EDITOR GRAFICO INTERATIVO PARA PROJETOS DE CIRCUITOS INTEGRADOS

Este exemplar corresponde a redação final da tese devidamente corrigida e defendida pela Sra. Thais Trevas Maciel e aprovada pela Comissão Julgadora.

Campinas, 4 de novembro de 1988

C. I. Z. Mammana

Prof. Carlos Ignácio Zamitti Mammana

Orientador

Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação, UNICAMP, como requisito parcial para obtenção do Título **MESTRE** em Matemática.

- novembro/1988 -

Meus sinceros agradecimentos

ao Prof. Carlos Ignácio Zamitti Mammana,

ao Fernando,

à Nilsa,

aos colegas Israel, Akira, José Geraldo, Armando, Ricardo, Marco e Márcio,

e a todos aqueles que me ofereceram seu apoio e depositaram sua confiança na realização deste trabalho.

CAPÍTULO 1

CAPÍTULO 2

CAPÍTULO 4

CAPÍTULO 6

CAPÍTULO 1

INTRODUÇÃO

I. INTRODUÇMO

1. MOTIVAÇMO

A idéia de se criar um ambiente de concepção de circuitos integrados voltado ao ensino surgiu há pouco mais de dois anos, quando foi proposto o curso de " Laboratório de Microeletrônica em Concepção de Circuitos Integrados " ministrado em fevereiro de 1986 na II Escola Brasileira e Argentina de Informática.

A preocupação principal foi de criar condições para que os alunos participassem efetivamente das atividades de projeto de um circuito integrado (CI) desde a sua especificação até o projeto geométrico ("lay-out"), utilizando recursos computacionais relativamente simples e que não exigissem grande experiência dos participantes.

O objetivo básico foi o de contribuir para a disseminação da capacitação em projetos de circuitos integrados, permitindo aos alunos a execução das etapas de projeto de um circuito integrado, simulando um ambiente profissional.

Por se tratar de um ambiente didático, evitou-se a elaboração de ferramentas que manipulassem todas as informações necessárias sem intervenção humana. O que se almejava era justamente a interação aluno/ferramenta, permitindo que o aluno não só participasse das etapas de projeto, mas também estivesse ciente da maneira com que determinados resultados eram obtidos através da ferramenta.

O ambiente escolhido para o sistema de concepção de CI's foi um microcomputador de 16 bits com 512 Kbytes de memória principal e uma unidade de disco tipo "WINCHESTER".

Optou-se por um ambiente de baixo custo e, de um modo geral, existente na maioria das universidades, tornando, assim, o produto acessível a toda comunidade.

Dentro deste contexto, o editor gráfico de que trata este trabalho foi especificado de modo a poder atender as necessidades básicas dos projetistas na tarefa de geração de "lay-out".

A utilização deste editor não está vinculada necessariamente ao sistema didático de projeto que foi mencionado acima.

Trata-se de uma ferramenta que pode ser utilizada isoladamente, sem que isto implique em prejuizo de suas características operacionais.

Como este editor executa a maioria das funções dos editores de categoria mais elevada, sua instalação em sistemas com "hardware" mais potente o torna competitivo nessa aplicação, pois as limitações atuais são fundamentalmente devidas ao "hardware" especificado, devendo-se incluir, por exemplo, uma placa controladora de vídeo com mais recursos , acelerador de "hardware" , etc.

Entretanto, levando em conta o "hardware" disponível para o seu desenvolvimento, este editor gráfico interativo cumpre bem o seu papel. São aproximadamente 35 funções distintas, que foram constantemente testadas e otimizadas.

Para melhor situar a tarefa de edição de "lay-out", torna-se interessante descrever o ciclo de realização de um circuito integrado [02] [07] [12] [24]. Isto será feito de forma sucinta na seção seguinte.

(Ver figura 1).

2. FASES DO PROJETO DE UM CIRCUITO INTEGRADO

2.1 ESPECIFICAÇÃO

Na etapa de especificação, o projetista procura definir os requisitos do sistema a ser desenvolvido como, por exemplo:

- as funções a serem executadas;
- o desempenho dinâmico;
- restrições de tamanho e consumo de energia;
- quantidade para produção;
- metodologia de projeto utilizada;
- partição do sistema.

Nesta fase o projetista define o seu problema.

CICLO DE REALIZACAO DE UM CIRCUITO INTEGRADO

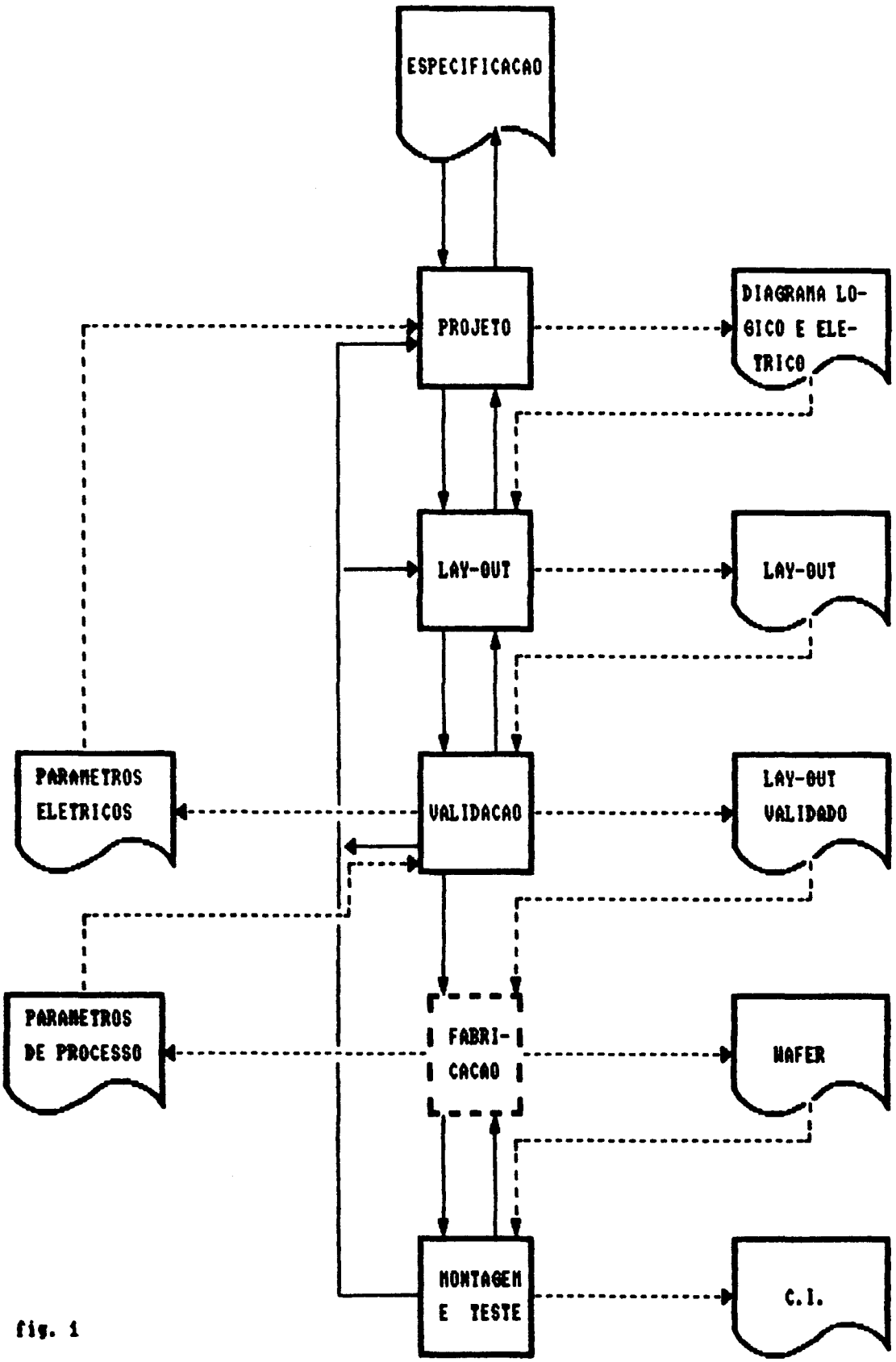


fig. 1

2.2 PROJETO

A etapa de projeto envolve:

a. Projeto e Simulação Funcional

O circuito é planejado a nível de grandes blocos (como registradores, memórias, blocos que realizam funções especiais, etc). São projetados os fluxos de dados entre os blocos e todos os sinais de controle a serem manipulados por estes blocos.

Com a simulação, assegura-se que o sistema se comporta como desejado antes mesmo do projeto detalhado ter início. Simuladores podem detectar falhas no projeto antes do circuito ser fabricado, evitando, assim, um alto custo de reprojeto.

b. Projeto e Simulação Lógica

Consiste no projeto a nível de portas lógicas e blocos lógicos elementares.

Nesta fase ocorre a expansão de cada um dos grandes blocos especificados no projeto funcional.

A simulação lógica nada mais é que exercitar o circuito nas condições em que este vai operar.

A descrição do circuito é dada na forma textual ou gráfica.

c. Projeto e Simulação Elétrica

Todos os blocos lógicos utilizados no projeto lógico são projetados a nível elétrico (a nível de transistor).

Quando um circuito é simulado eletricamente, procura-se, além de verificar a sua funcionalidade, determinar parâmetros elétricos como tempo de subida e de descida, atraso, corrente, etc. Estes parâmetros são determinados em função de variáveis de entrada do simulador que são dependentes da geometria do projeto e do processo utilizado.

2.3 "LAY-OUT"

A atividade de "lay-out" é a definição geométrica do projeto. É a meta final de qualquer sistema de projeto de circuito integrado.

O "lay-out" é formado por um conjunto de polígonos que são agrupados de acordo com o nível de máscara a que eles pertencem. Estes polígonos representam não apenas uma determinada função, mas também agem diretamente na fabricação do circuito integrado. Eles indicam quais as regiões do material semicondutor vão ou não receber o tratamento físico-químico em cada etapa do processo de fabricação.

Existe, então, uma clara ligação entre o "lay-out" e o processo físico de fabricação, a qual exige por parte do projetista a obediência às restrições do fabricante de circuito integrado.

Estas restrições dizem respeito às regras de projeto que mostram ao projetista as limitações do processo de fabricação. Elas podem ser definidas como relações geométricas entre os elementos do "lay-out".

Por isso, ao finalizar a fase de geração de "lay-out", o projetista deve ter todas as possíveis verificações esgotadas, reduzindo ao máximo a possibilidade de reprojeto.

A geração de um "lay-out" pode ser realizada através de ferramentas que levam a sua definição de forma :

- geométrica
- simbólica
- baseada em células
- baseada em procedimentos

Com o auxílio de ferramentas geométricas, o projetista cria as formas das máscaras de um circuito integrado através de comandos de manipulação de figuras rígidas. Em geral, estas ferramentas não levam em conta o significado elétrico destas figuras.

As ferramentas simbólicas também produzem saídas geométricas rígidas ou não, a um nível de hierarquia mais alto. O projetista trabalha com símbolos que passam a representar os dispositivos e suas interconexões.

O desenho é feito de modo simplificado, reduzindo, assim, a complexidade do projeto e contém informação sobre o significado elétrico das figuras.

As ferramentas baseadas em células trabalham com bibliotecas que guardam células previamente projetadas e testadas, realizando, portanto, uma função de acumulação de aprendizado.

A tarefa se resume na alocação conveniente destas células e sua interconexão.

As ferramentas baseadas em procedimentos possibilitam, através de uma linguagem própria, a descrição do "lay-out" sem referenciar tamanhos e posições dos elementos ou distância entre eles.

Esta descrição é feita através de equações ou através de elementos primitivos como, por exemplo, caixa("box"), fio, etc. e suas posições relativas.

Um arquivo de descrição da topologia do "lay-out" em conjunto com um arquivo de dados, que contém regras de projeto e valores das variáveis, geram uma célula. Portanto, o arquivo de descrição representa uma família de células a medida que o arquivo de dados vai sendo modificado.

Sendo o editor gráfico de que trata este trabalho destinado à geração de "lay-outs", trataremos este assunto em detalhe no capítulo II.

2.4 VALIDAÇMO

Na validação, faz-se uma comparação da geometria com o projeto original, ou seja, faz-se a extração do circuito elétrico a partir do "lay-out".

Em seguida são calculados os parâmetros elétricos que dependem do "lay-out" e cujos valores foram pressupostos na fase de simulação.

Nesta fase é feita também a verificação das regras de projeto. Estas regras são do tipo largura e separação mínimas de linhas de metal, largura e comprimento de transistores, etc.

2.5 FABRICAÇÃO

Uma vez validado o projeto, é gerada uma fita magnética que contém as informações geométricas distribuídas nos diferentes níveis de máscara. Tais informações passam por um pós-processamento (transformação de todas as figuras geométricas em pequenos retângulos) e são usadas para gerar as máscaras que serão utilizadas no processo de fabricação do CI.

2.6 MONTAGEM E TESTE

A montagem consiste na separação das pastilhas ("chips") que compõem a lâmina de semicondutor ("wafer") e no encapsulamento de cada uma destas pastilhas.

A aplicação de testes visa a verificação do processo de encapsulamento e a verificação da funcionalidade das pastilhas.

3. PLANO DE TESE

No primeiro capítulo é feita uma introdução ao trabalho desenvolvido situando-o dentro do ciclo de projeto de circuitos integrados.

O segundo capítulo descreve quatro editores gráficos disponíveis em universidades e também apresenta com um pouco mais de detalhes o editor gráfico de que trata este trabalho.

O terceiro capítulo contém a especificação dos requisitos de "software". A estrutura do programa é dada juntamente com a descrição detalhada de todas as funções nele contidas.

O quarto capítulo contém a especificação do projeto de "software". São apresentados os diagramas de fluxo de dados do programa, a estrutura de dados utilizada e a descrição dos principais procedimentos desenvolvidos e utilizados.

O quinto capítulo apresenta os resultados do teste de campo feito nos cursos das II e III EBAI.

O sexto capítulo apresenta comparações entre os editores gráficos descritos no capítulo II e o editor gráfico desenvolvido. Algumas sugestões de otimização do EGIPCIS também são fornecidas.

CAPÍTULO 2

EDITORES GRÁFICOS

II. EDITORES GRAFICOS

1. INTRODUÇÃO

O editor gráfico, classificado como ferramenta geométrica, é basicamente um sistema de desenho.

O projetista se utiliza de comandos do editor para criar, modificar e manipular os polígonos que compõem os níveis de máscaras.

O "lay-out" é feito diretamente na tela do terminal, geralmente partindo de um esboço manual.

Em alguns editores, existe a possibilidade de composição final do "lay-out" para gerar circuitos mais complexos. Isto é feito através das chamadas de células previamente projetadas a serem alocadas convenientemente.

Encerrada a tarefa de edição, o editor fornece um arquivo que descreve a geometria gerada em um formato adequado.

Atualmente, com a melhoria no desempenho dos microcomputadores, os editores gráficos deixaram de ser exclusividade dos grandes computadores com terminais gráficos ou de estações de projeto com "hardware" dedicado.

A nível profissional, as tendências estão voltadas para estas estações de trabalho com processadores dedicados, memória local, terminais gráficos de alta resolução e dispositivos gráficos de entrada ("tablet", "mouse", ...), como são os casos das estações MENTOR, SUN e CALMA, todas, porém, de custo muito elevado.

Embora existam algumas limitações quanto à velocidade de exibição e precisão de tela, limitações estas devidas somente ao "hardware" especificado, é perfeitamente viável a utilização de microcomputadores na realização de tarefas de auxílio ao projeto de circuito integrado como, por exemplo, edição gráfica, entrada esquemática, simulação, etc.

Dentre os diversos editores gráficos disponíveis em universidades, serão destacados os quatro mais utilizados.

Dois deles, o TEDMOS e o MICROEDITOR [20] [05], foram desenvolvidos para microcomputador de 16 bits e os outros dois foram desenvolvidos para VAX, o KIC [01] operando nos sistemas VMS e UNIX e o CAESAR [16] em sistema UNIX.

Os quatro itens seguintes apresentam cada um destes editores e o item 6 apresenta o EGIPCIS, Editor Gráfico Interativo para projeto de circuitos integrados de que trata este trabalho.

2. TEDMOS - TURBO EDITOR PARA CIRCUITOS INTEGRADOS MOS

O sistema TEDMOS II foi desenvolvido no Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro.

Ele contém ferramentas básicas que permitem não só a edição gráfica como também a verificação de regras de projeto e a simulação lógica.

Foi projetado para ser utilizado em ambiente acadêmico, com o intuito de contribuir para o ensino e aprendizado em projeto de circuitos integrados.

O TEDMOS foi projetado para a seguinte configuração de "hardware" :

- microcomputador de 16 bits, compatível com IBM-PC
- memória principal mínima de 512 Kbytes
- no mínimo 2 acionadores de disquete de 320 Kb ou disco tipo "WINCHESTER"
- monitor de vídeo monocromático com resolução de 640 X 200 pontos ou CGA 320 X 200 X 4 cores
- impressora (opcional) - compatível com padrão EPSON MX800 ou MX100

O editor gráfico do sistema TEDMOS trabalha sobre uma matriz que representa o "lay-out". Nesta matriz, cada elemento representa uma área de 1 X 1 lambda, onde lambda é definida como sendo uma unidade mínima de comprimento.

O tamanho da matriz depende da memória disponível, sendo que para memória de 512 Kbytes, o tamanho máximo que pode ser representado é 320 X 320 lambda. Em configurações de 640 Kbytes, é possível representar espaços de até 800 X 600 lambda.

O valor de lambda em micra é dado pelo usuário.

O TEDMOS é, então, voltado para edição de células.Ele não foi projetado para realizar a montagem da pastilha, já que esta tarefa iria requerer um tamanho de matriz muitas vezes maior.

O TEDMOS trabalha limitado até 8 niveis de máscara. Os códigos previamente estabelecidos pelo programa são :

D - difusão
P - polisilício
M - metal
C - contato
I - implante de campo

Com relação ao cursor, este se apresenta na forma de retângulo e seu conteúdo indica o trecho do "lay-out" sobre o qual serão feitas as operações de edição. O tamanho do cursor (retângulo) pode ser modificado até que se consiga envolver a parte do "lay-out" desejada.

Quanto à disponibilidade de cor existente na sua última versão, esta facilita a visualização e distinção dos níveis.

Para usuários que não dispõem de monitor de vídeo colorido, a opção continua sendo o preenchimento dos retângulos através de padrões de hachura.

O TEDMOS gera descrição do "lay-out" em formato CIF.

3. MICROEDITOR - EDITOR GRAFICO PARA MICROELETRONICA

O MICROEDITOR V2.3 (ME) foi desenvolvido pelo grupo de microeletrônica da pós-graduação em Ciências da Computação da Universidade Federal do Rio Grande do Sul.

É uma ferramenta que permite a edição interativa de células de circuitos integrados nas tecnologias NMOS e CMOS.

O ME foi projetado para a seguinte configuração de "hardware":

- microcomputador de 16 bits, compatível com IBM-PC
- memória principal de 512 Kbytes
- 1 disco flexível ou disco tipo "WINCHESTER"
- monitor de vídeo monocromático de 640 X 200 pontos
- "tablet" (opcional)
- "plotter" (opcional)

Os polígonos são descritos através de retângulos. Os códigos dos níveis são pré-estabelecidos pelo programa e não permitem alteração.

O ME trabalha com 7 níveis para tecnologia NMOS e 9 níveis para tecnologia CMOS. São eles :

NMOS :

P - polisilício
M - metal
D - difusão
B - pré-contato
I - implante
C - contato
G - camada para isolação ("overglass")

CMOS :

P - polisilício
M - metal 1
D - difusão N
B - difusão P
H - metal 2
V - via
S - poço
C - contato
G - camada para isolação ("overglass")

Como dispositivo de entrada é utilizado o teclado ou um "tablet" e como dispositivos de saída a tela monocromática e um "plotter" de 8 canetas.

O cursor se apresenta na forma de um ' X ' e um "menu" contendo todos os comandos disponíveis no editor é exibido à direita da tela.

Assim como o TEDMOS, o MICROEDITOR não é um editor hierárquico, ou seja, não consegue realizar a montagem final de um circuito integrado.

O MICROEDITOR gera a descrição do "lay-out" em formato semelhante ao CIF.

4. CAESAR - " VLSI CIRCUIT EDITOR "

O editor CAESAR foi desenvolvido na Divisão de Ciências da Computação da Engenharia Elétrica da Universidade da Califórnia.

Ele é um sistema interativo para criar e modificar projetos de circuitos integrados, nas tecnologias NMOS e CMOS.

Assim como os editores descritos anteriormente, o CAESAR não é um sistema que tem conhecimento das regras de projeto, propriedades elétricas ou mesmo conectividade. Ele é apenas um editor geométrico que permite o desenho dos níveis de máscaras de um circuito e que, diferenciando-se dos editores TEDMOS e MICROEDITOR, permite a hierarquização de células, o que possibilita a realização da montagem completa de uma pastilha.

O CAESAR opera em VAX UNIX e é configurado para terminais das linhas AED e OMEGA.

A entrada de dados é feita através do "tablet" (ou então do "joystick") e teclado.

Neste editor, o universo de trabalho é uma grade de 1 X 1 lambda. Tanto o movimento do cursor como medidas de distância são dados em números inteiros de lambda.

O valor de lambda em micra é dado pelo usuário.

O CAESAR trabalha com 7 níveis de máscara para tecnologia NMOS e 7 níveis para tecnologia CMOS. Os códigos que representam os níveis são previamente estabelecidos pelo programa e não permitem alteração. São eles :

NMOS :

p ou r - polisilício
d ou g - difusão
m - metal
i ou y - implante
b - "buried contact"
c - contato
o - camada de isolação

CMOS :

p ou r - polisilício
d ou g - difusão
m - metal
p ou y - implante P+
c - contato
w - poço P-well
o - camada de isolação

O editor utiliza um cursor na forma de retângulo, cujo tamanho pode ser modificado até que este consiga envolver a parte do "lay-out" a sofrer alguma alteração.

O CAESAR possui uma interface com o programa LYRA, que verifica se existe, no "lay-out" gerado, alguma violação nas regras de projetos.

O editor CAESAR gera a descrição do "lay-out" em formato CIF.

5. KIC - " A GRAPHICS EDITOR FOR INTEGRATED CIRCUITS "

O editor KIC foi desenvolvido na Universidade da California.

É um sistema interativo para criar e modificar projetos de circuitos integrados, nas tecnologias NMOS e CMOS.

Ele opera em VAX/VMS ou VAX/UNIX e é configurado para diversos terminais das linhas TEKTRONIX, AED, OMEGA e JUPITER.

A unidade de distância utilizada é o lambda e o "lay-out" gerado pode ser de até 200000 X 200000 lambda.

A relação entre lambda e micra é dada pelo usuário.

Os níveis de máscara são descritos através de retângulos, polígonos, fios, círculos e arcos.

O KIC trabalha com 25 níveis de máscara e se este número não for suficiente, existe a possibilidade de recompilação do programa fonte para aumentar o número máximo de níveis.

Existem, neste editor, os chamados atributos dos níveis que são : nome, tipo de preenchimento, cor, prioridade, visibilidade, contorno e "blinking".

- Nome(código), tipo de preenchimento e cor associados a qualquer nível podem sofrer alteração.
- O atributo de prioridade coloca os níveis na ordem de sobreposição conveniente ao projetista.
- A visibilidade permite a seleção dos níveis a serem visualizados na tela.
- Quando as figuras de um determinado nível são preenchidas por algum padrão, existe a possibilidade de se retirar o seu contorno.

- E por fim o "blinking" (pisca-pisca), que é uma maneira de ressaltar o nível de máscara.

Os atributos descritos acima podem ser modificados a qualquer momento da edição.

O arquivo de níveis fornecido pelo programa contém, inicialmente, os seguintes códigos :

NI - implante
ND - difusão
NP - polisilício
NM - metal
NC - contato
NG - camada de isolação

Fica a critério do usuário a modificação deste arquivo.

