING [1986]).

O problema de MIN-CUT definido dessa forma é NP-completo. Entretanto, devido à sua importância prática, muitas heurísticas foram desenvolvidas para obter uma partição quase ótima. Dentre as diversas heurísticas propostas na literatura, uma das mais conhecidas é a proposta por (KERNIGHAN [1970]).

Inicialmente, o algorítmo particiona o grafo em dois conjuntos de vértices de tamanhos aproximadamente iguais, e então troca os vértices entre os conjuntos para diminuir o número de conexões entre os conjuntos.

Essa técnica pode ser usada (SUARIS e KEDEM [1988]) na implementação de um algorítmo de posicionamento. Um algorítmo baseado nessa técnica particionaria, inicialmente, o circuito em dois conjuntos associando-os a duas áreas da placa. Na próxima etapa, cada uma dessas áreas seria particionada novamente. O processo continuaria, recursivamente até restar um único componente em cada conjunto. A figura (II.5) mostra algumas etapas desse algorítmo.

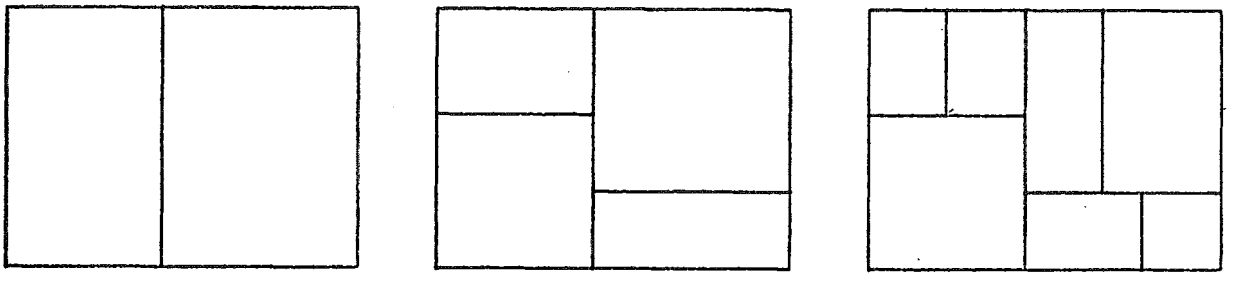


Figura II.5 - Posicionamento Baseado em Bisseção

II.3.8 - Posicionamento Dirigido por Força (FDP)

O método baseia-se no conceito de força resultante sobre os componentes (QUINN e BREUER [1979]). É montado um sistema de equações simultâneas baseado na topologia de interconexão dos componentes, cuja solução fornece a posição relativa ótima dos componentes na placa, tal que a força resultante sobre cada componente seja nula. As equações são montadas de forma a refletir forças atrativas entre componentes conectados e forças repulsivas entre componentes que não estão conectados.

A força resultante (QUINN e BREUER [1979]) sobre o componente i devida a todos os outros componentes da placa pode ser escrita como:

$$F_{xi} = \Sigma\,(Fat_x(i,j)+Frep_x(i,j)-FCMX,\ i=1,..,M \qquad (II.1)$$
$$F_{yi} = \Sigma\,(Fat_y(i,j)+frep_y(i,j)-FCMY,\ i=1,..,M$$

onde N é o número total de componentes dos quais i = 1,..,M são os M componentes móveis da placa, e i = N-M,..,N são os componentes fixos (conectores). Fat(i,j) é a força de atração entre os componentes i e j calculada pela lei de HOOKE. Se os componentes não estiverem concectados então Fat(i,j) = 0.

Frep(i,j) é a força de repulsão entre os componentes i e j cuja a direção é dada pelo vetor direção entre os componentes, e magnitude constante (definida experimentalmente).

FCMX e FCMY são os componentes nos eixos coordenados da resultante de todas as forças externas (força exercida pelos componentes fixos sobre os componentes móveis), atuando sobre o centro de massa CM, e é calculada como:

$$FCMX = \Sigma\ F_{xi}$$
$$FCMY = \Sigma\ F_{yi}$$

(QUINN e BREUER [1979] e FERREIRA [1989]) apresentam uma discussão detalhada da modelagem de cada parcela, bem como o seu efeito sobre o resultado final do método. A solução do sistema de equação são os pontos (xi,yi) de todos os componentes móveis da placa tais que $F_{xi} = 0$, e $F_{yi} = 0$, i=1,2,...,M, obtidos pelo método de Newton-Raphson Modificado (NRM) descrito em (STARK [1979]). É feita uma pequena modificação no método NRM para utilizar os valores das variáveis somente no final de cada iteração. Aplicando esta versão do NRM, cada novo ponto (X_{i+1}, Y_{i+1}) é obtido pela equação (II.2).

$$X_{i+1} = X_i - 0.5 * F_{xi} / D_{xi} \qquad (II.2)$$
$$Y_{i+1} = Y_i - 0.5 * F_{yi} / D_{yi}$$

onde F_{xi} e F_{yi} são obtidas pela equação (II.1) e D_{xi} e D_{yi} são respectivamente as derivadas parciais de F_{xi} e F_{yi} em relação a xi e yi. O fator 0.5 deve-se ao fato de que dois componentes quaisquer interagem com a mesma força ($F_{ij} = -F_{ji}$), sendo necessário mover cada um dos componentes apenas metade da distância devida.

O pseudo-código apresenta a forma resumida do algorítmo de posicionamento dirigido por força. O primeiro passo é definir as posições iniciais (xi, yi) dos componentes móveis através de um método qualquer (algorítmo CIP, gerar posições randomicamente, etc.). A posição inicial dos componentes praticamente não influência o resultado final do posicionamento, porém o número de iterações necessárias para alcançar esse posicionamento varia significativamente com as posições iniciais fornecidas ao algorítmo. A seguir apresentamos o algorítmo conforme proposto por (QUINN e BREUER [1979]).

```
DEFINIR a posição inicial (xi, yi) dos componentes;
REPITA
  PARA i:= 1 ATÉ M FAÇA
     Fxi:= Σ (Fatx (i,j) + Frepx (i,j));
     Fyi:= Σ (Faty (i,j) + Frepy (i,j));
     CALCULAR Dxi;
     CALCULAR Dyi;
     FCMX:= Σ Fxi;
     FCMY:= Σ Fyi;
     PARA i:= 1 ATÉ M FAÇA
        Fxi:= Fxi - FCMX/M;
        Fyi:= Fyi - FCMY/M;
        xi:= xi - 0,5 * Fxi/Dxi;
        yi:= yi - 0,5 * Fyi/Dyi;
        NORMA:= Σ ( Fxi  +  Fyi );
ATÉ QUE NORMA < erro;
```

(FERREIRA [1989]) apresenta uma descrição detalhada dos parâmetros do algorítmo, bem como a influência desses parâmetros no resultado do posicionamento.

II.4 - Características dos Algorítmos Clássicos

Analisando os algorítmos descritos na seção anterior, podemos observar que eles apresentam algumas características comuns:

i) a estratégia normalmente utilizada é transformar o problema de posicionamento em um problema de associação e utilizar a solução desse problema como solução do problema original.
ii) tratam o problema de posicionamento como um problema de programação matemática e/ou otimização em grafos.
iii) todos eles usam "força bruta", e muito pouco ou nenhum conhecimento empírico a respeito do problema, para achar a solução.
iv) o critério utilizado é minimizar uma dada função de custo, geralmente definida como o comprimento total dos fios.
v) tratam os componentes normalmente como se fossem pontos, sem considerar a sua área.
vi) não distinguem diferenças entre os componentes (ex.: memórias, CPUs, buffers, etc.).
vii) não são facilmente modificados, pois a estratégia do algorítmo se confunde com a sua programação.
viii) resolvem o problema de posicionamento de forma diferente do ser humano.

Esta tese propõe uma nova abordagem para problema de posicionamento de componentes eletrônicos em placas de circuito impresso, que possui as seguintes características:

i) pode ser aplicada a uma grande variedade de problemas, muito embora a implementação atual trate de placas digitais que possuem um grande número de barramentos.
ii) resolve o problema da forma semelhante à utilizada pelo ser humano.

iii) leva em consideração informações conhecidas sobre o problema no processo de busca da solução.
iv) a base de conhecimentos utilizada pelo sistema pode ser mais facilmente modificada e/ou expandida.
v) considera tanto a forma como a área dos componentes na obtenção do posicionamento.

CAPÍTULO III
Sistemas Especialistas Baseados em Conhecimento (SEBC)

O principal objetivo da Inteligência Artificial (IA) é transformar os computadores de máquinas que atualmente realizam tarefas laboriosas e repetitivas, em sistemas que realizem tarefas mais inteligentes e de forma semelhante ao ser humano (JOOBANI [1987]). Com os recentes avanços nas técnicas de IA, esse sonho está cada dia mais próximo da realidade. O sucesso dessas técnicas deve-se em grande parte aos chamados Sistemas Especialistas Baseados em Conhecimento (SEBC), cuja característica principal é a utilização maciça de conhecimento a respeito de um domínio específico, permitindo a esses sistemas imitarem o processo de solução de problemas dos seres humanos. A importância crescente das técnicas de IA, e em particular dos SEBCs pode ser observada pelas seguintes constatações (HAYES-ROTH [1984]):

. O número cada vez maior de aplicações utilizando SEBC (uns poucos na década de 70 comparado às mais de 100 aplicações existentes atualmente).
. O número cada vez maior de companhias comercializando esses sistemas e ferramentas relacionadas ao seu desenvolvimento. A quantidade de recursos gastos por vários governos e companhias para acompanhar o estado da arte neste campo.

Os esforços relacionados ao desenvolvimento de SEBC estão concentrados em três áreas:

Desenvolvimento de linguagens e ferramentas para capturar e representar o conhecimento humano (engenharia do conhecimento), dentre as quais devemos destacar as linguagens LISP, OPS5 e

PROLOG, e as ferramentas de desenvolvimento M1 e KEE (HARMON e KING [1985]).

. Desenvolvimento de máquinas capazes de executar linguagens de processamento símbolico tais como LISP e PROLOG mais eficientemente. Exemplos desses esforços são: a máquina LISP da Symbolics; e a máquina de inferência sequencial PROLOG do projeto japonês de quinta geração (JOOBANI [1987]).

. Desenvolvimento de aplicações para a solução de problemas específicos. Estes sistemas são usados atualmente numa grande variedade de aplicações que vão desde sistemas de diagnósticos até sistemas de auxílio a projeto e educação. (HARMON e KING [1985]) apresentam alguns sistemas especialistas baseados em conhecimento, dentre os quais devemos destacar os sistemas: DENDRAL, MYCIN, R1, TALIB, MACSYMA, etc.

A seção III.1 descreve de forma resumida a arquitetura dos sistemas baseados em conhecimento. O restante do capítulo descreve a linguagem PROLOG e a aplicabilidade de SEBC ao problema de posicionamento de componentes eletrônicos em placas de circuito impresso.

. Esses conhecimentos devem ser basicamente de natureza heurística ao invés de algorítmica, e manipular preferencialmente descrições símbólicas.
. Durante o processo de busca da solução, o especialista trabalha normalmente com informações incertas ou incompletas.
. É necessário avaliar várias soluções para o problema simultaneamente.
. Sejam problemas importantes, difíceis, e adequados à utilização das técnicas de IA.

III.2 - Arquitetura dos SEBC

Segundo o Prof. Edward Feigenbaum, da Universidade de Stanford, um sistema especialista pode ser definido como:

Definição V: "Um programa de computador que usa conhecimento e mecanismos de inferência para resolver problemas que sejam difíceis e requeiram um especialista humano para obter a solução. Os profissionais que projetam e constróem sistemas especialistas são chamados de Engenheiros do Conhecimento (HARMON e KING [1985])".

De acordo com a definição acima, os sistemas especialistas baseados em conhecimento (ou simplesmente sistemas especialistas) podem ser divididos em duas partes principais: a base de conhecimento; e a máquina de inferência. A figura (III.1) ilustra a arquitetura de um sistema especialista genérico.

A base de conhecimento consiste de fatos e heurísticas. Os fatos são conjuntos de informações publicamente disponíveis e compartilhados pelos especialistas, sobre os quais todos geralmente concordam. As heurísticas, por outro lado, são regras empíricas (obtidas por experiência e bom senso), normalmente desenvolvidas pelo próprio especialista (ou ensinadas por outro especialista) durante o seu período de aprendizado com o problema. As heurísticas permitem que o especialista descarte alternativas, reduzindo o espaço de busca,

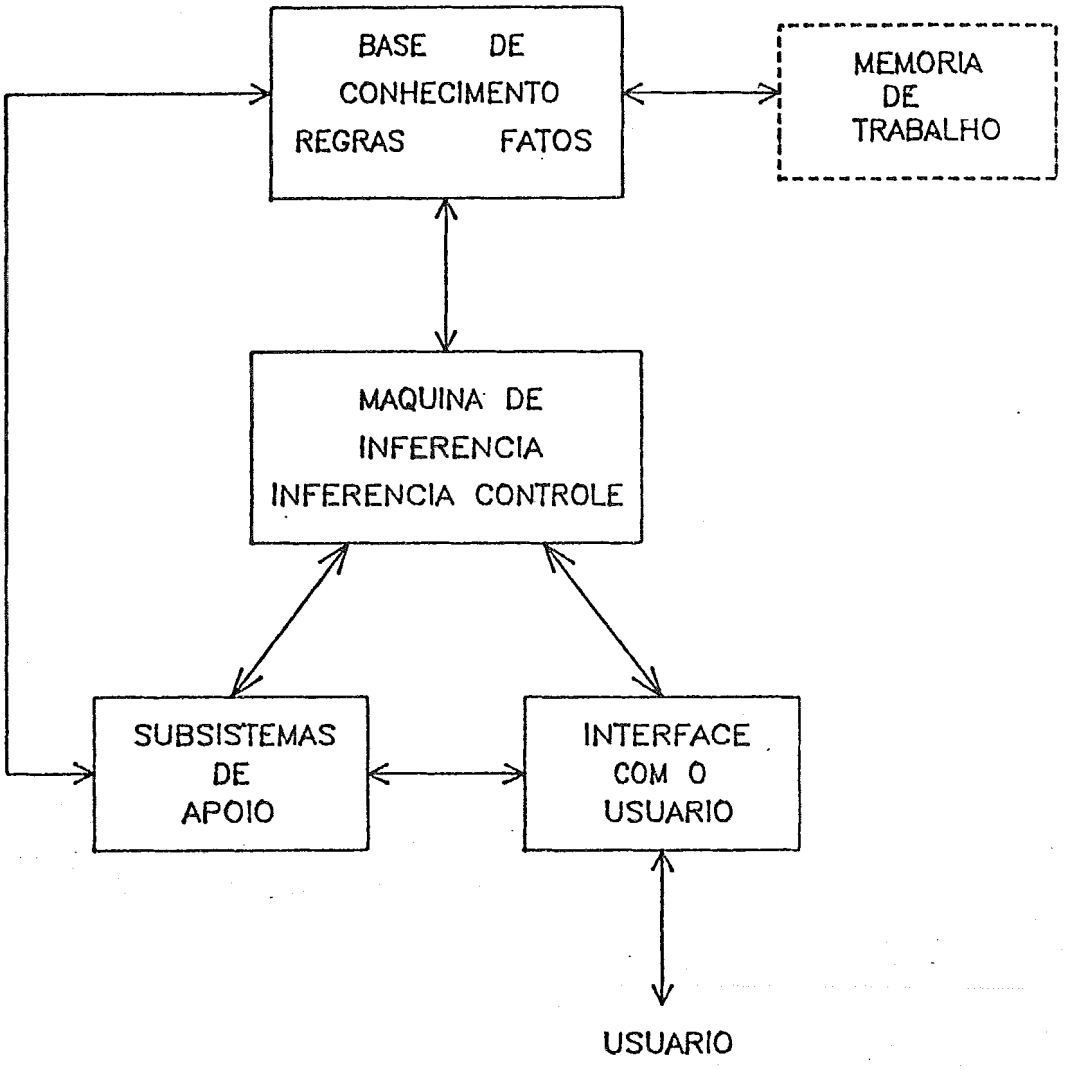


Figura III.1 - Arquitetura de um Sistema Especialista Genérico.

tornando o problema computacionalmente tratável. O desempenho de um sistema especialista depende diretamente do tamanho e da qualidade de sua base de conhecimento.

A máquina de inferência contém as estratégias de inferência e controle que o sistema especialista usa para manipular os fatos e as regras, bem como informações externas, para achar a solução ou soluções do problema (CARPINTEIRO [1988]).

III.2.1 - Formas de Representação do Conhecimento

As informações podem ser armazenadas na base de conhecimento de várias maneiras diferentes (HARMON e KING [1985]). Esta seção apresenta três métodos diferentes para codificar essas informações. Cada um possui vantagens e desvantagens. A escolha de um determinado método depende do problema em questão.

a) Redes Semânticas

As redes semânticas são consideradas o método de representação mais genérico, e também um dos mais antigos em IA. Nelas o conhecimento é representado por um conjunto de nós conectados entre si por meio de arcos, aos quais estão associados textos que identificam o conhecimento armazenado na rede. Os nós representam objetos e descritores, enquanto os arcos são usados para relacioná-los entre si. Não existe nenhuma restrição sobre como associar os textos aos arcos e aos nós. Entretanto, é comum utilizarmos algumas convenções:

. Objetos são entidades físicas que podem ser vistas ou tocadas (ex.: chapéu, casaco, pessoas, etc.), ou entidades conceituadas tais como ações, eventos, ou categorias abstradas (ex.: clima, números, etc.).

. Descritores fornecem informações adicionais sobre os objetos. Quente e bonito, por exemplo, armazenam informações sobre um determinado tipo de casaco.

. Arcos podem representar quaisquer relacionamentos, e muitas vezes estão associados a conceitos tais como: é-um, tem-um, exemplo-de, parte-de, etc..

A figura (III.2) ilustra a representação de uma base de conhecimento utilizando redes semânticas. As principais vantagens desse método são a sua flexibilidade e generalidade.

b) Quadros ("frames")

Quadros são estruturas que reúnem conhecimento sobre um conceito particular. Um quadro é uma descrição de um objeto que contém entradas ("slots") para todas as informações associadas ao objeto. As entradas podem conter valores, ponteiros para outros quadros, conjuntos de regras, e procedimentos (HARMON e KING [1985]). Por exemplo, um quadro do objeto pássaro poderia ter como valores: asas, bico, voar, e penas. Portanto, quando o sistema especialista se referir a pássaros, saberá que têm penas, asas, bico e que voa.

c) Regras

As regras, também chamadas regras de produção, são o método popular de representação de conhecimento utilizado nos sistemas especialistas (CARPINTEIRO [1988]). Muito embora a sintaxe das regras varie muito entre os diversos sistemas existentes, ela pode ser genéricamente escrita na forma: SE um conjunto de condições é satisfeito ENTÃO um conjunto de ações pode ser realizada (JOOBANI [1987]). A figura (III.3) mostra uma regra simples extraída do sistema especialista MYCIN (HARMON e KING [1985]).

Cada uma das quatro partes da premissa é chamada uma cláusula-SE. Nesse exemplo, a conclusão contém apenas uma ação, ou cláusula-ENTÃO. Em muitas situações, as ações de uma regra podem não ser completamente verdadeiras. No exemplo anterior, a conclusão está associada a um valor numérico (0.6), que reflete a probabilidade, ou a

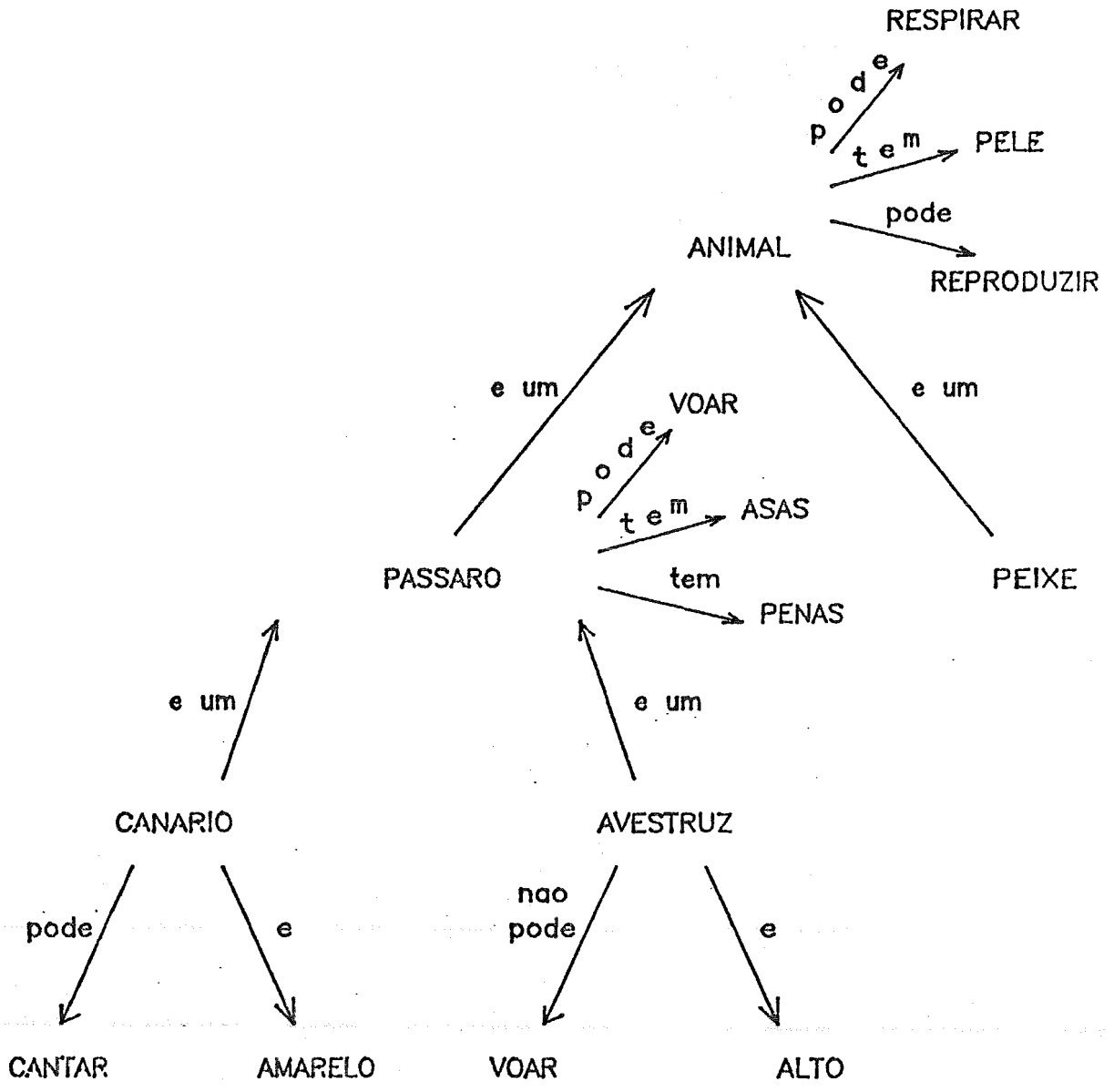


Figura III.2 - Exemplo de Rede Semântica

credibilidade da conclusão. Desta forma as regras podem ser usadas também para representar relacionamentos incertos (HARMON e KING [1985] e CARPINTEIRO [1988]).

Regra:

premissa
SE o ambiente da cultura é sangue E
a morfologia do organismo é circular E
o teste de oxidação de Gram é negativo E
o paciente é um hospedeiro comprometido

conclusão
ENTÃO há evidências (0.6) de que a identidade do organismo é Pseudomonas-aeruginosa.

figura III.3 - Exemplo de regra no sistema MYCIN.

III.2.2 - Máquina de Inferência

Nos sistemas especialistas, o processo de busca da solução é feito pela máquina de inferência, a qual é responsável por chamar e utilizar as informações armazenadas na base de conhecimento. A mesma é encarregada de executar três tarefas principais: examinar os fatos e regras da base de conhecimento, e acrescentar novos fatos sempre que possível; determinar a ordem em que a inferência é feita; e conduzir o processo de consulta ao usuário (HARMON e KING [1985]).

A máquina de inferência contém as estratégias de inferência e controle que o sistema usa para manipular os fatos e as regras. A estratégia de inferência determina a forma como novos fatos são obtidos a partir das regras e fatos conhecidos. Uma estratégia simples de inferência muito usada em sistemas especialistas (HARMON e KING [1985]) é a regra lógica chamada **modus ponens**. A partir da regra IF A ENTÃO B, modus ponens afirma que se A for verdadeiro, então é válido concluir que B é verdadeiro.

A estratégia de controle é responsável por duas tarefas principais: como as regras e os fatos estão armazenados numa base de conhecimento, o sistema deve

decidir onde o processo de inferência deve começar; Nos casos em que duas ou mais regras tiverem suas cláusulas-SE satisfeitas, a máquina de inferência deve escolher qual a cláusula-ENTÃO que será ativada em primeiro lugar. (HARMON e KING [1985]) apresenta algumas alternativas que podem ser utilizadas como estratégia de inferência e de controle

III.3 - O Ambiente PROLOG

As técnicas de programação convencionais têm sido utilizadas para desenvolver a maioria dos sistemas de processamento de dados existentes. Estes sistemas geralmente armazenam e processam informações utilizando algorítmos isto é, um conjunto de instruções simples, executadas passo-a-passo, que permite obter a solução correta se os dados de entrada estiverem corretos. As linguagens de programação utilizadas na implementação desses sistemas, chamadas linguagens imperativas, possuem duas características em comum: o programador deve descrever precisamente como o problema será resolvido; o método de solução do problema faz parte da própria implementação do sistema.

As linguagens declarativas possibilitam uma nova forma de programação. Nestas linguagens, o programador fornece apenas uma descrição do problema e as regras básicas para resolvê-lo, deixando para a máquina os detalhes de como obter a solução. Este modelo de programação apresenta as seguintes vantagens:

. o programador não desvia a sua atenção do problema para se preocupar com detalhes de implementação.

. os programas escritos em linguagem declarativa não dependem da arquitetura da máquina onde serão executados.

. cada definição individual isola um pequeno fragmento do problema, facilitando a sua compreensão.

. o sistema pode ser facilmente expandido, bastando apenas inserir novas definições ao conjunto das definições conhecidas.