Para facilitar a tarefa do usuário na geração do "lay-out",o KIC permite que a tela seja dividida em duas partes: a primeira contendo uma visão geral do "lay-out" e a segunda, detalhada e precisa, contendo apenas um trecho do "lay-out".

Qualquer modificação realizada neste trecho é visualizada nas duas partes da tela. Isto permite que o projetista esteja sempre a par do "lay-out" como um todo e que não se perca durante a edição de um circuito mais complexo.

O KIC permite hierarquia de células.

A descrição do "lay-out" é dada em um formato próprio (também chamado KIC) e sua conversão para o formato CIF é efetuada através do programa KICTOCIF, que faz parte do editor.

6. EGIPCIS - EDITOR GRAFICO INTERATIVO PARA PROJETOS DE CI's

O sistema EGIPCIS foi desenvolvido no Instituto de Microeletrônica do Centro Tecnológico para Informática.

É uma ferramenta que permite a edição interativa de células, a montagem interativa da pastilha e ainda a extração manual de parâmetros específicos para transistores MOS.

O EGIPCIS foi projetado para a seguinte configuração de "hardware" :

- 512 Kbytes de memória principal;
- 1 disco flexível;
- 1 monitor de vídeo com resolução de 640 X 200 pontos;
- 1 unidade de disco tipo WINCHESTER.

A unidade de distância utilizada é o micron e o tamanho do "lay-out" gerado depende somente da memória disponível.

Os níveis de máscara são descritos através de retângulos e polígonos.

O EGIPCIS trabalha com até 14 níveis, cujos códigos são previamente estabelecidos pelo programa. Todavia, a alteração destes códigos é permitida, uma vez que estes se encontram em um arquivo a parte, à disposição do usuário.

O arquivo de níveis fornecido pelo programa contém, inicialmente, os seguintes códigos :

CC - contato
CP - polisilício
CPI - implante P+
CPF - implante de campo
CNW - poço N
CPW - poço P
CA - nitreto
CNI - implante N+
CC2 - via
CM2 - metal 2
CG - pad 1
CG2 - pad 2
CP2 - polisilício 2

O atributo de visibilidade referente aos níveis de máscara pode ser acionado a qualquer momento durante a execução do programa.

Com relação ao cursor, este se apresenta na forma de cruz e seu movimento é dado nas direções x e y e também nas diagonais.

A descrição do "lay-out" é dada em formato CIF.

CAPÍTULO 3

ESPECIFICAÇÃO DOS REQUISITOS DE "SOFTWARE"

III. ESPECIFICAÇÃO DOS REQUISITOS DE "SOFTWARE"

1. INTRODUÇÃO

1.1 PROPÓSITO

Este capítulo tem como propósito a elaboração da primeira fase de desenvolvimento [18] [21], ou seja, a especificação dos requisitos da ferramenta computacional de auxílio ao projetista de circuitos integrados na fase de geração de "lay-out" de que trata este trabalho.

São apresentados o escopo do trabalho, o ambiente de implementação, características do usuário, dependências, interface com usuário, requisitos específicos e restrições gerais.

1.2 ESCOPO

Sistema EGIPCIS (Editor Gráfico Interativo para Projetos de Circuitos Integrados) proporciona ao usuário, através de primitivas gráficas, a facilidade de gerar, de modo interativo, o "lay-out" de um circuito integrado, independentemente da metodologia de projeto utilizada, desde que esta inclua interação do projetista.

Uma característica deste editor é o fato de ele dever operar em microcomputador de 16 bits, não só para diminuir a tarefa do computador central, mas também por tornar o produto acessível às universidades.

O EGIPCIS deve poder operar dentro do Sistema Didático de Projetos de Circuitos Integrados, o SDP. Porém, sua utilização não está vinculada necessariamente a este sistema. Trata-se de uma ferramenta que pode ser utilizada isoladamente, sem perda de suas características operacionais.

1.3 AMBIENTE DE TRABALHO

O requisito mínimo de "hardware" necessário à execução do EGIPCIS é um microcomputador de 16 bits incluindo:

. 512 Kbytes de memória principal;
. 1 unidade de disco flexível;
. 1 monitor de vídeo com resolução de 640x200 pontos.

Para projetos de maior complexidade, é indispensável a utilização de uma unidade de disco tipo "WINCHESTER".

Requisito mínimo de "software": sistema operacional MS-DOS.

1.4 ABREVIAÇÕES

CIF - " Caltech Intermediate Form ", formato de descrição da geometria do projeto ("lay-out").

MOSFET - "MOS Field Effect Transistor".

2. DESCRIÇÃO GERAL

2.1 UTILIZAÇÃO DO PROGRAMA NO CONTEXTO DE PROJETO DE CI's

O projeto de circuito integrado pode ser dividido em três grandes blocos :

. Sistema de simulação

. Sistema de geração de "lay-out"

. Pós-processamento

O editor gráfico EGIPCIS faz parte do sistema de geração de "lay-out", sistema este composto de:

. editor gráfico, para criar o "lay-out";

. DRC, para checar as regras de projeto;

. extrator de circuitos, para extrair o circuito equivalente ao "lay-out".

O esquema a seguir dá uma visão geral das diferentes fases do projeto de circuito integrado :

DIFERENTES FASES DO PROJETO DE CIRCUITOS INTEGRADOS

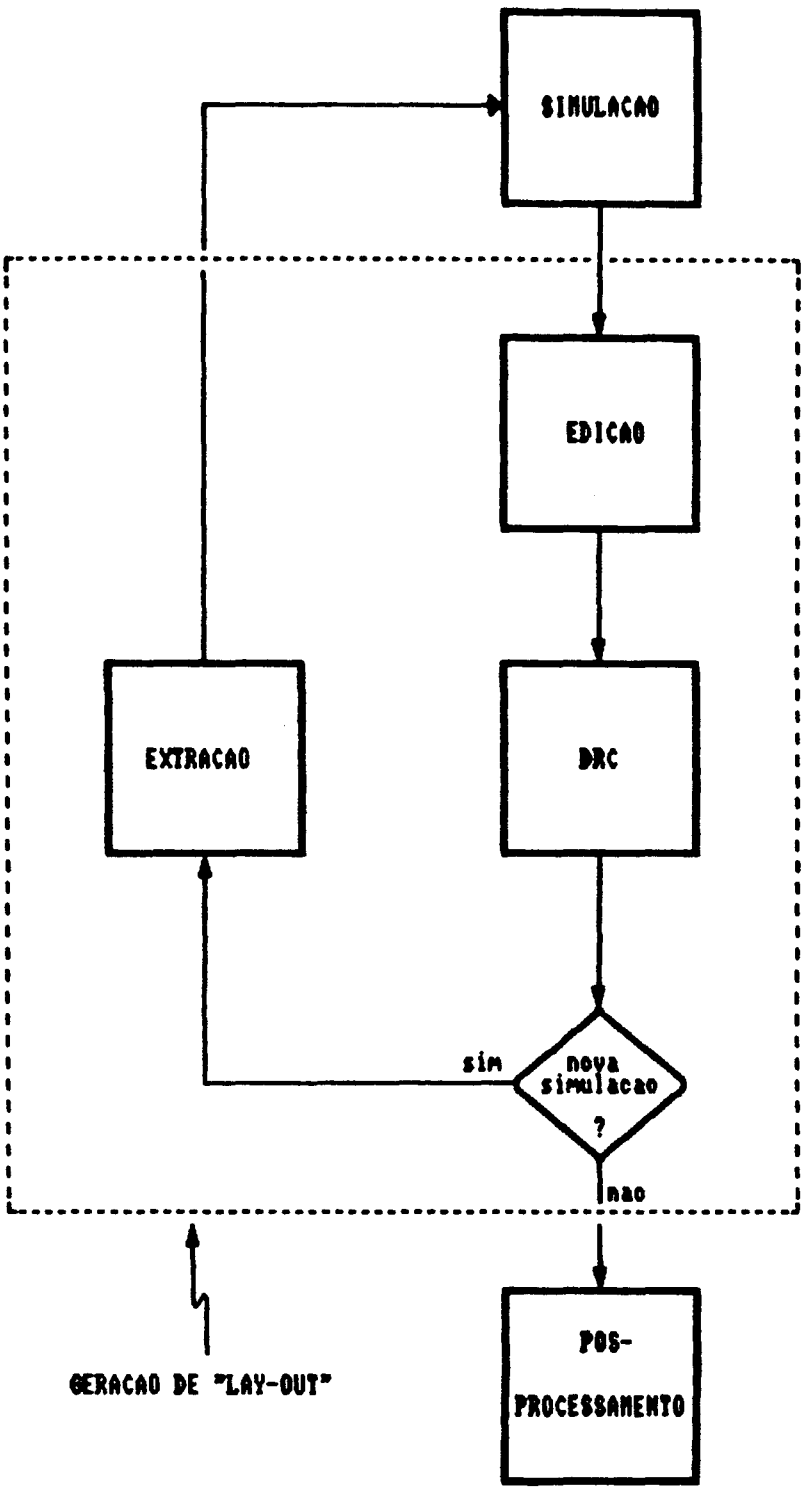


fig. 1

2.2 FUNÇÕES DO PROGRAMA

O editor gráfico EGIPCIS tem como funções básicas :

- edição interativa do "lay-out" de células que serão incorporadas a uma biblioteca;
- montagem interativa da pastilha através da alocação de células em posições determinadas pelo projetista;
- extração manual de parâmetros.

2.3 CARACTERÍSTICAS DO USUARIO

Podemos definir duas classes de usuários para este editor:

- projetistas de circuitos integrados que utilizam a ferramenta mais intensamente;
- alunos de pós-graduação das universidades, que utilizam a ferramenta a nível de aprendizado.

2.4 DEPENDÊNCIAS

O EGIPCIS foi escrito em Pascal Turbo versão 3.0 e sem a utilização de qualquer pacote gráfico. Todas as suas rotinas gráficas foram desenvolvidas utilizando-se de procedimentos primários embutidos na linguagem (PLOT: desenhar ponto; DRAW: desenhar linha) e de recursos de acesso direto à memória para abertura de janelas e limpeza de tela.

2.5 INTERFACE COM O USUARIO

A interface com o usuário é realizada através de:

- teclado;
- tela gráfica;
- tela alfanumérica;
- impressora;
- sistema de "menu".

A movimentação do cursor e a escolha de comandos são feitas através do teclado.

O cursor em forma de cruz, move-se nas direções x e y (horizontal e verticalmente) e também nas duas direções intermediárias a estas (diagonalmente), através das teclas situadas à direita do teclado.

São elas:

TECLAS	FUNÇÃO
↑	o cursor avança um passo para cima
↓	o cursor avança um passo para baixo
—›	o cursor avança um passo para a direita
‹—	o cursor avança um passo para a esquerda
HOME	o cursor avança um passo para cima e um para a esquerda (diagonal)
END	o cursor avança um passo para baixo e um para a esquerda (diagonal)
PG UP	o cursor avança um passo para cima e um para a direita (diagonal)
PG DN	o cursor avança um passo para baixo e um para a direita (diagonal).

O passo do cursor é dado de acordo com a mínima dimensão do "lay-out" e pode ser aumentado ou diminuido através das teclas "+" e "-". Isto se torna interessante quando grandes deslocamentos devem ser efetuados.

Os comandos disponíveis para o editor são controlados por "menus" auto-explicativos. Para cada comando, uma letra, assinalada em maiúsculo no "menu", é suficiente para efetuar a sua chamada.

Alguns comandos mais gerais são requisitados através das teclas de função F1, F2, ... F10 situadas à esquerda do teclado (Ver item 3.1.16).

Nas duas figuras que se seguem são mostradas a hierarquia dos "menus" que fazem parte do programa e a hierarquia dos comandos existentes nestes "menus", respectivamente.

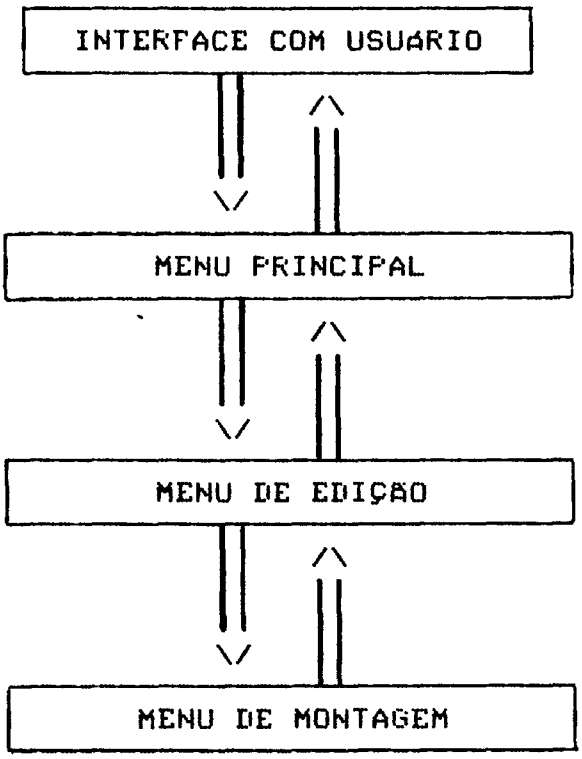


fig.2 - Hierarquia de "menus"

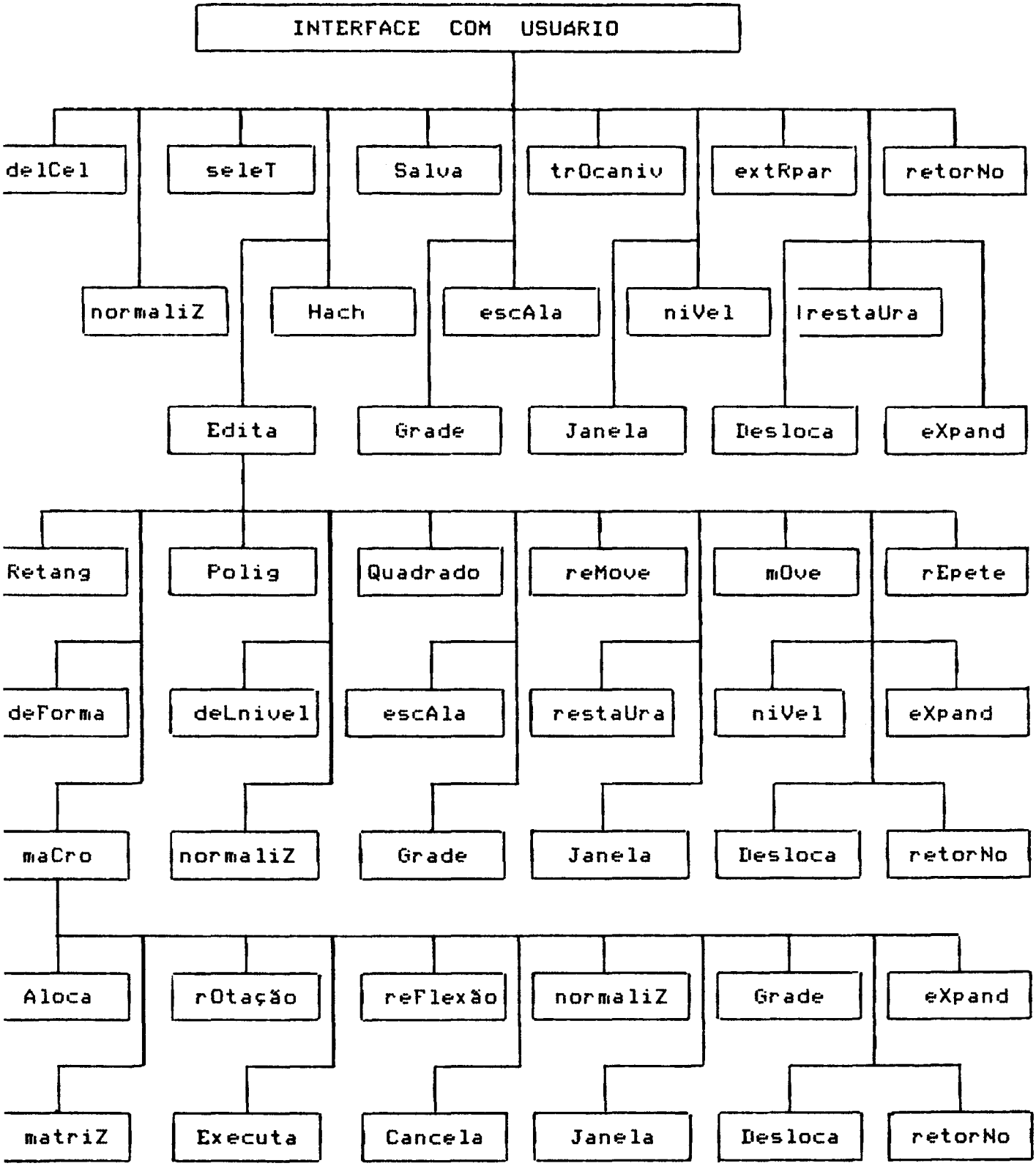


fig. 3 - Hierarquia dos comandos existentes nos "menus"

3. REQUISITOS ESPECIFICOS

3.1 REQUISITOS FUNCIONAIS

O EGIPCIS pode criar um "lay-out" novo ou modificar um já existente. Ele está preparado para receber arquivos contendo "lay-outs" descritos em formato CIF versão 2.0, um formato padrão internacional apresentado no item 3.1.9.

As descrições CIF geradas por outros programas ou mesmo aquelas geradas manualmente, devem estar de acordo com a versão 2.0, caso contrário o programa abortará ou até mesmo lerá o arquivo, mas talvez interpretando de forma não correta.

Depois de especificado o nome do "lay-out" a ser editado e a mínima dimensão para este "lay-out" (valor "default" = 0.5 micron), o EGIPCIS mostra sua tela, que é dividida em três regiões (ver figura 4) :

- Área de status e diálogo, na parte superior.
  Área reservada para (1) indicar posição do cursor em micra, (2) requisitar dados, (3) fornecer mensagens de advertência, (4) informar nome do projeto corrente.
- Área de trabalho, que chamaremos de janela de visualização, ao centro.
- Área de "menu", na parte inferior.
  Área reservada à exibição dos "menus" auto-explicativos, fornecidos pelo programa.

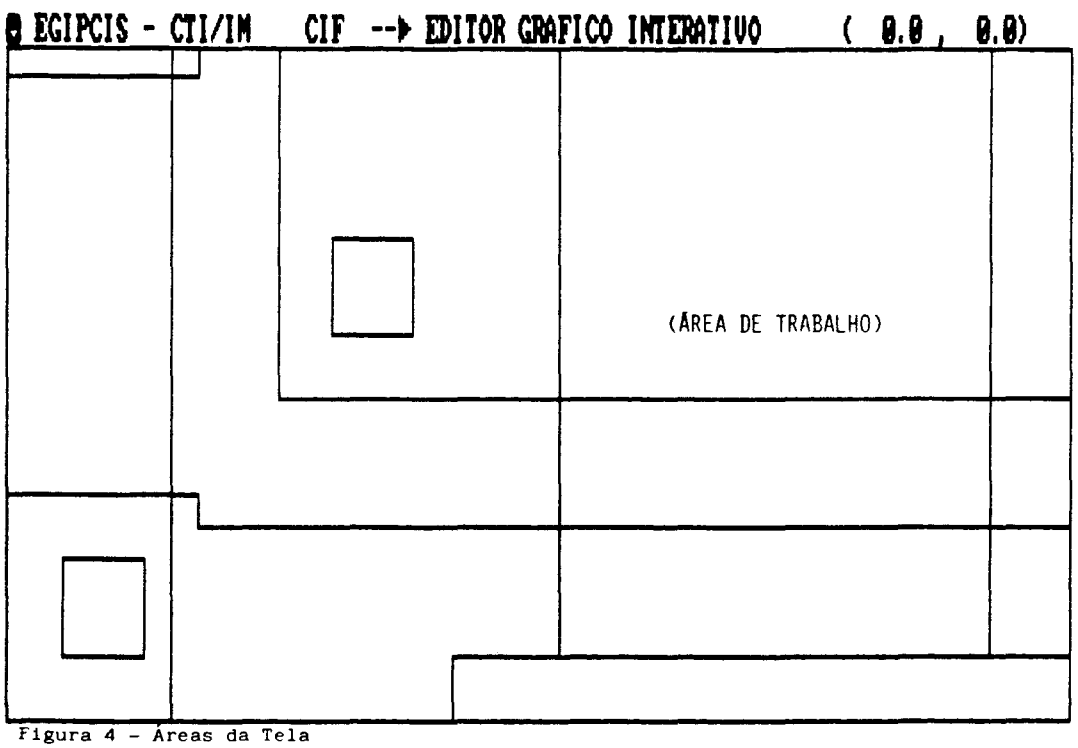


Figura 4 - Áreas da Tela

O primeiro "menu" apresentado é o "menu" principal, mostrado abaixo:

niVel Edita delCel Grade NormaliZ eXpand trOcaniv retorNo

esCala Janela seleT Salva restaUra Desloca extRpar Hach

Uma descrição de todos os comandos disponíveis neste editor para criação, modificação e manipulação de um "lay-out" é dada a seguir.

3.1.1 NIVEL

Para editar um "lay-out", deve ter sido feita previamente a seleção do nível de máscara no qual se vai trabalhar. Isto é feito uma vez que os níveis são criados separadamente (ver figura 5) .

Cada nível de máscara corresponde a um código definido, a princípio, pelo programa, mas que pode ser alterado pelo usuário.

Esta alteração é possível já que todos os níveis de máscara são apresentados em um arquivo à parte, chamado "niveis.eg". Este arquivo pode ser manipulado pelo usuário, permitindo que o programa se adapte a qualquer tecnologia.

Futuramente, para sistemas que possuam um vídeo colorido, os dados relacionados com cor também serão especificados neste arquivo, ou seja, a cada nível de máscara será atribuida uma cor correspondente, conforme a conveniência do usuário.

Uma vez selecionado o nível de trabalho, seu código passará a constar da linha de status.

Uma tabela com o nome e respectivo código de cada nível de máscara é mostrada a seguir. Estes níveis apresentados fazem parte do arquivo "default" que acompanha o programa.

NíVEL DE MÁSCARA	CóDIGO
contato	CC
polisilício 1	CP
implante P+	CPI
implante de campo	CPF
poço N	CNW
poço P	CPW
nitreto	CA
implante N+	CNI
via	CC2
metal 1	CM
metal 2	CM2
pad 1	CG
pad 2	CG2
polisilício 2	CP2

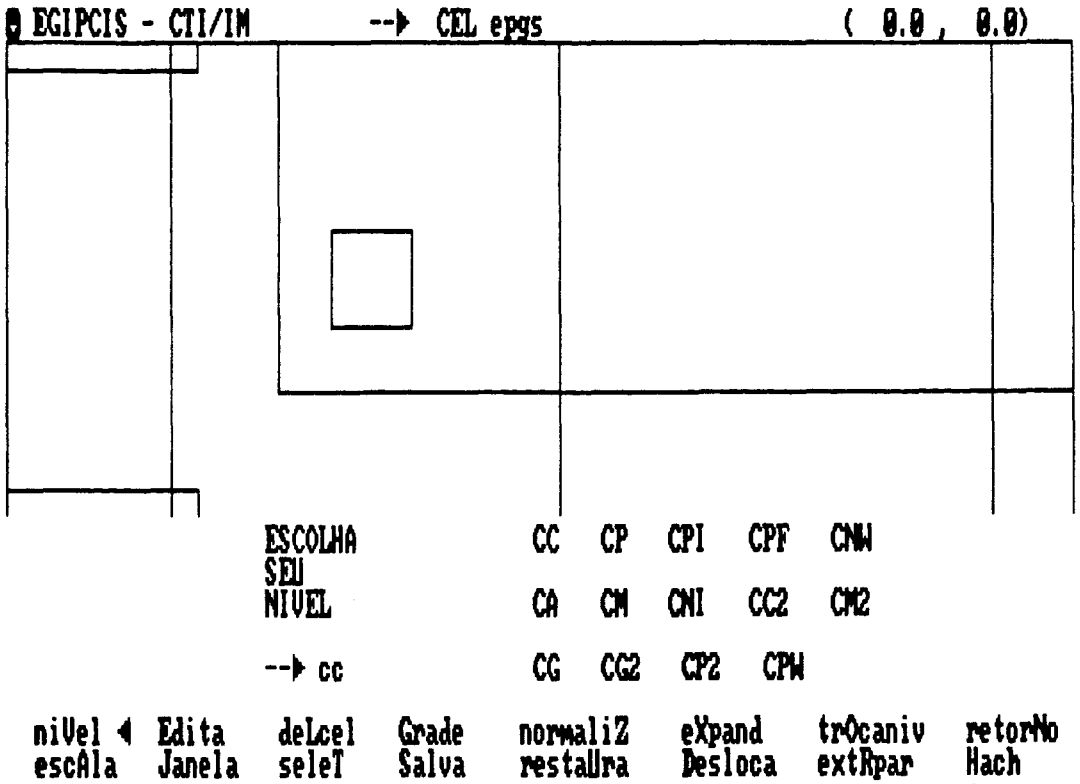


fig. 5 - Tabela de niveis

3.1.2 ESCALA

Transformação de escalamento [03] [09] [15] é aplicada a um conjunto de pontos com o objetivo de ampliar, reduzir ou mesmo distorcer uma figura. O ponto (x, y) da figura tem suas coordenadas escaladas ao longo do eixo x pelo fator de escala Sx e ao longo do eixo y pela fator de escala Sy, produzindo assim um novo ponto (x', y') tal que:

$$[\, x' \;\; y' \;\; 1 \,] = [\, x \;\; y \;\; 1 \,] \cdot \begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

O efeito da distorção é obtido quando os fatores de escala Sx e Sy tem valores diferentes. No EGIPCIS, o escalamento é feito somente com o objetivo de ampliação e redução de figuras. Para isto, foram atribuídos valores iguais a Sx e Sy, não afetando assim as proporções da figura.