O desenvolvimento de sistemas especialistas é feito normalmente utilizando ambientes ("shells") ou linguagens declarativas. Dentre as diversas linguagens existentes atualmente, deve-se destacar as linguagens OPS5 e PROLOG que têm sido utilizadas no desenvolvimento de vários sistemas especialistas (MORI et alii [1984], LIN e GAJSKI [1987] JOOBANI [1987] e DEJESUS et alii [1986]). A seguir apresentaremos, de froma resumida, as principais características da linguagem PROLOG escolhida para implementar o protótipo do sistema SAP.

III.3.1 - Fundamentos de PROLOG

A primeira versão oficial do PROLOG ("PROgramming in LOGic") foi desenvolvida na Universidade de Marselha na França, por Alain Colmerauher em 1970. Atualmente, PROLOG é uma ferramenta muito importante na programação de aplicações utilizando técnicas de IA, e em particular no desenvolvimento de sistemas especialistas (BORLAND [1988]).

As linguagens convencionais são, em geral, uma sequência pré-definida e fixa de instruções. O PROLOG, por outro lado, chega à solução do problema inferindo logicamente novas informações do conjunto de informações conhecidas. Um programa PROLOG é uma coleção de fatos e regras que permitem ao sistema inferir conclusões sobre esses fatos. O mundo conhecido do PROLOG é o conjunto finito de fatos e regras armazenados na base de conhecimento do sistema.

PROLOG é baseado em cláusulas de HORN, as quais são um caso particular de um sistema formal chamado lógica de predicados. Lógica de Predicados é uma maneira simples, elegante e poderosa de representar conhecimento (HARMON e KING [1985] cap. 4). Cláusulas são ou fatos ou regras, usados para exprimir os relacionamentos entre objetos.

Os fatos exprimem propriedades ou relações estáticas entre objetos, e são simbolizados em linguagem natural por uma sentença.

ex.:

1) A rede VCC está conectada ao pino 14 do componente U5.
2) A rede GND está conectada ao pino 7 do componente U5.
3) O componente U5 é um dispositivo eletrônico 74LS00.
4) O dispositivo 8086 é uma cpu.

Em PROLOG, o fato consiste do nome da propriedade ou relacionamento seguido do objeto, ou objetos escritos entre parênteses e terminado por um ponto. conforme mostra o exemplo abaixo. Chamam-se predicados e argumentos ao nome dos relacionamentos e dos objetos entre parênteses, respectivamente.

ex.:

1) conexão (VCC,U5,14)
2) conexão (GND,U5, 7)
3) dispositivo (U5,74LS00)
4) cpu (8086)

Um fato representa uma única ocorrência da propriedade ou da relação entre objetos, e como podemos notar ele é auto-explicativo, ou seja, o sistema não necessita buscar nenhuma confirmação para o fato. Portanto, os fatos são usados como base para o processo de inferência (BORLAND [1988]).

As regras permitem ao sistema inferir novos fatos a partir de fatos já conhecidos. Regras expressam propriedades ou relações entre objetos que são verdadeiras quando um conjunto de outras propriedades ou relações é verdadeiro. (BORLAND [1988] cap. 3) descreve a sintaxe da linguagem PROLOG. O exemplo abaixo mostra uma regra que

pode ser usada para agrupar componentes de memória RAM ou ROM numa placa de circuito impresso.

Ex.:

. em linguagem natural

SE ((C_1 é memória RAM E
C_2 é memória RAM) OU
(C_1 é memória ROM E
C_2 é memória ROM)) E
C_1 e C2 estão ligados no mesmo barramento
ENTÃO agrupe C_1 e C_2 no barramento B_1

. escrito em PROLOG

```
Agrupe (B1, C1, C2): -
   Ligado_A_Barramento (B1, C1),
   Ligado_A_Barramento (B1, C2),
   RAM(C1),
   RAM(C2).

Agrupe (B1, C1, C2): -
   Ligado_A_Barramento (B1, C1),
   Ligado_A_Barramento (B1, C2),
   ROM(C1),
   ROM(C2).
```

Uma regra é constituída de uma cabeça e um corpo. A cabeça é separada do corpo por intermédio de dois pontos seguidos de um hífen. Tal separador é pronunciado SE. O corpo é constituído de um ou mais objetivos separados por vírgulas, onde cada objetivo é uma cláusula PROLOG. Para satisfazer uma regra, o interpretador tenta satisfazer cada objetivo do seu corpo, partindo da esquerda para a direita, procurando-os na base de conhecimento (fatos e regras). Se todos os objetivos forem satisfeitos, então a cabeça da regra é satisfeita.

III.3.2 - Mecanismo de Inferência

Programar em PROLOG consiste em definir um conjunto de informações conhecidas, as quais estão armazenadas na base

de conhecimento (formada de fatos e regras), e realizar perguntas sobre estas informações. A máquina de inferência embutida no interpretador do sistema contém o mecanismo de inferência responsável por pesquisar e recuperar informações armazenadas pelo programa associando perguntas às possíveis respostas. PROLOG tenta inferir se uma hipótese (ou seja, uma pergunta) é verdadeira comparando-a ao conjunto das informações conhecidas. Seja a base de conhecimento abaixo:

```
pai(josé, antonio).
pai(antonio, júlio).
pai(antonio, solimá).
mãe(jadir, júlio).
mãe(jadir, solimá).
pai(ary, sandra).
pai(ary, artur).
mãe(moema, sandra).
mãe(moema, artur).

avo(Avo, Neto):- pai(Avo, X), pai(X, Neto).
avo(Avo, Neto):- pai(Avo, X), mãe(X, Neto).
```

A sintaxe das perguntas é semelhante à declaração dos fatos exceto que precedida de um ponto de interrogação e um hífen, conforme o exemplo abaixo:

ex:

```
. em linguagem natural
antonio é pai de júlio?
antonio é pai de sandra?

. em PROLOG
?-pai(antonio, júlio).
?-pai(antonio, sandra).
```

Se houver alguma cláusula, ou conjunto de cláusulas, na base de conhecimento que permita validar a hipótese, o sistema responderá SIM, caso contrário a resposta é NÃO. No

exemplo anterior o sistema responde afirmativamente para **antonio é pai de júlio?** e negativamente para **antonio é pai de sandra?.**

As perguntas são feitas utilizando fatos (como no exemplo anterior) ou regras. Para responder a um fato, a máquina de inferência percorre a base de conhecimento, a partir do topo até o final, identificando cada ocorrência do fato tal que os predicados e os argumentos correspondentes sejam idênticos. Este processo é chamado de unificação. Sempre que a pergunta unificar com alguma ocorrência na base de conhecimento a resposta é SIM. Caso contrário a resposta é NÃO.

Para validar uma regra, a máquina de inferência tenta inicialmente unificar a pergunta com alguma ocorrência da regra. Se a unificação for satisfeita, a máquina reinicia o processo sobre cada um dos objetivos do corpo da regra tentando prová-los. Se todos os objetivos forem verdadeiros então a regra é satisfeita. Caso contrário, o interpretador PROLOG realiza um retrocesso ("backtrack") e procura a próxima ocorrência da regra. O processo é repetido tantas vezes quantas forem necessárias até findarem as ocorrências, ou até que uma delas seja satisfeita.

Outra característica importante do PROLOG, além de responder perguntas, é a sua capacidade de avaliar alternativas e encontrar todas as soluções possíveis de um problema. O mecanismo de retrocesso permite que o interpretador retorne de um objetivo e investigue outras formas para resolver o problema. Assim, nos exemplos abaixo a máquina de inferência deve descobrir todas as ocorrências que satisfazem às perguntas.

ex:

```
. em linguagem natural
jadir é mãe de quem?
quem é o avô de solimá?

. em PROLOG
?-mãe(jadir, Filhos).
?-avô(Avô, solimá).
```

Utilizando a base de conhecimento anterior teríamos as seguintes respostas:

```
?-mãe(jadir, Filhos).
Filhos = júlio
Filhos = solimá

?-avô(Avô, solimá).
Avô = josé
```

No exemplo acima, **Filhos** é uma variável livre, pois não está associada a nenhum valor. Ao unificar uma cláusula, a máquina de inferência associa as variáveis livres ao valor dos atributos correspondentes. O mecanismo de retrocesso permite identificar todas as ocorrências da base de conhecimento que satisfazem à pergunta, e consequentemente obter todas as soluções para a resposta.

PROLOG é uma linguagem poderosa que oferece muitos recursos para a implementação de programas que utilizam técnicas de IA. Por esta razão ela vem sendo utilizada no desenvolvimento de vários SEBCs (HARMON e KING [1985]), e em particular no desenvolvimento de ferramentas de CAD/CAE (MORI et alii [1984], DEJESUS et alii [1986] e JOOBANI [1987]).

CAPÍTULO IV
Estratégia do SAP

O capítulo II apresentou o resumo de alguns critérios e técnicas (algorítmos) que podem ser utilizados para decidir onde e em que ordem os componentes são posicionados na PCI. Este capítulo mostra quais as técnicas e critérios utilizados pelo sistema SAP, e de que forma eles estão organizados dentro da estratégia adotada para resolver o problema de posicionamento.

IV.1 - A Estratégia

Definição VI: Estratégia é a arte de aplicar os meios disponíveis com vista à consecução de objetivos ou tarefas específicas (FERREIRA [1986]).

No caso do SAP, a tarefa em questão é determinar o posicionamento "ótimo" de componentes eletrônicos dentro de uma PCI, de acordo com uma das métricas descritas anteriormente. A proposta do SAP é observar o especialista humano e utilizar uma estratégia semelhante para resolver o problema.

No período de janeiro a outubro de 1988 foi implantado no Departamento de Eletrônica do CEPEL o sistema CADD para a confecção de leiautes de placas de circuito impresso (PIMENTEL e BANDIM [1988]), que contou com a participação de dois especialistas em leiaute. Durante este período, o autor teve a oportunidade de observar e analisar a estratégia utilizada por estes especialistas, tendo concluído que:

i) os especialistas inicialmente particionam os circuitos em módulos funcionalmente independentes, levando em consideração a estrutura dos barramentos, e a funcionalidade dos módulos.

ii) eles determinam o posicionamento relativo dos módulos na PCI baseado na função de cada bloco, a posição e o número de conectores externos, e outras condições.
iii) realizam o posicionamento dos componentes pertencentes aos módulos na placa.
iv) otimizam o posicionamento relativo para minimizar o comprimento total dos fios aumentando a roteabilidade da placa.

Da mesma forma que a estratégia humana, a estratégia do SAP está organizada em três etapas: particionamento, posicionamento relativo e otimização. Estas etapas foram implementadas utilizando regras empíricas para representar o conhecimento do especialista sobre as diversas etapas do problema. Nas situações onde não foi possível obter regras (ou por que não existem, ou por que é difícil obtê-las), e em alguns casos, por motivo de eficiência, o sistema SAP utiliza algorítmos convencionais. Cabe notar que inserindo novas regras na base de conhecimento, a parte do circuito tratado por algorítmos tende a diminuir.

O restante deste capítulo descreve em detalhes as funções que são executadas durante cada etapa e as situações onde são utilizadas regras e/ou algorítmos. O capítulo V descreve a implementação do sistema.

IV.2 - Definições

Embora a estratégia utilizada no sistema SAP seja de uso geral, a implementação atual trata de placas digitais estruturadas em barramentos. Estas placas são compostas normalmente pelos seguintes elementos: componentes, redes, barramentos, conectores e módulos. Para entendermos a estratégia do SAP, é necessário apresentarmos primeiro algumas definições.

IV.2.1 - Elementos que Compõem a Placa

As placas de circuito impresso são formadas de conjuntos de dispositivos eletrônicos, chamados componentes. Conjuntos interconectados de componentes

definem uma placa, e podem ser circuitos integrados (ex.: memórias, microprocessadores, amplificadores operacionais, etc.), componentes analógicos (ex.: resistores, capacitores, transformadores, etc.), ou conectores. Os conectores são usados normalmente para entrada e saída de sinais na placa (comunicação com o mundo externo).

Os componentes têm pinos de conexão localizados normalmente nas laterais. Vários conjuntos destes pinos estão conectados entre si formando redes de sinais. Como cada componente contém dois ou mais pinos, geralmente cada componente pertence a duas ou mais redes. Quando um conjunto de redes carrega uma informação de um mesmo tipo, ele é chamado de barramento (ex.: barramento de dados; e barramento de endereços), e cada rede do conjunto é representada pelo nome do barramento seguido de um número que identifica a rede.

ex.: barramento A ==> rede A0
rede A1
:
rede A15

barramento D ==> rede D0
rede D1
:
rede D7

No sistema SAP os componentes são agrupados de acordo com suas características funcionais formando módulos, com o objetivo de reduzir a complexibilidade do problema.

IV.2.2 - Classificação das Redes

As redes podem ser classificados quanto ao número de componentes interligados em (ODAWARA et alii [1983]):

redes globais ou do sistema - são redes que interligam muitas partes do circuito. Em geral, essas redes são mais críticas que as demais redes do circuito, e devem ser tratadas, portanto, de forma prioritária (ex.: VCC, GND, clock, etc.)

. redes locais ou randômicas - são todas as redes que não pertencem a barramentos e não são redes globais. Estas redes interligam normalmente um conjunto pequeno de componentes. Na maioria dos circuitos, mais de 80% das redes interligam no máximo três componentes (GOTO [1978] e WEIS [1988]).

IV.2.3 - Conectividade

A conectividade de um barramento é uma medida do afastamento entre barramentos e conectores. Ela é definida como o número de componentes entre o barramento e o conector. Na figura (IV.1) o barramento A está ligado diretamente ao conector externo (conectividade = 0). Os barramentos B e C possuem conectividade igual a um e dois respectivamente.

A conectividade de um barramento tem grande influência na forma como os seus componentes serão posicionados na PCI, ou seja, quanto menor for a conectividade do barramento, tanto mais próximo do conector devem ser posicionados os componentes ligados ao barramento. Por outro lado, componentes ligados a barramentos com conectividade muito grande (fracamente conectados a conectores) devem ser posicionados na região central da PCI.

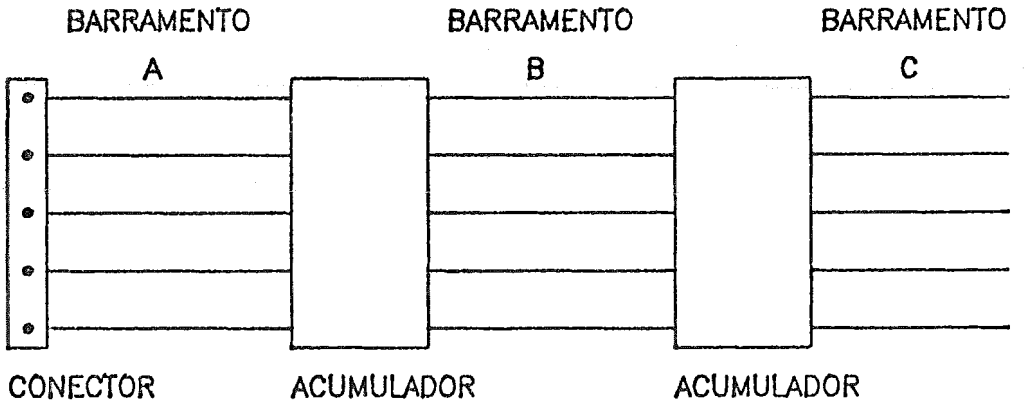


Figura IV.1 - Conectividade dos Barramentos.

Os barramentos podem ser classificados quanto à conectividade em: primários e secundários. Barramentos primários são aqueles que estão conectados diretamente a um conector e secundários são os que não possuem ligaçao direta com os mesmos. Na figura (IV.1) o barramento A é primário, enquanto que os barramentos B e C são secundários.

IV.2.4 - Comprimento

Outro atributo importante de um barramento (ODAWARA et alii [1987]) é o seu comprimento, que é definido como o número de componentes ligados a ele. Os barramentos podem ser classificados quanto ao comprimento em duas categorias: barramentos globais ou do sistema e barramentos locais.

Os barramentos globais são aqueles que estão interligados com um número muito grande de componentes tais como: CPUs, RAMs, ROMs, processadores de entrada/saída, etc.. Os barramentos locais estão ligados a poucos componentes. (ODAWARA et alii [1987]) determinou experimentalmente que um valor de comprimento igual a dez é ótimo para separar os barramentos locais e globais. Desta forma, barramentos com comprimentos maiores do que dez são globais, enquanto que todos os outros são locais. A figura (IV.2) mostra o mapa de comprimentos para duas placas reais. A curva a) foi extraída de uma placa digital micro-processada. Os picos nos valores de 20 e 30 correspondem a barramentos globais. A curva b) foi obtida de uma placa de memória. O pico no valor de comprimento de 35 corresponde ao barramento de endereço que conecta todos os elementos de memória. Todos os circuitos digitais estruturados em barramento tem curvas características semelhantes (ODAWARA et alii [1987]) às mostradas na figura (IV.2).

Do ponto de vista do posicionamento, os componentes ligados a um barramento local devem ser posicionados próximos uns dos outros a fim de economizar espaço para o roteamento. No caso dos barramentos globais a estrutura do barramento tem uma influência maior na qualidade final do leiaute do que minimizar o comprimento físico do

barramento. De acordo com o comprimento e a conectividade, os barramentos podem ser classificados em:

- barramentos primários globais (BPG)
- barramentos primários locais (BPL)
- barramentos secundários globais (BSG)
- barramentos secundários locais (BSL)

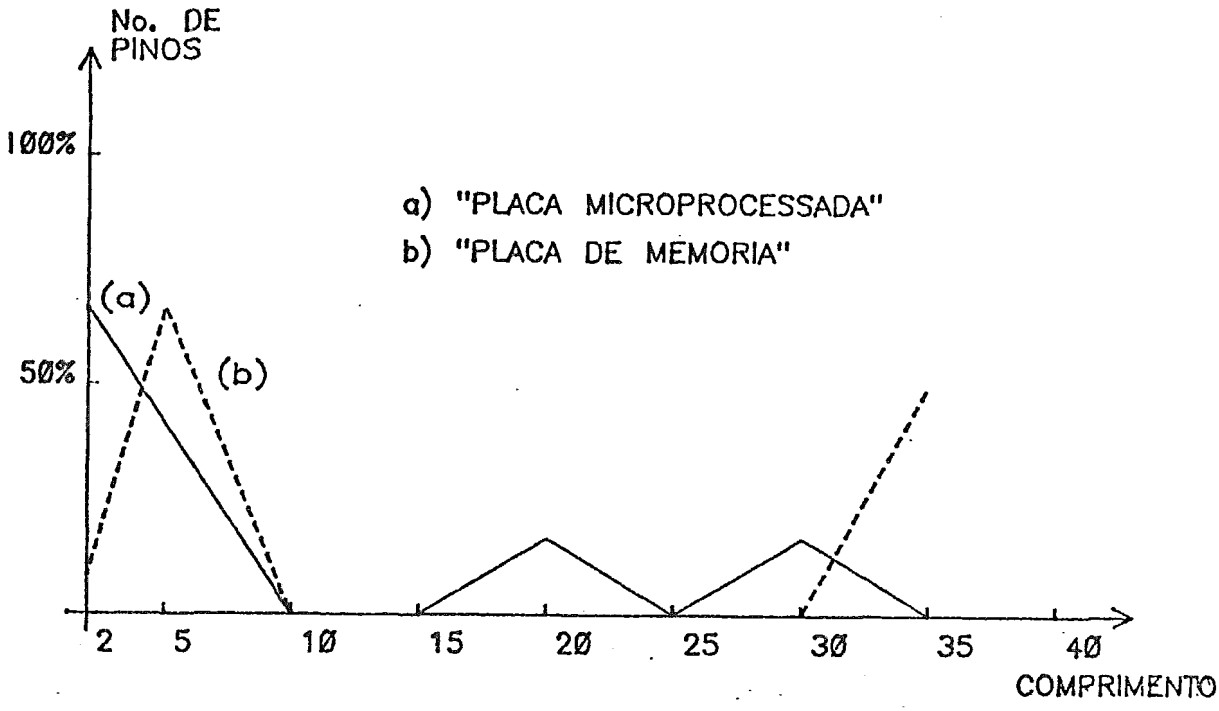


Figura IV.2 - Mapa de Comprimentos

Os BPGs e BPLs estão conectados diretamente aos conectores externos. Portanto, os componentes ligados a esses barramentos devem ser posicionados durante as primeiras etapas do processo de posicionamento. Os componentes ligados aos BSGs e BSLs devem ser posicionados no espaço que restar da placa.

IV.2.5 - Largura

A largura de um barramento é definida como o número de redes do barramento, e está relacionada à área de placa que será gasta para rotear os barramentos. Ela não tem importância decisiva na qualidade final do leiaute, e geralmente é usada pelos especialistas em leiaute como um

critério auxiliar na escolha de duas soluções que parecem igualmente boas.

IV.3 - Particionamento

Atualmente, a maioria dos circuitos eletrônicos é constituída de componentes análogicos e digitais, bem como circuitos integrados LSI e VLSI ("Very Large Scale Integration") cujos pinos estão normalmente conectados a barramentos. Do ponto de vista do posicionamento, é conveniente identificar dois grupos que possuem características distintas: uma é a parte estruturada em barramentos que consiste das redes que pertencem a barramentos e dos componentes ligados a essas redes, tais como: CPUs memórias e processadores de entrada/saída; o restante do circuito é formado pelas redes randômicas e pelos componentes que não estão ligados a barramentos.

No sistema SAP, a primeira etapa do particionamento consiste em separar a parte randômica da parte estruturada em barramentos. A parte randômica não é tratada nessa etapa devido à dificuldade de identificar regras gerais que permitam agrupar (fundir) os componentes pertencentes a ela. Na implementação atual do SAP, essa tarefa deve ser feita manualmente pelo usuário. Realizar o particionamento manual da parte randômica é fortemente recomendado, embora não seja necessário, tendo em vista que o sistema é capaz de tratar esse tipo de componentes. Todavia, foi observado que em alguns casos o particionamento manual melhora significativamente a qualidade do leiaute final.

Os componentes da parte estruturada em barramentos são agrupados (fundidos) formando módulos com o objetivo de diminuir a complexibilidade do problema e garantir que componentes fortemente relacionados sejam posicionados próximos uns dos outros, minimizando o comprimento total dos fios dos barramentos. Esta etapa é realizada através de um conjunto de regras empíricas para:

. agrupar estruturas para as quais o senso comum conhece a solução, tais como bancos de memórias ligadas ao mesmo barramento.

. agrupar componentes que possuem muitas ligações em comum.

. agrupar componentes que possuem a mesma função e estejam ligados aos mesmos barramentos.

IV.4 - Posicionamento Relativo dos Componentes

O posicionamento relativo dos componentes ligados a barramentos é realizado em três etapas: primeiro é feito o posicionamento relativo (ordenação) dos componentes dentro dos módulos; depois os módulos são posicionados dentro da placa; e finalmente os componentes são posicionados dentro da placa.

IV.4.1 - Ordenação Dentro dos Módulos

Para alguns módulos tais como bancos de memória, o posicionamento relativo interno dos componentes é conhecido (matriz de componentes). Nesses casos, é ativada uma regra que realiza a ordenação dos componentes dentro dos módulos. Todos os módulos que não forem ordenados por este processo têm o seu componente semente, que é o componente com maior número de ligações com os demais, posicionado no centro do módulo, deixando livre os outros componentes para serem posicionados numa etapa posterior.

IV.4.2 - Posicionamento dos Módulos na Placa

O posicionamento dos módulos dentro da placa é realizado por um conjunto de regras e algorítimos. As regras posicionam os módulos ligados a barramentos primários numa posição adjacente ao conector, visando minimizar o comprimento desses barramentos, e liberar o centro da placa para os barramentos secundários. A posição relativa dos módulos ligados aos BSLs e BSGs é determinada pelo método dirigido por forças (FDP), descrito na seção II.3.8, mantendo os módulos já posicionados fixos em suas posições.

Determinado o posicionamento relativo, a próxima etapa é eliminar as superposições de módulos que por acaso existam. (WIPFLER [1982], ODAWARA et alii [1987] e SHA e DUTTON [1985]), descrevem três métodos que podem ser

utilizados para resolver o problema. A técnica utilizada no SAP consiste em dividir a área da placa em uma matriz de células ("slots") e associar módulos a células, minimizando a distorção total do posicionamento relativo (QUINN e BREUER [1979]). O apêndice C apresenta um circuito de teste, o resultado do posicionamento relativo, e da etapa de eliminação das superposições aplicadas sobre esse circuito.

IV.5 - Posicionamento dos Componentes na Placa

A inserção dos componentes na placa é feita em três etapas:

. inicialmente são posicionados os componentes pertencentes aos módulos ordenados na etapa anterior.

. os componentes sementes dos módulos não-ordenados são fixados no centro do módulo, e a posição relativa dos outros componentes da parte estruturada em barramentos é obtida usando o algorítmo FDP.

. finalmente são posicionados os componentes pertencentes à parte randômica do circuito.

IV.6 - Otimização

A última etapa é otimizar o posicionamento final (incluindo todos os componentes da placa), para diminuir o comprimento total dos fios e equilibrar a sua distribuição sobre a placa, evitando congestionamentos. Essa otimização é feita pelo algorítmo de Steinberg (STEINBERG [1961]) e modificado por (ODAWARA et alii [1982]) para tratar componentes (ou módulos) de formas e tamanhos variados, utilizando a função de custo descrita na seção II.2.3.

CAPÍTULO V
Implementação

O projeto e implementação do SAP foram realizados em quatro etapas distintas:

. familiarização com o problema.
. aquisição inicial de conhecimento junto ao especialista.
. escolha de um formalismo para representar esse conhecimento.
. modificação modular e incremental do sistema, baseado num conjunto de casos típicos, até alcançar um desempenho satisfatório.

A familiarização com o problema de posicionamento de componentes eletrônicos em placas de circuito impresso ocorreu durante o projeto do sistema CADD para confecção de leiautes de PCIs (COSTA e PIMENTEL [1986]). Durante a implementação do CADD no Departamento de Eletrônica (DPET-CEPEL) foram realizadas várias entrevistas com os especialistas responsáveis pela operação do sistema, tendo sido supervisionado o desenvolvimento do leiaute de várias placas (PIMENTEL e BANDIM [1989]).

A partir das entrevista e da observação da metodologia de trabalho dos especialistas, foi possível identificar a estratégia e um conjunto de regras básicas utilizadas para realizar o posicionamento. A linguagem PROLOG foi escolhida como formalismo para representar o conhecimento do especialista.

Posteriormente, devido à familiaridade adquirida com o problema, foi possível modificar e otimizar o sistema, baseado nos casos testes, sem necessitar da ajuda do especialista.

V.1 - O Sistema CADD

O SAP é um dos módulos do sistema CADD (Sistema de Auxílio ao Projeto de Circuitos Eletrônicos), que foi

desenvolvido dentro do programa de infraestrutura para microeletrônica, visando auxiliar a etapa de confecção de máscaras de circuitos integrados (COSTA e PIMENTEL [1984]). Entretanto, devido à sua generalidade, o sistema CADD vem sendo utilizado também na confecção de leiautes de placas de circuito impresso (COSTA e PIMENTEL [1986]). Foram confeccionadas até o momento mais de vinte placas.

O sistema CADD, mostrado na figura (V.1), compõe-se de três módulos principais: um editor gráfico responsável pela edição e modificação interativa dos leiautes; o SAP que gera automáticamente o posicionamento ótimo dos componentes dentro da placa; e o programa PLOTA responsável pela confecção das artes finais que serão enviadas para a fabricação. As seções V.1.1 e V.1.2 descrevem de forma resumida o editor gráfico e o gerador de arte final. O restante do capítulo descreve a implementação do SAP.

V.1.1 - O Editor Gráfico (EDITH)

O EDITor gráfico Hierárquico - EDITH - é utilizado dentro do sistema CADD para ajudar o usuário na preparação e modificação dos leiautes, definir o estado inicial da placa para o SAP (regiões proibidas, área da placa, posicionar os componentes fixos, etc.), e se necessário, modificar o resultado do posicionamento automático.

O editor é composto fisicamente de um monitor colorido de alta resolução, um dispositivo apontador (ex.: ratinho ou "mouse", mesa digitalizadora, etc.), e um teclado. A tela do monitor está dividida em três regiões conforme ilustra a figura (V.2). A maior parte da tela é dedicada à confecção do leiaute, ou seja, é sobre essa região que são desenhadas as PCIs. Abaixo da região de leiaute está a área destinada ao diálogo com o usuário, que é utilizada para requisitar informações necessárias à execução dos comandos e/ou enviar informações sobre os mesmos. O "menu" de comandos está localizado à esquerda da região de leiaute, e contém todos os comandos disponíveis no editor. Estes comandos podem ser ativados através de um código mnemônico digitado no teclado, ou utilizando o dispositivo apontador para selecioná-los no "menu".

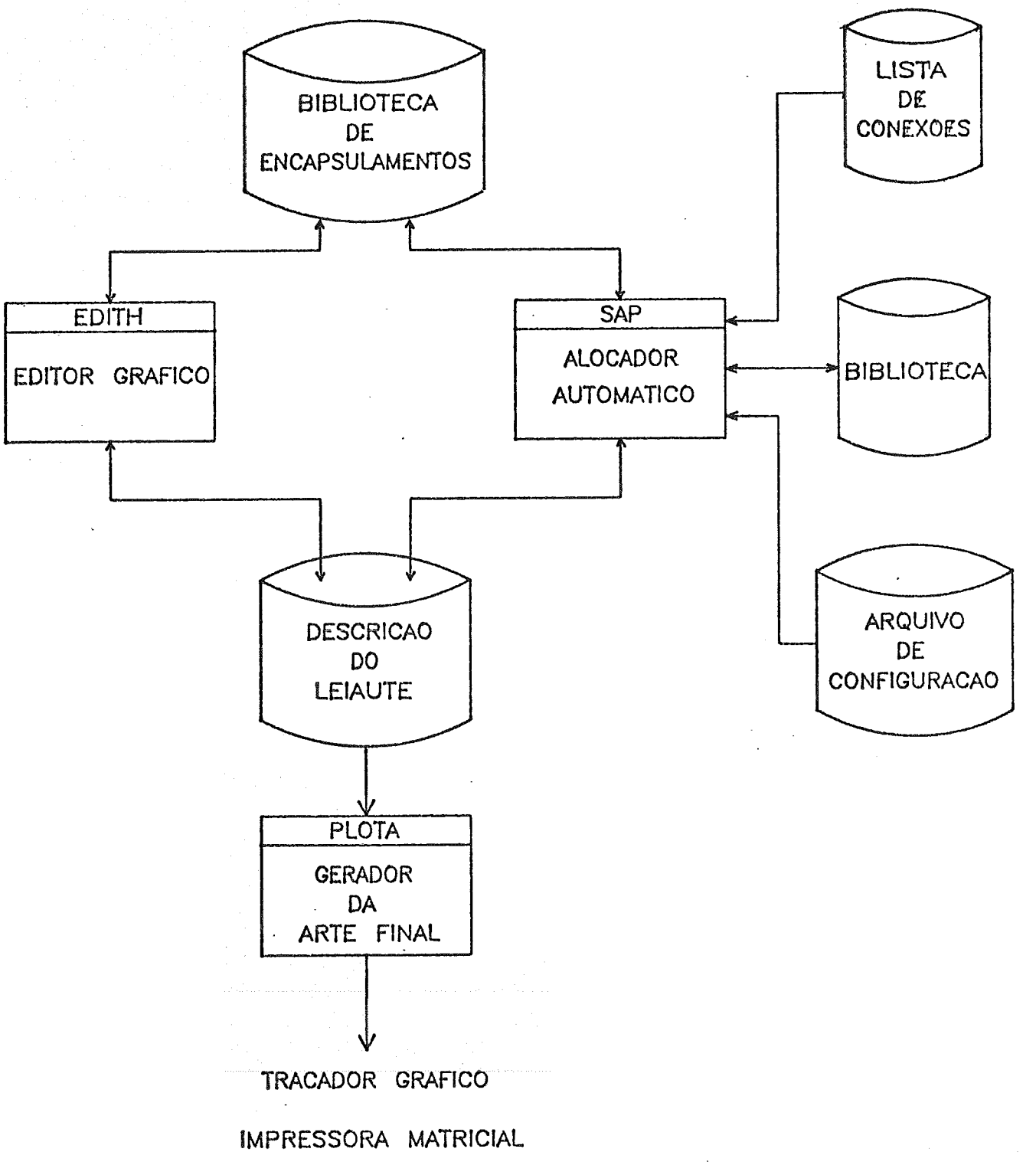


figura V.1 - Arquitetura do Sistema CADD.

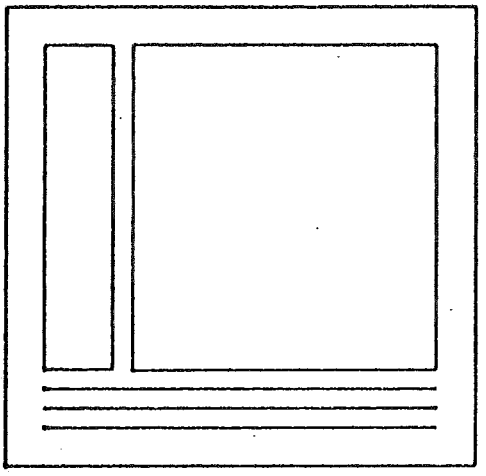


Figura V.2 - Tela do Editor EDITH.

O editor fornece comandos para: criar fios, retângulos, polígonos, arcos e textos; manipular (eliminar, mover e copiar) essas primitivas gráficas; definir parâmetros de desenho tais como largura dos fios, camada onde as primitivas serão desenhadas, etc.; realizar operações sobre a tela (ampliar, reduzir e mover); e trazer encapsulamentos da biblioteca e posicioná-los no desenho. (PIMENTEL e BANDIM [1989]) descrevem um roteiro para utilização do sistema CADD na confecção de PCIs. Informações sobre o seu projeto e implementação podem ser encontradas em (COSTA e PIMENTEL [1984] e COSTA e PIMENTEL [1986]). Dentre as diversas características do EDITH, deve-se destacar:

. a biblioteca de símbolos que contém a maioria dos encapsulamentos utilizados na confecção de PCIs.

. a possibilidade de criar e modificar os símbolos da biblioteca de acordo com as necessidades do leiaute.

. a existência de comandos para armazenar e recuperar leiautes, permitindo a reutilização destes, na confecção de circuitos mais complexos.

. a generalidade dos comandos existentes, que permite a sua utilização em um universo maior

de aplicações, tendo sido utilizado inclusive na preparação das figuras da tese.

V.1.2 - O Gerador de Arte Final (PLOTA)

Após o término do leiaute, a próxima etapa é a preparação das artes finais que serão enviadas para a fabricação. A confecção da arte final é realizado pelo programa PLOTA que recupera o desenho armazenado pelo editor gráfico e gera o desenho da arte final. Esse desenho pode ser feito sobre vários tipos de papel (ex.: vegetal, poliéster, papel branco simples, etc.), e em formato A1 ou A2.

Na versão atual, o programa dispõe de um único módulo de interface que controla um dispositivo traçador gráfico DIGICON modelo TDD-21R. Entretanto, podem ser escritos módulos para controlar outros tipos de traçadores gráficos, e até mesmo outros tipos de dispositivos tais como impressoras matriciais. O apêndice E mostra o leiaute de duas placas de circuito impresso dupla-face confeccionadas inteiramente no sistema CADD. Nestas placas, o posicionamento dos componentes e a interligação dos pinos foram feitos manualmente, utilizando o editor gráfico.

V.1.3 - O Alocador Automático (SAP)

Para implementar a estratégia proposta no capítulo IV em um programa de computador, foi utilizada a arquitetura mostrada na figura (V.3), onde cada uma das etapas da estratégia é mapeada em um módulo de programa. As etapas de fusão, ordenação e posicionamento relativo foram agrupadas no módulo de programa ALEX que corresponde à parte inteligente do sistema. Estas etapas contém um conjunto de regras que representam o conhecimento do especialista, e algorítmos para resolver as situações em que não foram possíveis identificar regras. As etapas de eliminação de superposições, posicionamento dos componentes e otimização são inteiramente algorítmicas.

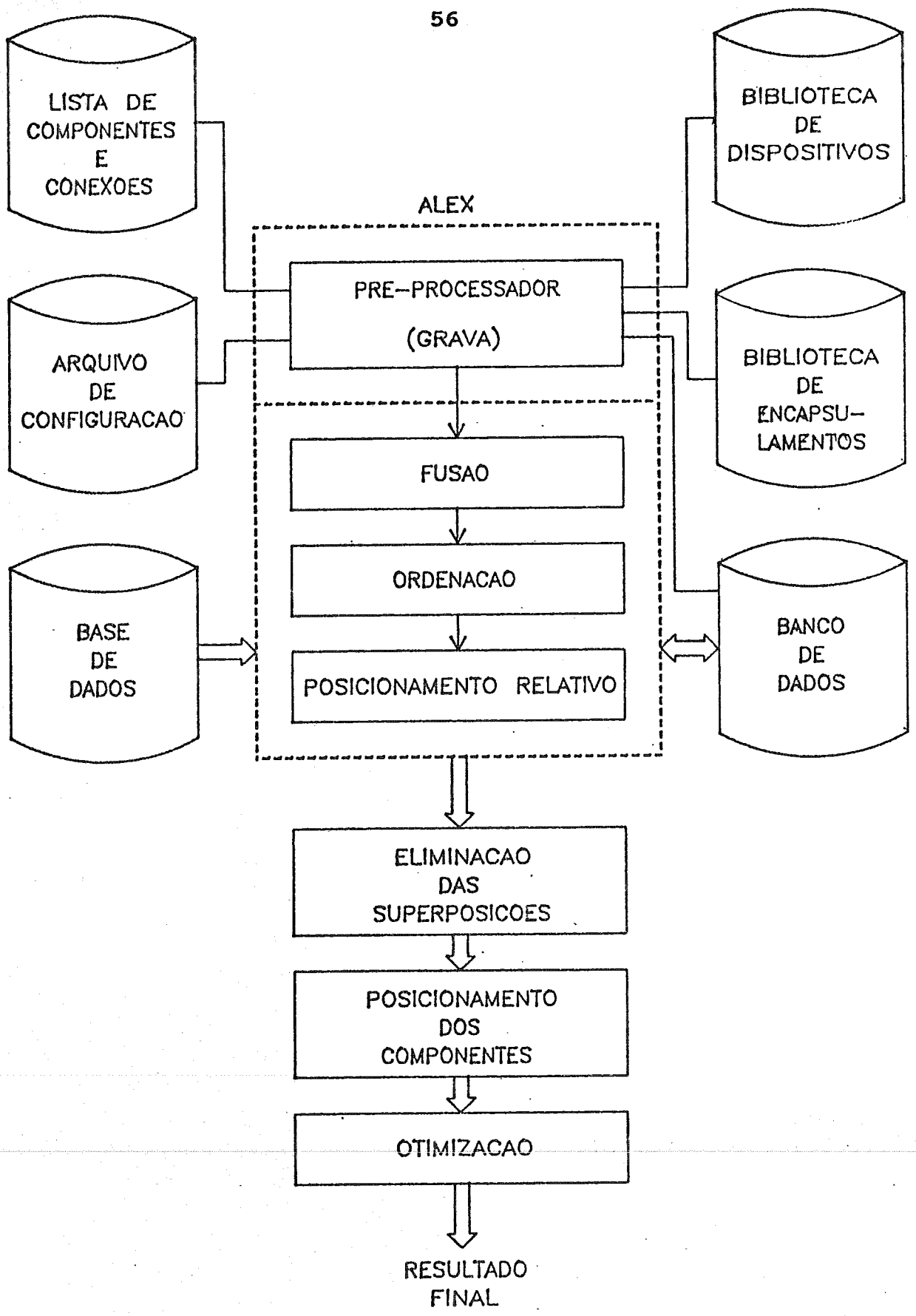


Figura V.3 - Arquitetura do SAP

A seguir é apresentada uma descrição detalhada de cada módulo de programa que compõe o alocador, além do projeto lógico e físico do banco de dados.

V.2 - Projeto do Banco de Dados

Em muitas aplicações, é necessário armazenar grandes quantidades de dados e informações em um computador por um longo período de tempo. Dados que são armazenados dessa forma são denominados bancos de dados(DB), e os programas que permitem usar e modificar esses dados são chamados Sistemas Gerenciadores de Banco de Dados - SGBD (ULLMAN [1980], cap.1).

Como os computadores manipulam informações na forma de bits e bytes, e os seres humanos utilizam geralmente símbolos e conceitos abstratos, a representação do banco de dados deve ser feita em três níveis diferentes de abstração: o modelo conceitual do banco de dados; o modelo lógico; e o modelo físico.

O nível mais baixo de abstração é o banco de dados físico que está relacionado à forma como os dados são armazenados no computador. O banco de dados lógico é uma representação abstrata das informações armazenadas no banco de dados físico. O nível mais alto de representação é o modelo conceitual que consiste em identificar e listar as informações que descrevem a aplicação à qual o banco de dados se destina.

A próxima seção descreve o modelo conceitual do banco de dados do SAP, e o método utilizado para obtê-lo. As seções V.2.2 e V.2.3 descrevem o modelo lógico e o modelo físico do banco de dados respectivamente.

V.2.1 - O Modelo Conceitual

Especificar o modelo conceitual de um banco de dados consiste em analisar o domínio do problema, identificar, classificar e abstrair informações relativas a ele. (SHLAER e MELLOR [1988]) propõem um método de análise chamado Modelagem da Informação (MI), o qual é particularmente adequado para obter e formalizar o conhecimento a respeito de um determinado domínio. A idéia básica da MI é descrever

objetos, atributos, e relacionamentos entre objetos que sejam relevantes para o problema. Este método foi utilizado no projeto do banco de dados do SAP.

Um objeto é a abstração de um conjunto de entidades do mundo real tal que todas as entidades (ou ocorrências) do conjunto têm as mesmas características, e estão sujeitas às mesmas regras. Os objetos possuem um ou mais atributos de forma que cada atributo diz respeito a apenas uma única característica do objeto. Um conjunto de atributos que permita identificar cada ocorrência do objeto é chamado de chave. A tabela (V.1) lista alguns dos objetos pertencentes à base de dados do SAP. A descrição detalhada de todos os objetos, seus atributos, e os relacionamentos entre esses objetos estão mostrados no apêndice A.

- placas
- módulos
- componentes (ex.: U15)
- encapsulamentos (ex.: DIP24)
- redes (ex.: ADD0)
- barramentos (ex.: ADD)
- pinos
- dispositivo (ex.: 6264)

Tabela V.1 - Objetos do SAP

Um relacionamento é uma abstração de um conjunto de associações entre tipos diferentes de objetos. Durante o processo de análise foram identificados os seguintes relacionamentos:

- pertence a => este relacionamento serve para associar alguns objetos tais como: componentes e módulos; redes e barramentos; pinos e encapsulamentos.
- é-do-tipo - associa componentes e dispositivos, componentes e encapsulamentos.

. está-conectado-a - este relacionamento associa um pino de um componente à rede que está ligada nele.

Os relacionamentos envolvendo somente dois objetos podem ser classificados em três tipos fundamentais, dependendo do número de ocorrências dos objetos que participam do relacionamento. Em outras palavras, alguns relacionamentos são:

. um-para-um (1:1)
  estado TEM UM governador.
. um-para-muitos (1:M)
  barramento POSSUI redes.
  módulos POSSUI componentes.
. muitos-para-muitos (M:M)
  módulos ESTÃO LIGADOS A barramentos.
  componentes ESTÃO LIGADOS A redes.

Na MI, a representação da informação é feita de duas formas: a forma gráfica é chamada de Diagrama de Estrutura da Informação (DEI) e a textual é o Documento de Especificação de Objetos e Relacionamentos (DEOR) (SHLAER e MELLOR [1988]). O DEI é baseado nas várias formas de diagramas de entidades - relacionamentos amplamente utilizados no desenvolvimento de sistemas de computação. Seu objetivo é representar de forma simples e objetiva a declaração dos objetos e atributos, e facilitar a vizualização dos relacionamentos entre os objetos. O DEOR contém todas as informações disponíveis sobre o problema. Ele inclui a descrição detalhada de cada objeto e relacionamento, bem como a definição de cada um de seus atributos. Obviamente, estes documentos incluem uma grande quantidade de informação e portanto deve ser organizado de forma útil, porém completa. O apêndice A apresenta de forma resumida o DEI e o DEOR relativos ao banco de dados do SAP.

V.2.2 - O Modelo Lógico

Em geral, os SGDBs fornecem uma linguagem de alto nível que permite descrever o modelo conceitual em termos de um modelo de dados, que corresponde a obter modelo lógico do banco de dados. A escolha do modelo de dados é uma tarefa difícil, principalmente por duas razões: deve ser capaz de descrever os aspectos importantes do mundo real; e permitir uma implementação eficiente do modelo conceitual em um modelo físico. Dentre os diversos modelos de dados existentes, os mais importantes são (ULLMAN [1980]):

. hierárquico - este modelo de dados é uma árvore onde os nós representam os objetos. Os filhos de um nó estão associados aos seus pais por um relacionamento.

. rede - este modelo corresponde a um grafo dirigido onde os nós representam os objetos e os arcos são os relacionamentos.

. o modelo relacional é baseado no conceito técnico de relações (ULLMAN [1980]), e são representados normalmente como tabelas.

No método MI, os objetos são modelados como tabelas, onde cada ocorrência do objeto ocupa uma linha da tabela, e cada coluna representa uma característica, ou atributo, deste objeto. A figura (V.4) ilustra alguns objetos do SAP. Nesta figura, um dos atributos da rede é o nome de um barramento. Esta coluna define um relacionamento entre redes e barramento do tipo:

a rede A PERTENCE AO barramento B.

REDE

nome (*)	tipo	número de pinos	nome do barramento
add0	sinal	36	add
add1	sinal	36	add
gnd	terra	53	""
clk	clock	9	""
int	sinal	3	""

BARRAMENTO

nome (*)	largura	compri mento	conecti vidade	tipo
add	16	14	1	bsg
dt	8	6	0	bpl
pa	8	2	0	bpl

DISPOSITIVO

nome (*)	tipo
8088	cpu
z80	cpu
6264	ram
2732	rom
8251	sio

OBS : (*) - chave dos objetos

Figura V.4 - Exemplo de Objetos

A MI modela os relacionamentos (1:1) duplicando a chave de um dos objetos e inserindo-a como um novo atributo do outro objeto conforme mostrado no exemplo abaixo.

ex.:

. Estado
- nome do estado (*)
- população

. Governador
- nome do governador (*)
- data em que assumiu o cargo
- nome do Estado (R)

No exemplo acima, (*) identifica as chaves, e (R) indica que o atributo refere-se a um relacionamento.

Num relacionamento (1:M), uma ocorrência de um objeto A pode estar associada a uma ou mais ocorrências do objeto B. Estes relacionamentos são modelados usando a chave do objeto A como um novo atributo do objeto B, conforme ilustra o seguintes exemplo:

ex.:

. Barramento
- nome do barramento (*)
- largura
- comprimento
- conectividade
- tipo

. Rede
- nome da rede (*)
- número de pinos
- tipo
- nome do barramento (R)

Nos relacionamentos (M:M), toda ocorrência do objeto A está associada a uma ou mais ocorrências do objeto B, e toda ocorrência do objeto B está associada a uma ou mais

ocorrências de A. Para modelar os relacionamentos (M:M) deve-se construir uma tabela auxiliar, chamada de tabela de correlação, que contém as chaves de cada objeto.

ex.:

. TBM (Tabela barramentos x módulos)
- nome do barramento (*)
- nome do módulo (*)
- número de sinais do barramento ligados ao módulo.

. Conexão (entre redes e componentes)
- nome da rede (*)
- nome do componente (*)
- número do pino (*)

V.2.3 - O Modelo Físico

Os objetos identificados no modelo conceitual foram agrupados de acordo com a sua natureza. O primeiro grupo é a biblioteca do sistema e inclui a definição dos encapsulamentos, seus pinos, e os dispositivos eletrônicos. Esses objetos estão armazenados estaticamente na base de dados e representam informações gerais que podem ser utilizadas por várias PCIs. O segundo grupo compreende os objetos que contém informações específicas sobre a PCI que está sendo confeccionada no momento.

O ambiente Turbo PROLOG V2.0 é capaz de criar e manipular dois tipos de bases de dados: uma base de dados externa cuja organização física (em disco) é uma árvore-B; e uma base de dados interna que é armazenada na memória na forma de tabelas (BORLAND [1988]).

A base de dados externa foi escolhida para implementar a biblioteca do sistema por várias razões, entre as quais a sua capacidade de armazenar grandes volumes de dados, e recuperá-los eficientemente. As chaves dos objetos da biblioteca são utilizadas como chave de busca para a árvore-B. As entidades armazenadas na base de dados interna são tratadas como fatos pelo interpretador PROLOG e portanto podem ser utilizadas diretamente pela máquina de

inferência embutida no interpretador. Essa propriedade, aliada à sua organização física, torna a base de dados particularmente adequada para armazenar os objetos da base de fatos e os relacionamentos do SAP. O mapeamento desses objetos em fatos PROLOG é feito da seguinte forma:

. os nomes dos objetos e relacionamentos correspondem aos predicados.
. os atributos desses objetos e relacionamentos são os argumentos dos fatos.

O exemplo abaixo relata o trecho do programa que implementa a base de fatos do SAP cujo os objetos estão descritos no apêndice A. Esses fatos serão utilizados pelas regras empíricas embutidas no sistema.

i) relacionamentos
conexão (nome rede, nome comp, número do pino).
tbm (nome bar, nome mod, núm. de ligações).

ii) objetos
placa (nome, encaps, xe, yi, xd, ys, nmod, ncomp, nredes, nbar, npinos, direção).
módulo (nome, xe, yi, xd, ys, ox, oy, tipo comp, mobil, direção, tipo bar).
componente (nome, encaps, desposit, tipo comp, direção, xe, yi, xd, ys, ox, oy, nomemod, mobil).
rede (nome, tipo de rede, npinos, nomebar).
barramento (nome, largura, comprimento, conectividade, tipo bar).

V.3 - O Alocador Especialista (ALEX)

A primeira etapa no processo de posicionamento do SAP é determinar as posições relativas dos componentes pertencentes à parte estruturada em barramentos. O módulo de programa ALEX, responsável por esta tarefa, compreende regras e algorítmos que levam em consideração a estrutura

dos barramentos, as características funcionais dos componentes e a disposição dos conectores externos.

No ALEX, o conhecimento empírico obtido do especialista em leiaute está armazenado na forma de regras de produção, principalmente devido a:

. unidade de conhecimento do tipo SE <condições> ENTÃO <ações>, é adequada à representação do conhecimento do especialista (JOOBANI [1987], e ODAWARA et alii [1987]).
. facilidade de adicionar, modificar e remover as regras da base de conhecimento.

O ALEX consiste de duas partes: uma parte algorítmica escrita em Pascal e uma parte baseada em conhecimento escrita em PROLOG. A razão para esta divisão é que algorítmos são muito mais fáceis de serem implementados em linguagens convencionais, e são processados num tempo muito menor.

V.3.1 - Arquitetura do ALEX

A parte inteligente foi implementada dentro do ambiente TURBO PROLOG V2.0 (BORLAND [1988a] e BORLAND [1988b), e segue a arquitetura genérica proposta para sistemas especialistas mostrada na figura (III.1). A base de conhecimento absorvida do especialista está armazenada na forma de cláusulas (fatos e regras), enquanto a máquina de inferência está embutida no próprio compilador PROLOG.

As diversas tarefas realizadas pelo ALEX estão distribuídas por quatro especialistas encarregados de propôr soluções relativas à sua área de atuação. Cada especialista trata apenas de alguns aspectos do problema. A posição relativa dos componentes pertencentes à parte estruturada em barramentos é obtida ativando-os na sequência mostrada na figura (V.5). O restante dos ítens desta seção explica o funcionamento de cada especialista, bem como os métodos e critérios utilizados por cada um deles.