As ampliações e reduções são sempre feitas em múltiplos de 2 e podem ser requeridas sucessivamente até atingir a escala desejada.

Vale ressaltar que, quanto maior a complexidade do "lay-out", maior o tempo requerido para a exibição deste, no caso de uma ampliação e que também as reduções excessivas podem tornar impraticável sua edição devido à baixa resolução de tela.

Dependendo do fator de escala escolhido, o trabalho de edição em cima de um "lay-out" reduzido é perfeitamente viável.

Uma última consideração a respeito deste comando é o fato da posição onde se encontra o cursor ser transladada automaticamente para o canto inferior esquerdo da janela de visualização quando ocorrerem ampliações e reduções, fornecendo, assim, um referencial para o posicionamento do cursor.

Ver figura 6.

3.1.3 HACH

Permite que os polígonos que compõem os níveis de máscara sejam preenchidos com padrões pré-estabelecidos pelo programa.

O mesmo comando HACH cancela o preenchimento.

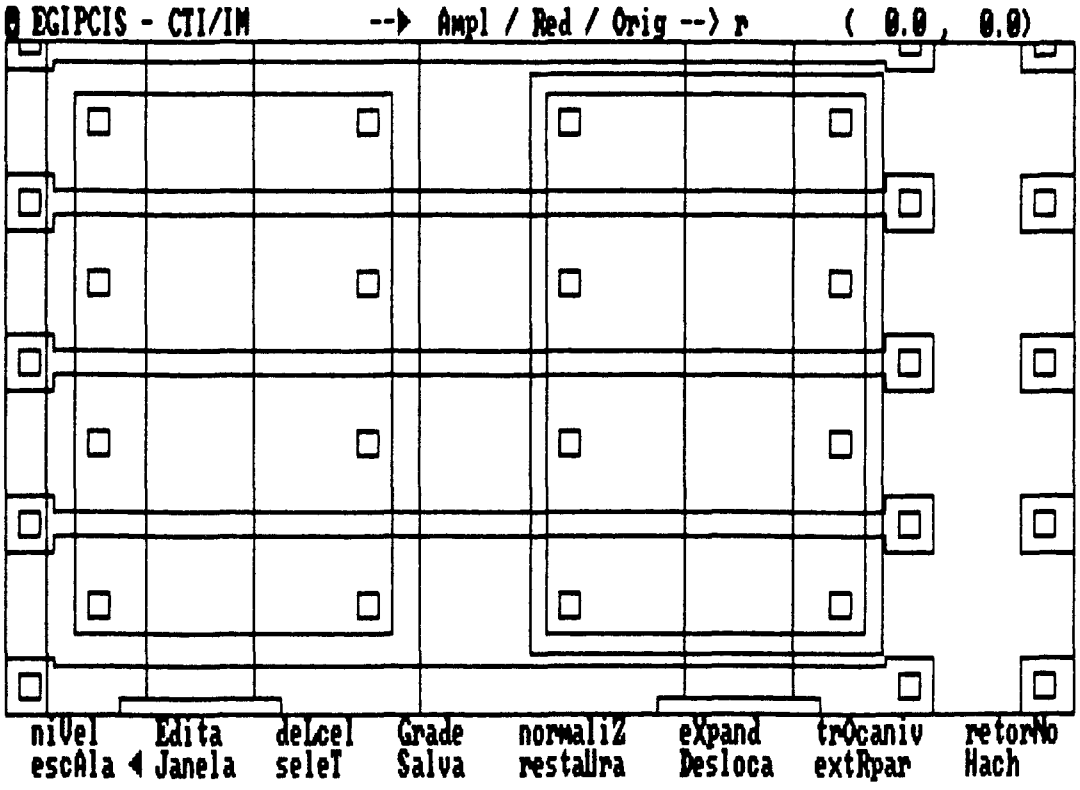


fig. 6 - Redução do "lay-out"

3.1.4 DESLOCA

Transformação de deslocamento [03] [09] [15] é aplicada a um conjunto de pontos com o objetivo de movê-los de uma posição a outra.

O ponto (x, y) de uma figura é transladado para uma nova posição através da adição de Tx unidades ao longo do eixo x e de Ty unidades ao longo do eixo y , produzindo assim um novo ponto (x', y') tal que:

$$[x'\ y'\ 1] = [x\ y\ 1] \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ Tx & Ty & 1 \end{vmatrix}$$

A janela de visualização representa apenas uma parte do "lay-out" inteiro. No EGIPCIS, o usuário tem a possibilidade de visualizar partes de interesse do "lay-out", através de deslocamentos que podem ser efetuados nas direções x, y e nas diagonais (45 graus). Define-se por unidade de tela a própria janela de visualização.

O deslocamento é feito por unidades de tela e caso o usuário não forneça o número de unidades a deslocar por vez, é utilizado o valor 1 como "default".

As teclas utilizadas para requerer o deslocamento são as mesmas utilizadas na movimentação do cursor.

3.1.5 JANELA

Esta é uma outra opção para visualização de partes do "lay-out". Este comando leva vantagem em relação ao DESLOCA, uma vez que a posição fornecida pelo usuário é prontamente alocada na origem da janela de visualização ao invés da ocorrência de deslocamentos sucessivos até se alcançar a parte do "lay-out" desejada.

3.1.6 NORMALIZ

Permite que o "lay-out" seja exibido totalmente na janela de visualização (ver figura 7).

Não é possível a edição da figura normalizada; o objetivo é de apresentar somente uma visão global do projeto.

Executando o comando mais uma vez, a normalização é retirada, dando lugar novamente ao "lay-out" em suas dimensões reais.

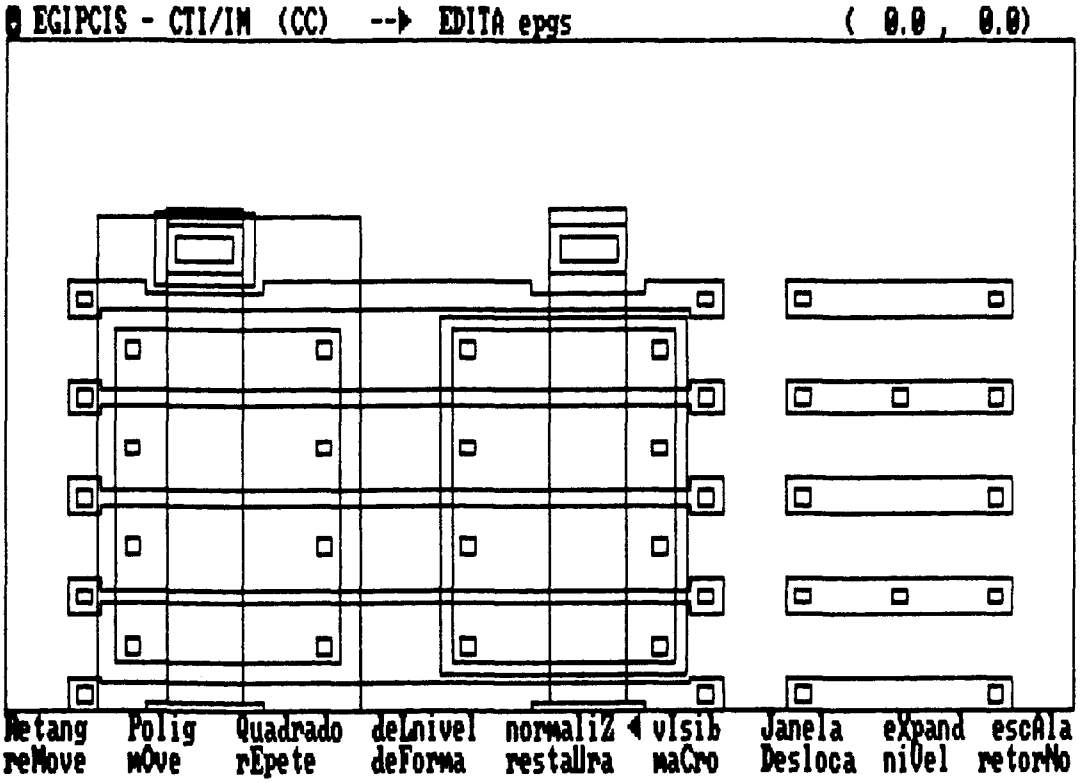


fig. 7 - Normalização do "lay-out"

3.1.7 SELET

Visualização seletiva dos níveis de "lay-out".

É exibido somente o desenho correspondente ao nível requisitado. Para visualização de mais de um nível de "lay-out", utiliza-se a visualização acumulada, onde cada nível selecionado passa a ser sobreposto aos anteriores (ver figuras 8 e 9).

Dentro desta opção é possível, ainda, a exibição de grade e a normalização do "lay-out" em relação à tela.

A edição não é permitida dentro desta opção.

3.1.8 DELCEL

Mediante confirmação do usuário, todos os níveis de "lay-out" são apagados da janela de visualização e da estrutura de dados.

O arquivo aberto para conter a descrição daquele "lay-out" continua a existir, porém vazio.

3.1.9 SALVA

O projeto que está sendo editado é salvo no arquivo especificado inicialmente sem, contudo, encerrar a edição.

Como foi dito anteriormente, o formato utilizado para descrever a geometria do projeto é o formato CIF.

O arquivo CIF é composto de uma sequência de caracteres. Este arquivo contém uma lista de comandos seguidos por um ponto final. Os comandos são separados por ponto e vírgula.

A tabela a seguir descreve os comandos aceitos pelo formato CIF e a maneira pela qual são apresentados dentro do arquivo.

COMANDO	FORMA
Polygon	P v1 v2 ... vn
Box	B comp larg centro direção
Round flash	R diâmetro centro
Wire	W larg v1 v2 ... vn
Layer	L nome
Start Symbol Definition	DS índice escala
Finish Symbol Definition	DF
Delete Symbol Definition	DD índice
Call Symbol	C índice transformação
End	E
Comentários	dígito + texto

3.1.10 GRADE

Desenha ou apaga a grade de referência. A sua dimensão original é especificada na ativação do editor gráfico EGIPCIS, (a dimensão da grade é igual à mínima dimensão do "lay-out"), sendo ampliada ou reduzida nas mesmas proporções do desenho quando utilizada a opção ESCALA.

A dimensão da grade pode ser alterada de acordo com a necessidade do usuário, utilizando-se as teclas ")" e "(" (maior e menor). Através da tecla ")", a grade é ampliada de duas vezes, ou seja, será retraçado no vídeo uma grade com duas vezes menos pontos do que a anterior. O mesmo conceito é válido inversamente para a tecla "(".

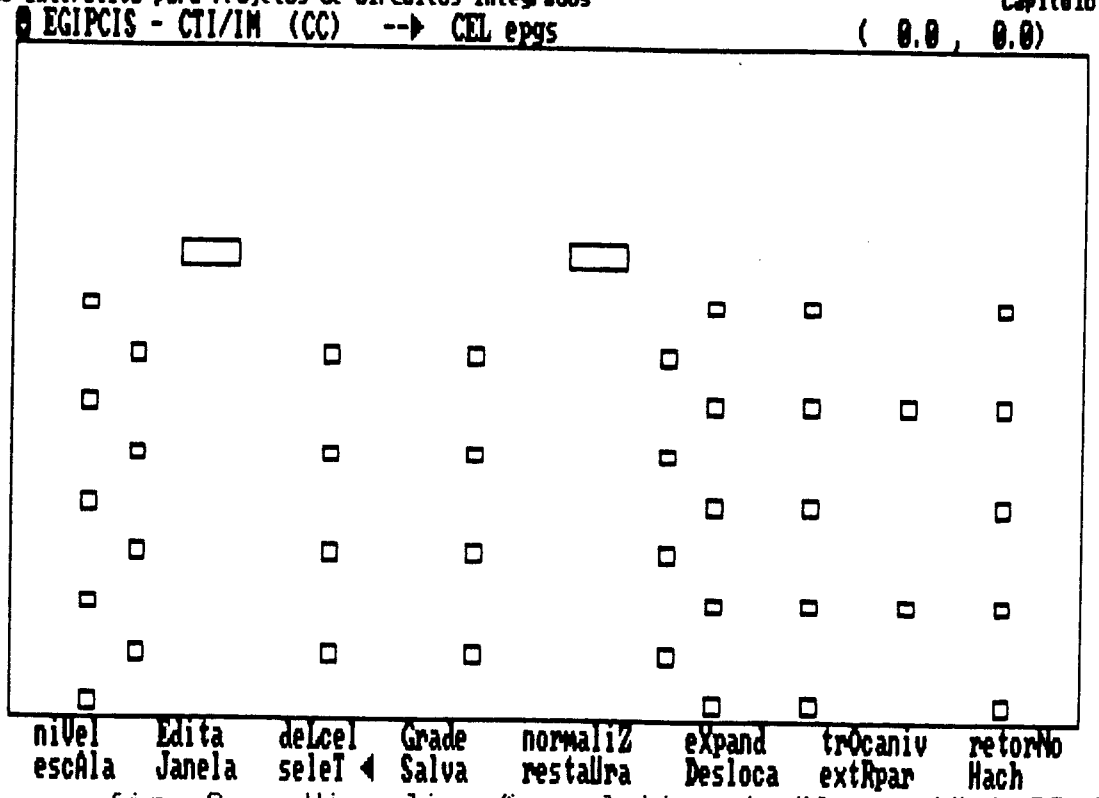


fig. 8 - Visualização seletiva do "lay-out" (CC)

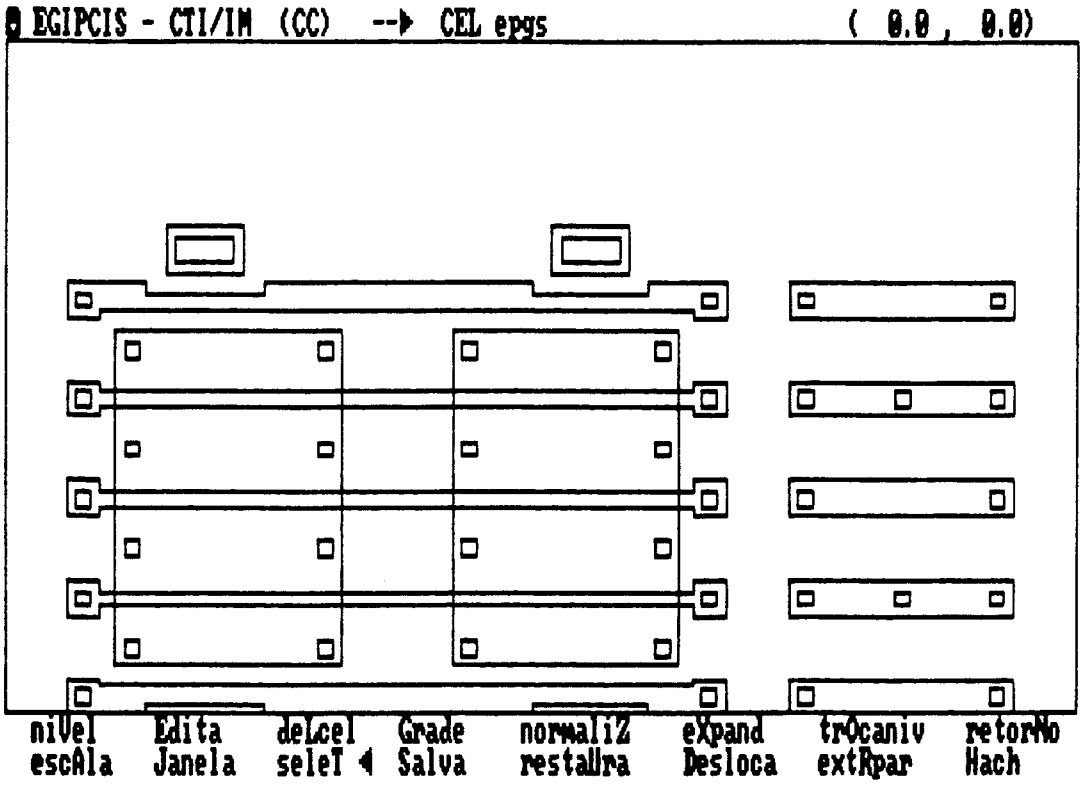


fig. 9 - Visualização acumulada (CC + CP + CA)

3.1.11 RESTAURA

Pelo fato de estarmos trabalhando com um único plano de tela, na remoção de uma figura, a área existente sob ela não é recomposta imediatamente. Para esta finalidade é utilizada a opção RESTAURA, que efetua a recomposição do desenho conforme o desejo do usuário.

3.1.12 TROCANIV

Esta opção é utilizada quando o usuário deseja que todas as figuras de um determinado nível sejam incorporadas a um outro nível.

Transferir o nível A para o nível B significa que todas as figuras que compõem o nível A passam a constar do nível B e o nível A, consequentemente, torna-se vazio.

3.1.13 EXPAND

Durante a edição de um "lay-out", é permitida a chamada de macro-células, ou simplesmente macros, previamente projetadas e verificadas. Estas macros passam a compor o "lay-out".

Dependendo do número de macros a serem chamadas e manipuladas, o tempo requerido para exibição do "lay-out" pode se tornar muito grande. Assim sendo, foi incorporado ao EGIPCIS a opção de trabalhar com o contorno das macro-células, com o objetivo de reduzir o tempo de exibição e diminuir a quantidade de informação na tela durante a montagem de circuitos complexos.

O comando EXPAND, por sua vez, expande as macros, ou seja, exibe todo o conteúdo das macros existentes.

Executando o mesmo comando mais uma vez, a expansão é retirada, dando lugar novamente aos contornos.

3.1.14 EXTRPAR

Extração de parâmetros específicos para transistores MOS, que serão utilizados posteriormente na simulação elétrica.

Os parâmetros medidos são (ver figura 10) :

a. Área de dreno e fonte
Para o cálculo da área dos polígonos correspondentes aos drenos e fontes adota-se a filosofia de dividir o mesmo em retângulos através dos quais são medidas suas áreas. A área total é a soma das áreas de todos os sub-polígonos.
Para a obtenção da área de um retângulo qualquer, basta indicar com o cursor os dois vértices que formam a diagonal.
À medida que as áreas são calculadas, os retângulos são hachurados e os valores mostrados na linha de status.
(ver figura 11)

b. Perímetro de dreno e fonte de um MOSFET
Para o cálculo do perímetro de um polígono é necessário indicar com o cursor todos os seus vértices de maneira consecutiva. A medida do perímetro é encerrada quando o primeiro vértice é reapontado pelo cursor.

c. Dimensões do canal de um MOSFET
O comprimento/largura do canal é obtido através da medida de um ou mais segmentos.
Uma medida de segmento é efetuada apontando com o cursor os seus extremos.
Conforme os segmentos são indicados e medidos, o EGIPCIS mostra os resultados na linha de status.

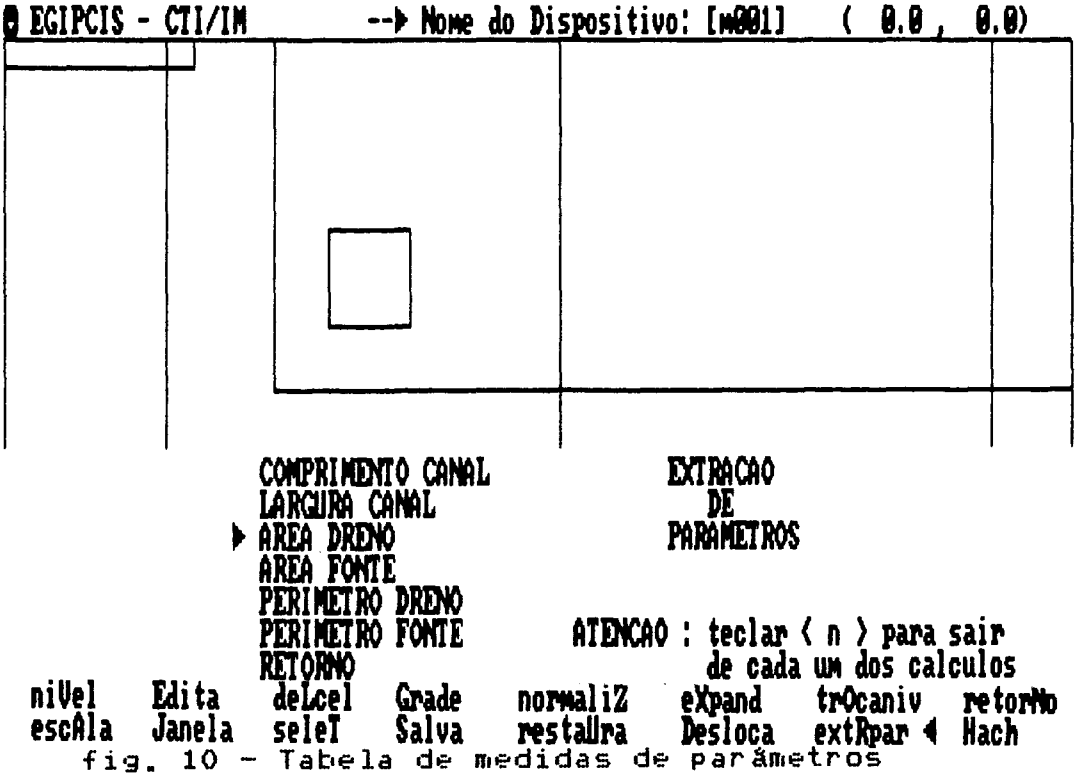


fig. 10 - Tabela de medidas de parâmetros

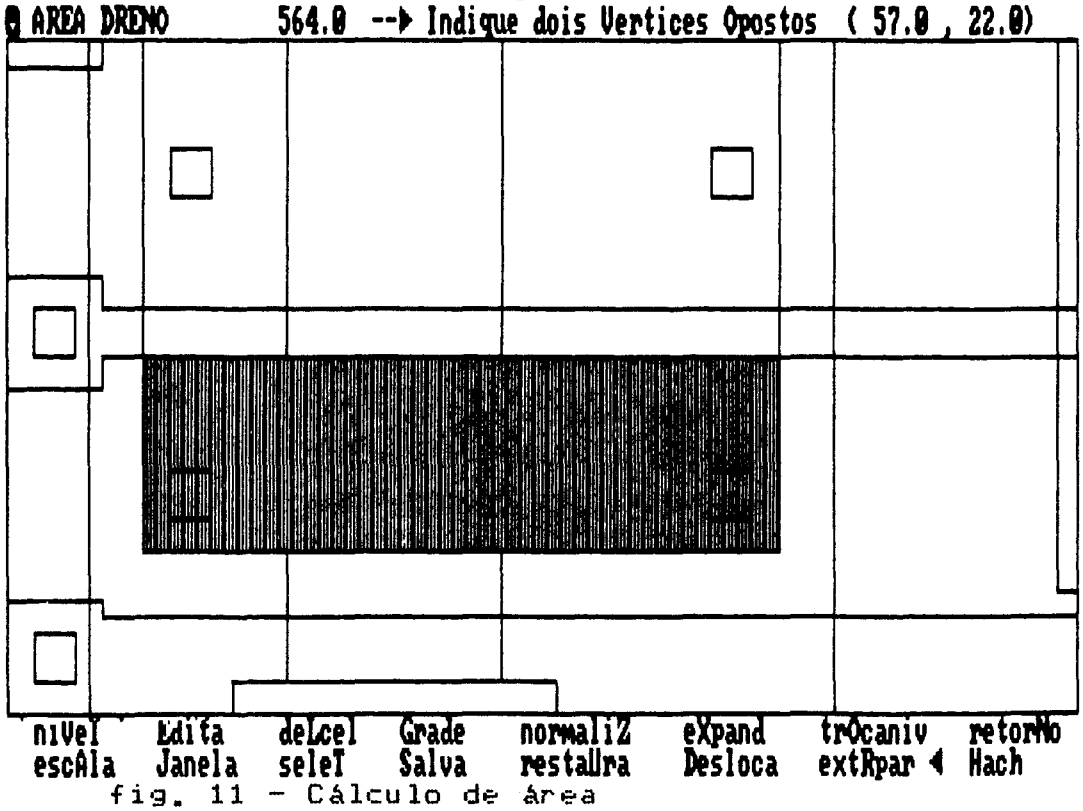


fig. 11 - Cálculo de área

3.1.15 RETORNO

Encerra edição do "lay-out" em questão. O EGIPCIS relembra o usuário da tarefa de salvar o projeto.

3.1.16 EDITA

Ao optar pela edição propriamente dita, um novo menu, contendo comandos de criação e modificação de "lay-out", é apresentado.

Retang Polig Quadrad deLnivel normaliZ vIsib Janela eXpand escAla

reMove mOve rEpete deForma restaUra maCro Desloc niVel retorNo

Os recursos são descritos a seguir.

3.1.16.1 RETANG

São requeridos dois pontos para definir um polígono de 4 lados de qualquer tamanho.

O primeiro ponto posiciona o retângulo, enquanto que o segundo ponto define a altura e a largura relativa ao primeiro. Ao se posicionar o primeiro ponto, um pequeno quadrado é exibido nesta posição para servir como referência e o programa passa a informar, na linha de status, o deslocamento do cursor em termos absolutos a partir deste ponto de referência, permitindo ao usuário um maior controle na dimensão do retângulo em construção.

Determinado o outro ponto, o retângulo é, então, desenhado.

É permitido o cancelamento do primeiro ponto definindo o segundo sobre ele.

Caso as dimensões do retângulo excedam os limites da janela de visualização, um simples toque do cursor na borda desta janela é o suficiente para gerar um deslocamento automático, possibilitando, assim, a continuação da construção do retângulo. O tamanho deste deslocamento é equivalente a meia janela de visualização.

3.1.16.2 POLIG

Polígonos são criados simplesmente posicionando seus vértices com o cursor em qualquer lugar da janela de visualização.

No primeiro vértice posicionado, é exibido um pequeno quadrado para servir como referência ao usuário e, na linha de status, passa-se a informar os deslocamentos do cursor em relação ao eixo x e ao eixo y, auxiliando o dimensionamento das arestas do polígono.

As arestas do polígono são desenhadas à medida que este vai sendo criado.

Durante a criação do polígono, é permitido o cancelamento da última aresta desenhada através da tecla DEL. Esta operação pode ser repetida quantas vezes for necessária até chegar, se for o caso, ao primeiro vértice, cancelando, assim, o polígono todo.

O polígono gerado pode ser aberto ou fechado.

Para a obtenção de um polígono fechado, o último vértice deve coincidir com o primeiro, enquanto que para um polígono aberto o último vértice deve ser posicionado duas vezes consecutivas.

Caso a posição para algum vértice do polígono exceda os limites da janela de visualização, um simples toque do cursor na borda desta janela é o suficiente para gerar um deslocamento automático, possibilitando, assim, a continuação da construção do polígono. O tamanho deste deslocamento é equivalente a meia janela de visualização.

3.1.16.3 QUADRAD

Este comando foi implementado para facilitar a criação de uma sequência de contatos, todos de mesmo tamanho.

O usuário fornece a dimensão do quadrado em micra e indica com o cursor a posição desejada para a sua alocação.

Este quadrado pode ser repetido quantas vezes for necessário através do posicionamento do cursor nos pontos desejados. O tamanho máximo de quadrado é 9 (nove) micra; para dimensões maiores que esta, deve ser utilizado a opção RETANG.

3.1.16.4 REMOVE

Permite que uma figura qualquer - quadrado, retângulo ou polígono, seja apagada da janela de visualização e da estrutura de dados.

Esta operação é realizada indicando um dos vértices da figura com o cursor (certificando-se de que este vértice não é comum a outras figuras) e pressionando a tecla DEL.

A figura em questão é assinalada através de linhas pontilhadas e uma confirmação é solicitada ao usuário.

Uma vez apagada a figura, o que se encontrava sob ela não é recomposto automaticamente. É preciso utilizar, se for conveniente, o comando RESTAURA.

Ver figuras 12 e 13.

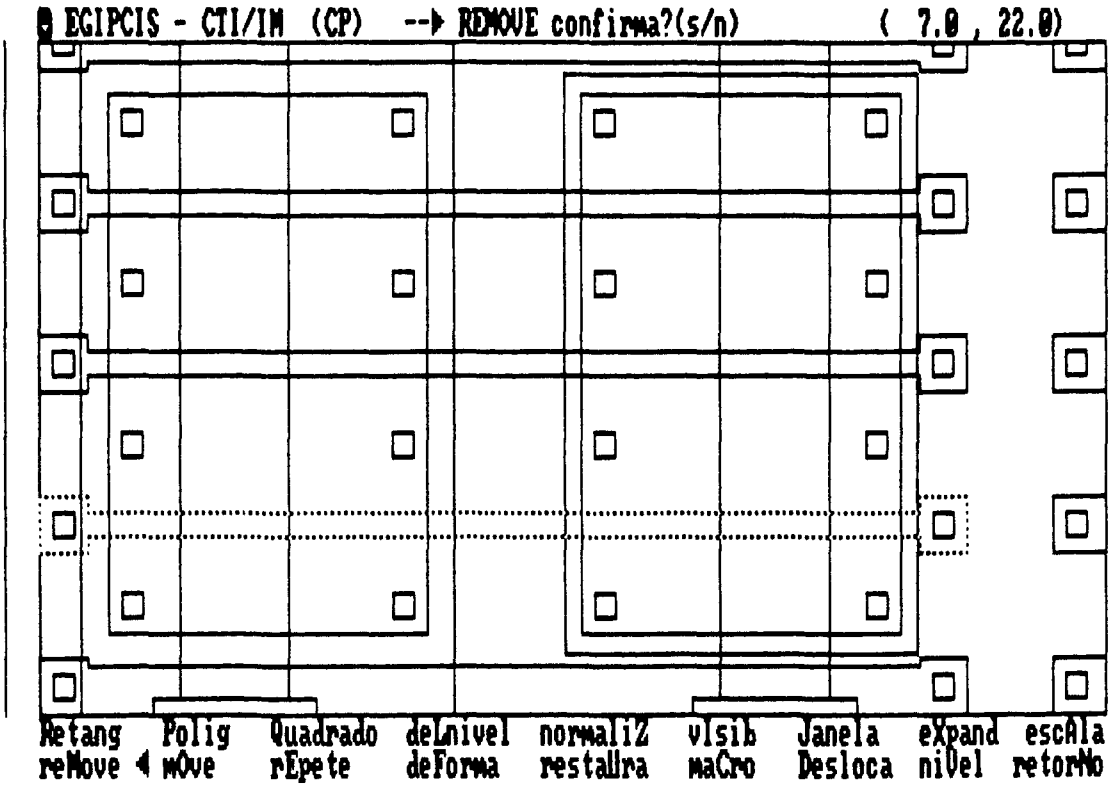


fig. 12 - Remoção de uma figura

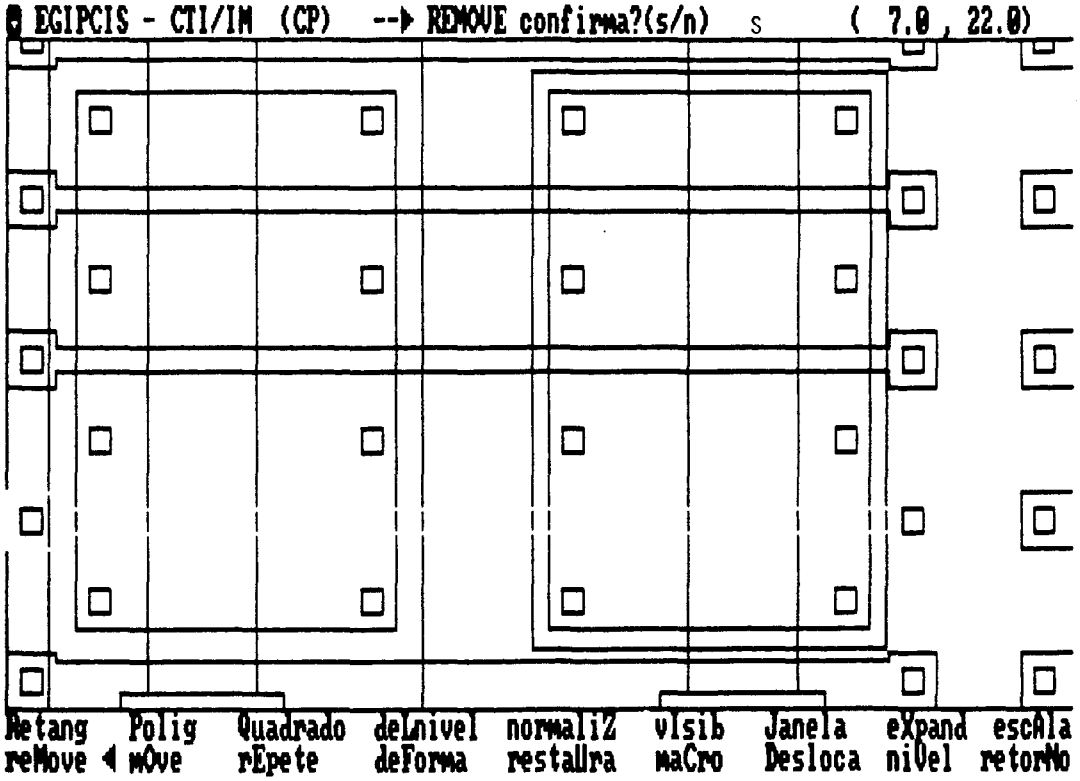


fig. 13 - Confirmação da remoção da figura

3.1.16.5 DEFORMA

Qualquer figura pode ser deformada.

O vértice a sofrer alteração deve ser apontado pelo cursor. Um pequeno quadrado é exibido neste vértice para servir como referência.

O cursor é então deslocado até a posição desejada para o novo vértice. Uma confirmação é solicitada ao usuário. (Ver figuras 14 e 15).

3.1.16.6 REPETE

A repetição é feita identificando um vértice qualquer da figura através do cursor e posicionando-o no novo local onde a figura deverá ser repetida (ver fig. 16 e 17).

3.1.16.7 MOVE

Uma figura qualquer pode ser mudada de posição.

Isto ocorre identificando, através do cursor, um vértice qualquer da figura e posicionando-o no novo local para onde a figura deverá ser transladada.
(ver figuras 18 e 19)

3.1.16.8 DELNIVEL

Apaga da janela de visualização e da estrutura de dados todo o nível corrente de trabalho.

A operação é efetuada somente após uma confirmação do usuário.