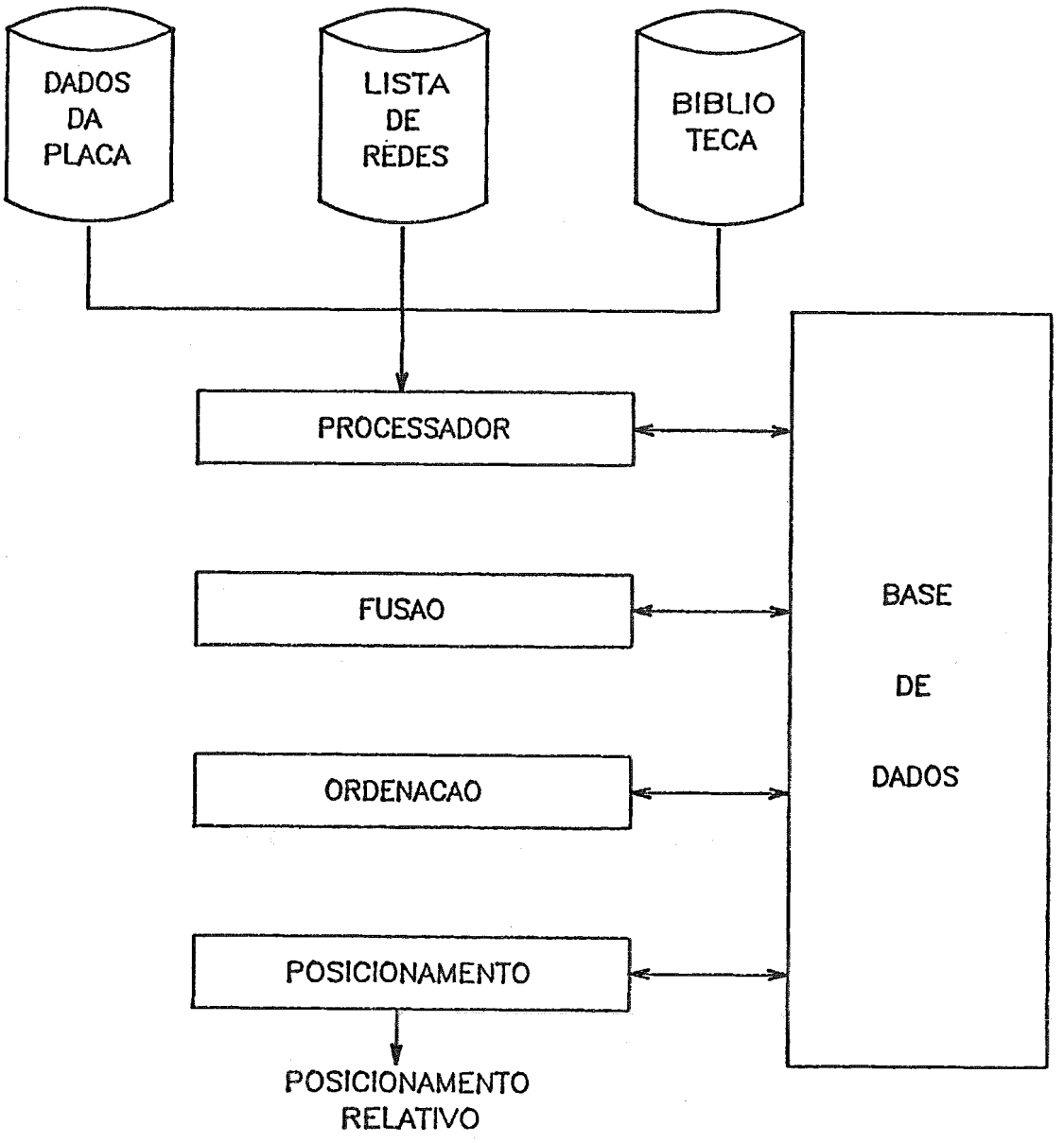


Figura V.5 - Organização dos Especialistas.

V.3.2 - Pre-processador Especialista (GRAVA)

A principal função do pre-processador GRAVA é ler os dados de entrada do sistema e construir a base de dados que será utilizada por todos os outros módulos do SAP. Os dados de entrada compreendem:

. o arquivo de configuração que contém informações relativas à placa de circuito impresso, tais como: área útil; lista de componentes fixos (normalmente concectores); e a lista de redes especiais (ex.: Vcc e Gnd). A implementação atual reconhece apenas as redes de alimentação e terra como especiais.
. o arquivo de ligações que descreve a topologia do circuito e contém as listas de redes e componentes.
. a biblioteca do sistema que contém a descrição física dos encapsulamentos e o tipo funcional dos dispositivos eletrônicos.

ex.:

- encapsulamentos
  DIP14 - 14 pinos
  área (-127,-127,1651,889)
  posição dos pinos

- dispositivos
  . 8088 - cpu
  . 6264 - memória RAM
  . 2732 - memória ROM

A partir dos dados de entrada, o especialista GRAVA extrai um conjunto de informações que será utilizada pelos demais especialistas. Inicialmente ele pesquisa na biblioteca do sistema o tipo do dispositivo e o encapsulamento de cada componente. Em seguida, é percorrida a lista de redes e identificados todos os barramentos

existentes no circuito, os quais são classificados em um dos quatro tipos citados na seção IV.2 (BPL, BPG, BSL e BSG). Os barramentos são então ordenados em função da seguinte lista de prioridades:

BPL, BPG, BSG e BSL

Caso dois barramentos pertençam ao mesmo tipo, o barramento escolhido será o de maior comprimento (número de componentes). Se o empate persistir, então é escolhido qualquer um deles aleatoriamente.

De posse dos componentes e dos barramentos, o GRAVA associa um módulo a cada componente e monta a tabela de barramentos x módulos que contém, todos os módulos (e por conseguinte os componentes) pertencentes à parte do circuito estruturada em barramentos. Esta tabela será usada posteriormente no processo de tomada de decisões dos outros especialistas.

V.3.3 - Especialista em Fusão de Módulos (RFM)

O especialista RFM agrupa dois ou mais componentes que estejam fortemente conectados entre sí. Eles são agrupados formando módulos, que são tratados como uma unidade. Para agrupar os componentes, o RFM leva em consideração algumas informações tais como o tipo funcional do módulo, a sua forma, o tamanho da PCI, e o tamanho e forma dos componentes. Deste modo, pode-se garantir que todos os módulos possam ser posicionados na placa. A figura (V.6) mostra um exemplo de fusão. A CPU e as duas SIOs estão conectadas ao mesmo BSG. Portanto são fundidas em um único módulo. A PIO não é fundida junto com os demais pois está fortemente conectada ao conector.

A figura (V.7) mostra alguns exemplos de regras existentes no especialista RFM. Essas regras foram escritas em linguagem natural, ao invés de PROLOG, para facilitar a compreensão das mesmas.

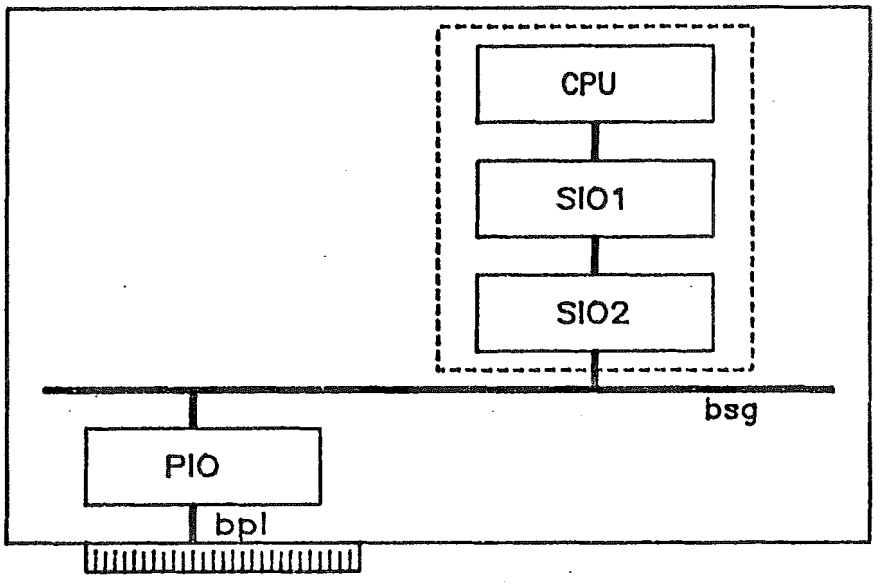


Figura V.6 - Exemplo de Fusão.

```
R1: SE   ((existe 1 módulo num BPL) E
          (este módulo também está ligado a um BSL) E
          (existe 1 módulo nesse mesmo BSL))
    ENTÃO (funda-os em um módulo no BPL)

R2: SE   ((existem 2 módulos num barramento B1) E
          (os 2 módulos são memórias RAM)
    ENTÃO (funda os 2 módulos)

R3: SE   ((existem 2 módulos num BPG) OU
          (existem 2 módulos num BSG)) E
           (eles não estão fortemente conectados mais
          ninguem) E
          (a área dos 2 módulos cabe na PCI))
    ENTÃO (funda os 2 módulos)
```

Figura V.7 - Exemplo de Regras do RFM.

V.3.4 - Especialista em Ordenação de Módulos (ROM)

O especialista em ordenação-ROM recebe os módulos fundidos pelo RFM e ordena os componentes dentro destes módulos considerando o número de linhas conectadas ao

barramento e a função dos componentes. Atualmente, ROM contém regras para ordenar bancos de memória, e módulos ligados a barramentos primários.

Os bancos de memória são ordenados por uma regra que reconhece o tipo do módulo (ex.: memória RAM), e ativa um algorítmo responsável pela construção da matriz de memórias.

Os módulos ligados a barramentos primários são ordenados pelo método descrito em (ODAWARA et alii [1983]). A estrutura das ligações entre barramentos e componentes é mapeada numa árvore onde a raiz (parte superior da árvore) contém os componentes que possuem mais ligações com o barramento, enquanto a parte inferior corresponde aos componentes que possuem menos ligações com o barramento. A figura (V.8a) mostra um barramento primário B de largura igual a oito, e comprimento igual a cinco, cuja estrutura foi mapeada na árvore da figura (V.8b). A construção da árvore baseia-se na seguinte estratégia: seja L(C,B) o conjunto das ligações do componente C com o barramento B, N(L(C,B)) o número de ligações em L(C,B), e M o módulo que está sendo ordenado. Para todo par de componentes C_i e C_j pertencente a M tal que $N(L(C_i,B)) > N(L(C_j,B))$, se $L(C_i,B)$ contém $L(C_j,B)$ então C_j é inserido na árvore como filho de C_i, senão C_j é inserido como irmão.

Os módulos pertencentes a BPLs podem conter tanto componentes ligados diretamente ao conector quanto componentes que não possuem ligações diretas com este. O segundo tipo corresponde aos componentes inseridos no módulo pela regra R1 da figura (V.7), os quais pertenciam originariamente a um BSL. O componente já posicionado que possuir o maior número de ligações com o BSL será utilizado como raiz da sub-árvore que contém os componentes pertencentes ao BSL. este procedimento é repetido até que todos os componentes do módulo tenham sido inseridos na árvore.

A posição relativa dos componentes dentro do módulo é obtida agrupando os componentes que ocupem a mesma profundidade na árvore, e empilhando os vários níveis na

ordem crescente de suas profundidades, conforme ilustra a figura (V.9).

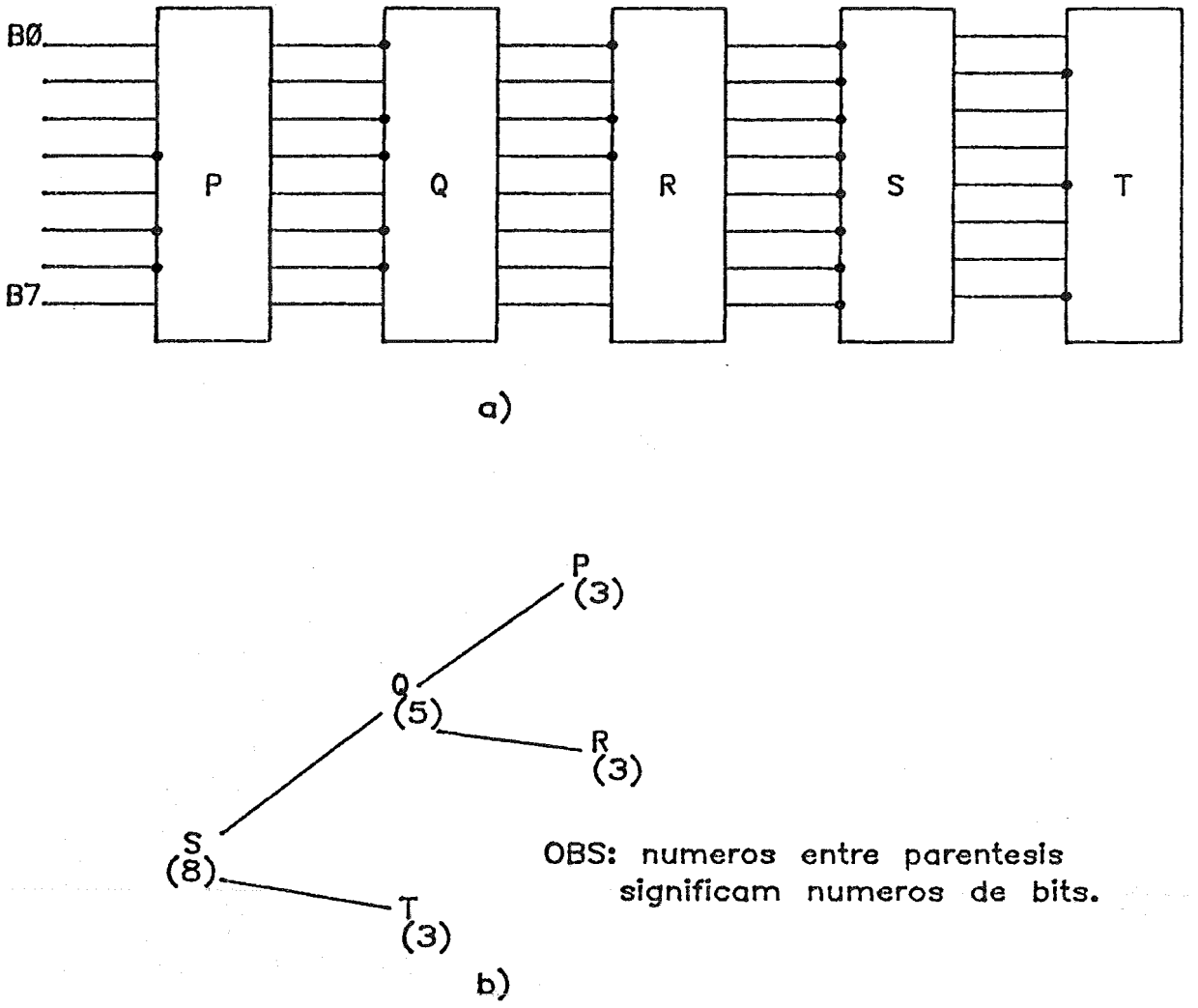


Figura V.8 - Árvore de Estrutura dos Barramentos

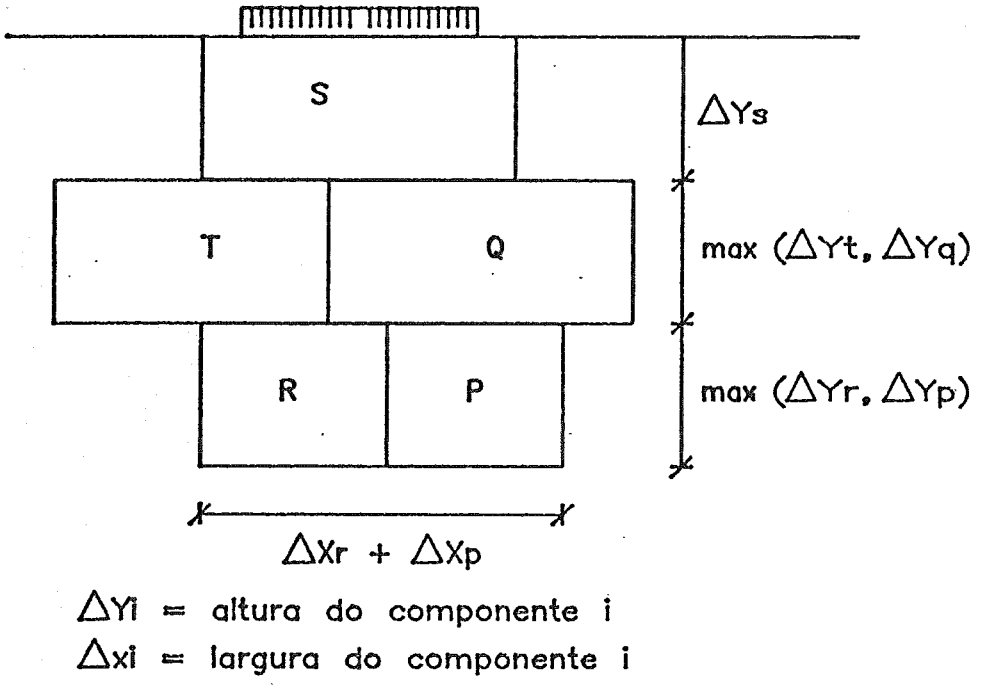


Figura V.9 - Posição Relativa Dentro do Módulo

Abaixo estão mostradas as duas regras do especialista em ordenação ROM, explicadas anteriormente.

```
R1: SE    (tipo do módulo é RAM).
    ENTÃO (gerar matriz de memórias).

R2: SE    ((módulo pertence a BPL) OU
           (módulo pertence a BPG))
    ENTÃO ((gerar árvore) E
           (determinar posições relativas)).
```

V.3.5 - Especialista em Posicionamento (RPM)

Este especialista é responsável por determinar a posição relativa dos módulos na placa. Para executar sua tarefa, RPM considera várias características da placa e do

módulo, entre as quais o tamanho, a forma do módulo e a posição dos conectores. Ele é formado basicamente por três tipos de regras para:

- posicionar módulos pertencentes a BPGs.
- posicionar módulos pertencentes a BPLs.
- posicionar módulos pertencentes a BSGs e BSLs.

Para determinar a posição relativa dos módulos pertencentes aos BPGs e BPLs, RPM calcula o centro de gravidade CGc dos pinos do conector que estão ligados ao módulo. Em seguida, o módulo é posicionado adjacentemente ao conector tal que o centro de gravidade CGm do módulo fique alinhado com CGc. A figura (V.10) mostra o posicionamento de um módulo pertencente a um BPL.

Os módulos pertencentes a BPGs e BPLs são, então, fixados em suas posições. Em seguida, é disparada uma regra que identifica todos os módulos pertencentes aos BSLs e BSGs, e aplica o algorítmo FDP descrito na seção II.3.8 para determinar as suas posições relativas. A força de atração entre os módulos é calculada em função do número de ligações com os barramentos. As redes randômicas são ignoradas durante esta fase do posicionamento.

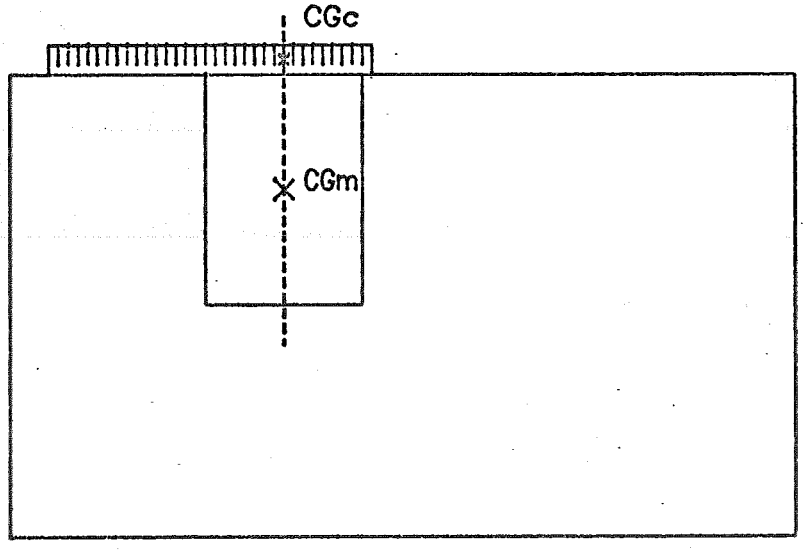


Figura V.10 - Posicionamento de Módulo Pertencentes aos BPGs e BPLs.

Os primeiros testes com o algorítmo FDP mostraram que em todas as placas apresentadas o número de superposições entre módulos era muito grande, ocorrendo em alguns casos superposições quase total de módulos. Obviamente, o excesso de suporposições prejudica a qualidade do posicionamento relativo, e reduz a eficiência do método de resolução de superposições.

A versão atual do SAP utiliza o algorítmo FDP com a seguinte modificação: após executar o FDP conforme descrito anteriormente, é efetuada mais uma etapa de processamento que consiste em aplicar uma forca de repulsão sobre todos os módulos que estiverem superpostos. Esta força de repulsão tem módulo constante e direção determinada pelos centros dos respectivos módulos.

Essa modificação permite que o algorítmo FDP utilize a forma e a área dos módulos para minimizar as superposições. Note que para os casos onde ocorre superposição de módulos, a força de repulsão $Fr_{(i,j)}$ é dada pela seguinte expressão:

$$Fr_x(i, j) = C * dX_{ij} / Ds \qquad (V.1)$$

$$Fr_y(i, j) = C * dY_{ij} / Ds \qquad (V.2)$$

onde:

$$dX_{ij} = (X_i - X_j)$$

$$dY_{ij} = (Y_i - Y_j)$$

$$Ds = |dX_{ij}| + |dY_{ij}|$$

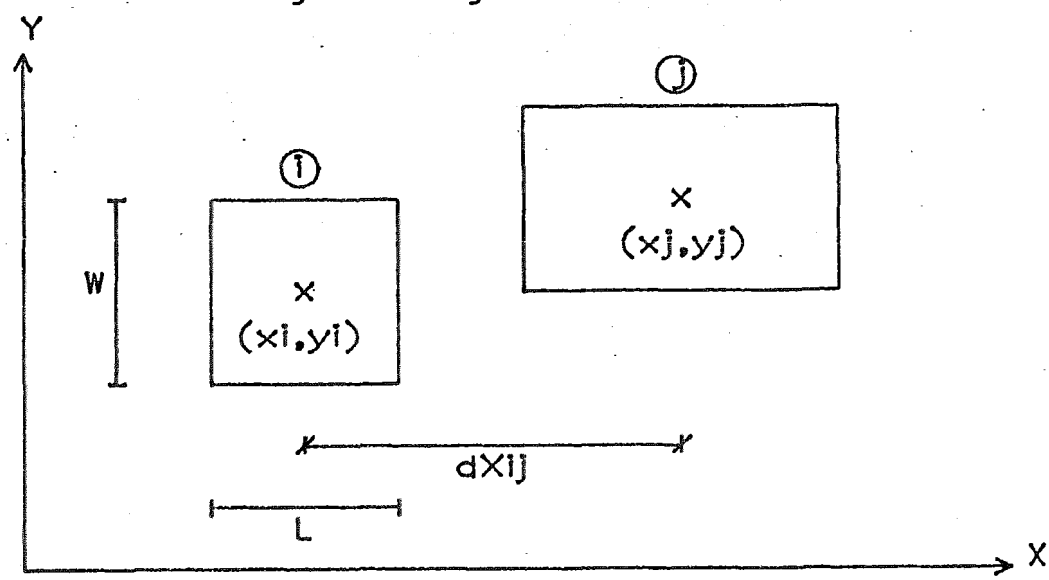


Figura V.11 - Definição do Vetor Posição

V.4 - Eliminação das Superposições

(QUINN e BREUER [1987]) propõe um algorítmo de eliminação, de superposições que consiste em dividir a área da placa em células ("slots"), e associar os módulos às células, minimizando a distorção total do posicionamento relativo. O custo c_{ij} de associar o módulo m_i à célula l_j é calculado pelo quadrado da distância euclidiana entre a posição relativa de m_i (obtida na etapa anterior), e a posição de l_j, ou seja:

$$c_{ij} = (m_{ix} - l_{jx})^2 + (m_{iy} - l_{jy})^2 \qquad (V.3)$$

Impondo a condição de que todos os módulos possuam o mesmo tamanho, então determinar a associação de menor distorção total é equivalente a resolver um problema de associação linear (KUHN [1055]), sujeito à matriz de custo $[c_{ij}]$. Entretanto, essa restrição é muito forte, e em geral prejudica a qualidade do leiaute final.

A implementação da etapa de eliminação de superposições é feita pelo algorítmo de atribuição proposto por (ODAWARA et alii [1982]), cujo pseudo-código está mostrado na figura (V.12). Este algorítmo utiliza basicamente o método anterior com algumas técnicas adicionais para tratar módulos de tamanhos e formatos variados, do seguinte modo:

- módulos que ocupam mais de uma célula são posicionados na ordem decrescente de suas áreas, ou seja, os módulos maiores são posicionados primeiro.
- se dois módulos se sobrepuserem, o menor deles é retirado da placa.
- módulos que ocupar uma única célula são posicionados pelo algorítmo de associação linear.

Dado em grupo de n-módulos candidatos $M = \{m_1,,\ldots,m_n\}$ e um grupo de p-células candidatas $S = \{s_1,s_2,\ldots,s_p\}$, então o algorítmo associa módulos a células da seguinte forma:

```
ORDENAR M  em ordem decrescente de área;
SELECIONAR  o módulo m_i de maior área;
ENQUANTO  área (m_i) > área (1 célula) FAÇA
    PARA j:=1 ATÉ n FAÇA
        CALCULAR CUSTO (m_i, l_j);
        k:=1;
        CONTINUA:= VERDADE;
        ENQUANTO (k <= n) e CONTINUA FAÇA
           DETERMINAR k-ésima associação de custo
                   mínimo;
           SE M_i não sobrepõe outro módulo M_s
           ENTÃO POSICIONAR M_i
                 CONTINUA:=FALSO;
           SENÃO
           SE  área (M_i) > área (M_s)
           ENTÃO REMOVER M_s
                 POSICIONAR M_i
                 INSERIR M_s no conjunto M;
           k:= k + 1;
    SELECIONAR  o  próximo  módulo M_i  de  maior  área;
POSICIONAR os módulos restantes usando o algorítmo de
           associação linear;
```

Figura V.12 - Algorítmo de Atribuição

O algorítmo de associação linear foi implementado pelo método proposto em (BALL [1980]).