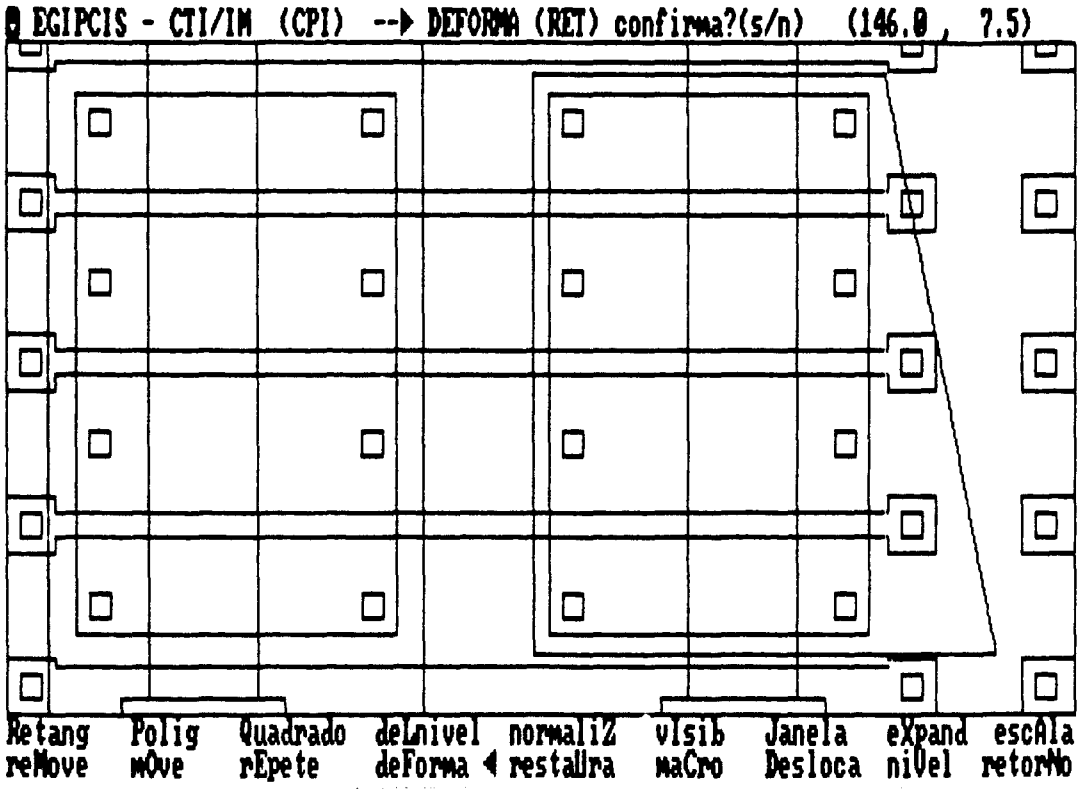


fig. 14 - Deformação de uma figura

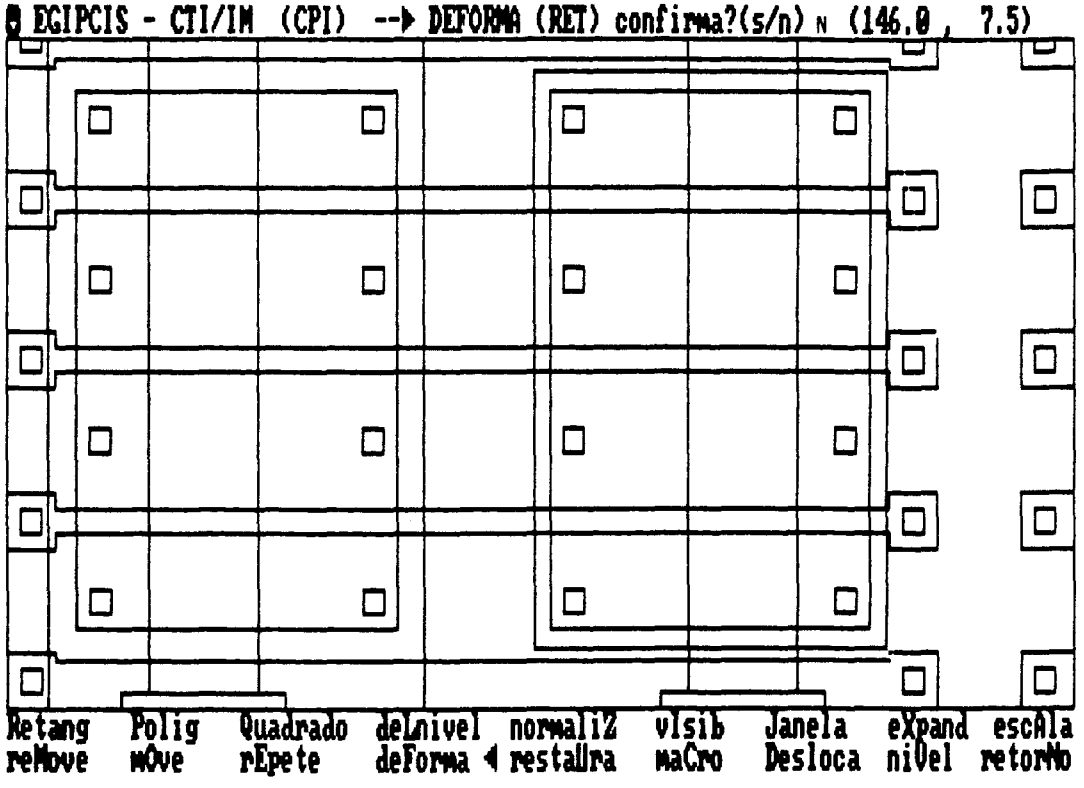


fig. 15 - Não configuração da deformação da figura

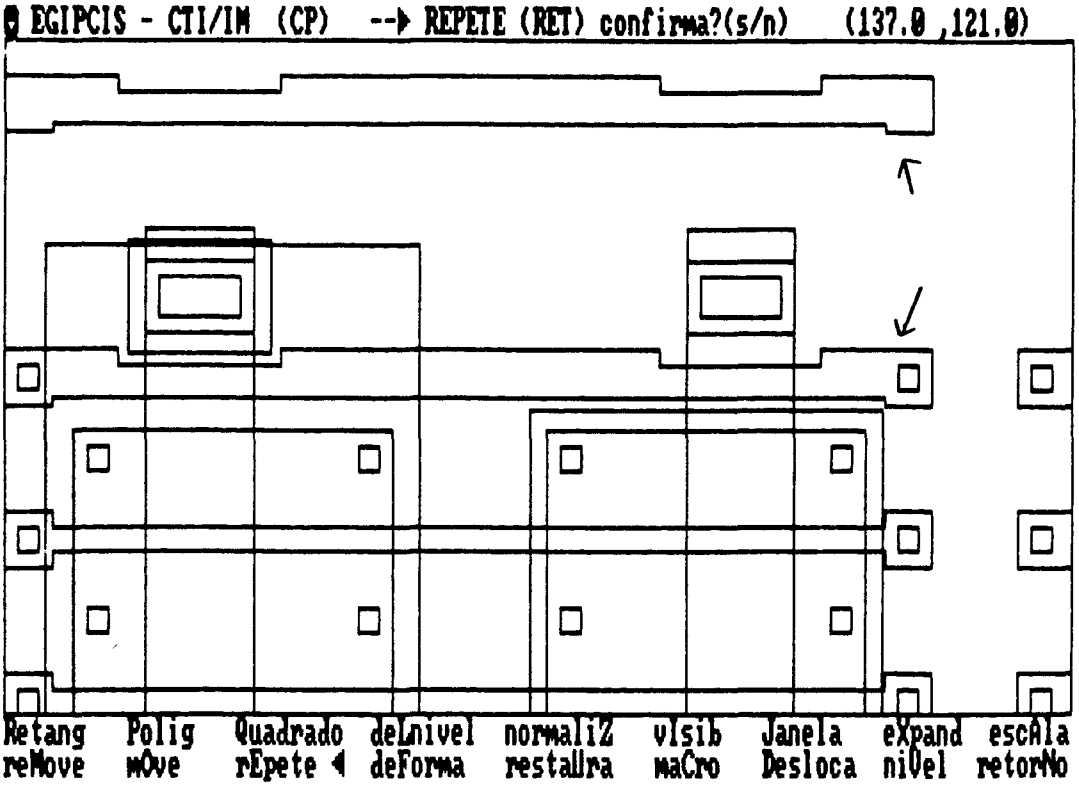


fig. 16 - Repetição de uma figura

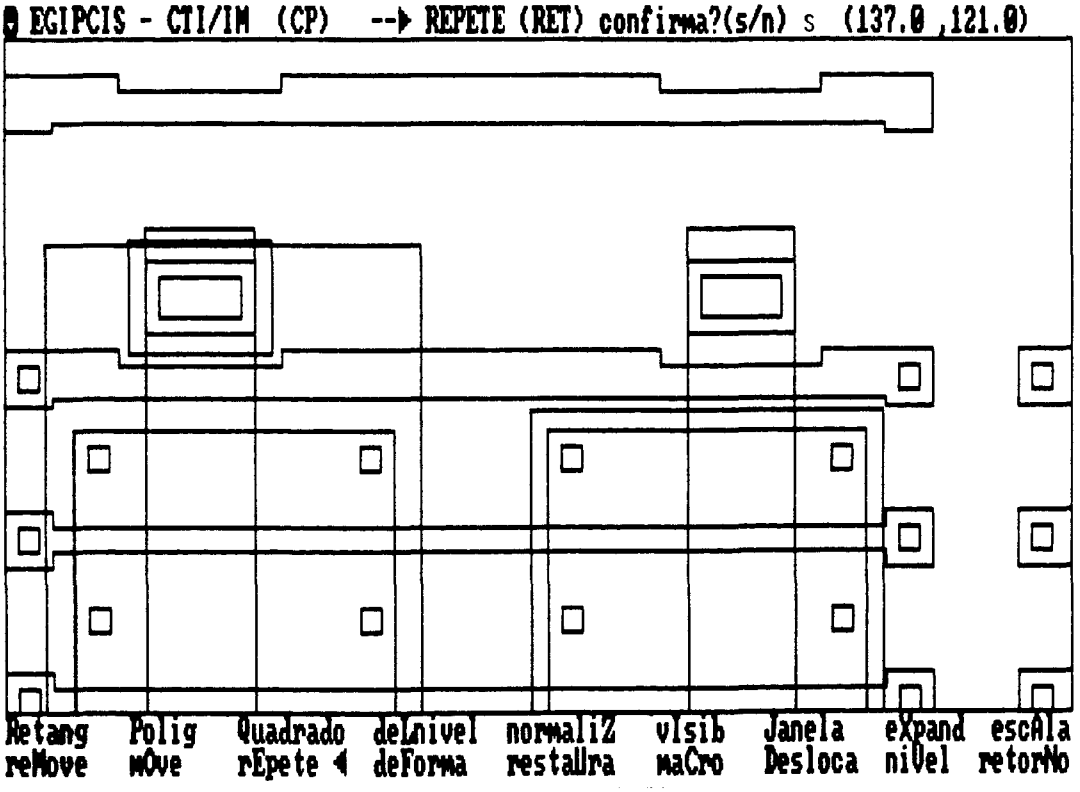


fig. 17 - Confirmação da repetição da figura

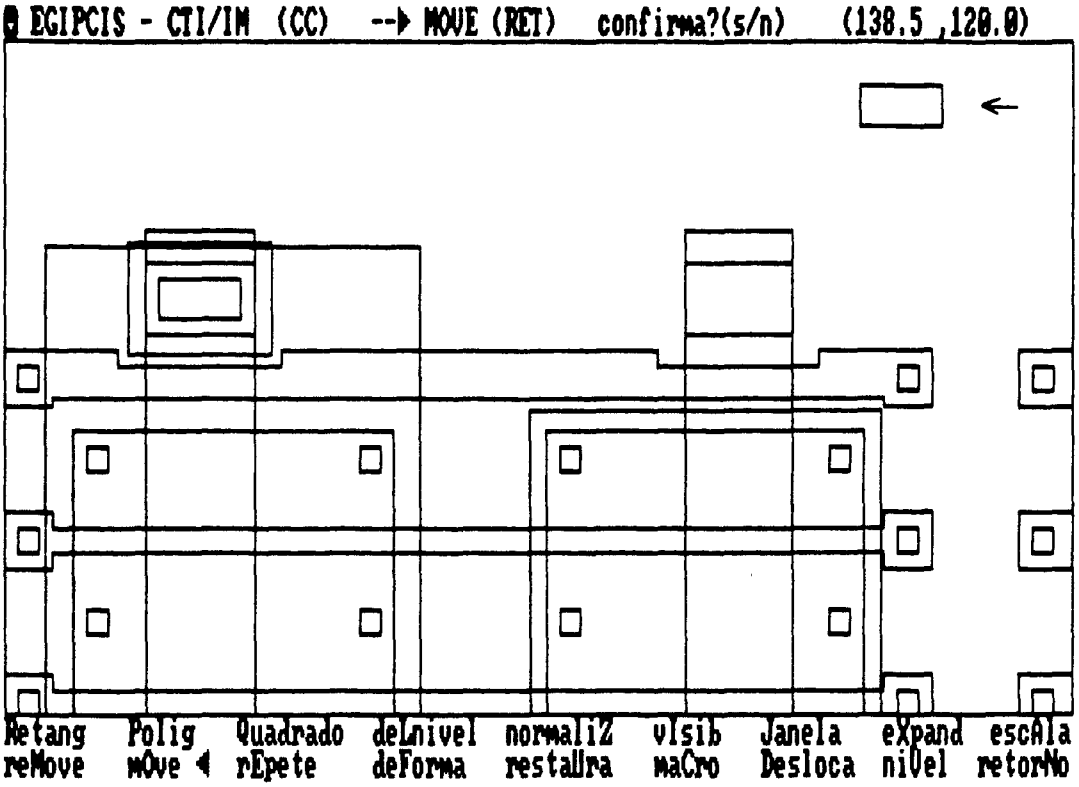


fig. 18 - Translação de uma figura

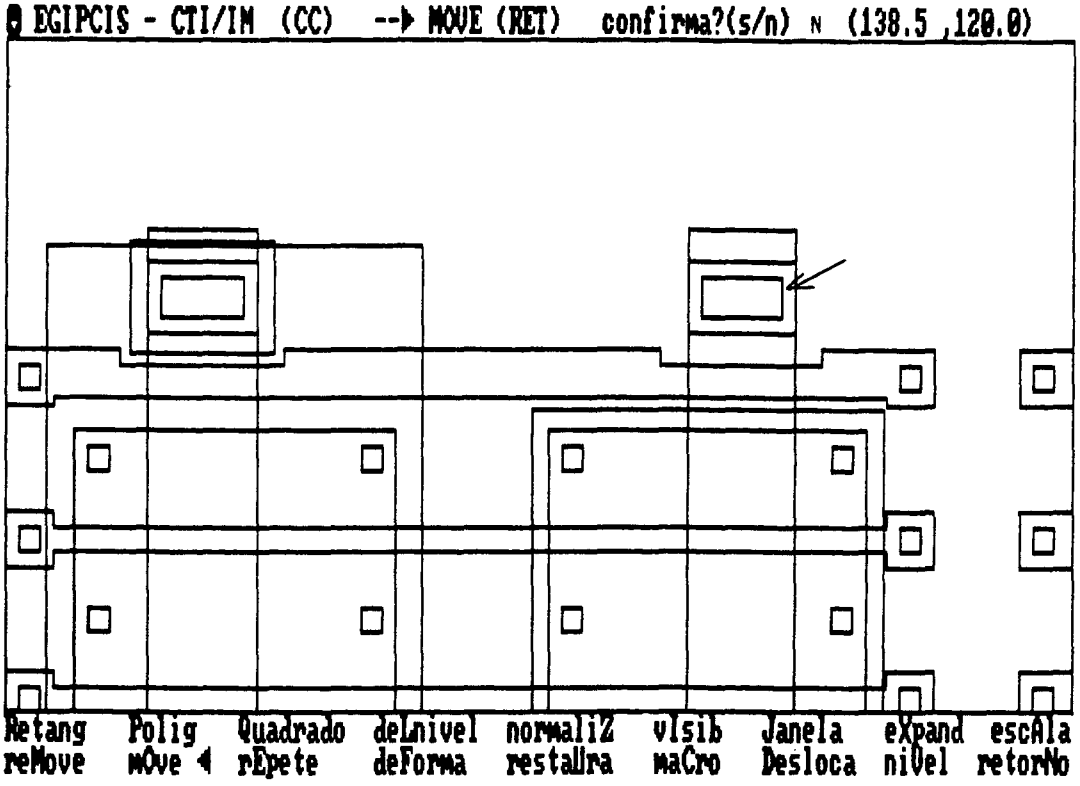


fig. 19 - Não confirmação da translação da figura

3.1.16.9 VISIB

Esta opção permite ao usuário atribuir ou não visibilidade para determinados níveis de máscara.

Isto significa que apenas aqueles níveis selecionados pelo usuário serão exibidos na tela e manipulados pelo programa. O fato de bloquear manipulação nos níveis não visíveis evita possíveis erros de edição.

O objetivo desta opção é tornar mais clara a informação exibida na tela além de reduzir o tempo de redesenho, limitando-se a mostrar apenas os níveis necessários para uma determinada operação.

Inicialmente, o programa atribui visibilidade para todos os níveis de máscara que constam do arquivo "niveis.eg", ficando a critério do usuário qualquer modificação.
(ver figuras 20 e 21)

3.1.16.10 RETORNO

Retorna ao "menu" anterior.

3.1.16.11 MACRO

Permite a composição do "lay-out" a partir de macro-células já existentes, através das seguintes etapas:

- Especificação da dimensão final aproximada do "lay-out".
- Especificação do nome da macro a ser incluída no projeto. Uma busca no diretório é realizada com a finalidade de verificar se existe o arquivo que contém a descrição da macro requerida.
- Aplicação, se for o caso, de transformação do tipo rotação e/ou reflexão à macro em questão.
- Alocação da macro na posição desejada.

  Geralmente, a parte do "lay-out" exibida na janela de visualização é consideravelmente pequena, comparada ao "lay-out" como um todo, sendo praticamente impossível obter uma visão do projeto.

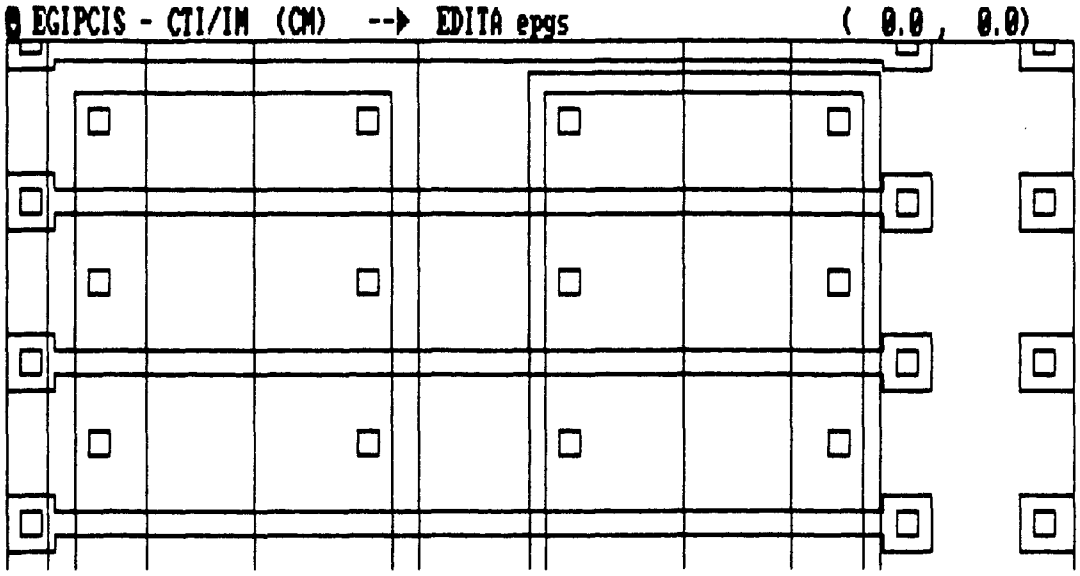


fig. 20 - Atribuição de visibilidade : níveis CA,CM,CNW

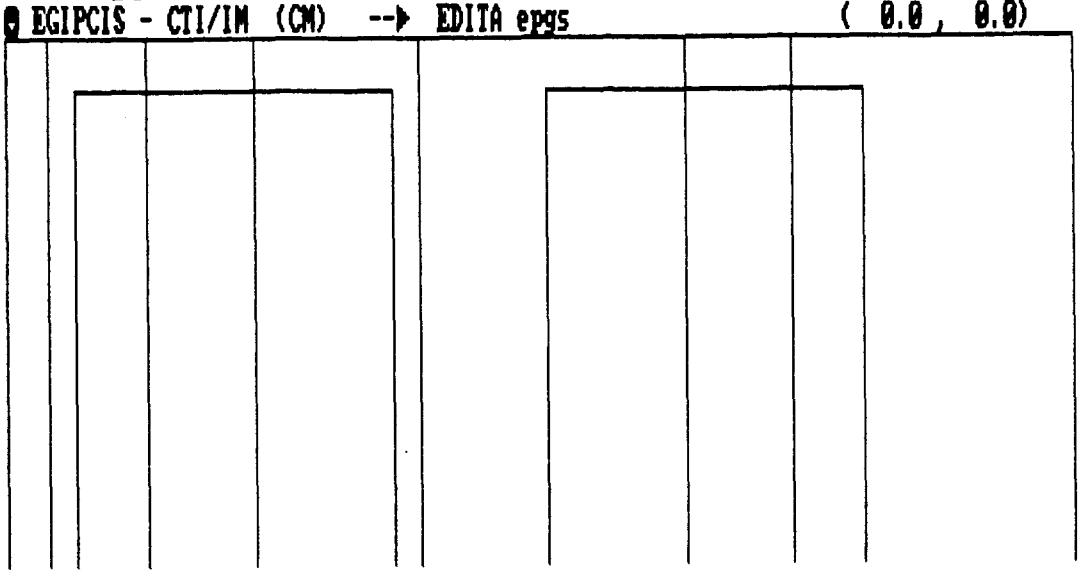


fig. 21 - Atribuição de visibilidade : níveis CC,CA,CP

Por este motivo, a tarefa de montagem/composição é realizada partindo do princípio de que a dimensão total do projeto é igual à dimensão da janela de visualização. Assim, é possível um acompanhamento visual por parte do usuário da montagem a ser efetuada. Se existir a necessidade de visualizar alguma região do "lay-out" em suas dimensões reais, deve-se fazer uso dos comandos NORMALIZ, DESLOCA e JANELA, já descritos anteriormente nos itens 3.1.6, 3.1.4 e 3.1.5 respectivamente.

Com a finalidade de diminuir o tempo gasto na montagem, somente os contornos das macros são exibidos, porém com possibilidade de expansão das macros através do comando EXPAND descrito no item 3.1.13.

Esta opção exibe um novo menu, que proporciona diversos recursos a serem descritos a seguir.

Aloca (-,-) rOtação Executa Cancela Salva Grade retorNo

matrIz (-,-) reFlexão normaliZ Janela eXpand Desloca

3.1.16.11.1 ROTAÇÃO

Os pontos (x, y) de uma figura são girados de um ângulo A em torno da origem do sistema de coordenadas [03] [09] [15].

A rotação é feita no sentido anti-horário e de ângulo A múltiplo de 90 graus (ver figura 22).

Esta transformação de rotação é representada matricialmente como:

$$[x'\ y'\ 1] = [x\ y\ 1] \cdot \begin{vmatrix} \cos A & \text{sen } A & 0 \\ -\text{sen } A & \cos A & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

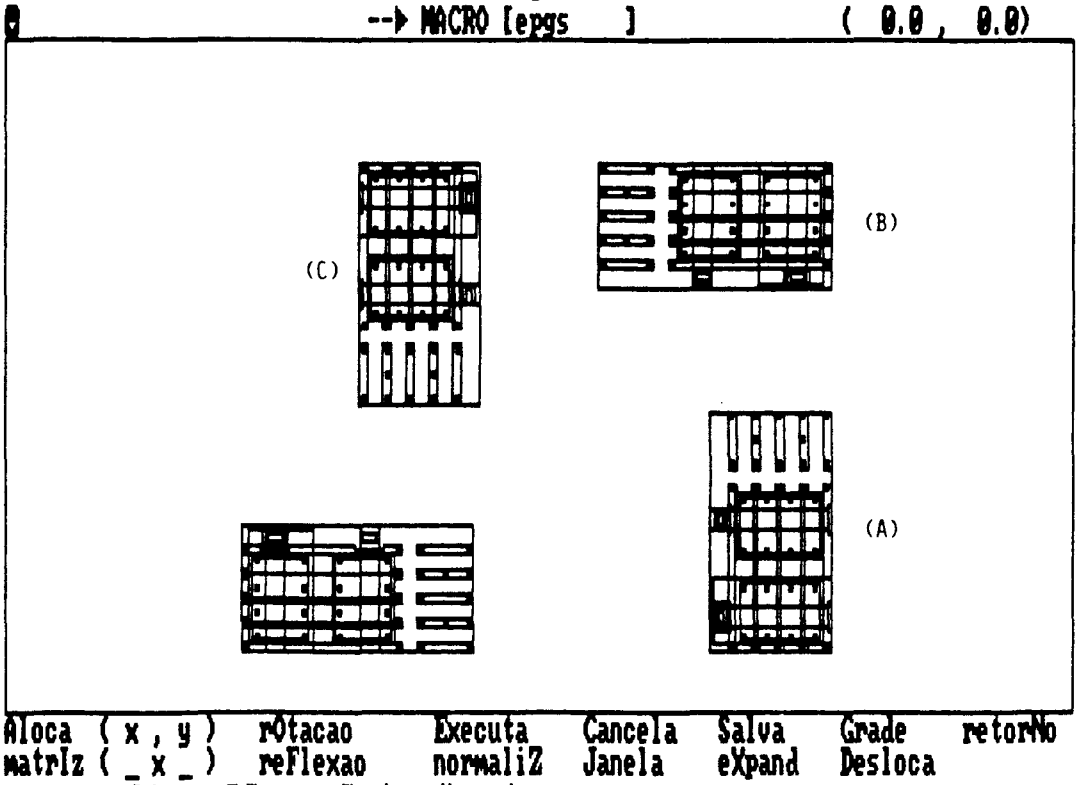


fig. 22 - Rotação da macro
a) 90 graus b)180 graus c)270 graus

3.1.16.11.2 REFLEXAO

A imagem refletida de uma figura pode ser gerada através da aplicação da transformação de escalamento, atribuindo valores negativos para o fator de escala Sx ou para o fator de escala Sy, dependendo do eixo utilizado para a reflexão [03] [09] [15].
(Ver figura 23)

A reflexão de um ponto (x, y) em torno do eixo x gera um novo ponto (x, y) tal que:

$$[x'\ y'\ 1] = [x\ y\ 1] \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

A reflexão de um ponto (x, y) em torno do eixo y gera um novo ponto (x', y') tal que:

$$[x'\ y'\ 1] = [x\ y\ 1] \cdot \begin{vmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

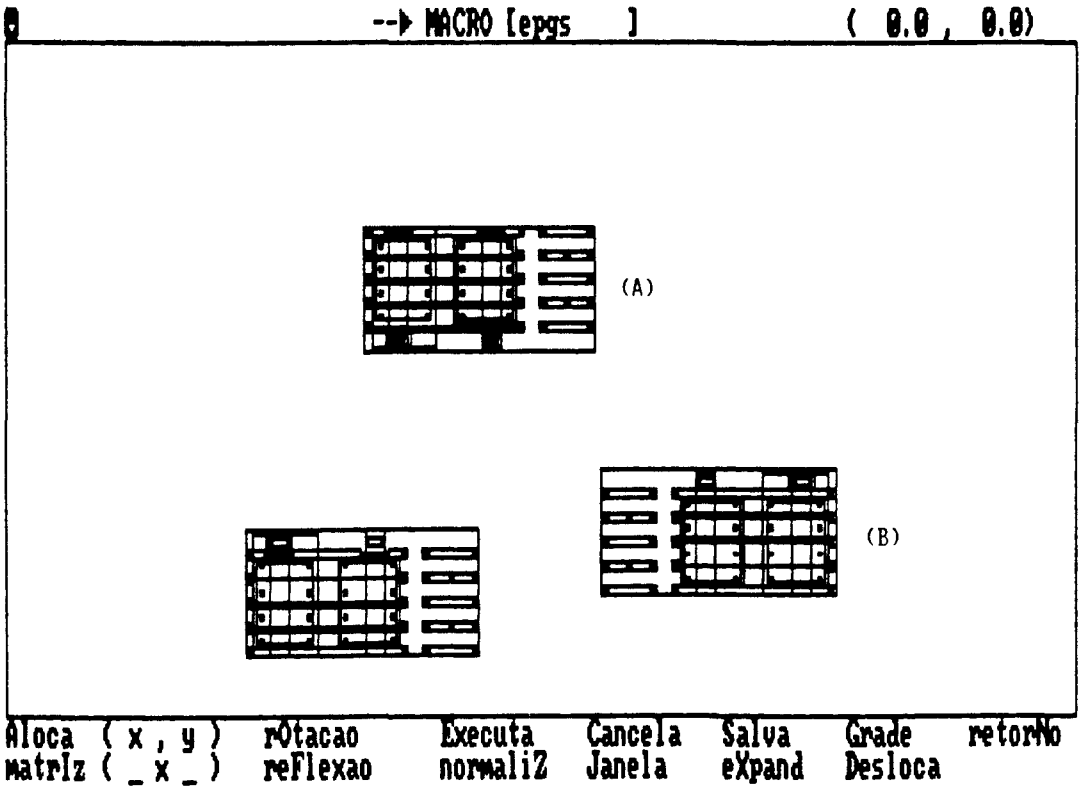


fig. 23 - Reflexão da macro a) em torno do eixo x
b) em torno do eixo y

3.1.16.11.3 ALOCA

Determina a posição onde a macro-célula é incluída no projeto. A macro é alocada através da transformação de translação aplicada a ela [03] [09] [15].

O usuário pode fornecer esta posição ou através do cursor ou via teclado, o que evita deslocamentos sucessivos até encontrar a posição desejada. Se a posição não for especificada, a macro será alocada no ponto (0, 0).

3.1.16.11.4 EXECUTA

Com este comando, a estrutura de dados do programa é atualizada, passando a constar dela os dados da nova macro que foi incluída no projeto.

Estes dados se referem à sequência de transformações aplicadas à macro em questão que, através do processo de concatenação, passa a ser representada por uma única transformação [03] [09] [15].

O propósito básico de concatenar transformações é que é mais eficiente aplicar uma única transformação a um determinado ponto do que uma série de transformações, uma após a outra.

Usando este conceito, a única matriz de transformação armazenada é a matriz resultante da combinação das transformaçòes aplicadas.

Supondo que o ponto (x, y) de uma figura sofre uma rotação de ângulo A em torno da origem, uma reflexão em torno do eixo y e é transladado de Tx unidades ao longo do eixo x e Ty unidades ao longo do eixo y. Um novo ponto (x', y') é gerado de modo que:

$$[x'\ y'\ 1] = [x\ y\ 1] \cdot \begin{vmatrix} -\cos A & \operatorname{sen} A & 0 \\ \operatorname{sen} A & \cos A & 0 \\ Tx & Ty & 1 \end{vmatrix}$$

$$\begin{vmatrix} -\cos A & \operatorname{sen} A & 0 \\ \operatorname{sen} A & \cos A & 0 \\ Tx & Ty & 1 \end{vmatrix} \Rightarrow M$$

onde M é a matriz resultante da combinação das transformações:

$$M = \begin{vmatrix} \cos A & \operatorname{sen} A & 0 \\ -\operatorname{sen} A & \cos A & 0 \\ 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix} \cdot \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ Tx & Ty & 1 \end{vmatrix}$$

Uma observação importante é que a ordem da sequência de transformações não é destruida pela concatenação. Uma vez invertida a ordem, o resultado obtido não seria o mesmo.

(Ver figura 24)

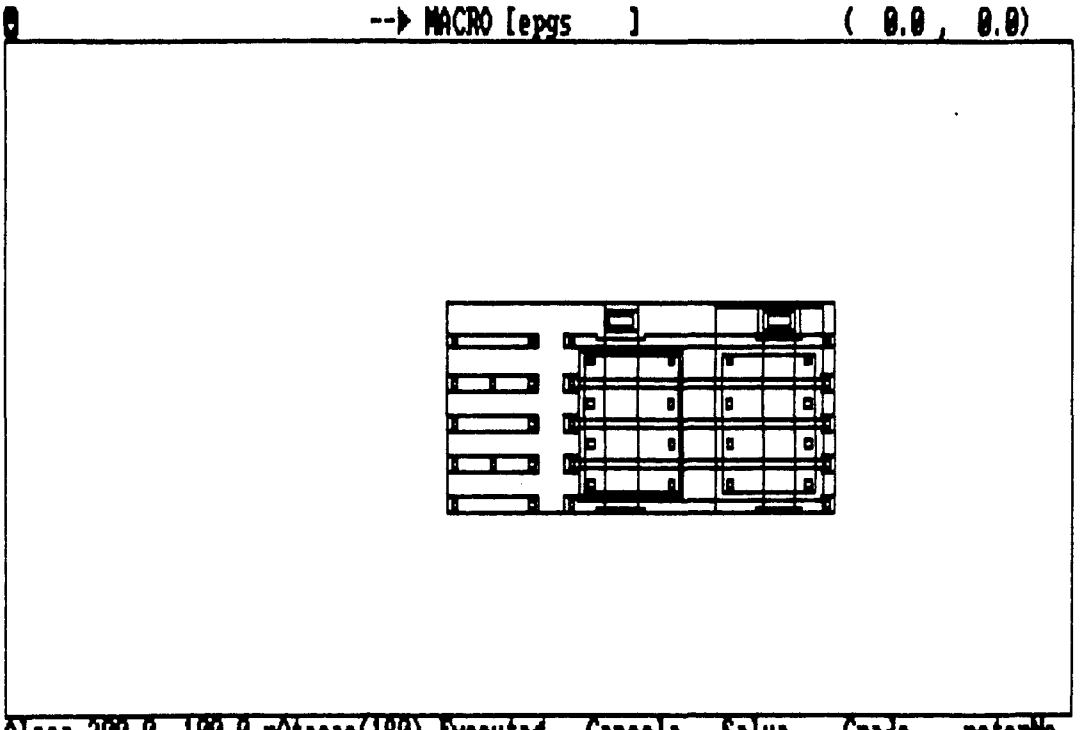


fig. 24 - Sequência de transformações
Rotação de 180 graus + Reflexão X +
+ Translação (300,100)

3.1.16.11.5 MATRIZ

Tem como objetivo replicar a macro corrente, dispondo-as em linhas (fileiras), na forma de vetor, ou mesmo em linhas e colunas, na forma de matriz.

O intuito é auxiliar o usuário com relação a repetições sucessivas de uma mesma macro-célula.

Os dados requeridos pelo programa são:

- espaçamento entre macros, em micra;
- número de repetição das macros;
- posição inicial para a sequência de repetições, através de cursor ou teclado.

A macro aplicada pode ter sofrido, anteriormente, transformação de rotação e/ou de reflexão.