V.5 - Posicionamento dos Componentes na Placa

Finalmente, os componentes têm a sua posição relativa dentro da placa determinada da seguinte forma:

i) para os módulos que foram ordenados a posição do componente dentro da placa (x_{cp}, y_{cp}) é obtida compondo a posição do módulo (x_{mp}, y_{mp}), com a posição do componente dentro do módulo (x_{cm}, y_{cm}), fixando-o nessa posição.

$$x_{cp} = x_{mp} + x_{cm}$$
$$y_{cp} = y_{mp} + y_{cm}$$

ii) fixar o componente semente que possui o maior número de ligações com os outros componentes dos módulos não-ordenados no centro do módulo.
iii) executar o algorítmo FDP sobre os demais componentes dos módulos não-ordenados. Utilizar como k_{ij} o número de ligações compartilhadas com os barramentos, ou seja, desconsiderar as redes randômicas.
iv) mantendo fixos os componentes pertencentes aos módulos ordenados, associar componentes às células minimizando a distorção total do posicionamento relativo.

Ao final desta etapa, a posição relativa de todos os componentes ligados a barramentos terá sido determinada. Além disso, quanto melhor for o seu resultado, melhor será a qualidade do layout final (ODAWARA et alii [1987]). O apêndice C mostra o resultado do posicionamento dos componentes do circuito PT1.

Falta agora determinar o posicionamento dos componentes da parte radômica. Seus componentes são inseridos na placa, e sua posição relativa é determinada pelo algorítmo FDP, mantendo fixos os componentes já posicionados. A eliminação das superposições é realizada pelo método descrito anteriormente.

V.6 - Otimização

A otimização do posicionamento final é feita basicamente pelo algorítmo SB descrito na seção II.3.2, modificado para tratar módulos de tamanhos variados. Esse algorítmo pode ser resumido em três passos: seleção dos módulos candidatos; cálculo da função de custo; algorítmo de troca de posições.

A seleção de módulos candidatos consiste em identificar todos os grupos de módulos mutuamente disconectados (STEINBERG [1961]). Para cada grupo, todos os seus módulos são retirados da placa e trocados de posição utilizando o algorítmo de atribuição descrito na seção V.4. O custo do novo posicionamento é calculado pela função O TWL descrita na seção II.2.3. Se a troca reduziu o custo total dos posicionamento, então ela é aceita, em caso contrário ela é rejeitada. O processo é repetido indefinidamente, até que ocorra a saturação da função objetivo OTWL.

No estado atual do trabalho, a etapa de otimização se encontra implementada, mas ainda não foi inserida no protótipo do sistema.

CAPÍTULO VI
Resultados Experimentais

O alocador automático SAP pode ser dividido basicamente em três modulos de programa: o módulo ALEX responsável pela parte inteligente do sistema; o algorítmo FDP que determina posicionamentos relativos; e o algorítmo de ATRIBUIÇÃO responsável pela eliminação de superposicões e otimização do leiaute final. Tendo em vista facilitar os testes de cada módulo, optou-se por implementar o protótipo do sistema de forma que cada um destes módulos utilizasse o mesmo formato padrão para entrada e saída dos resultados. Assim, é possível observar o efeito de cada etapa isoladamente. A sintaxe deste arquivo está descrita no apêndice B.

VI.1 - Metodologia de Testes

A metodologia de testes empregada para validar os módulos do SAP consistiu de três etapas: validar a parte algorítmica; refinar a base de conhecimento; realizar o posicionamento de uma placa real.

Para validar a parte algorítmica, os módulos FDP e ATRIBUIÇÃO foram submetidos a um conjunto de casos sintéticos simples, para os quais a solução ótima era conhecida. Dessa forma, foi possível avaliar o resultado obtido corrigindo possíveis erros de implementação e ajustar as variáveis de controle destes algorítmos entre as quais o erro de convergência e a constante de repulsão. A experiência adquirida durante essa fase auxiliou a análise dos mesmos quando submetidos a casos mais complexos.

A base de conhecimento foi avaliada e posteriormente refinada, utilizando-se duas placas reais intensamente estruturadas em barramentos cujo os resultados estão mostrados nos apêndices C e D.

Finalmente, realizou-se um teste completo sobre que incluiu todos os módulos do sistema. Este teste utilizou a placa digital mostrada no apêndice D formada por: uma CPU, memórias, periféricos de entrada/saída, etc. Devido à sua

generalidade, foi possível observar o comportamento do sistema sujeito a várias situações reais.

VI.2 - Análise dos Resultados

VI.2.1 - Análise da Parte Algorítmica

1) Teste do Algorítmo FDP

Os apêndices B.4 e B.8 mostram os resultados obtidos aplicando-se o algorítmo FDP a dois casos sintéticos. Inicialmente, foi realizado o posicionamento do circuito de teste CT1. Neste circuito, os componentes foram inicialmente colocados em posições opostas às posições ótmas. O apêndice B.4 mostra que FDP trocou os componentes de posição, convergindo para a solução ótma.

Da mesma forma que no exemplo anterior, os componentes do circuito de teste CT2 foram pré-posicionados em regiões tais que dificultassem a convergência do algorítmo. Note que os componentes CMP5 e CMP6, os quais estão fortemente ligados ao conector CON3 foram trazidos para junto deste, e neste processo arrastaram também o componente CMP7, minimizando assim o comprimento total das ligações.

2) Algorítmo de ATRIBUIÇÃO

O primeiro teste deste algorítmo consistiu em aplicá-lo sobre o posicionamento relativo mostrado em B10, extraído do artigo onde o método foi proposto. A solução obtida pela implementação do SAP é a mesma apresentada por (ODAWARA et alii [1982]) e mostrada em B10.

Numa segunda fase, o método foi aplicado sobre os posicionamentos relativos obtidos pelo FDP, e produziram os resultados mostrados em B5 e BB9 respectivamente. Note que em B9, a superposição total de CMP1 com CMP4 resultou numa distorção grande do posicionamento relativo. Os resultados experimentais mostraram que posicionamentos relativos com níveis de superposição maiores que trinta porcento não são em geral preservados (distorção mínima) pelo método de ATRIBUIÇÃO.

VI.2.2 - Avaliação da Base de Conhecimento

A figura do apêndice C.2 mostra o estado da placa de teste PT1 imediatamente antes de atuar a regra do RPM que ativa o algorítmo FDP. Os módulos XMOD61 a XMOD64 são dispositivos de comunicação paralela tipo 8255 que estão fortemente conectados aos barramentos. Portanto foram posicionados junto aos mesmos. Os módulos existentes no canto superior esquerdo da figura correspondem aos módulos pertencentes a BSGs e SGLs os quais terão suas posicões relativas determinadas pelo FDP. FMOD2 é um banco de memórias RAM. XMOD82 é uma CPU. Estes módulos, junto com FMOD1 estão conectados a um barramento de endereços, e portanto ocupam a posição central da placa visando facilitar o seu roteamento. O restante do apêndice mostra o resultado das demais etapas do processo.

Comparando as figuras dos apêndices C.3 e C.4 podemos ver que o método de eliminação de superposições embutido no algorítmo FDP reduziu as superposições existentes, facilitando a tarefa do algorítmo de ATRIBUIÇÃO.

VI.2.3 - Teste Completo

A placa de testes PT2 mostrada no apêndice D refere-se a uma placa digital microprocessada utilizada para avaliar o resultado da iteração de todos os módulos de programa do SAP. Neste exemplo, FMOD1, FMOD2, e FMOD3 são buffers que realizam a interface com o mundo externo estando fortemente ligados aos conectores por BPLs. Estes módulos foram ordenados pela regra do especialista em ordenação (ROM) que trata de BPLs, e posicionados próximos aos conectores pelo RPM. FMOD5 e FMOD6 são dois bancos de memórias ordenadas pelo especialista ROM.

VI.3 - Conclusões

Observando a placa de teste PT2, podemos chegar às seguintes conclusões:

. a atuação dos especialistas reduziu à metade o número de módulos que serão posicionados

algoritmicamente. Assim, espera-se que o tempo gasto no processamento seja reduzido.

. a estrutura dos barramentos foi preservada, facilitando a tarefa de roteamento.

. o leiaute gerado é bastante organizado, e semelhante ao produzido pelos especialistas humanos.

. o método FDP mostrou-se adequado para determinar posicionamentos relativos.

. o método de ATRIBUIÇÃO distorce o posicionamento relativo para níveis de superposição acime de trinta procento.

CAPÍTULO VII
Conclusão

Este trabalho apresenta o desenvolvimento de um alocador automático que utiliza técnicas de inteligência artificial e algorítmos para obter o posicionamento de componentes eletrônicos em placas de circuito impresso.

As técnicas existentes anteriormente têm tratado a etapa de posicionamento como um problema de otimização. Portanto, não têm sido capazes de obter resultados satisfatórios para todos os tipos de PCIs, e em especial para o caso de placas digitais estruturadas em barramentos. Este trabalho propôs uma nova abordagem para o problema que possui as seguintes características:

. utiliza algorítmos e conhecimento, de acordo com uma estratégia previamente escolhida, procurando explorar o melhor das duas abordagens.
. tenta resolver o problema de forma semelhante à utilizada pelos especialistas humanos.
. as primeiras etapas do posicionamento (particionamento e posicionamento relativo) estão sujeitas a um conjunto de restrições mais fracas do que as restrições impostas pelas técnicas clássicas, evitando eliminar desnecessáriamente algumas soluções para o problema.
. além das interconexões entre os componentes (listas de redes), o sistema considera também outras informações, tais como: a estrutura dos barramentos e as características funcionais dos componentes. Estas informações estão armazenados numa base de conhecimento na forma de fatos e regras do PROLOG, sendo facilmente expandidas e/ou modificadas.
. pode ser aplicada a uma grande variedade de problemas, embora a implementação atual trate de PCIs digitais estruturadas em barramentos.

VII.1 - Resultados Obtidos

O protótipo encontra-se atualmente instalado num microcomputador tipo IBM-PC/AT com sistema operacional IBM-DOS3.1, no DPET-CEPEL. Os testes com a primeira versão do protótipo mostram que:

. os resultados obtidos com o protótipo são melhores que os obtidos com a técnica clássica proposta em (BREUER e QUINN [1987]) que consiste em utilizar o algorítmo FDP seguido do algorítmo de superposições.
. desabilitar os especialistas do ALEX piora significativamente a qualidade final do leiaute, o que permite concluir que as regras existentes atualmente representa (pelo menos em parte) o conhecimento do especialista humano sobre o problema.

Algumas das inovações existentes no projeto do SAP são:

. utilização de um novo paradigma para modelagem da arquitetura de sistemas de CAD/CAE.

SISTEMAS = CONHECIMENTO + ALGORÍTMOS

. identificação da estratégia (particionamento, posicionamento relativo e otimização) utilizada pelos especialistas humanos na busca da solução do problema de posicionamento.
. obtenção de um pequeno conjunto de regras que representam o conhecimento e a experiência do especialista.
.a utilização do método conhecido como Modelagem da Informação mostrou ser de grande utilidade no auxílio à etapa de análise e especificação de bancos de dados.

O resultado dos primeiros testes realizados sobre as duas placas dos apêndices C e D, mostram que a técnica proposta é bastante promissora. Entretanto, um número maior de testes devem ser realizados, antes de chegarmos a um conlusão definitiva.

VII.2 - Sugestões para Trabalhos Futuros

Os trabalhos futuros a serem realizados no SAP podem ser classificados em duas categorias: "otimização" da versão atual; e utilização do sistema em outros tipos de problemas de posicionamento.

a) Otimização da versão atual - Algumas das modificações que podem ser realizadas no SAP para melhorar o seu desempenho e aumentar a sua abrangência para atender um número maior de classes de PCIs são as seguintes:

. No SAP, todas as situações que não forem identificadas pelos especialista do ALEX serão tratadas pela parte algorítmica do sistema, ou seja, o desempenho do sistema é sempre melhor ou igual ao desempenho obtido com os métodos clássicos. Portanto, é desejável identificar e adicionar novas regras à base de conhecimento para aumentar a "esperteza" do ALEX no caso de outros tipos de placas digitais, e investigar novas regras aplicáveis a placas analógicas.

. Uma característica inconveniente do método de eliminação de superposições empregado refere-se ao fato de impor como restrição que os componentes sejam sempre posicionados dentro de células ("slots") da placa. Uma solução alternativa é utilizar o método proposto por (WIPFLER, WIESEL e MLYNSKI [1982]). Essa solução apresenta duas vantagens sobre o método anterior: elimina a necessidade de dividir a PCI em células; permite implementar o mecanismo de decisão das linhas de corte utilizando regras, portanto dentro do espírito da tese. Entretanto,

a identificação de um conjunto de regras aceitáveis promete ser uma tarefa bastante trabalhosa.

. PCIs que possuam um número grande de componentes, onde a maioria destes pertence à parte randômica, pode degradar bastante o desempenho do sistema, visto que o circuito será tratado quase exclusivamente de forma algorítmica. Para solucionar este problema, propõe-se aplicar o algorítmo de crescimento de sementes descrito em (ODAWARA et alii [1982] e GOTO [1981]) sobre os componentes da parte randômica, ao invés de utilizar o algorítmo FDP seguido do método de eliminação de superposições.

b) utilização em outros problemas de posicionamento - devido à generalidade da arquitetura do sistema, o SAP pode ser modificado para ser utilizado em um número maior de aplicações. Estas modificações consistem basicamente em obter novos fatos e regras para a base de conhecimento. Uma classe de problemas bastante importante está relacionada à confecção automática de leiautes de circuitos integrados. O sistema SAP poderia ser utilizado, com algumas alterações na base de conhecimento e na arquitetura do ALEX, para auxiliar as etapas de definição da planta baixa e posicionamento de componentes em circuitos integrados que utilizem uma metodologia de projeto tipo "gate-array" ou "standard-cell".

REFERÊNCIAS BIBLIOGRÂFICAS

AHO, A. V., GAREY, M. R., HUANG, F. K., (1977), "Rectilinear Steiner Trees: Efficient Special Cases of Algorithms", Networks, N° 7, pp 37-58.

BALL, M. O., DERIGS, U., (1980), "Analysis of Alternatives for Implementing Matching Algorithms", Reprint Series, R143-WP82226, Institute Bur Okonometrie und Operations Research, Rheinische Friedrich - Wilhelms - Universitat.

BARATZ, A. E., (1981), "Algorithms for Integrated Circuit Signal Routing", Ph.D. Dissertation, Massachusetts Institute of Technology - MIT, August 1981.

BHAWMIK, S., PALCHAUDHURI, P., (1989), "Expert System to Configure Global Design for Testability Structure in VLSI Circuit", Microprocessors and Microsystems, Vol. 13, N° 7, pp 462.

BORLAND, (1988a), "Turbo Prolog Reference Manual V2.0", Borland International Inc.

BORLAND, (1988b), "Turbo Prolog Users Guide V2.0", Borland International Inc.

BOXLEITNER, W., (1989), "How to Defeat Eletrostatic Discharge", IEEE Spectrum, August 1989, pp 36-40.

BREUER, M. A., (1972), "Design Automation of Digital Circuits: theory and techniques", Ed. New York Prentice-Hall, Vol. 1, chapter 5, 1972.

CARPINTEIRO, O. A. S., (1988), "Um Sistema Especialista para Escolha dos Fatores de Escala do TNA", Dissertação de Msc., COPPE-UFRJ, Prog. de Eng. de Sistemas, março de 1988.

CHANG, S., (1972), "The Generation of Minimal Trees With Steiner Topology", Journal of the Association for Computer Machinery, Vol. 4, N^{o} 19, october 1972, pp 669-711.

COSTA,R.S., PIMENTEL,J.C.G., (1984) "Editor Gráfico para Projeto de Circuitos Integrados, IEEE LATINCON 84, México.

COSTA, R. S., PIMENTEL, J. C. G., (1986), "Editor Gráfico para Auxílio a Projeto de Circuitos Eletrônicos", 6o Congresso Brasileiro de Automática, pp 526-531.

DEJESUS, E. J., CALLAN, J. P., WHITEHEAD, C. R., (1986), "PEARL: An Expert System for Power Supply Layout", 23^{a} Design Automation Conf. Proc., pp 615-621.

FERREIRA, A. A. P., (1989), "Estudo de Algorítmos Posicionadores para o Leiaute de Placas de Circuito Impresso", TFC, Dpto. Eng. Elétrica, PUC-RJ, Publicação Interna, janeiro de 1989.

FERREIRA, A. B. H., (1986), "Novo Dicionário Aurélio da Lingua Portuguesa", segunda edição, Ed. Nova Fronteira.

FISK, C. J., CASKEY, D. L., WEST, L. E., (1967), "ACCEL: Automated Circuit Card Etching Layout", Proceedings IEEE, Vol. 55, 1971-1982, november 1967.

FORBES, R., (1987), "Heuristic Acceleration of Force-Directed Placement", 24^{a} Design Automation Conf. Proc., pp 735-740.

GOTO, S., (1981), "An Efficient Algorithm for the Two-Dimensional Placement Problem in Electrical Circuit Layout", IEEE Trans. on Circuits and Systems, Vol. 28, N^{o} 1, pp 12-18.

GOTO, S., KUH, E. S., (1978), "An Approach to the Two-Dimensional Placement Problem in Circuit Layout", IEEE Trans. on Circuits and Systems, Vol. 25, N° 4, pp 208-214.

HANAN, M., KURTZBERG, J. M., (1972), "A Review of the Placement and Quadratic Assignment Problems", SIAM Review, Vol. 14, No 2, pp 324-342.

HANAN, M., WOLFF, P. K., ANGULE, B. J., (1976), "A Study of Placement Techniques for Computer Logic Graphs", 13a Design Automation Conf. Proc., pp 214-224.

HARMON, P., KING, D., (1985), "Expert Systems: Artificial Intelligence in Business", John Wiley and sons, Inc.

HAYES-ROTH, F., (1984), "Knowledge-Based Expert Systems", IEEE Computer, Vol. 17, N° 10, october 1984, pp 263-273.

HUHNS, M. N., ACOSTA, R. D., (1988), "ARGO: A System for Design by Analogy", IEEE Expert, fall 1988, pp 53-68.

JOOBANI, R., (1987), "An Artificial Intelligence Approach to VLSI Routing - WEAVER", Kluwer Academic Publishers

KERNIGHAN e LIN, B. W., LIN, S., (1970), "An Efficient Heuristic Procedure for Partitioning Graphs", Bell System Thecnical J., Vol. 49, No 2, pp 291-307.

KOO, S., (1985), "Fact-Based Expert Systems", VLSI Design, june 1985, pp 122-123.

KUHN, H. W., (1955), "The Hungarian Method for the Assignment Problem", Naval Research Logistics Quarterly, Vol. 2, pp 83-97.

LIN, S., GAJSKI, D. D., (1987), "LES: A Layout Expert System", 24ª Design Automation Conf. Proc., pp 672-679.

LOOSEMORE, K. J., (1979), "Automated Layout of Integrated Circuits", Proceedings of the IEEE International Symposium on Circuits and Systems, pp 665-668.

MORI, H., MITSUMOTO, K., FUJITA, T., GOTO, S., (1984), "Knowledge-Based VLSI Routing System - WIREX", Proceedings of the International Conference on Fifth Generation Computer Systems, Ed. by ICOT, pp 383-388.

ODAWARA, G., IIJIMA, k>, KIYOMATSU, T., (1982), "Arbitrarily-Sized Module Location Tecnique in the LOP System", 19ª Design Automation Conf. Proc., pp 718-726.

ODAWARA, G., IIJIMA, K., KIYOMATSU, T., (1983), "Partitioning and Placement Technique for Bus-Structured PWB", 20ª Design Automation Conf. Proc., pp 449-456.

ODAWARA, G., IIJIMA, K., KIYOMATSU, T., HAMURO, T., DAI, Y., (1987), "A Rule-Based Placement System for Printed Wiring Boards", 24ª Design Automation Conf. Proc., pp 777-785.

ODAWARA, G., IIJIMA, K., WAKABAYASHI, K., (1985), "Knowledge-Based Placement Technique for Printed Wiring Boards", 22ª Design Automation Conf. Proc., pp 616-622.

PIMENTEL, J. C. G., BANDIM, C. J., (1989), "CADD - Sistema para Confecção de Layouts de Circuitos Eletrônicos", Centro de Pesquisas de Energia Elétrica - CEPEL, Dpto. de Eletrônica, Relatório Técnico Interno.

PREAS, B. T., KARGER, P. G., (1986), "Automatic Placement: A Review of Current Techniques", 23ª Design Automation Conf. Proc., pp 622-629.

QUINN, N. R., BREUER, M. A., (1979), "A Forced Directed Component Placement Procedure for Printed Circuit Boards", IEEE Trans. on Circuits and Systems, Vol. 26, Nº 6, pp 377-388.

SCHWEIKERT, D. G., (1976), "A Two-Dimensional Placement Algorithm for the Layout of Eletrical Circuits", 13ª Design Automation Conf. Proc., pp 408-414.

SHA, L., DUTTON, R.W., (1985), "An Analytical Algorithm for Placement of Arbitrarily Sized Rectangular Blocks", 22a. Design Automation Conf.Proc., pp 602-608.

SHING, M. T., HU, T. C., (1986), "Computational Complexity of Layout Problems", Layout Design and Verification, Ed. T. Ohtsuki, Elsevier, Science Publishers B. V., chapter 8.

SHLAER, S., MELLOR, S. J., (1988), Object-Oriented Systems Analysis: modelling the world in data", Yourdn Press, Prentice-Hall.

STARK, P. A., (1979), "Introdução aos Métodos Numéricos", Ed. Interciência, 1979, cap. 4.

STEELE, R. L., (1987), "An Expert System Application in Semicustom VLSI Design", 24ª Design Automation Conf. Proc., pp 679-686.

STEINBERG, L., (1961), "The Backboard Wiring Problem: a placement algorithm", SIAM Review, Vol. 3, Nº 1, pp 37-50.

SUARIS, P. R., KEDEM, G., (1988), "An Algorithm for Quadrisection and its Application to Standard Cell Placement", IEEE Trans. on Circuits and Systems, Vol. 35, N^{o} 3, pp 294-303.

ULLMAN, J. D., (1980), "Principles of Database Systems", Computer Science Press, Inc., (1980).

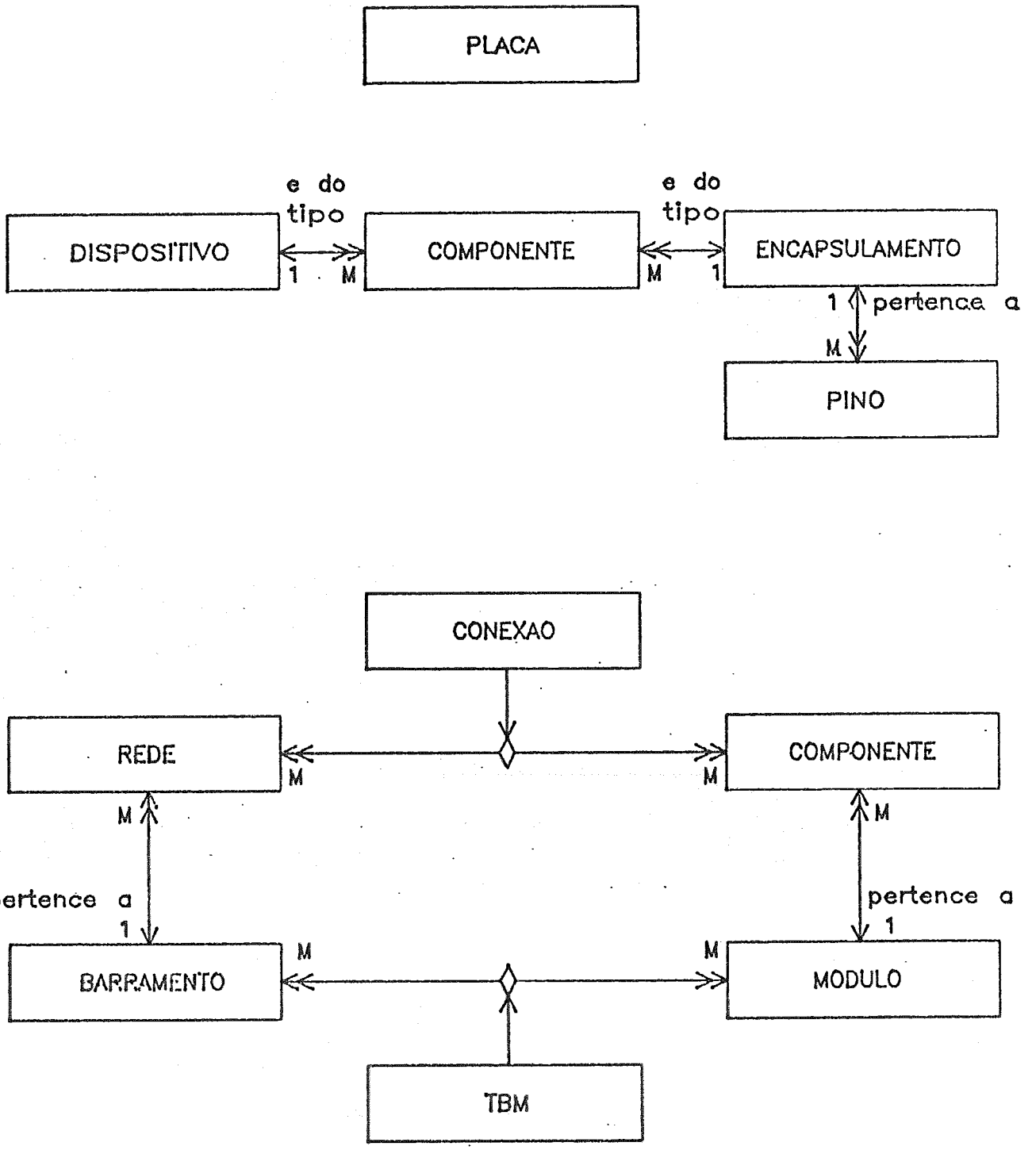
WIES, B. J., MYLINSKI, D. A., (1988), "A Graphtheoretic Approach to the Relative Placement Problem", IEEE Trans. on Circuits and Systems, Vol. 35, march 1988, pp 286-293.

WILSON, D. C., SMITH, R. J., (1976), "An Analytic Technique for Router Comparasion", 13^{a} Design Automation Conf. Proc., pp 251-258.

WIPFLER, G. J., WIESEL, M., MLYNSKI, D. A., (1982), "A Combined Force and Cut Algorithm for Hierarchcal VLSI Layout", 19^{a} Design Automation Conf. Proc., pp 671-677.