(Ver figuras 25 e 26)

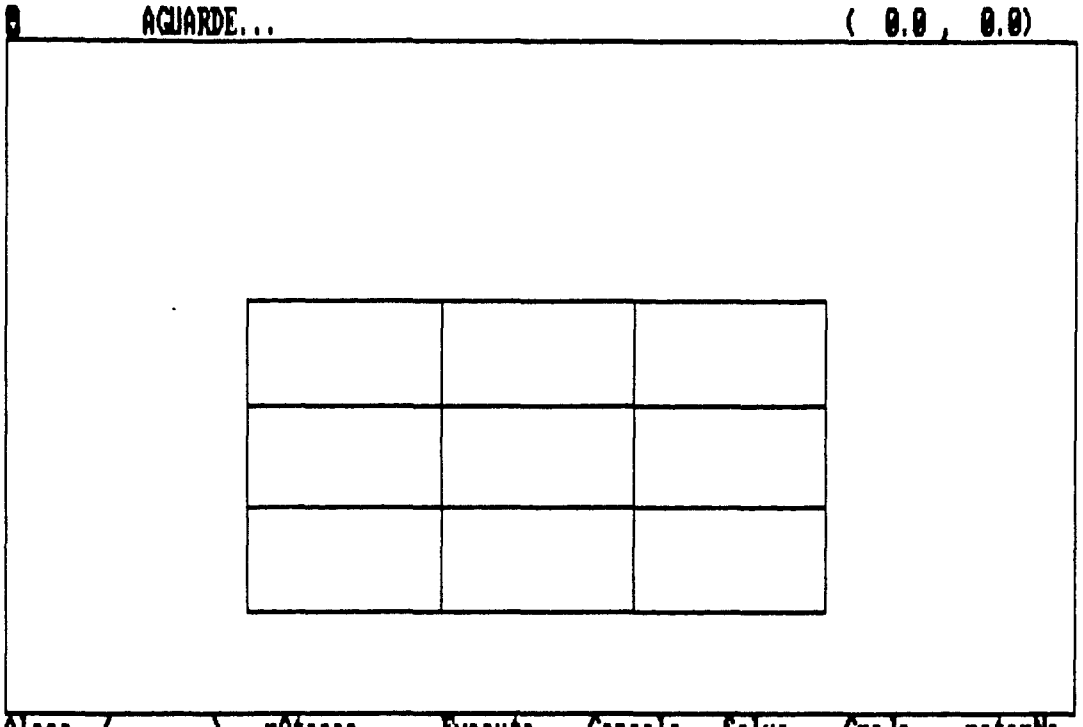


fig. 25 - Matriz de macro-células

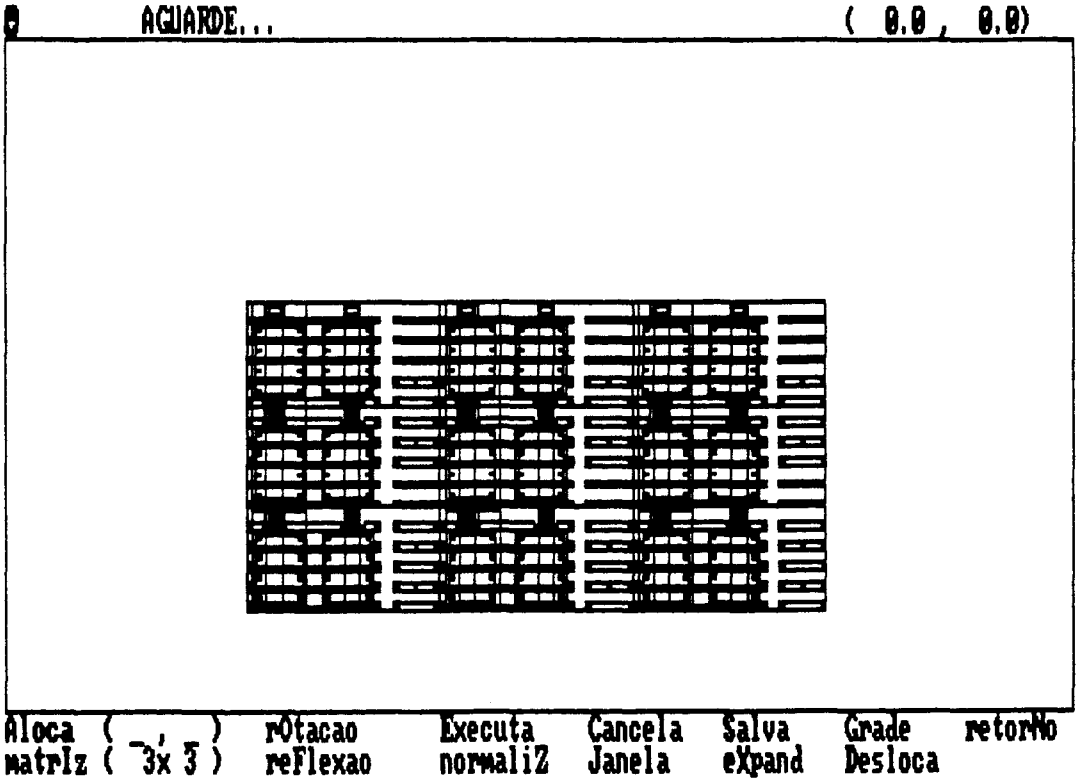


fig. 26 - Matriz de macro-células expandidas

3.1.16.11.6 CANCELA

Permite o cancelamento da última operação efetuada na macro mediante confirmação do usuário.

Se nesta última operação foi utilizado o comando MATRIZ (item 3.1.16.11.5) para replicar a macro em uma determinada sequência, o comando CANCELA atua de maneira diferente. Em vez de toda a sequência de macros ser cancelada, somente a última macro da sequência é que o será. Se o usuário quiser cancelar toda a sequência, deve requisitar o comando CANCELA n vezes, onde n é o número de vezes que a macro foi replicada.

Agindo deste modo, o EGIPCIS permite que apenas algumas macros sejam canceladas.

Uma observação importante é o fato deste cancelamento ocorrer obrigatoriamente do fim para o começo da sequência (ver figura 27).

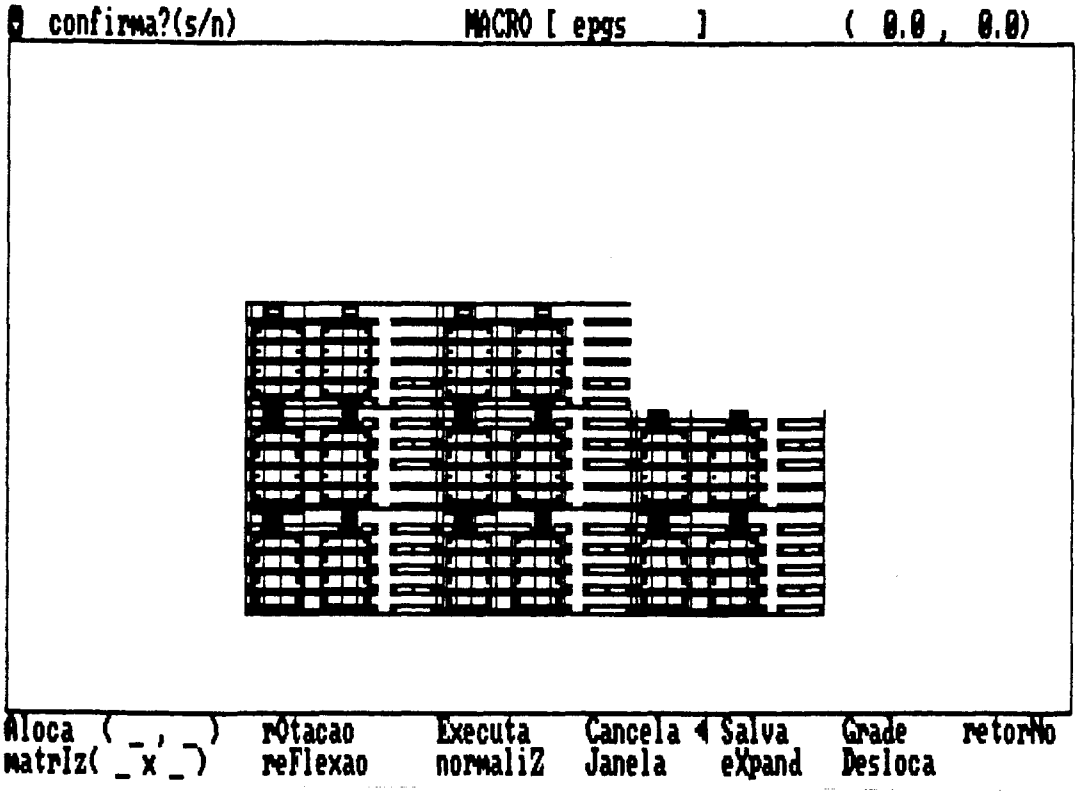


fig. 27 - Ocorrência de um cancelamento na matriz de células

3.1.16.11.7 RETORNO

Retorna ao pedido de especificação do nome da macro a ser incluída no projeto. Caso o usuário não queira incluir outros elementos, deve teclar (RETURN) ao invés de fornecer algum nome.

O programa volta a apresentar, então, o menu que contém as primitivas de edição.

3.1.17 USO DAS TECLAS DE FUNÇÃO:

As teclas de função podem ser utilizadas em qualquer momento da edição, independente do "menu" que se apresenta na tela.

Elas executam funções mais gerais do programa como:

TECLA	FUNÇÃO
F1	calcular o ponto máximo (x,y) do "lay-out".
F2	fazer cópia, na impressora, do que se encontra na janela de visualização.
F3	calcular distâncias, a partir de 2 pontos fornecidos pelo usuário. Recursos de deslocamento podem ser utilizados na busca dos pontos desejados.
F4	modificar o nome do "lay-out" que está sendo editado.
F5	fazer cópia, em arquivo de nome diferente, do "lay-out" que está sendo editado.
F6	ativação/desativação do preenchimento dos níveis com padrões previamente determinados pelo programa.
F7	fornecer informações sobre as macros que foram incluídas no projeto.
F8	controle do projeto (ainda não implementado -- ver cap. VI, item 3.7)
F9	normalização/não normalização do "lay-out".
F10	restauração da tela.

3.2 REQUISITOS DE DESEMPENHO

3.2.1 MONO-USUARIO

O desenvolvimento do editor foi feito em microcomputador de 16 bits, por isso sua utilização é feita por um único usuário por máquina.

3.2.2 ARQUIVOS MANUSEADOS

3.2.2.1 ARQUIVOS PARA LEITURA

O EGIPCIS trabalha com apenas dois arquivos de leitura, um que contém a descrição em CIF do "lay-out" e o outro que contém os códigos dos níveis de máscara que serão utilizados pelo programa. Todas as informações necessárias à edição são encontradas nestes arquivos e nos dados fornecidos pelo usuário, via teclado, ao longo da execução do programa.

3.2.2.2 ARQUIVOS PARA IMPRESSAO

. arquivo de saida

No final da edição, o próprio arquivo de leitura é atualizado, se for conveniente ao usuário.

. arquivo para cópia

O usuário pode também fazer uma cópia do "lay-out" que está sendo gerado em um arquivo de nome diferente do original.

. arquivo de controle

Existirá também um arquivo de controle de projeto, que fará a contabilidade do uso do editor no projeto em questão.

3.2.3 TABELA DE VARIAVEIS

A tabela de variáveis é composta de :

- vetores para guardar nomes de arquivos (arquivo contendo descrição do "lay-out", arquivo de cópia, arquivo de controle).
- vetor para guardar os códigos dos níveis de máscara.
- vetor para guardar os vértices de um polígono, quando seu preenchimento se fizer necessário.
- vetor para guardar as transformações aplicadas a uma macro.
- lista de coordenadas, contendo os vértices das figuras.
- lista de conteúdo, contendo informações de mais baixo nível de cada célula (transformações aplicadas tipo rotação, reflexão, alocação).
- lista de células, contendo as informações gerais de cada célula (nome, valores máximos e mínimos).

3.3 RESTRIÇÕES GERAIS

1. O EGIFCIS está adaptado para monitor monocromático, mas pode ser modificado para trabalhar com monitor de vídeo colorido. Porém, devido à restrições da linguagem PASCAL TURBO, o resultado pode não ser satisfatório. Isto porque a linguagem permite apenas três modos gráficos:
   - "Graphcolormode", que ativa a tela gráfica colorida de 320 x 200 pontos.
   - "Graphmode", que ativa a tela gráfica branco e preto de 320 x 200 pontos.
   - "Hires", que ativa a tela gráfica de 640 x 200 pontos, com o preto como cor de fundo e uma outra cor selecionada pelo usuário.

   Desta maneira, é possível obter ou um "lay-out" com uma boa resolução mas com todos os níveis de uma mesma cor, ou então um "lay-out" com níveis de cores diferentes porém com baixa resolução.

2. A interface para utilização de dispositivos de entrada como, por exemplo, "mouse" e "tablet", ainda não foi incorporada ao programa.

3. O código que representa um nível de máscara é declarado como sendo uma cadeia de, no máximo, três caracteres.

4. O nome do "lay-out" a ser editado e também das macros a serem incluídas nele são compostos de, no máximo, dez caracteres, sendo que até 8 caracteres se destinam ao nome e o restante à extensão, que é opcional.

5. O número máximo de níveis de máscara que o EGIPCIS suporta é 14.

6. O número máximo de vértices de um polígono é 100.

OBSERVAÇÃO: Os limites dos itens 3, 4, 5 e 6 acima podem ser alterados no programa fonte sem acarretar problema no desempenho deste.

CAPÍTULO 4

ESPECIFICAÇÃO DO PROJETO DE "SOFTWARE"

IV. ESPECIFICAÇÃO DO PROJETO DE "SOFTWARE"

1. INTRODUÇÃO

1.1 PROPÓSITO

Este capítulo tem como propósito definir o editor gráfico interativo de que trata este trabalho em detalhe suficiente para permitir sua codificação [18] [21].

São apresentados o escopo do trabalho, os diagramas de fluxo de dados, a estrutura de dados adotada e os principais procedimentos desenvolvidos.

1.2 ESCOPO

Como foi visto no capítulo III, o editor gráfico EGIPCIS gera, a partir de primitivas gráficas, o "lay-out" de um circuito integrado. Ele foi desenvolvido para operar em microcomputador de 16 bits e a linguagem utilizada foi a PASCAL.

2. DESCRIÇÃO DO PROJETO

2.1 DIAGRAMA DE FLUXO DE DADOS

O programa EGIPCIS foi elaborado de forma modular visando seu teste, manutenção e posteriores implementações de comandos.

Dentro desta filosofia de modularidade, foi gerado durante a fase de projeto o diagrama de fluxo de dados em seus diversos níveis de abstração, partindo do modelo fundamental do sistema até o nível detalhado para programação.

Define-se diagrama de fluxo de dados como sendo uma técnica gráfica para representar fluxo de informação.

Na figura 1, nota-se que o modelo fundamental do sistema representa o programa inteiro como uma bolha e suas entradas e saidas como arcos.

Nas figuras seguintes, sucessivos refinamentos são aplicados até se chegar a um nível grande de detalhamento.

A partir daí, os procedimentos já podem ser escritos em pseudo-código e posteriormente codificados na linguagem estabelecida inicialmente.

MODELO FUNDAMENTAL DO SISTEMA

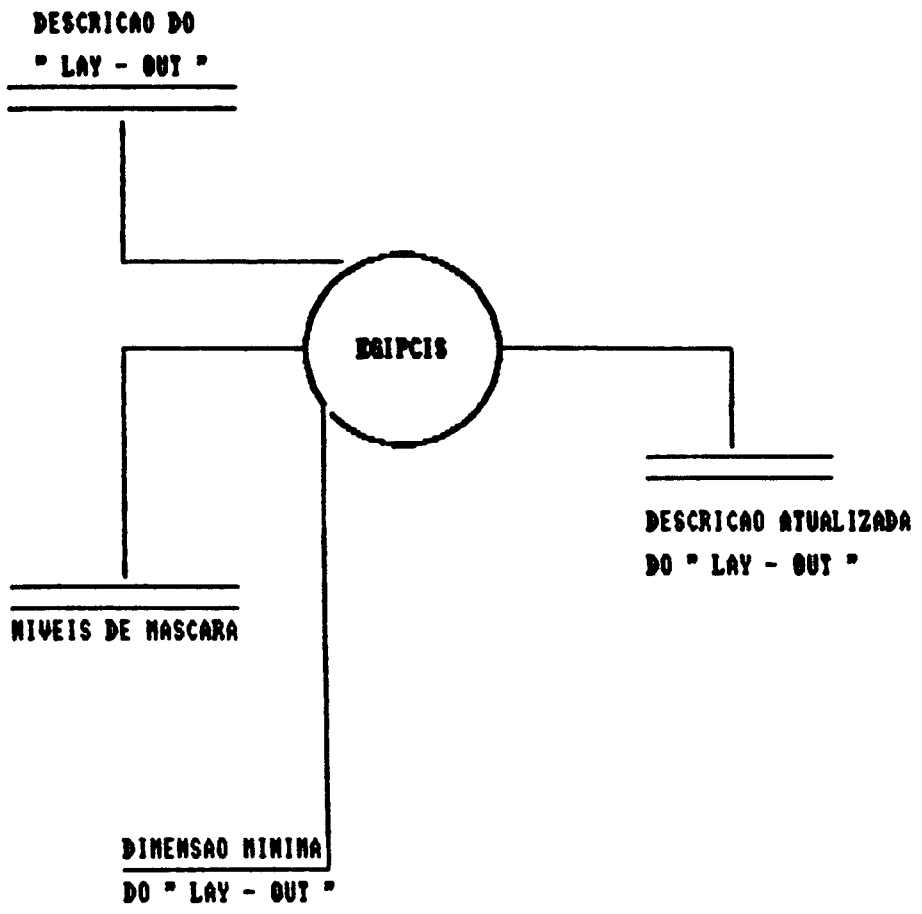


fig. 1

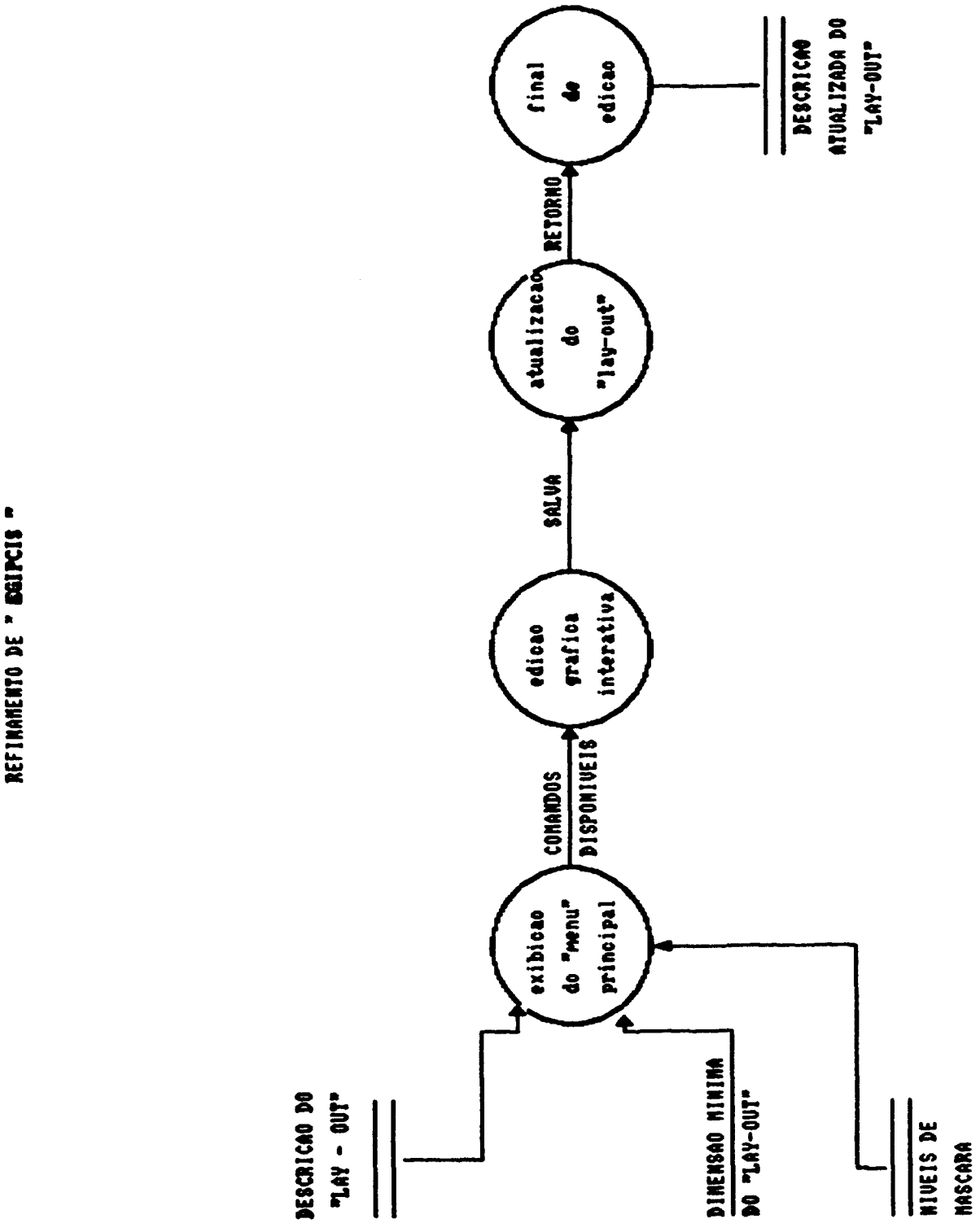

REFINAMENTO DE " EGIPCIS "
DESCRICAO DO "LAY - OUT"
DIMENSAO MINIMA DO "LAY-OUT"
NIVEIS DE MASCARA
exibicao do "menu" principal
COMANDOS DISPONIVEIS
edicao grafica interativa
SALVA
atualizacao do "lay-out"
RETORNO
final de edicao
DESCRICAO ATUALIZADA DO "LAY-OUT"

fig. 2

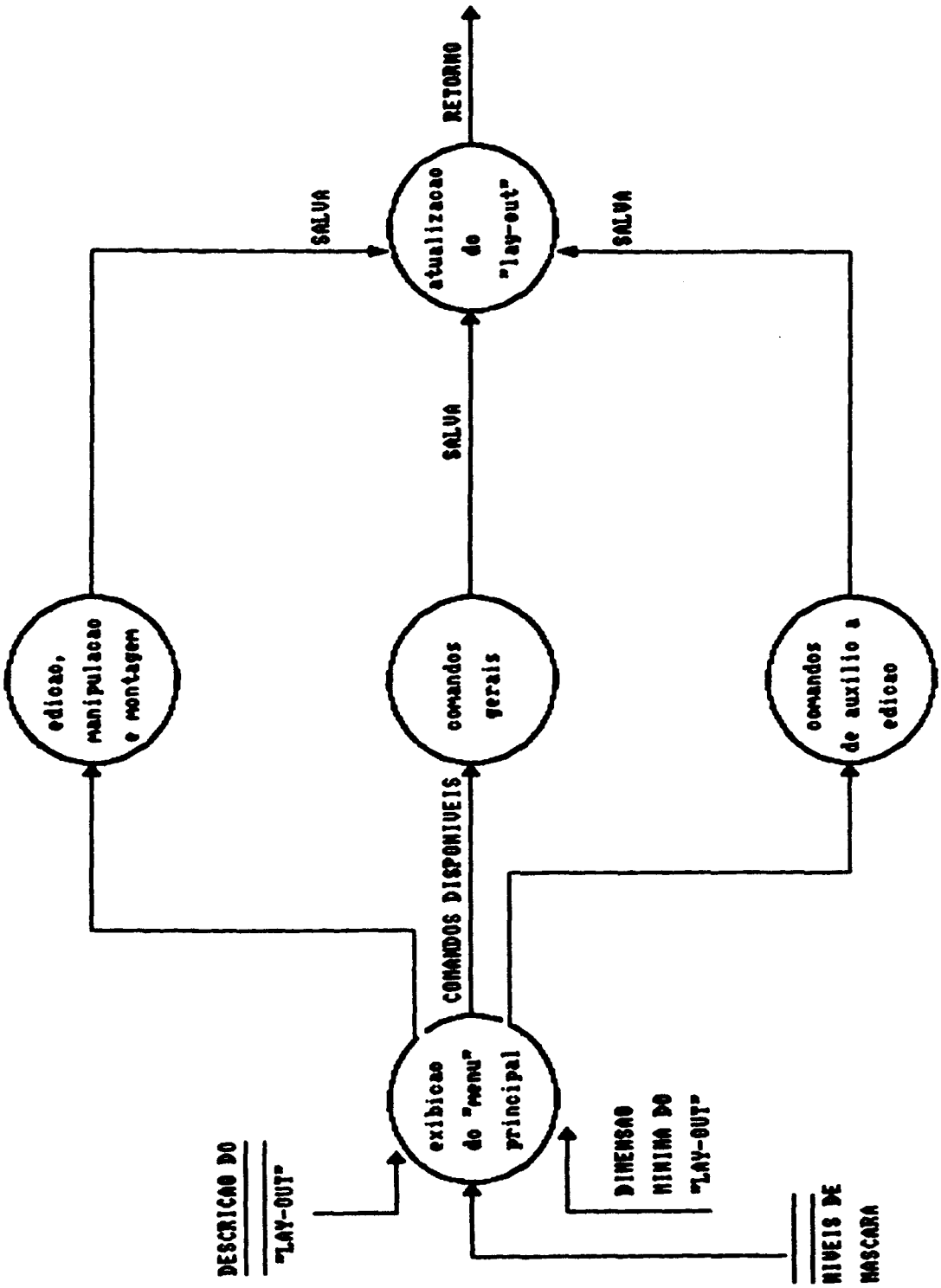

REFINAMENTO DE "EDICAO GRAFICA INTERATIVA"
DESCRICAO DO "LAY-OUT"
exibicao do "menu" principal
DIMENSAO MINIMA DO "LAY-OUT"
NIVEIS DE MASCARA
COMANDOS DISPONIVEIS
edicao, manipulacao e montagem
comandos gerais
comandos de auxilio a edicao
SALVA
SALVA
SALVA
atualizacao do "lay-out"
RETORNO