APÊNDICE A
Modelagem da Base de Dados

1) Diagrama de Estrutura de Objetos



2) Documento de Especificação de Objetos e Relacionamentos.

a) Especificação dos Objetos:

. PLACA
- nome da placa (*)
- nome do encapsulamento => define o padrão de placa que será utilizado para confeccionar o leiaute (ex.: padrão IBM-PC).
- área útil => região disponível para o leiaute.
- número de módulos
- número de componentes
- número de redes
- número de barramentos
- número de pinos
- direção pré-definida para os componentes.
- dimensao X dos slots.
- dimensao Y dos slots.

. MÓDULO
- nome do módulo (*)
- definição do contorno (x_e, y_i, x_d, y_s)
- posição dentro da placa (o_x, o_y)
- função (ex.: ram, rom, etc.)
- mobilidade (ex.: fixo, móvel)
- direção pré-definida para os seus componentes
- tipo do barramento que comanda o módulo (BPG, BPL, BSG, BSL).

. COMPONENTE
- nome (*)
- tipo de encapsulamento (ex.: DIP 28) (R)
- tipo do dispositivo (ex.: 2732) (R)
- função do componente (ex.: rom)
- direção do componente
- definição do contorno
- posição dentro do módulo (o_x, o_4)
- nome do módulo (R)

- mobilidade

. ENCAPSULAMENTO
- nome (*)
- definição do contorno
- origem (o_x, o_y)
- número de pinos

. PINO
- código do pino (*)
- número do pino
- nome do encapsulamento (R)
- tipo do pino => define sua forma
- posição dentro do encapsulamento (o_x, o_y)

. REDE
- nome (*)
- tipo da rede (ex.: clock)
- número de pinos
- nome do barramento (R)

. BARRAMENTO
- nome (*)
- largura
- comprimento
- conectividade
- tipo de barramento (BPG, BPL, BSG, BSL)

. DISPOSITIVO
- nome do dispositivo (*) (ex.: 80386)
- tipo do dispositivo (ex.: cpu)

b) Especificação dos relacionamentos

. CONEXÃO => este relacionamento representa o fato de redes estarem ligadas aos pinos dos compomentes.

- nome da rede (*)
- nome do componente (*)

- número do pino (*)

. TBM (tabela barrametno x módulos) => relaciona os barramentos aos módulos que estão ligados a esses barramentos.

- nome do barramento (*)
- nome do módulo (*)
- número de ligações

. PERTENCE_A => associa um objeto A a um objeto B tal que A faz parte do objeto B. Por exemplo: componentes e módulos; redes e barramentos; e pinos e encapsulamentos.

. É_DO_TIPO => associa um objeto A a um objeto B tal que o conjunto de todos os atributos de B são um subconjunto dos atributos do objeto A (ex.: componentes e dispositivos: e componentes e encapsulamentos).

APÊNDICE B
Resultado dos Testes com os Algorítmos

B.1 - Sintaxe do Arquivo de Definição de Circuitos

<controno da placa> <num. de módulos>
<diemnsao X dos Slots> <dimensao Y dos slots>

<nome do modulo> <num. de componentes> <contorno>
<nome do componente> <contorno> <mobilidade>
<nome do componente> <contorno> <mobilidade>
.
.
.

<nome do modulo> <num. de componentes> <contorno>
<nome do componente> <contorno> <mobilidade>
<nome do componente> <contorno> <mobilidade>
.
.
.

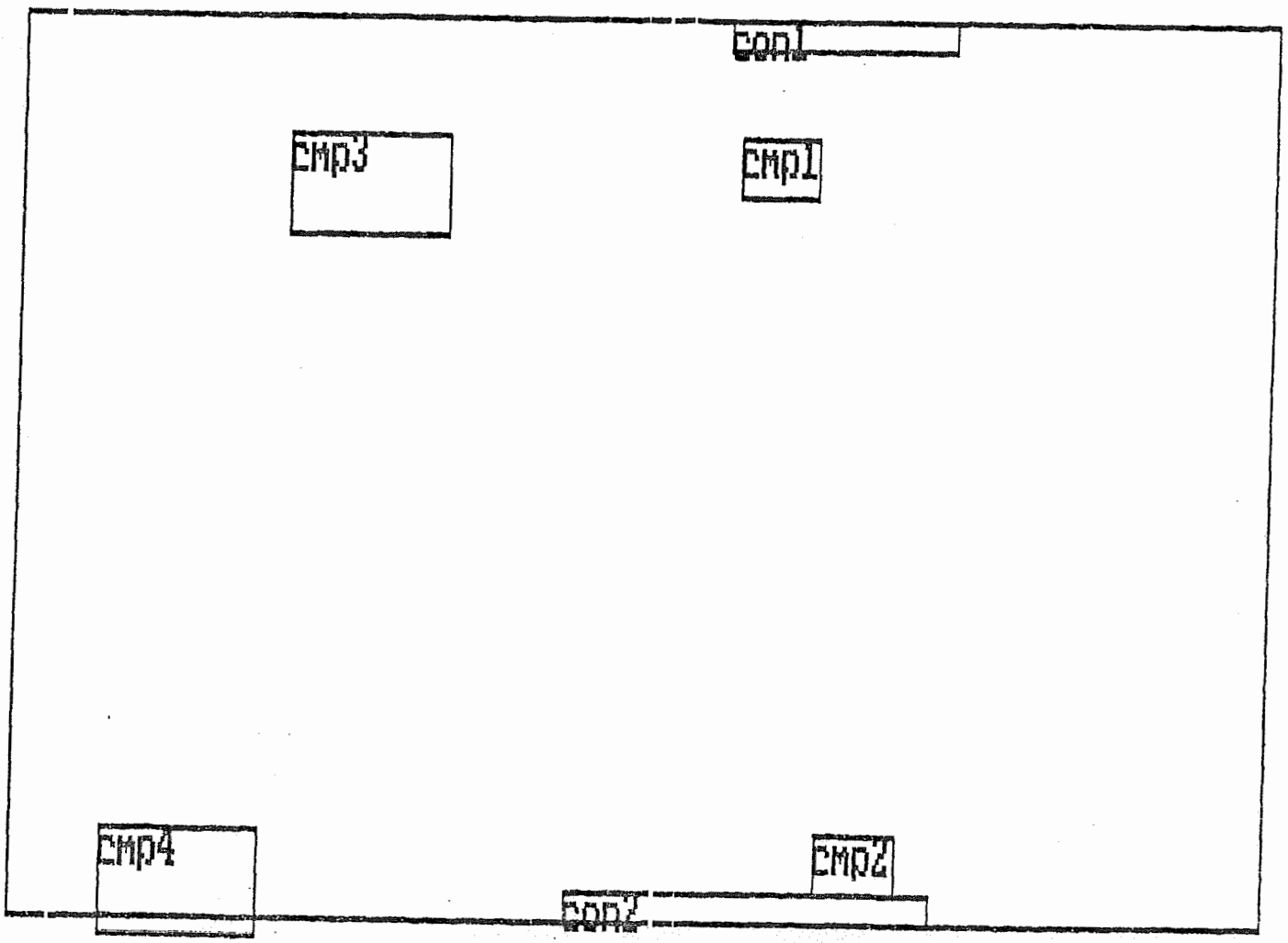
C12
C13 C23
C14 C24 C25
C15 C25 C35 C36
.
.
.

OBS: Cij é o numero de ligacoes do componente i com o componente j.

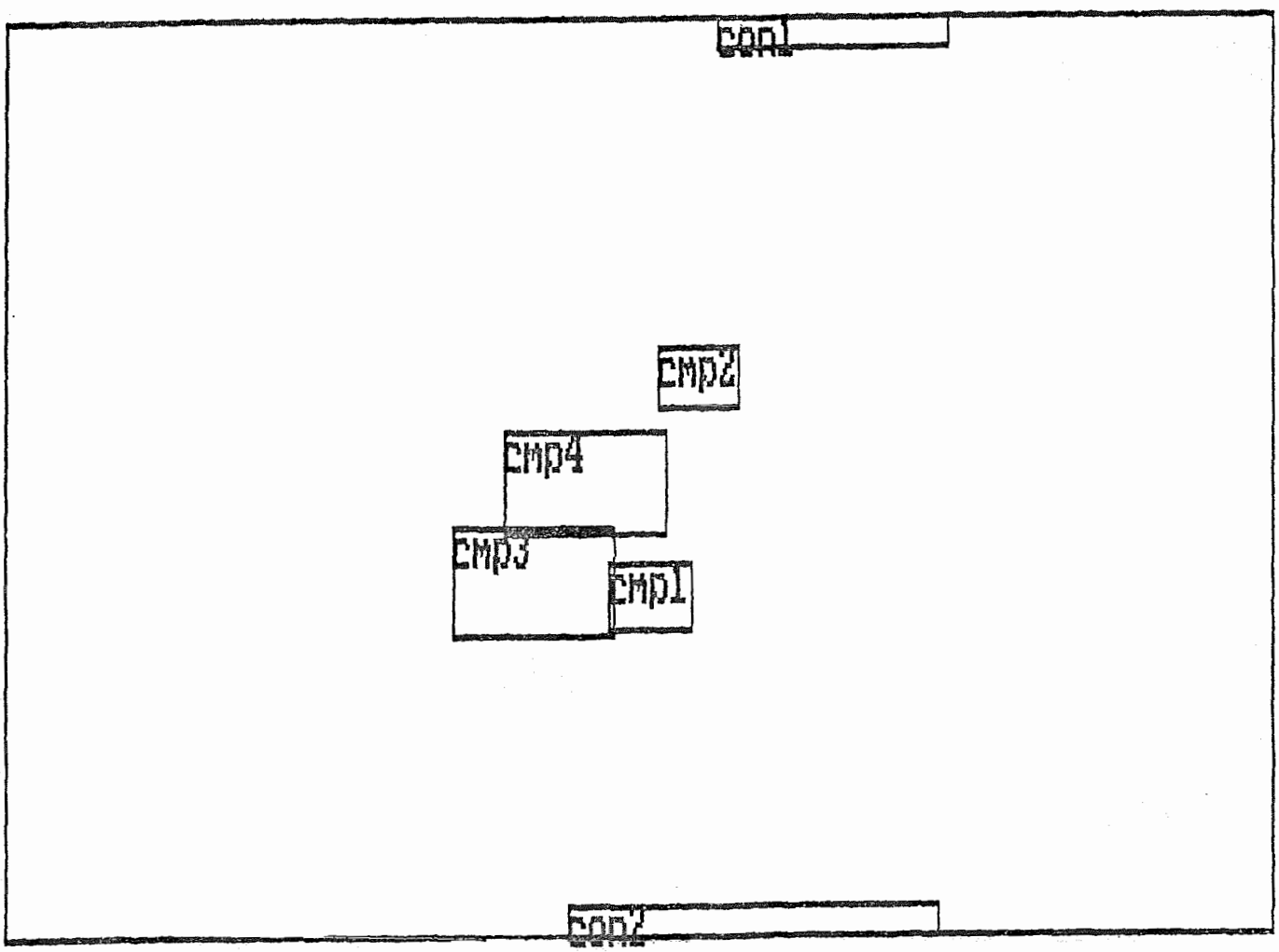
B.2 - Arquivo de descrição do circuito de testes CT1

```
0.000    0.000    28.050   15.460   1
1.778    1.016

algtst 6   0.000    0.000    28.050   15.460
con1    15.720    0.000   20.800   0.508      0
con2    12.460   14.950   20.590   15.460     0
cmp1    16.0 2.0 17.778 3.016 1
cmp2    18.0 14.0 19.778 15.016 1
cmp3    6.0 2.0 9.556 3.778 1
cmp4    2.0 14.0 5.556 15.778 1

0
3 6
6 3 0
0 0 4 1
0 0 3 3 0
```

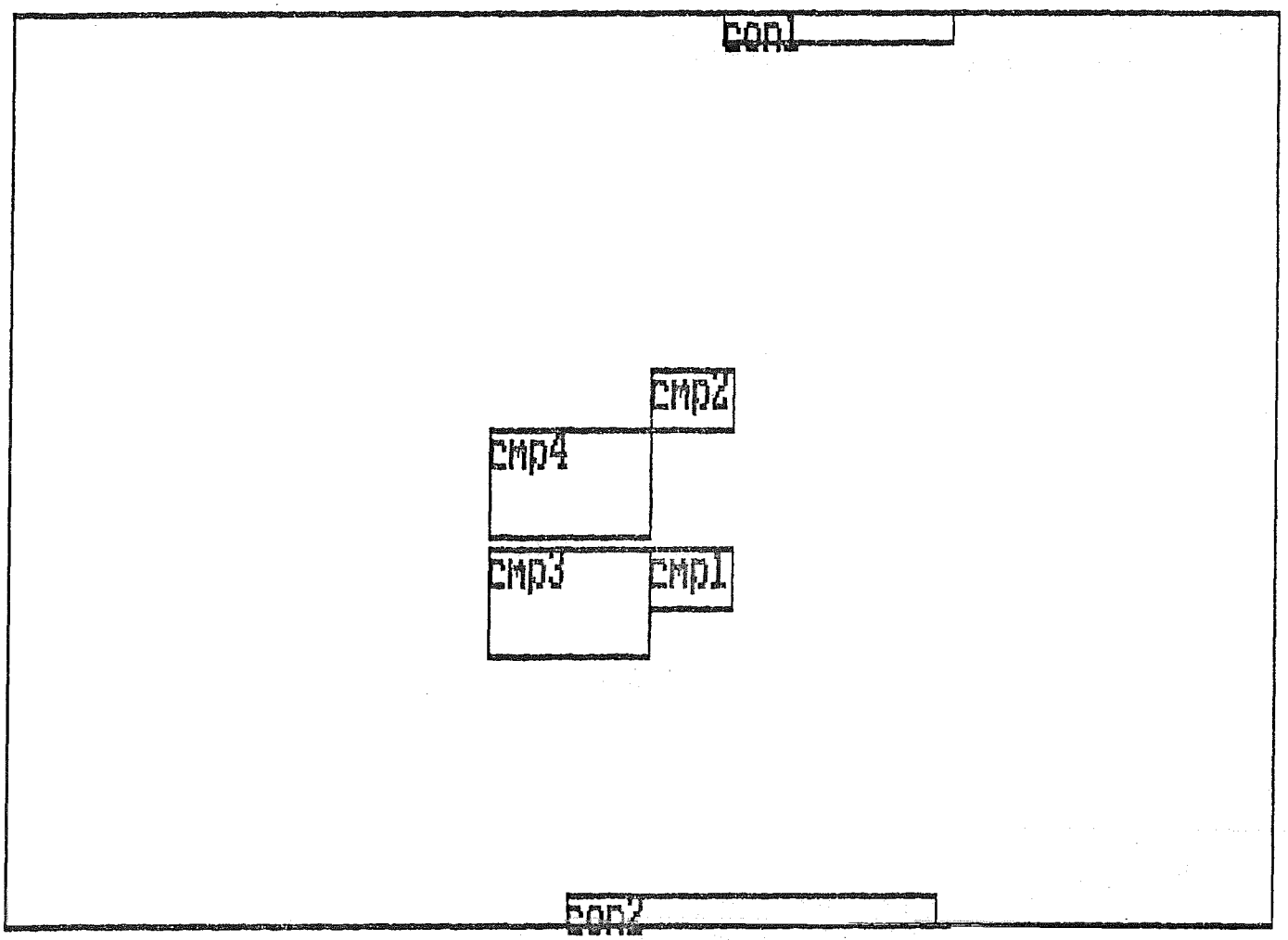
B.3 - Posicionamento Inicial de CT1



B.4 - Posicionamento Relativo de CT1



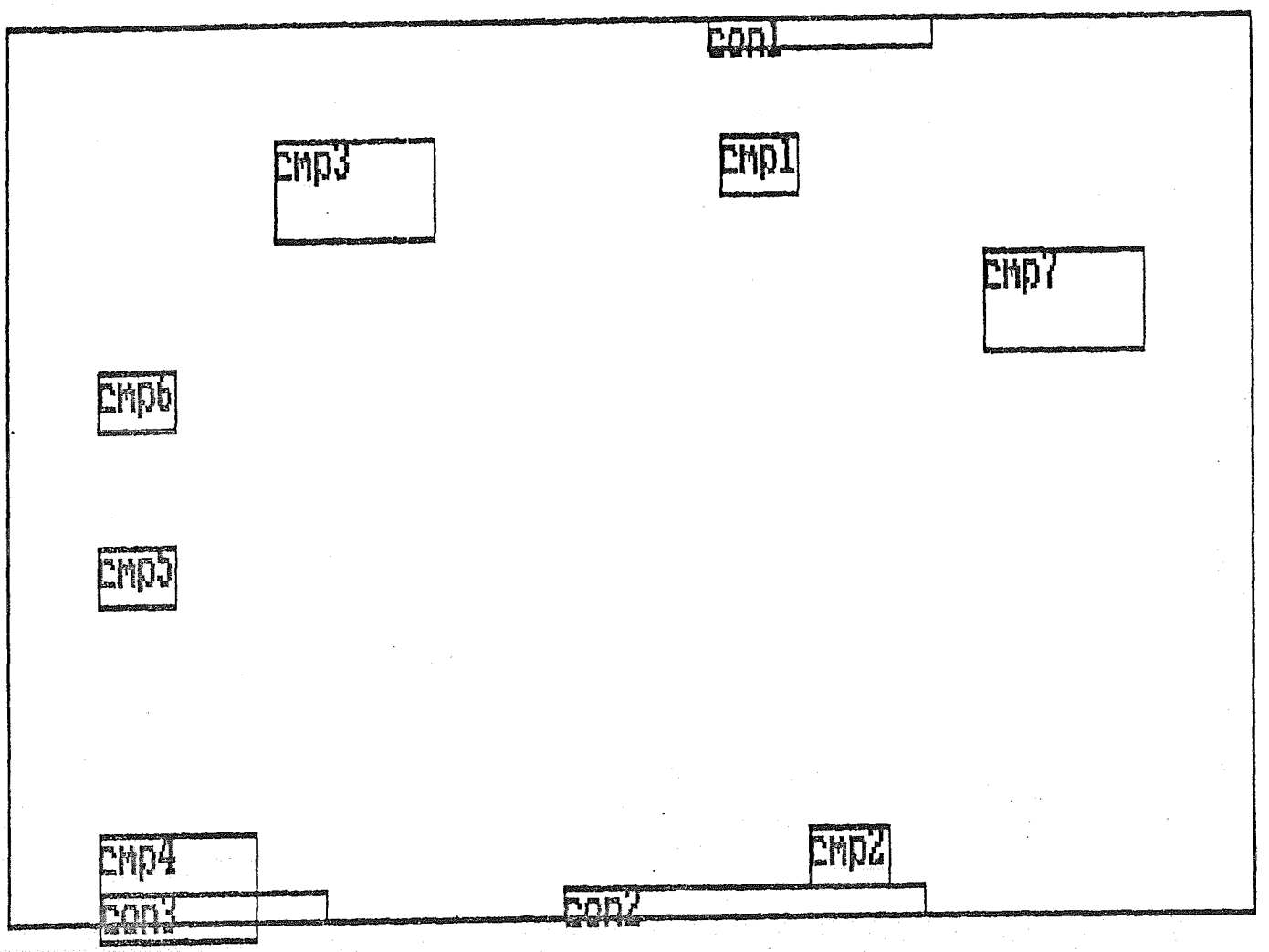
B.5 - Eliminação das superposições de CT1



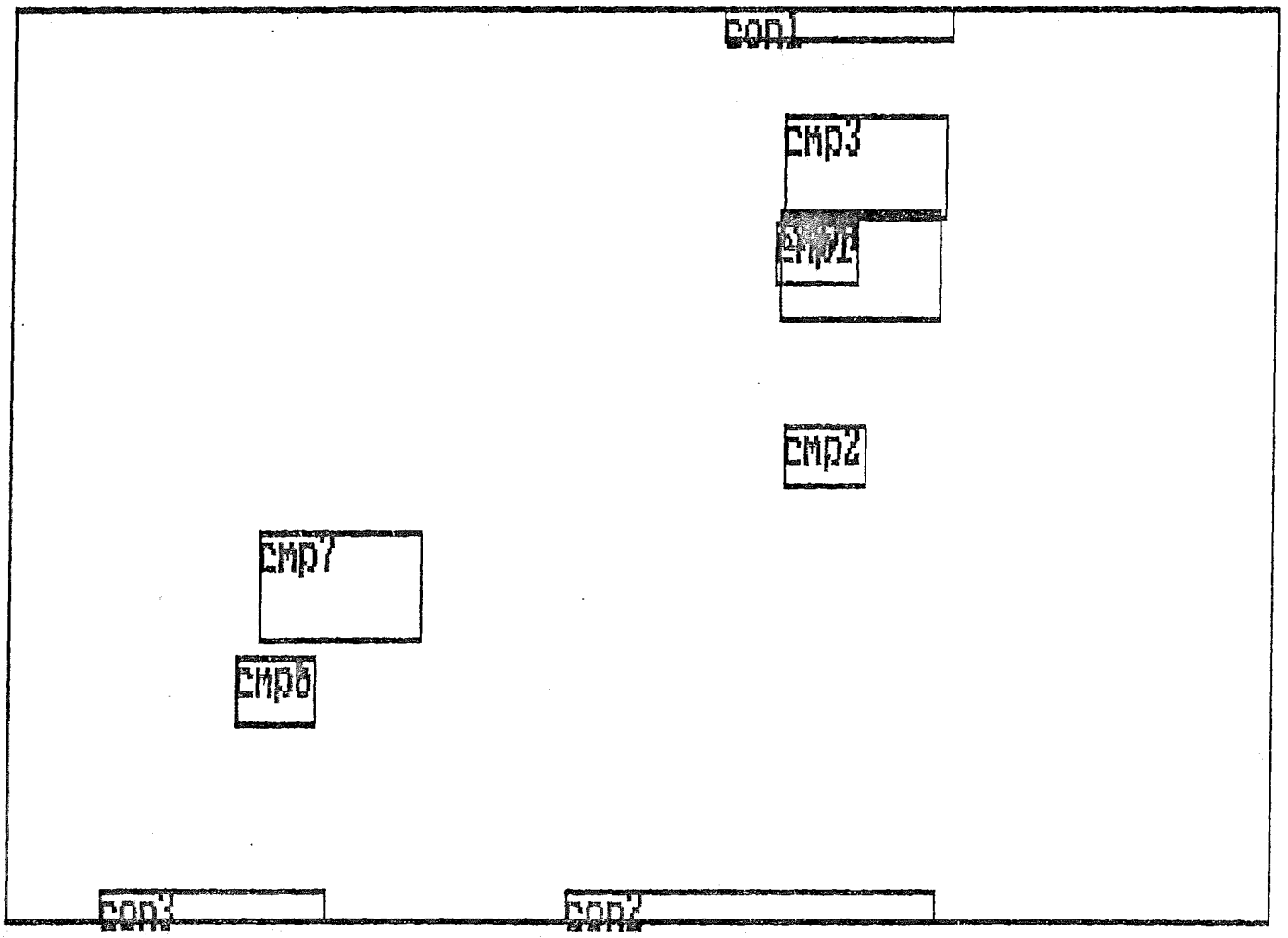
B.6 - Arquivo de descrição do circuito de testes CT2

```
0.000    0.000    28.050   15.460   1
1.778    1.016

algtst 10   0.000    0.000    28.050   15.460
con1    15.720    0.000   20.800   0.508      0
con2    12.460   14.950   20.590   15.460     0
cmp1    16.0   2.0 17.778 3.016 1
cmp2    18.0 14.0 19.778 15.016 1
cmp3    6.0 2.0 9.556 3.778 1
cmp4    2.0 14.0 5.556 15.778 1
cmp5    2.0 9.0 3.778 10.016 1
cmp6    2.0 6.0 3.778 7.016 1
cmp7    22.0 4.0 25.556 5.778 1
con3    2.082   14.950    7.162   15.460    0

0
6 3
3 6 0
0 0 4 1
0 0 3 3 0
0 0 0 0 0 0
0 0 0 0 0 0 0
0 0 1 0 0 0 4 4
0 0 0 0 0 0 4 4 0
```

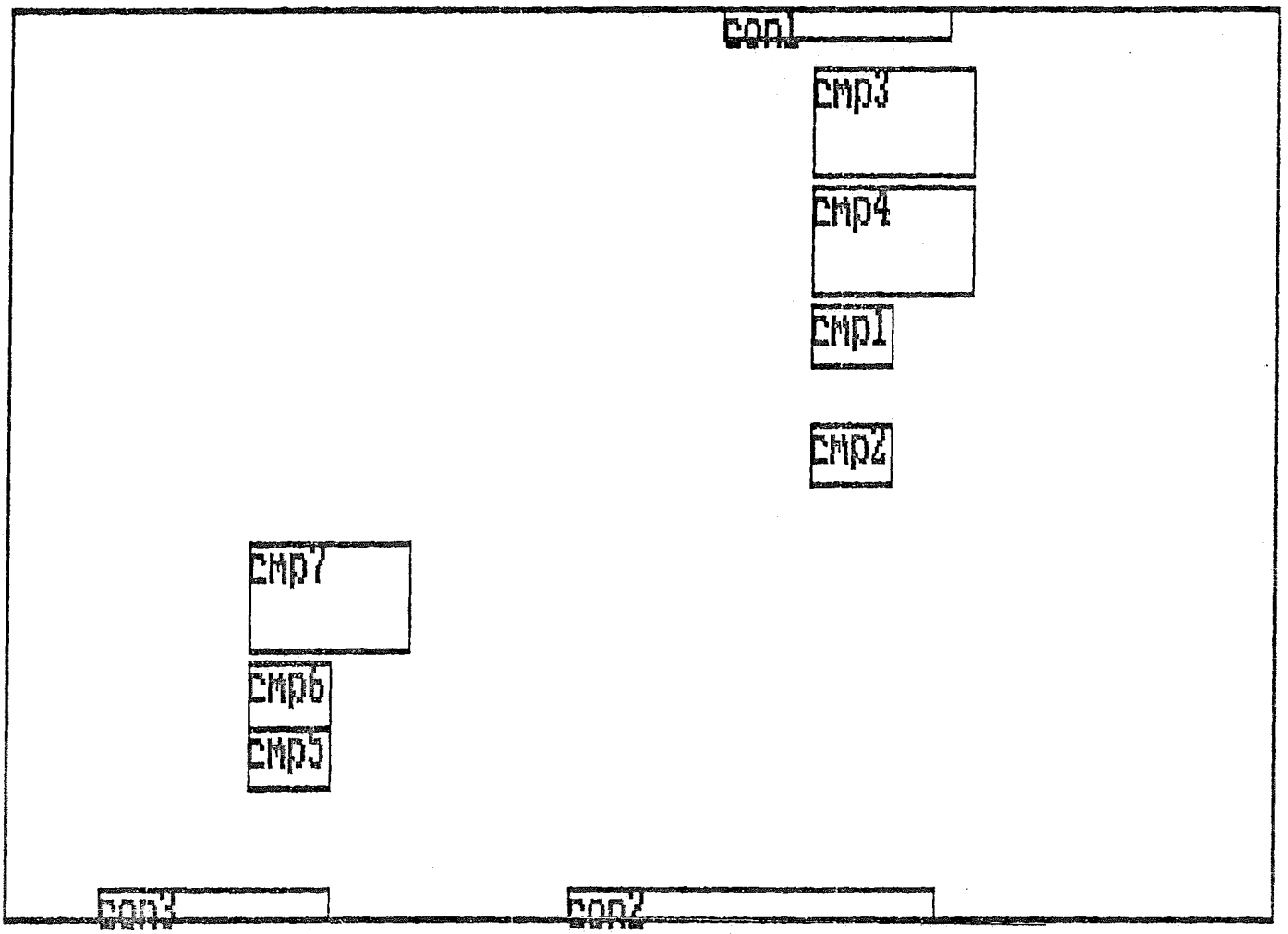
B.7 - Posicionamento Inicial de CT2



B.8 - Posicionamento Relativo de CT2

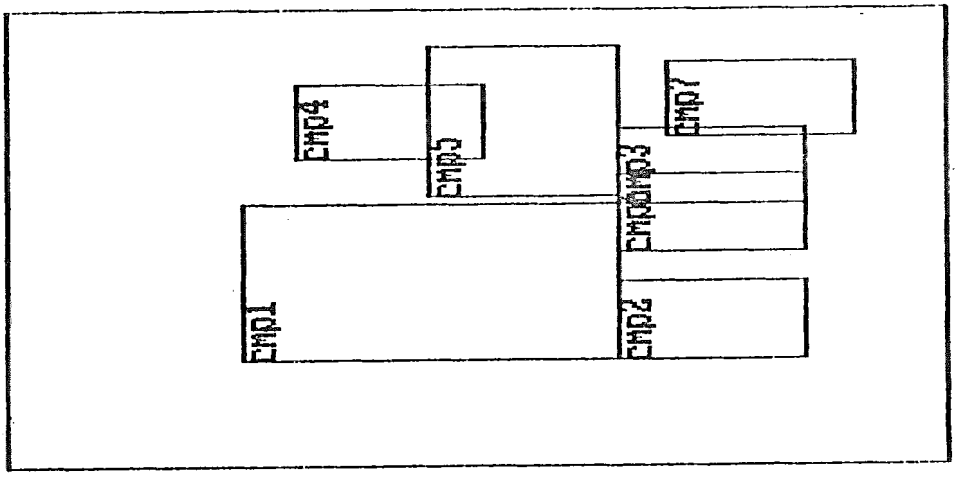


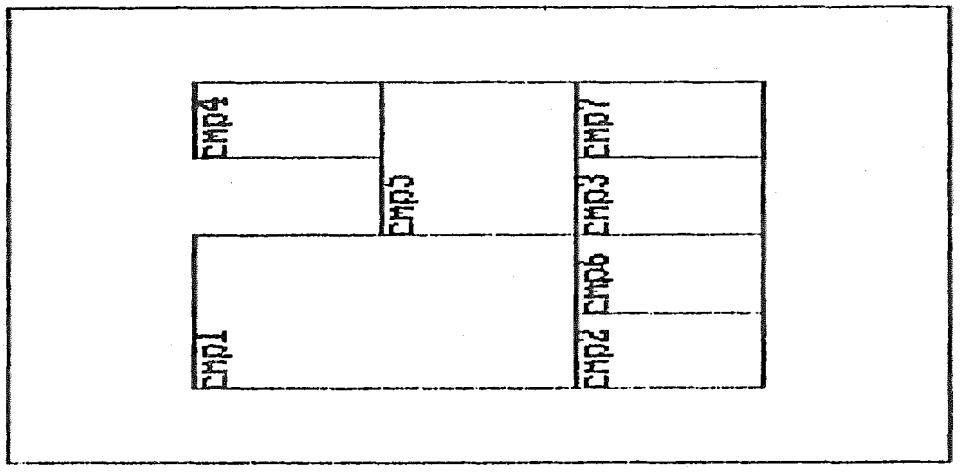
B.9 - Eliminação das superposições de CT2



B.10 - Teste Proposto por (ODAWARA et alii [1982])

```
0.  0.  6.0 10.0 2
1.0 2.0

mod1     4 0.0 0.0 6.0 10.0
cmp1     1.4 2.5 3.4 6.5 1
cmp2     1.4 6.5 2.4 8.5 1
cmp3     3.4 6.5 4.4 8.5 1
cmp4     4.0 3.1 5.0 5.1 1

mod2     3 0.0 0.0 6.0 10.0
cmp5     3.5 4.5 5.5 6.5 1
cmp6     2.8 6.5 3.8 8.5 1
cmp7     4.3 7.0 5.3 9.0 1
```





APÊNDICE C
Placa de Testes
da Base de Conhecimento

C.1 - Arquivo de Definição do Circuito

```
0.000   0.000   23.000  24.500  1
1.778   1.016

aline  16  0.000   0.000   23.000  24.500
xmod83 -0.127  -0.127  2.413   0.889     1
xmod99 -0.127  -0.127  1.905   0.889     1
xmod82 -0.127  -0.127  4.953   1.651     1
xmod57 -0.127  -0.127  2.413   0.889     1
xmod77 -0.127  -0.127  2.413   0.889     1
xmod17 0.373   13.990  1.135   22.120    0
xmod18 0.373   2.381   1.135   10.510    0
fmod1  0.000   0.000   4.833   4.833     1
xmod84 0.000   0.000   3.556   1.778     1
fmod2  0.000   0.000   3.556   3.556     1
xmod74 1.135   16.020  3.675   17.040    21
xmod62 1.135   15.640  6.215   17.420    21
xmod68 1.135   21.290  2.913   22.310    21
xmod61 1.135   20.720  6.215   22.500    21
xmod64 1.135   6.064   6.215   7.842     21
xmod63 1.135   7.080   6.215   8.858     21

0.4
3.067 1.8
2.667 0 2.667
0 0 0 2.179
0 0 0 0 0
0 0 0 0 0 0
3.517 0 2.667 2.667 0 0 0
6.917 1.4 5.4 2.667 0 0 0 3.517
6.917 1.8 5.8 2.667 0 0 0 3.517 9.25
0 0 0 2.179 2.179 4.5 0 0 0 0
```

0.45 0 0 2.179 2.179 19.5 0 0.45 0.45 0.45 6.679

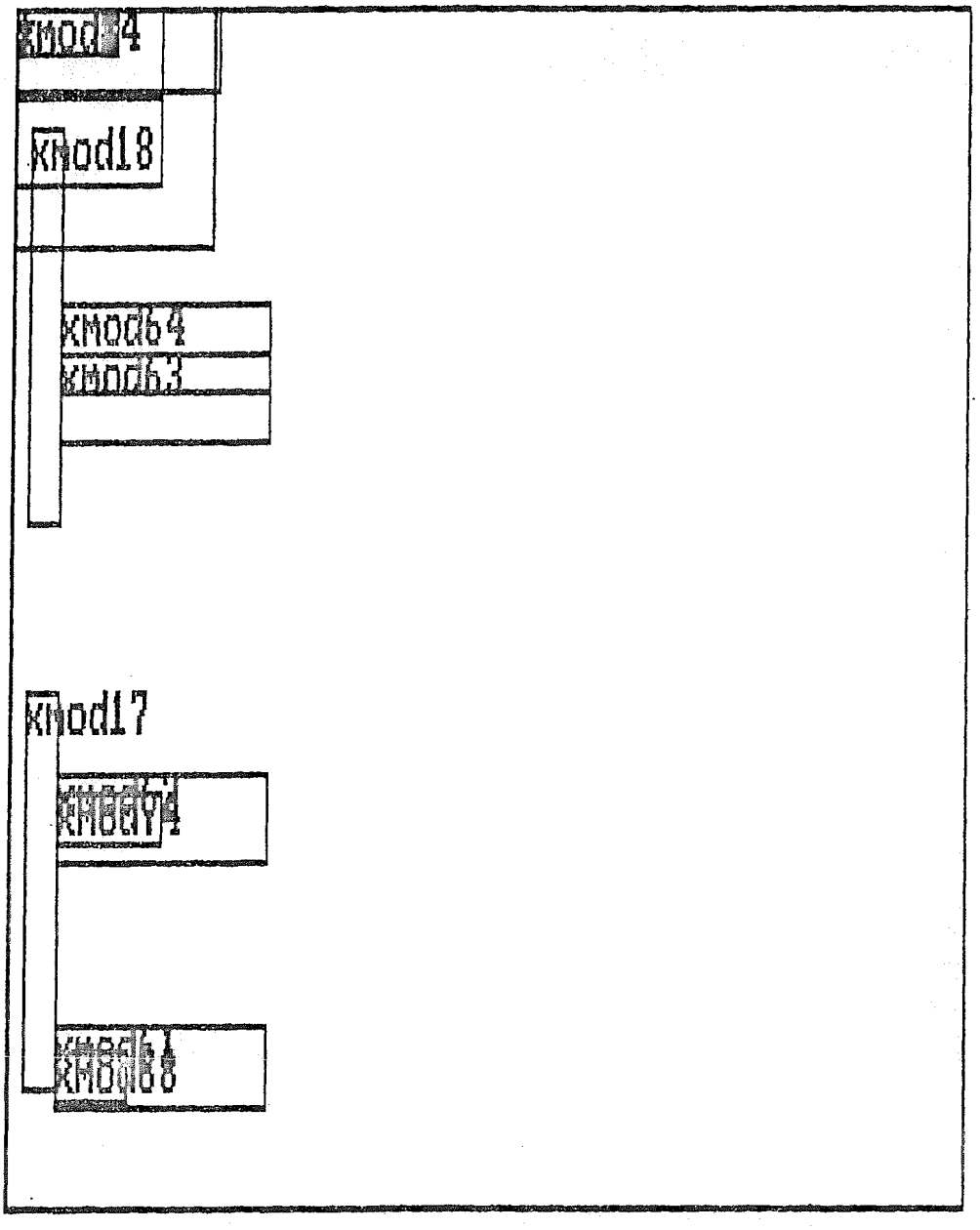
0 0 0 0 0 2.667 0 0 0 0 0 0

0.45 0 0 2.179 2.179 20 0 0.45 0.45 0.45 2.18 2.63 2.7

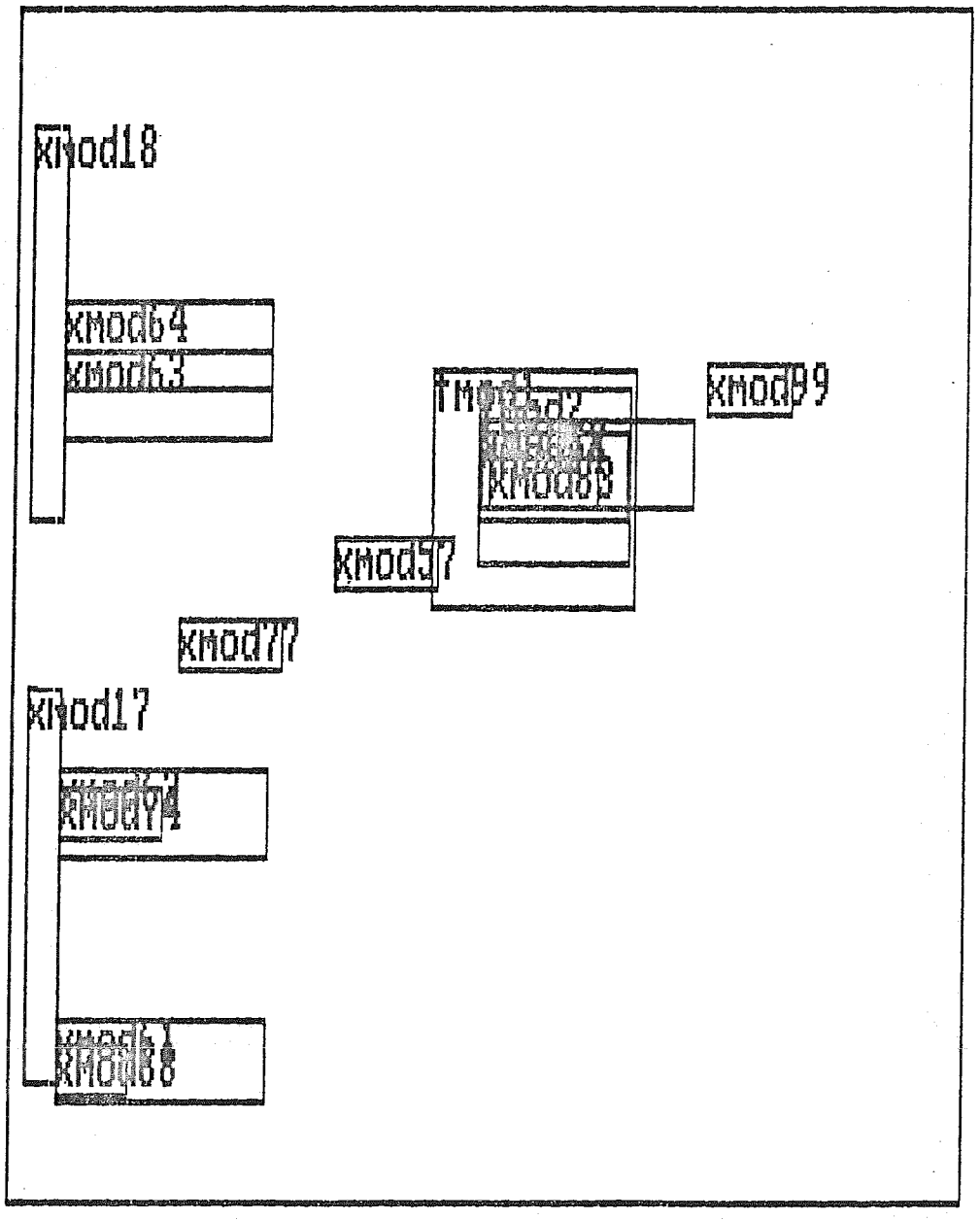
0.45 0 0 2.2 2.2 0 23.33 0.45 0.45 0.45 2.2 2.6 0 2.6

0.45 0 0 2.2 2.2 0 24 0.45 0.45 0.45 2.2 2.6 0 2.6 2.6

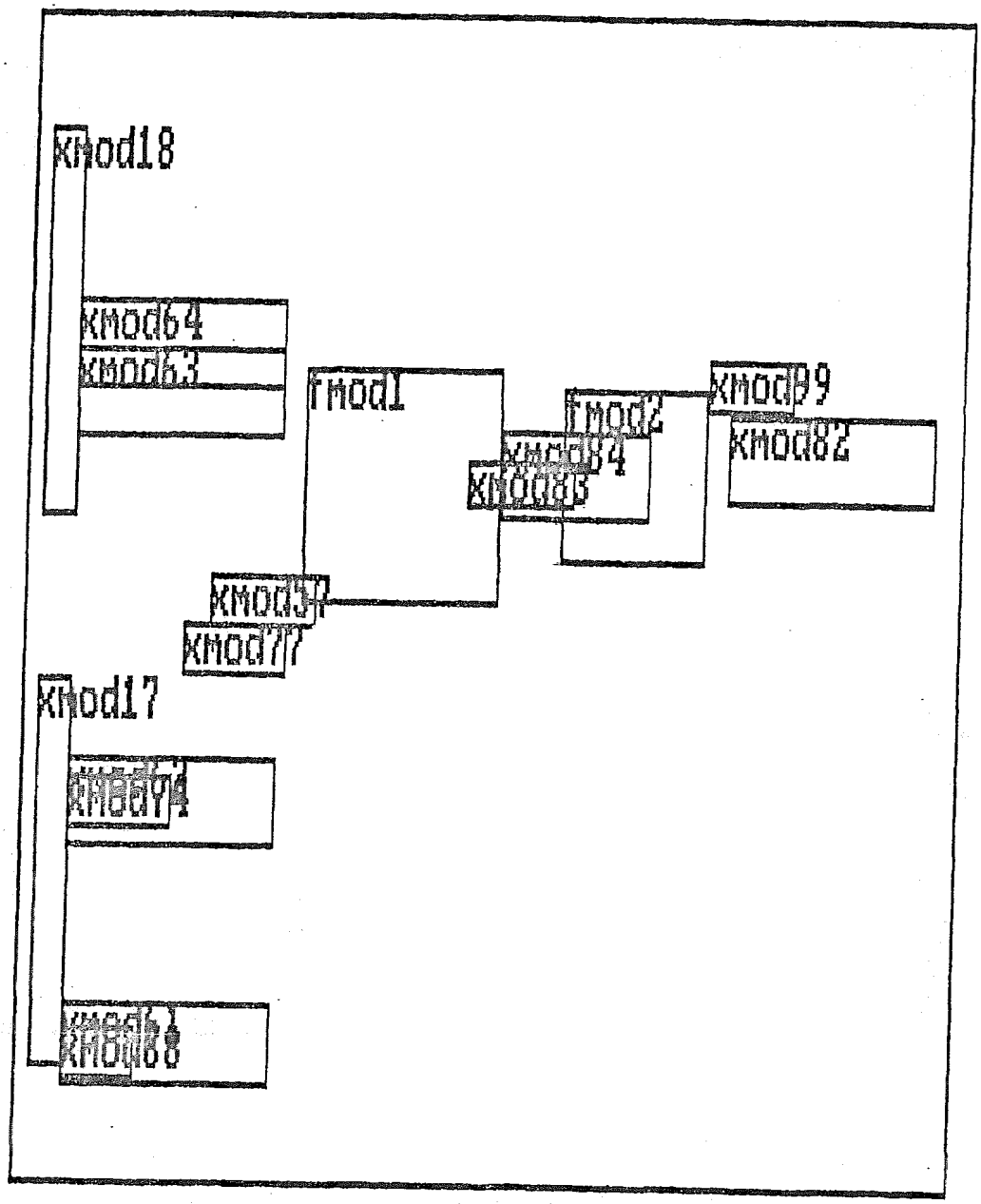
C.2 - Resultado do Posicionamento Antes de Disparar Regra Relativa ao FDP.