fig. 3

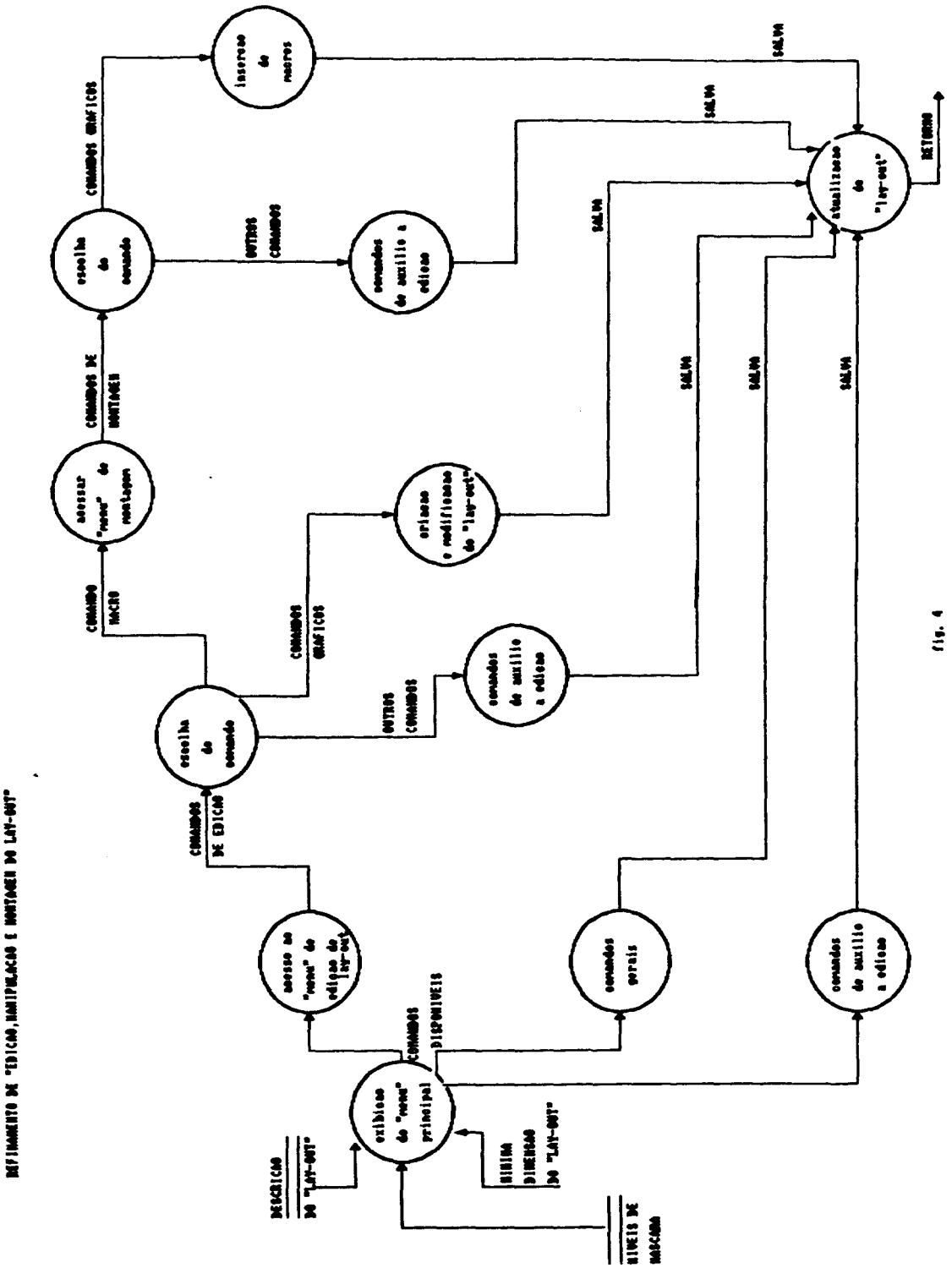

REFINAMENTO DE "EDICAO,MANIPULACAO E MONTAGEM DO LAY-OUT"
inserção de nomes
escolha do comando
comandos de auxílio à edição
atualização do "lay-out"
acessar "nome" de montagem
criação e modificação do "lay-out"
comandos de auxílio à edição
escolha do comando
acesso ao "nome" de edição do lay-out
comandos gerais
comandos de auxílio à edição
exibição do "nome" principal
COMANDOS GRÁFICOS
OUTROS COMANDOS
COMANDO MACRO
COMANDOS DE MONTAGEM
COMANDOS DE EDIÇÃO
COMANDOS DISPONÍVEIS
SAÍDA
RETORNO
DESCRIÇÃO DO "LAY-OUT"
REGIÃO DIRIGIDA DO "LAY-OUT"
NÍVEIS DE MÁSCARA

fig. 4

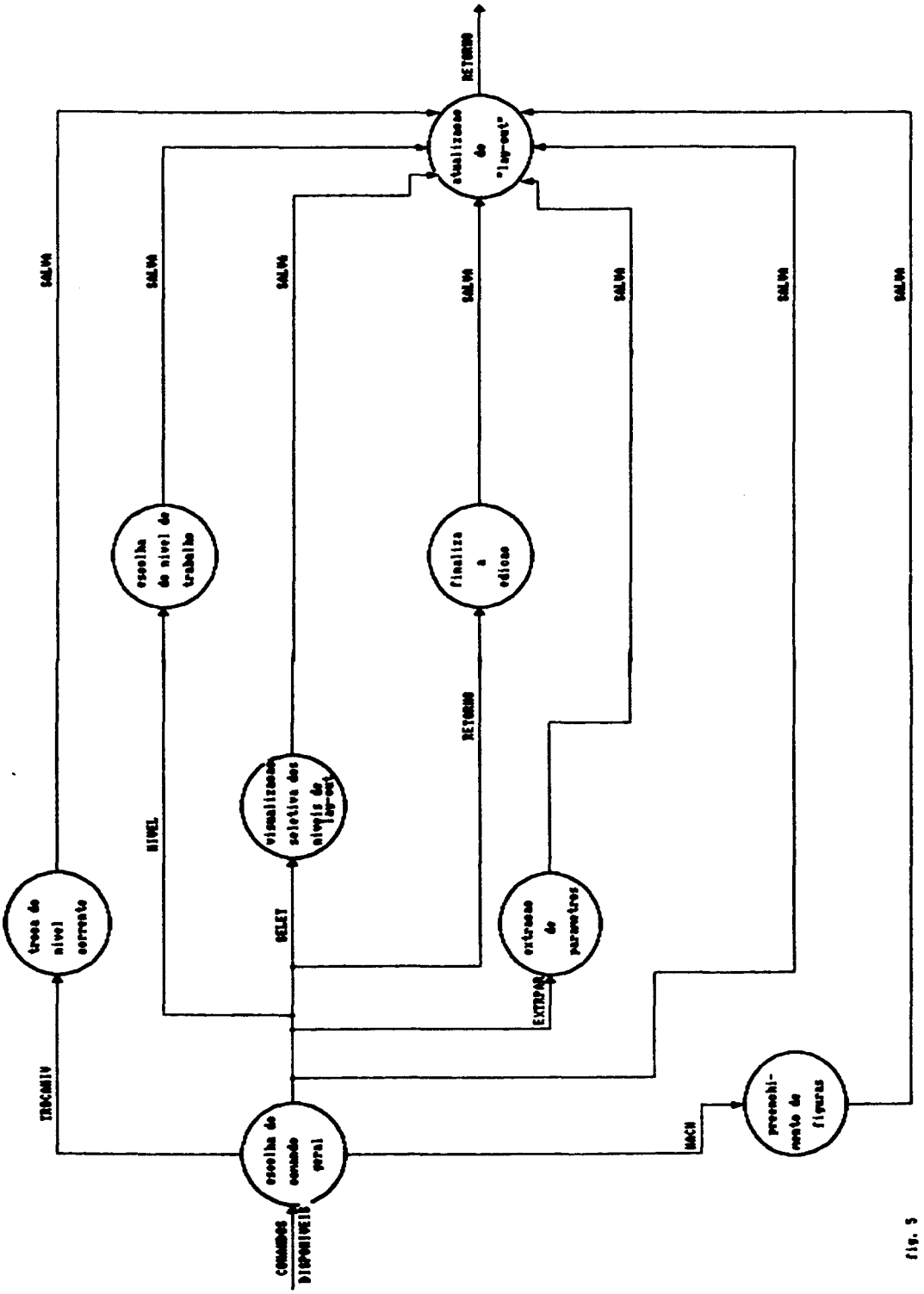

REFINAMENTO DE "COMANDOS GERAIS"
COMANDOS DISPONÍVEIS
escolha do comando geral
troca do nível corrente
escolha do nível de trabalho
visualização seletiva dos níveis de imp-ext
finaliza a edição
entrada de parametros
preenchimento de figuras
atualização do "imp-ext"
RETORNO
SALVA
NÍVEL
SELET
TROCANIV
MACH
EXTPAR

fig. 5

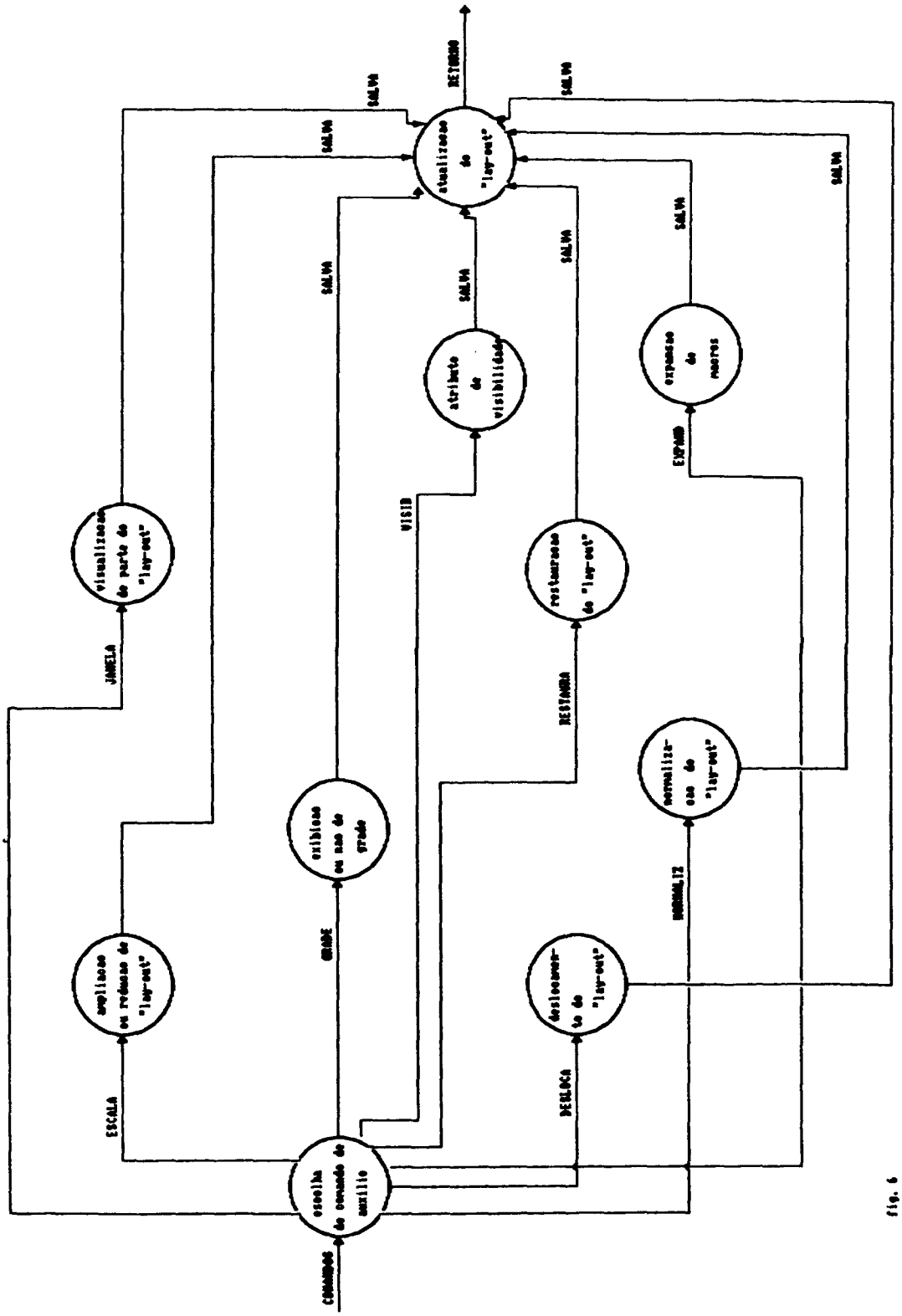

REFINAMENTO DE "COMANDOS DE AUXÍLIO A EDIÇÃO"
COMANDOS
escolha do comando de auxílio
ESCALA
ampliação ou redução do "lay-out"
JANELA
visualização de parte do "lay-out"
GRADE
exibição ou não de grade
VISIB
atributo de visibilidade
DESLOCA
deslocamento do "lay-out"
RESTAURA
restauração do "lay-out"
NORMALIZ
normalização do "lay-out"
EXPANDE
expansão de nomes
SALVA
atualização do "lay-out"
RETORNO

fig. 6

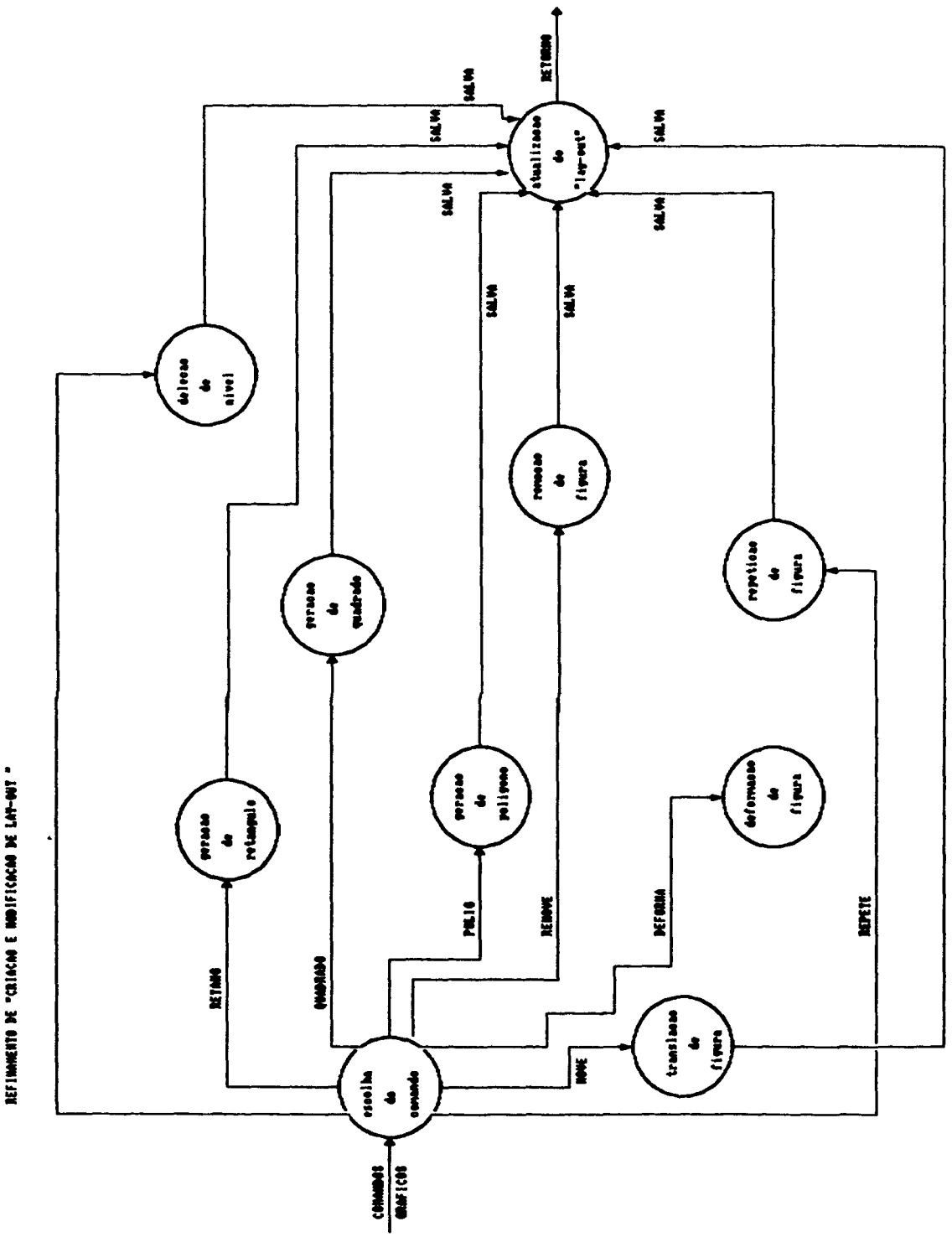
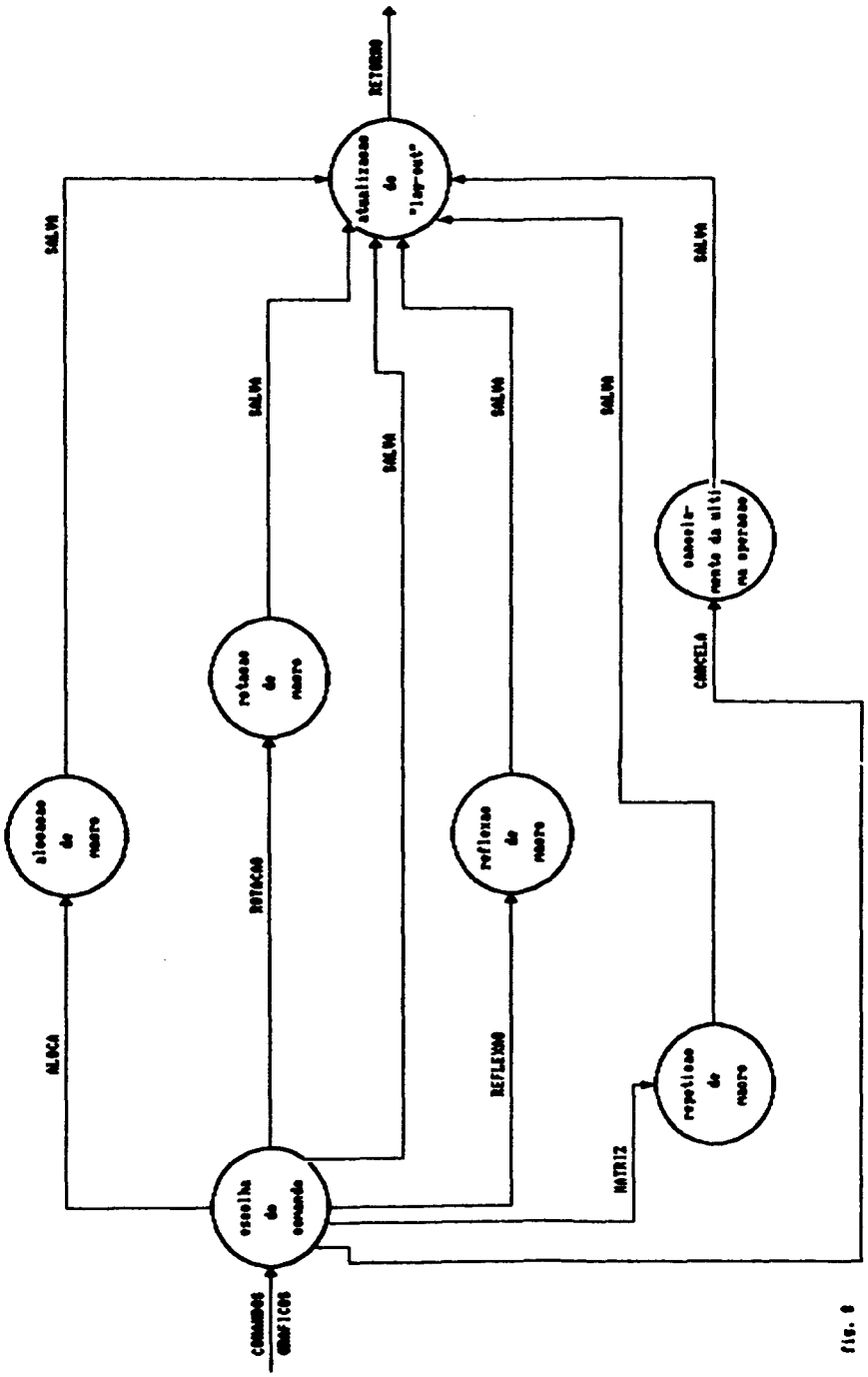

REFINAMENTO DE "CRIACAO E MODIFICACAO DE LAY-OUT"
COMANDOS GRAFICOS
escolha de comando
apaga de nível
apaga de retangulo
apaga de quadrado
apaga de poligono
apaga de figura
repetição de figura
deformação de figura
translação de figura
atualização de "lay-out"
RETORNO
SALVA
RETANG
QUADRADO
POLIG
REMOVE
MOVE
DEFORMA
REPETE

fig. 7



REFINAMENTO DE "INSERCAO DE MACROS NO LAY-OUT"

fig. 8

2.2 ESTRUTURA DE DADOS

2.2.1 USO DE LISTAS SEQUENCIAIS

A estrutura de dados apresentada pelo EGIPCIS é formada por 3 listas sequenciais :

1. Lista de Células

   No seu primeiro elemento são armazenados dados do "lay-out" que está sendo gerado e, nos demais elementos, os dados de cada uma das macro-células que são incluídas neste "lay-out" (ver ítem 2.2.3).

   Existem macro-células que são chamadas mais de uma vez para compor o "lay-out", porém elas são armazenadas na estrutura uma única vez, evitando, assim , a redundância de dados. Se nenhuma macro-célula for incluída no "lay-out", a lista de células passa a ser composta por um único elemento, ou seja, por uma única célula, que é o próprio "lay-out". Na figura 9, este elemento recebe o nome de RAIZ.

2. Lista de Conteúdo

   Cada elemento da lista de células aponta para uma lista de conteúdo.