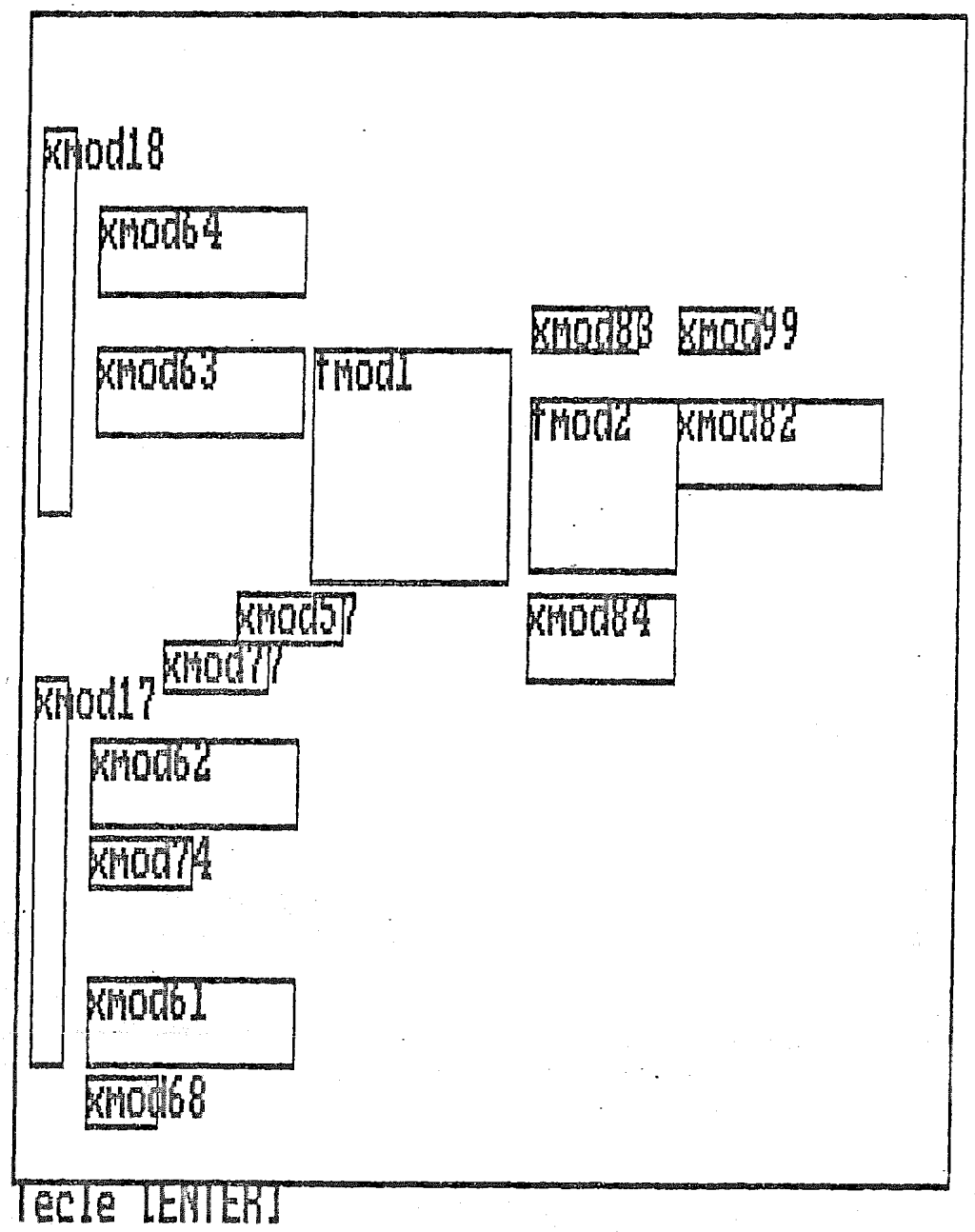
C.3 - Posicionamento após FDP



C.4 - Eliminação Parcial das Superposições



C.5 - Eliminação Total das Superposições



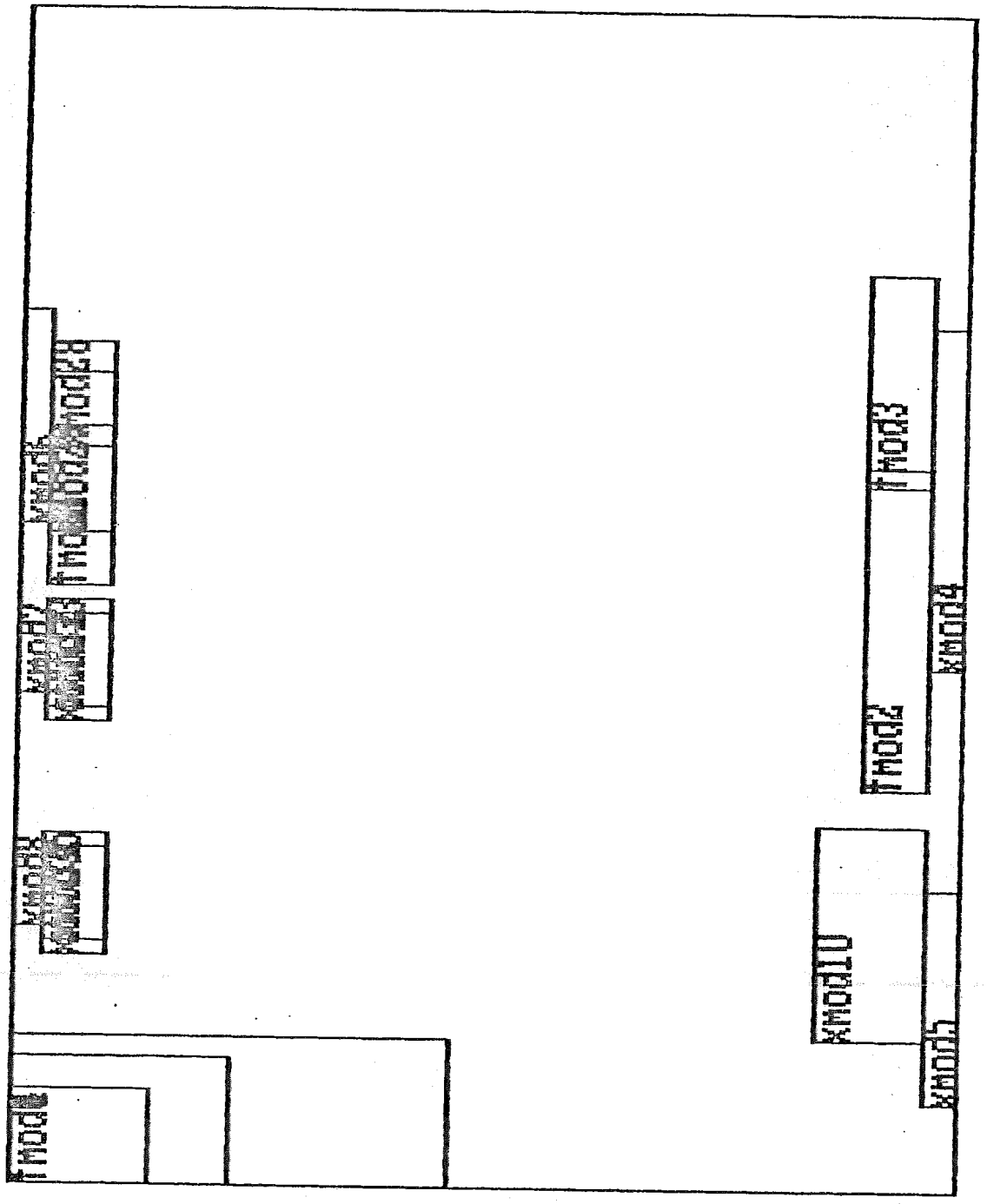
OAPÊNDICE D
Teste Completo

D.1 - Arquivo de Definição do Circuito

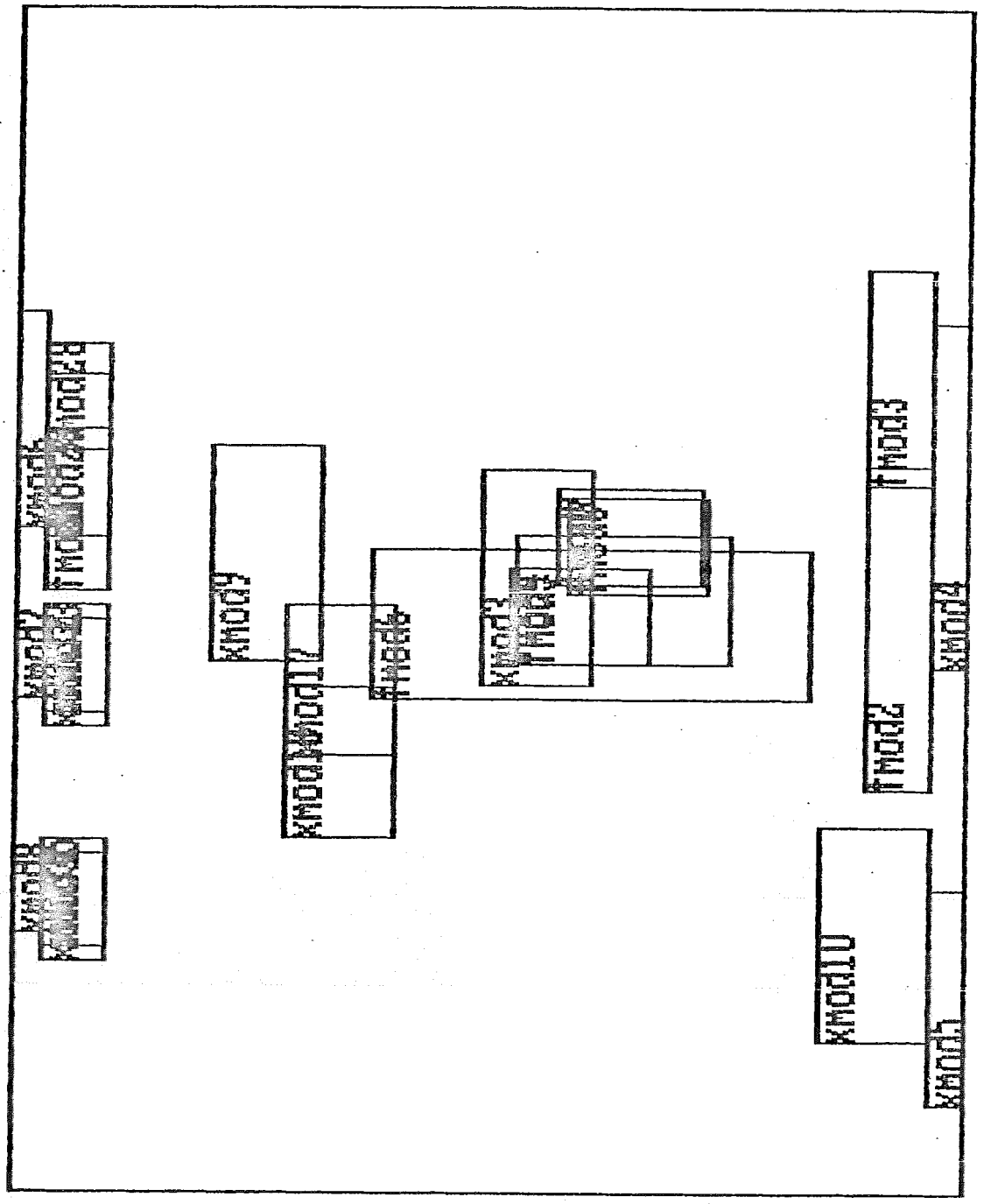
```
0.000    0.000    28.050   15.460   1
1.778    1.016

odaw     24   0.000    0.000    28.050   15.460
xmod3    -4.953   -1.651   0.127    0.127      1
xmod18   -3.429   -1.651   0.127    0.127      1
xmod6    15.720   -0.000   20.800   0.508      0
xmod4    12.460   14.950   20.590   15.460     0
xmod9    -4.953   -1.651   0.127    0.127      1
xmod5    2.082    14.950   7.162    15.460     0
xmod7    11.670   -0.000   13.070   0.508      0
xmod8    6.147    -0.000   7.544    0.508      0
xmod17   -3.429   -1.651   0.127    0.127      1
fmod4    0.000    0.000    2.272    2.272      1
fmod7    0.000    0.000    2.272    2.272      1
fmod8    0.000    0.000    2.272    2.272      1
fmod5    0.000    0.000    3.048    3.556      1
fmod6    0.000    0.000    3.556    7.112      1
xmod29   15.460   0.508    18.000   1.524     21
xmod28   17.500   0.508    20.030   1.524     21
fmod1    14.190   0.508    19.270   1.524     21
fmod2    9.574    13.940   17.190   14.950    21
fmod3    16.760   13.940   21.840   14.950    21
xmod10   3.606    13.170   8.686    14.950    21
xmod35   5.801    0.508    8.341    1.524     21
xmod34   5.455    0.508    7.995    1.524     21
xmod32   10.980   0.508    13.520   1.524     21
xmod33   11.320   0.508    13.860   1.524     21

0
0 0
0 0 0
0 2.0 0 0
```

```
0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0 0
0 2.0 0 0 2.0 0 0 0
10.67  2.0 0 0 2.667 0 0 0 2.0
10.67 0 0 0 0 0 0 0 0 10.67
0 0 0 0 0 0 0 0 0 0 7.0
0 2.0 0 0 2.667 0 0 0 2.0 5.667 5.0 5.0
0 2.0 0 0 2.667 0 0 0 2.0 5.667 0 13.0  5.667
0 0 8.0 0 8.0 0 0 0 0 0 0 0 0 0
0 0 8.0 0 8.0 0 0 0 0 0 0 0 0 0 0
0 0 8.0 0 8.0 0 0 0 0 0 0 0 0 0 0 0
0 2 0 16 2.667 0 0 0 2 5.667 0 0 5.667 5.667 0 0 0
0 0 0 15 0 0 0 0 0 0 10 7 5 0 0 0 0 0
0 2 0 0 2.667 24 0 0 2 2.667 0 0 2.667 2.7 0 0 0 2.7 0
0 4 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0
0 4 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 4 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 4 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

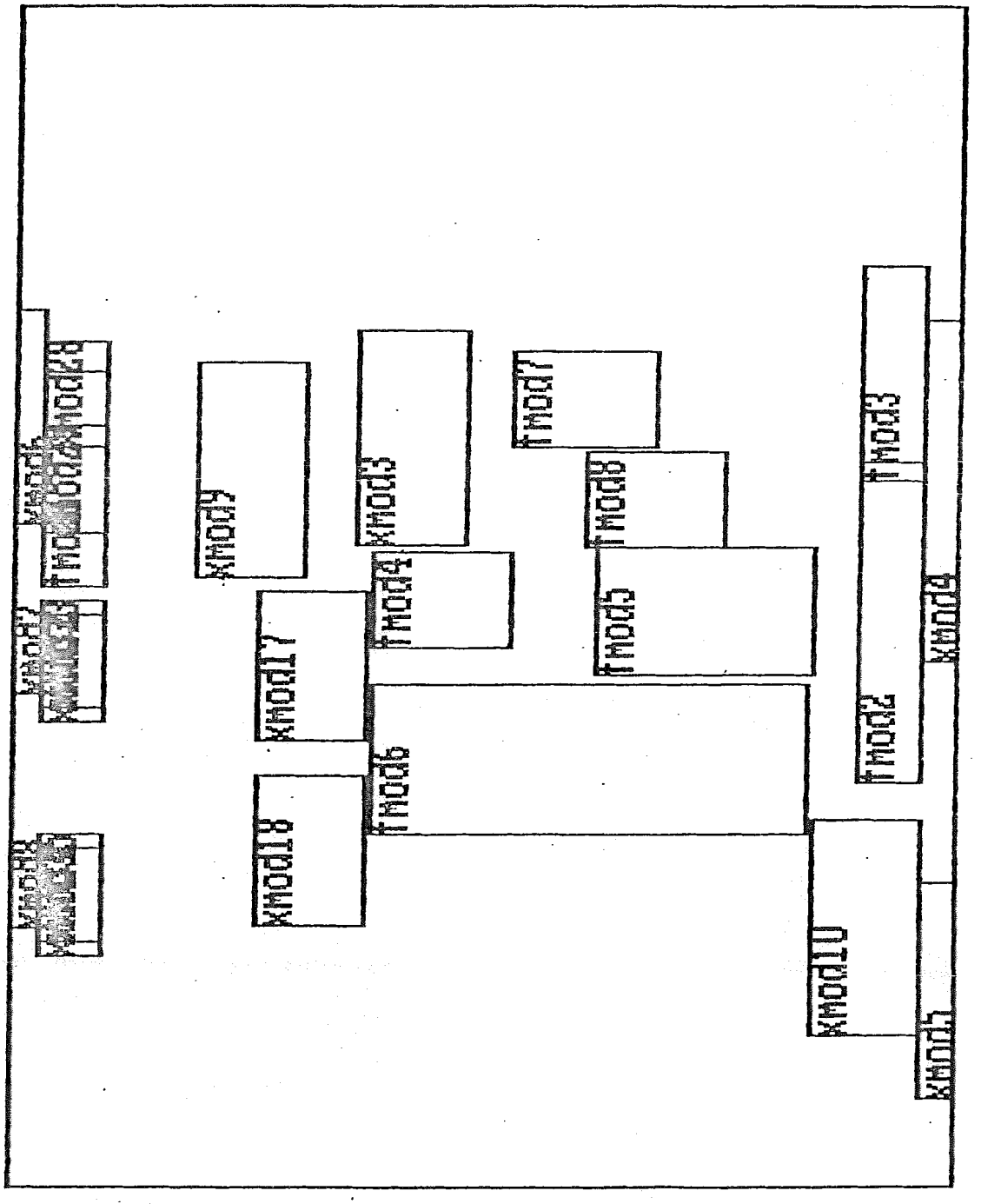
D.2 - Resultado do Posicionamento Antes de Disparar Regra Relativa ao FDP.



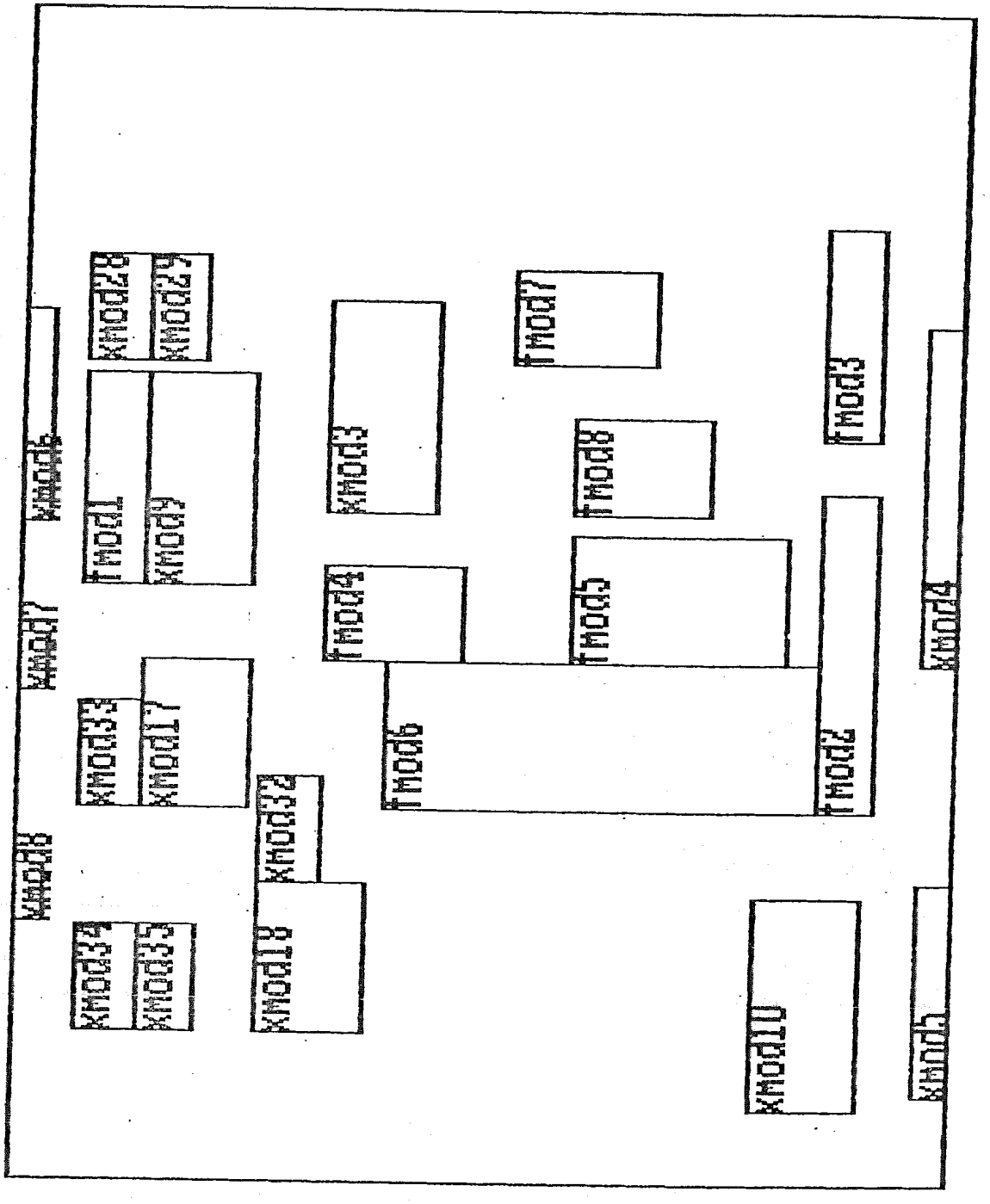
D.3 - Posicionamento após FDP



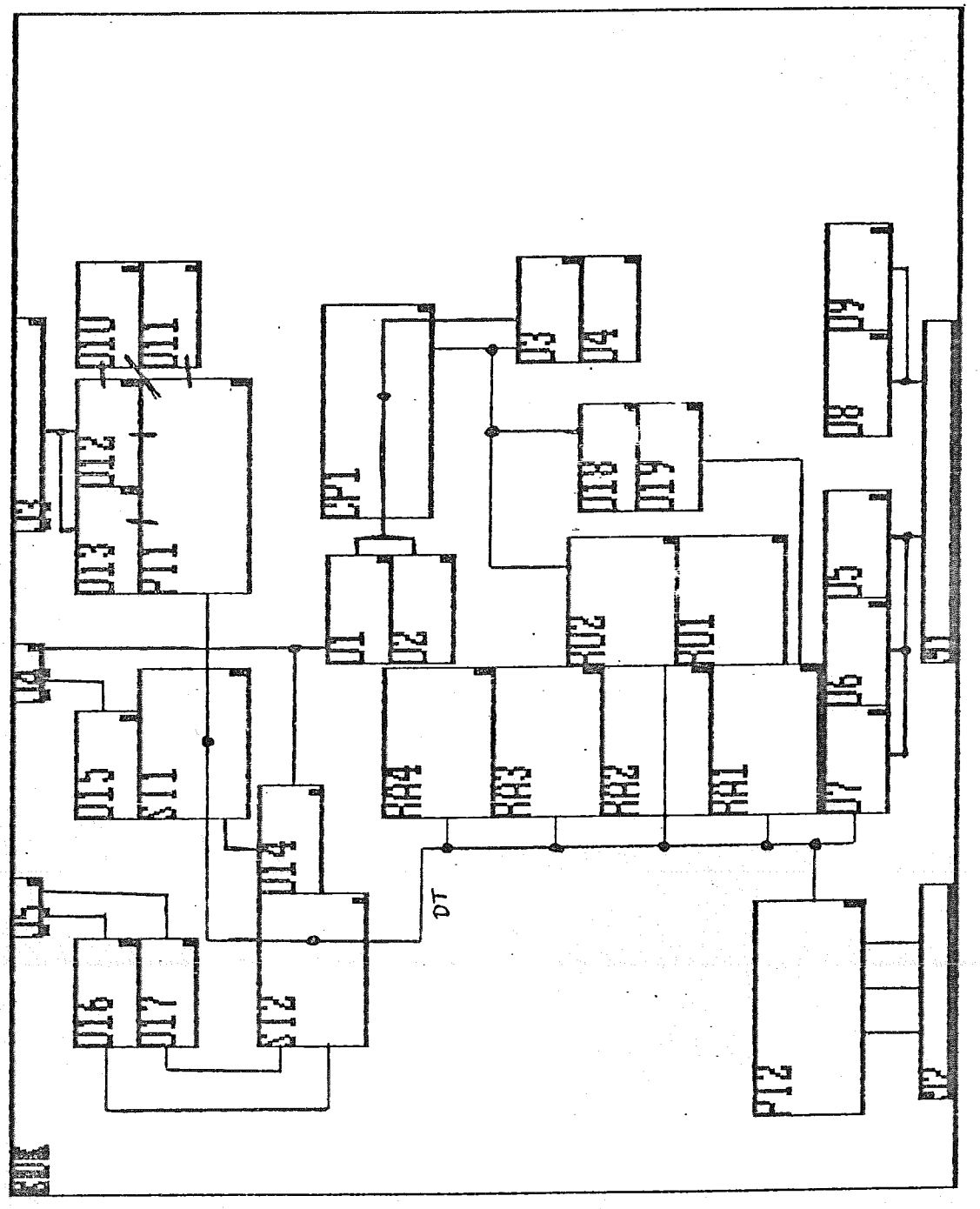
D.4 - Eliminação Parcial da Superposições



D.5 - Eliminação Total das Superposições

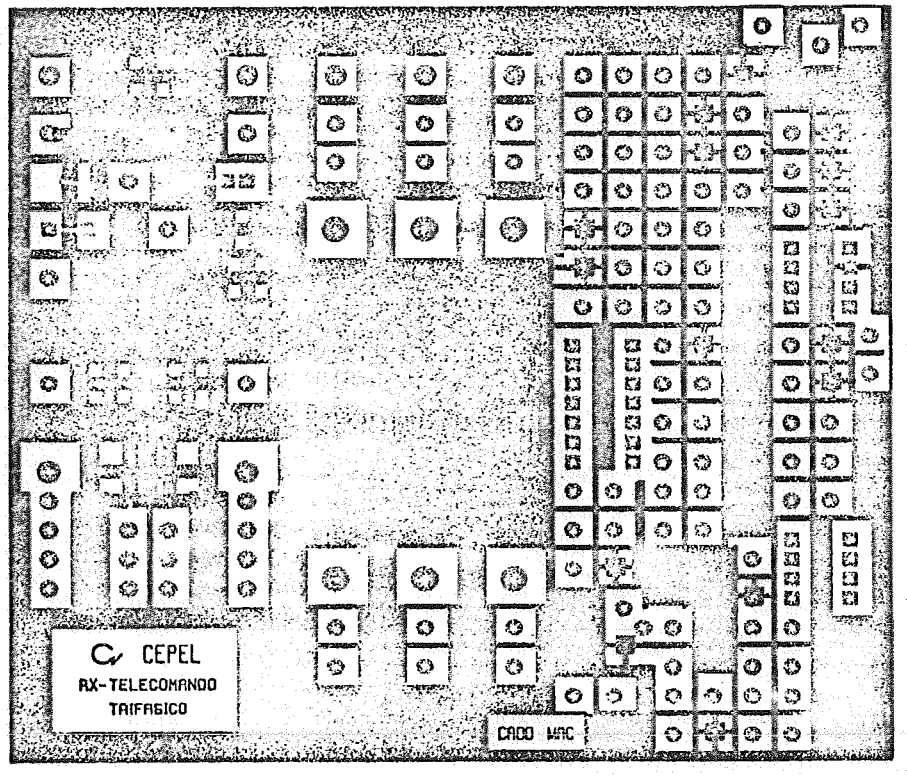


D.6 - Posicionamento dos Componentes Na PCI

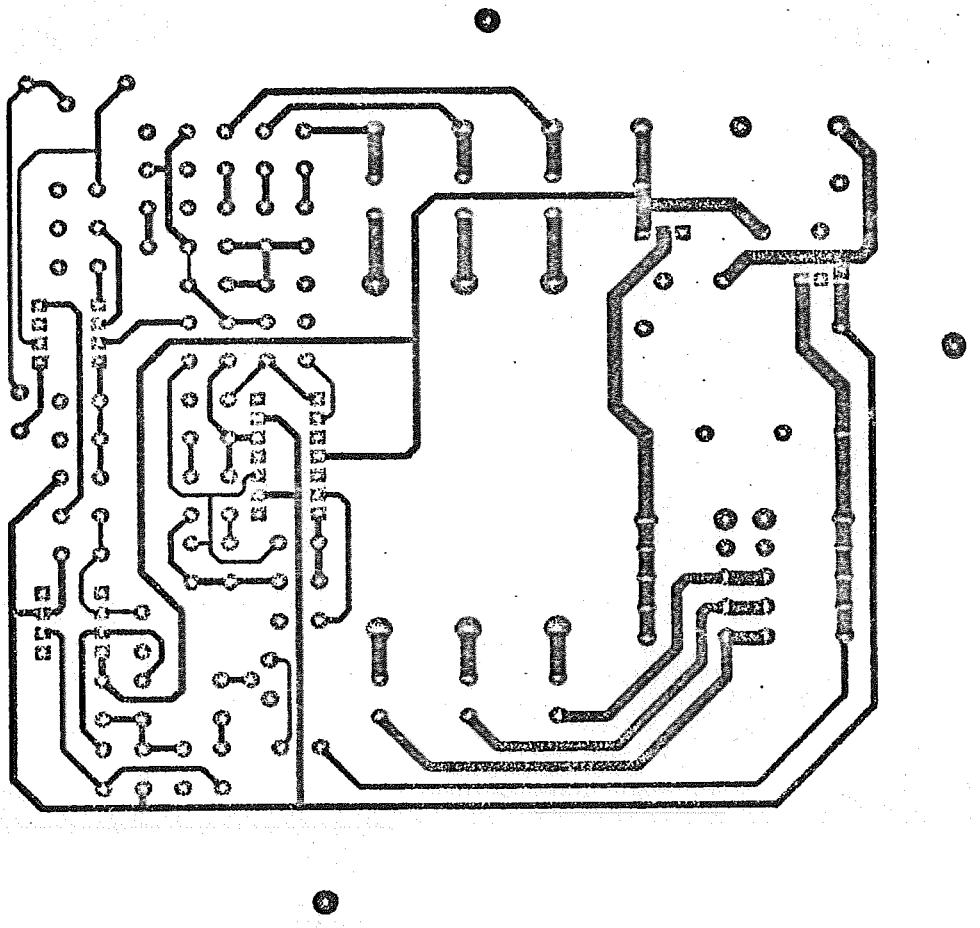


APÊNDICE E
Leiaute de Placas
Confeccionadas no Sistema CADD

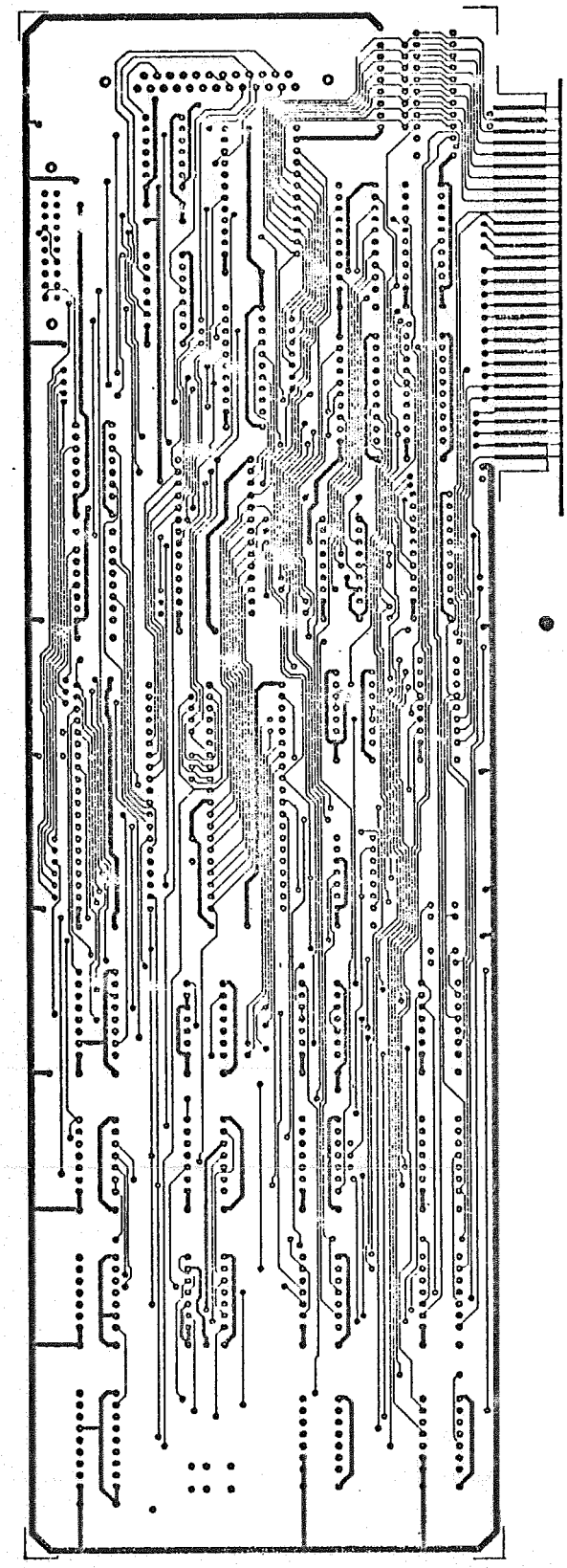
E.1 - Face de Componentes da Placa Analógica.



E.2 - Face de Solda Da Placa Analógica



E.3 - Face de Componentes da Placa Digital



E.4 - Face de Solda da Placa Digital