   Esta lista contém dois tipos de elementos : um para armazenar dados de cada nível de máscara (poli, difusão, contato, etc), que chamaremos de elemento E1 na figura 9, e um outro para armazenar informações de cada uma das macro-células requisitadas para compor o "lay-out", que chamaremos de elemento E2. Estas informações estão a um nível um pouco mais baixo se comparadas àquelas contidas nos elementos da lista de células. Elas se referem às transformações geométricas aplicadas à macro-célula (ver ítem 2.2.4).

3. Lista de Coordenadas

   Cada elemento E1 da lista de conteúdo, elemento este que representa um nível de máscara, aponta para uma lista de coordenadas.

   A lista de coordenadas armazena todos os vértices das figuras que compõem os níveis de máscara (ver ítem 2.2.5).

2.2.2 APONTADORES

A lista de células está ligada à lista de conteúdo através do apontador INFORME.

A lista de conteúdo, por sua vez, está ligada ou à lista de coordenadas, através de apontador PTO, ou à lista de conteúdo de uma outra célula, através do apontador CONTENTS. Isto acontece porque a lista de conteúdo contém dois tipos distintos de informação: um que está relacionado com os níveis de máscara da célula e o outro que está relacionado com as chamadas das células. O apontador PTO aponta para a lista de coordenadas que contém os vértices das figuras, enquanto que o apontador CONTENTS aponta para uma outra lista de conteúdo que fornece a descrição da célula.

As ligações entre elementos da lista de células são feitas através dos apontadores NEXT e LAST.

As ligações entre elementos da lista de conteúdo são feitas através dos apontadores PROXNIVEL e BACK.

As ligações entre os elementos da lista de coordenadas são feitas através dos apontadores PROXIMO e ATRAS.

Estes apontadores permitem que as listas sejam percorridas para frente e para trás, respectivamente.

Ver figura 9.

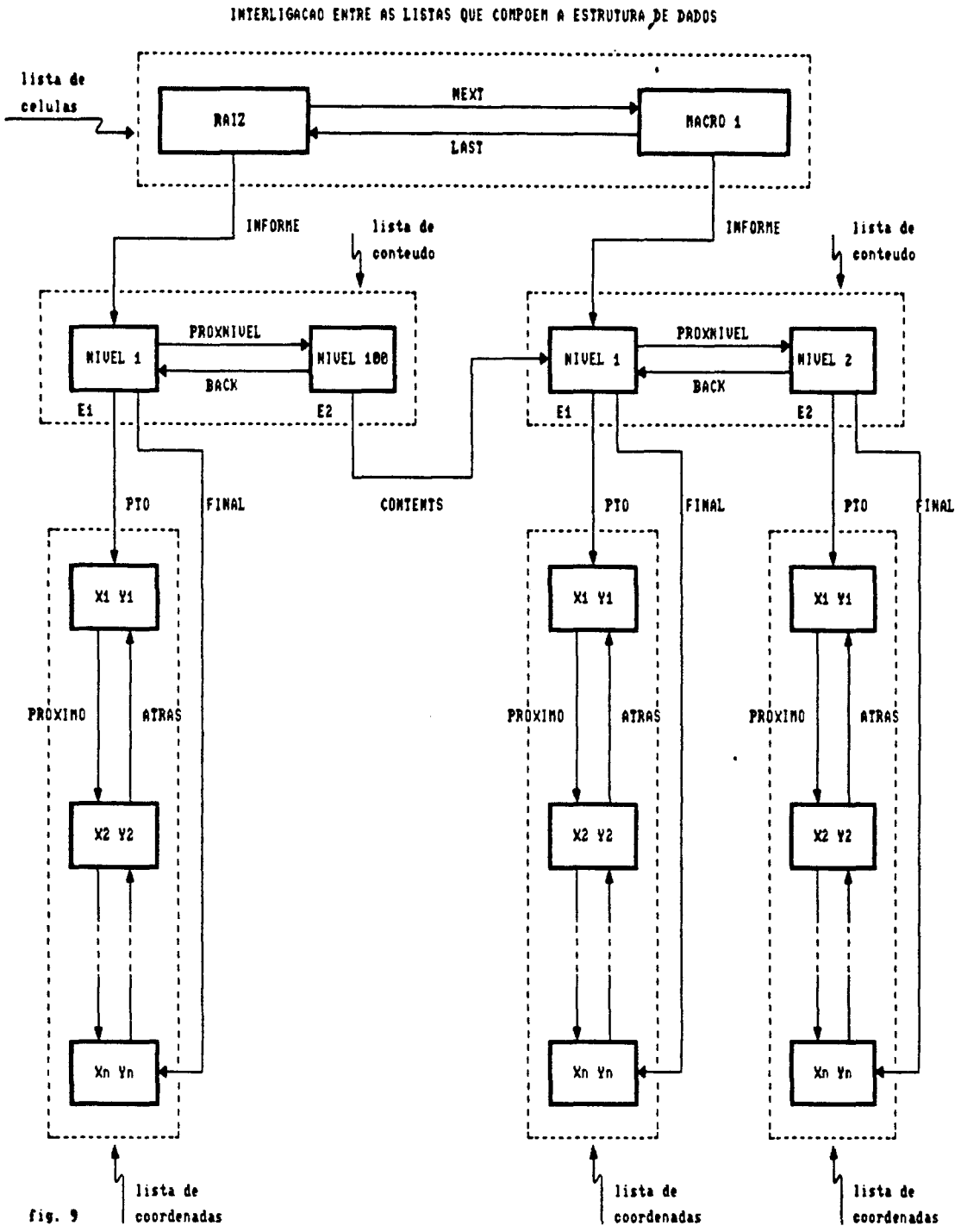
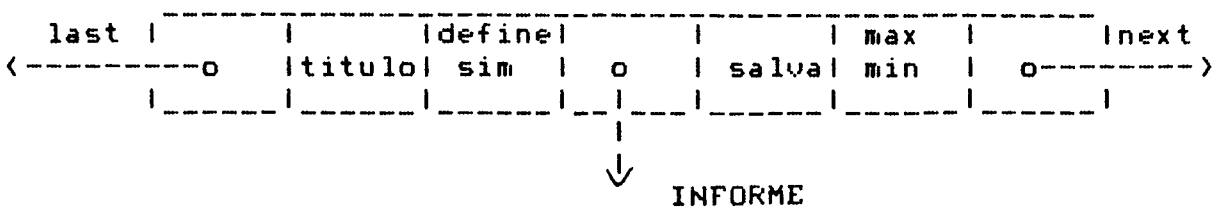

INTERLIGACAO ENTRE AS LISTAS QUE COMPOEM A ESTRUTURA DE DADOS
lista de celulas
RAIZ
NEXT
LAST
MACRO 1
INFORME
lista de conteudo
INFORME
lista de conteudo
NIVEL 1
PROXNIVEL
BACK
NIVEL 100
E1
E2
NIVEL 1
PROXNIVEL
BACK
NIVEL 2
E1
E2
PTO
FINAL
CONTENTS
PTO
FINAL
PTO
FINAL
X1 Y1
PROXIMO
ATRAS
X2 Y2
Xn Yn
X1 Y1
PROXIMO
ATRAS
X2 Y2
Xn Yn
X1 Y1
PROXIMO
ATRAS
X2 Y2
Xn Yn
lista de coordenadas
lista de coordenadas
lista de coordenadas

fig. 9

2.2.3 LISTA DE CÉLULAS



A lista de células contém elementos formados por 7 campos. Dois deles são apontadores para elementos desta mesma lista, um apontando para o elemento anterior e o outro para o elemento seguinte (LAST e NEXT).

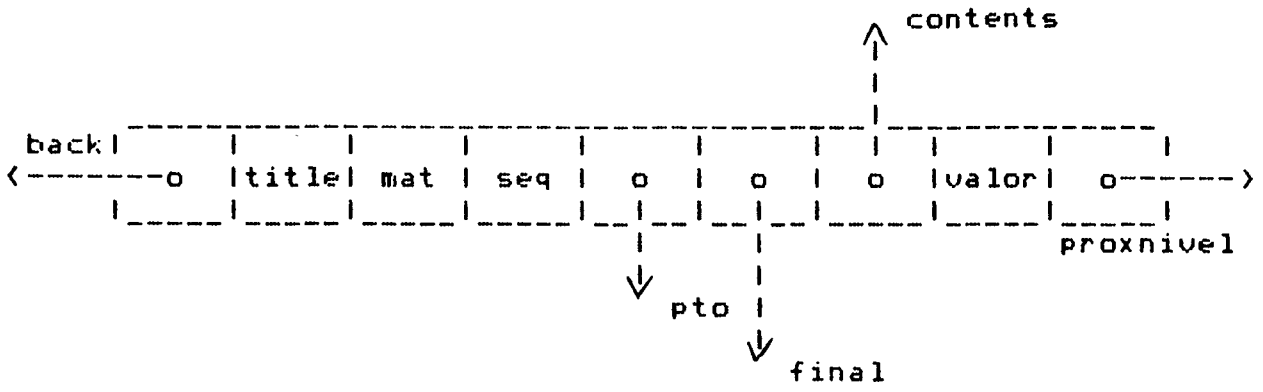
Existe ainda, um terceiro apontador, só que este aponta para a lista de conteúdo (INFORME). É a partir dele que se obtém todo o conteúdo da célula, sejam níveis de máscara e/ou macro-células.

Os outros 4 campos guardam o nome da célula, o número atribuído à célula quando esta é descrita em CIF, os valores máximos e mínimos da célula e ainda uma variável condicional que permite que uma macro-célula, embora chamada várias vezes para compor o "lay-out", seja salva apenas uma vez.

A estrutura da lista é declarada como sendo:

```
ponteiro =  ^ células;

células  =    record

              titulo : texto;

              definesim : integer;

              salva : boolean;

              next, last : ponteiro;

              informe : pointer;

              maxmin : vetor4;
end;
```

2.2.4 LISTA DE CONTEÚDO



A lista de conteúdo contém elementos formados por 9 campos. Dois destes campos são apontadores para elementos desta mesma lista, um apontando para o elemento anterior e o outro para o elemento seguinte (BACK e PROXNIVEL).

Existem, ainda, três outros apontadores:

1. apontador para a lista de coordenadas que contém os vértices das figuras daquele nível (PTO);

2. apontador para o final da lista de coordenadas facilitando, assim, a tarefa de inserção de figuras na lista (FINAL);

3. apontador para uma outra lista de conteúdo, quando se trata de chamada de macro (CONTENTS).

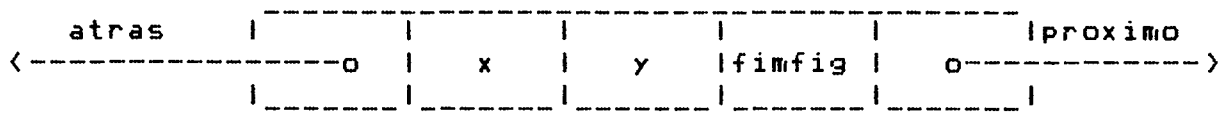
O campo denominado VALOR é utilizado para indicar o nível de máscara. Nele, uma variável numérica é armazenada, uma vez que o programa associa uma variável a cada nível. O fato de atribuir a esta variável um valor maior que 100 significa que o elemento representa uma macro-célula. Valores de 1 a 14 representam os níveis de máscara.

Os campos TITLE, MAT e SEQ são utilizados no caso do elemento representar uma macro. TITLE guarda o nome da macro, MAT guarda a matriz transformação resultante aplicada à macro e SEQ a sequência exata das transformações aplicadas.

A estrutura da lista é declarada como sendo:

```
pointer = ^ niveis;

niveis  =  record

           valor  : integer;

           pto, final : apontador;

           proxnivel, back, contents : pointer;

           mat : mat3;

           seq : string [5];

           title : texto;
end;
```

2.2.5 LISTA DE COORDENADAS



A lista de coordenadas contém elementos formados por 5 campos. Dois campos são apontadores para elementos desta mesma lista, um apontando para o elemento anterior e o outro para o elemento seguinte.

Os campos x e y guardam, respectivamente, a abscissa e a ordenada de um vértice da figura.

O campo FIMFIG guarda uma variável condicional que indica o último vértice de cada figura.

A estrutura da lista é declarada como sendo:

```
apontador =  ^ coordenada;

coordenada =  record

              x, y : integer;

              proximo, atras : apontador;

              fimfig : boolean;
end;
```

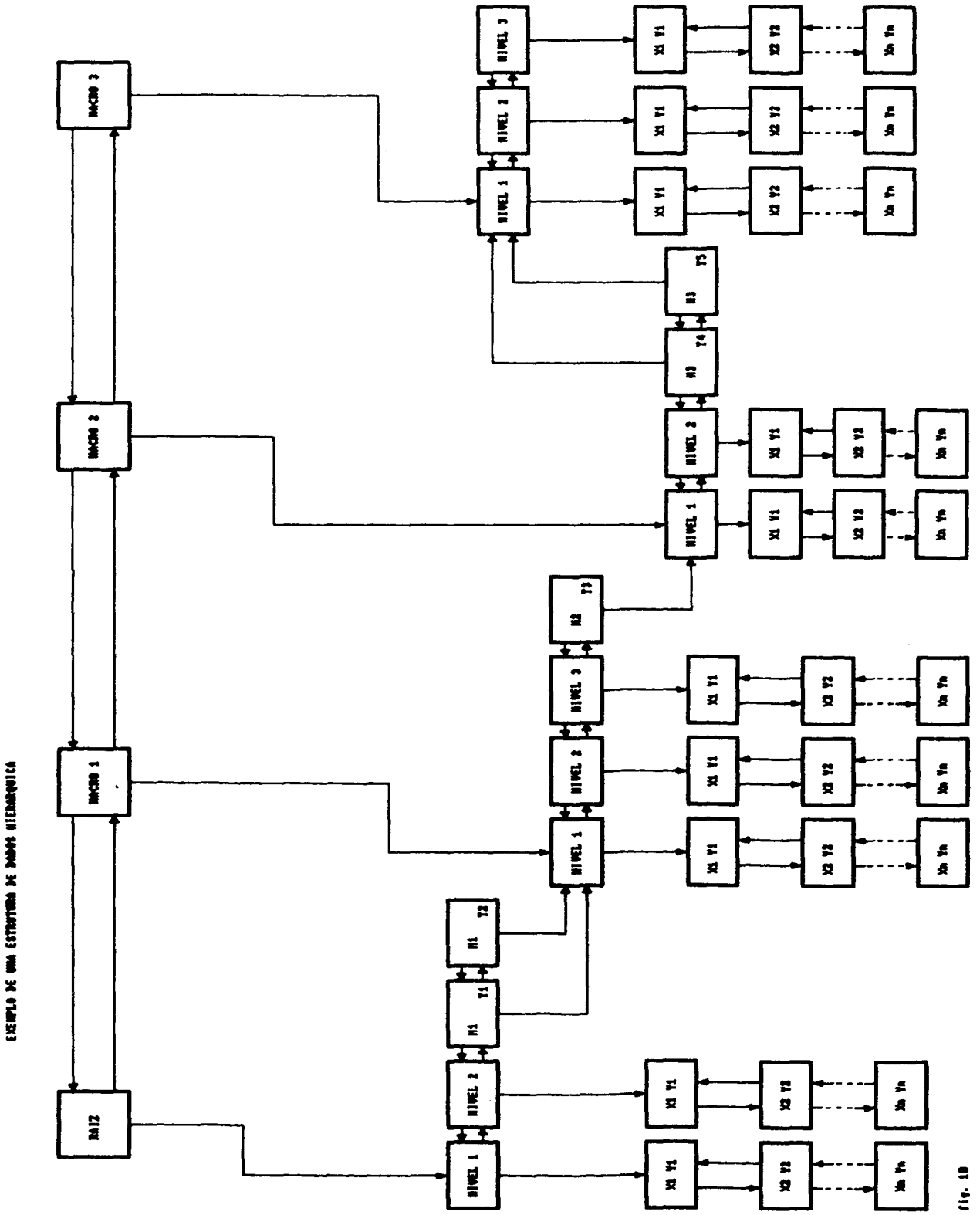

EXEMPLO DE UMA ESTRUTURA DE DADOS HIERÁRQUICA
RAIZ
NÍVEL 1
NÍVEL 2
NÍVEL 3
X1 Y1
X2 Y2
Xn Yn
T1
T2
T3
T4
T5

fig. 10

2.2.6 HIERARQUIA DE CÉLULAS

O "lay-out" de um circuito consiste de geometrias primitivas como retângulos e polígonos e/ou de células previamente criadas (que em última análise também são compostas por geometrias primitivas), que são chamadas segundo uma transformação geométrica - translação, rotação, reflexão e/ou composição destas.

Uma célula pode ser chamada diversas vezes num mesmo "lay-out" através de transformações diferentes. Também, uma célula pode ser composta por outras células, o que nos leva a ver o "lay-out" final como uma composição hierárquica de células.

A chamada de células dentro de um projeto é análoga à chamada de um procedimento num programa principal. Devemos lembrar que um procedimento também pode chamar outros procedimentos. Uma diferença, porém, é que as chamadas de células não podem ser recursivas.

A possibilidade do uso de células dentro de um "lay-out" acelera o projeto, visto que o projetista dispende esforço apenas uma vez na sua definição, para posteriormente alocá-las como "caixa preta" no "lay-out" final.

A nível de estrutura de dados, uma célula está expandida apenas uma vez, quando ocorre a sua primeira chamada. Este fato tráz outras vantagens como, por exemplo, a enorme economia de memoria principal quando o "lay-out" é carregado para manipulação e a diminuição no tempo para execução desta operação. Também, devido ao fato de o arquivo contendo a descrição do "lay-out" (no nosso caso, o CIF) aceitar definição de células (símbolos), há também economia de memória em disco. Estas vantagens são determinantes quando se trabalha com microcomputadores onde a memória principal e a memória de massa são limitadas.

Um exemplo de composição hierárquica de células é dado a seguir nas figuras 11 e 12.

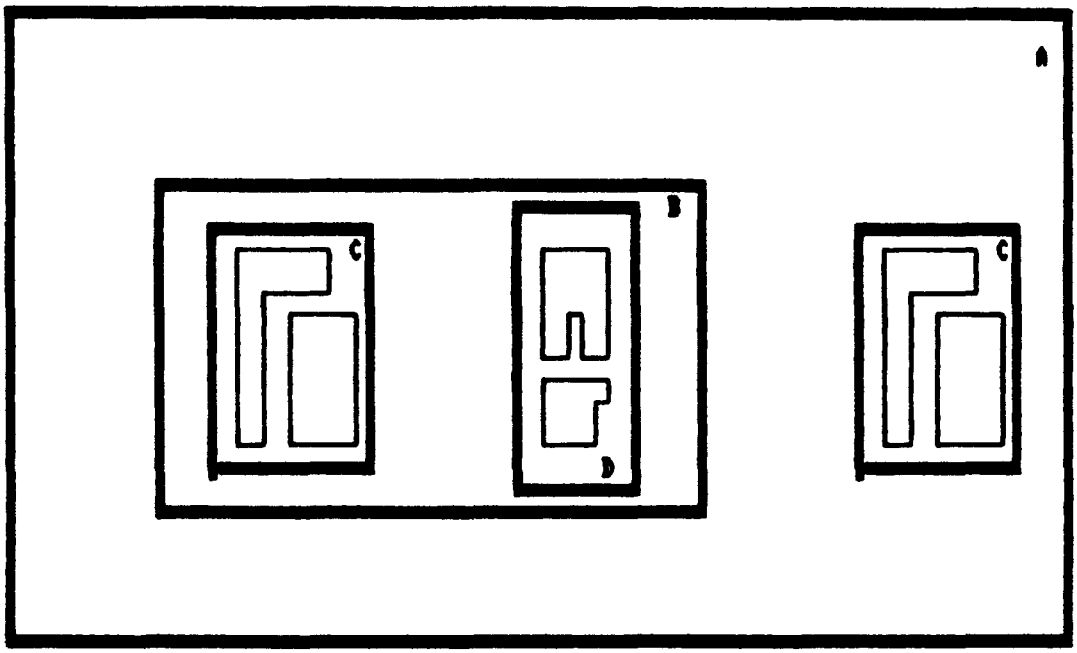
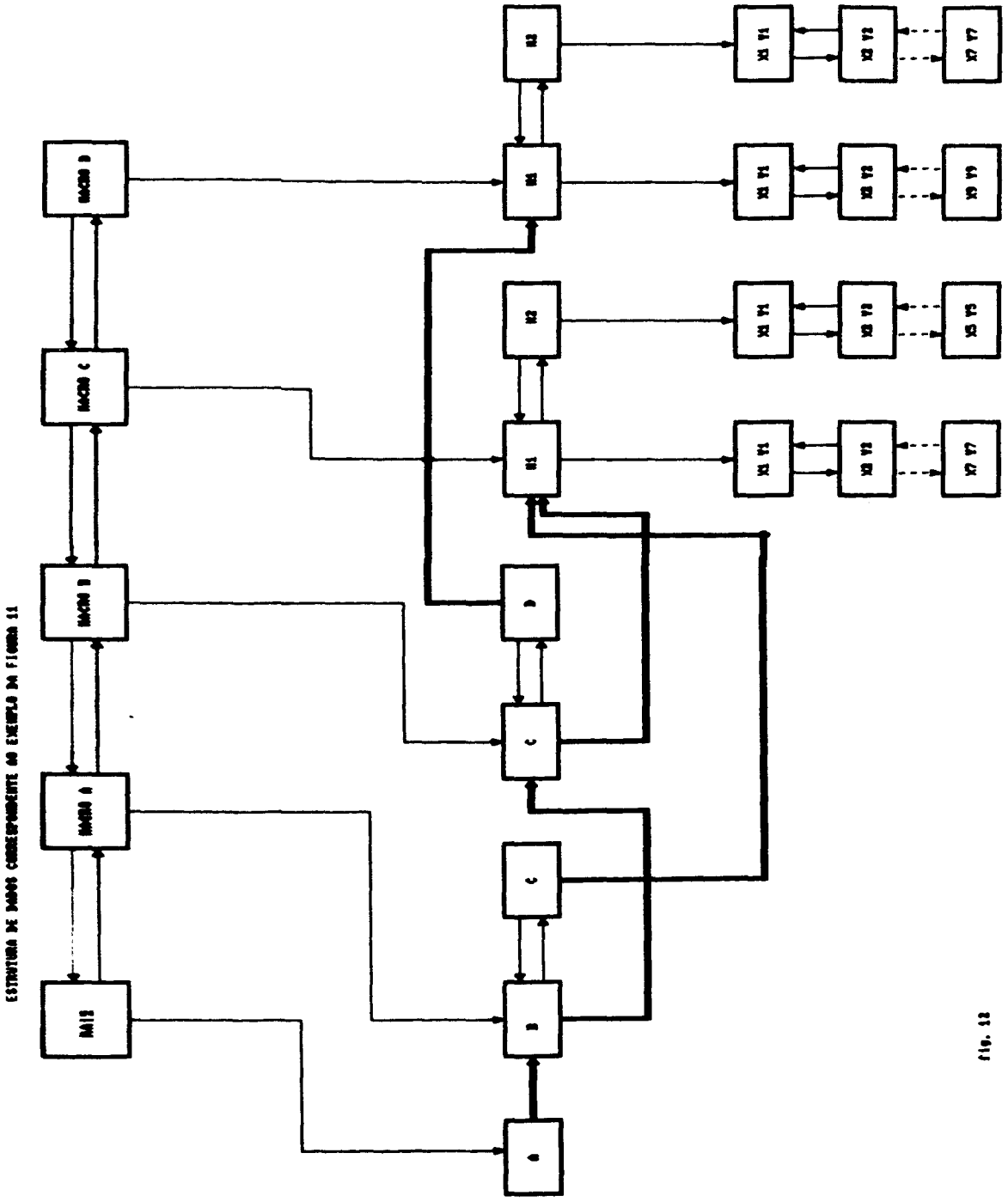


fig. 11 EXEMPLO DE COMPOSICAO HIERARQUICA DE UM "LAY-OUT"



3. PROCEDIMENTOS

Os principais procedimentos do EGIPCIS são descritos a seguir :

3.1 PROCEDIMENTO DE GERAÇÃO DE RETANGULO

PASSO 1 :

Ler a posição para o primeiro ponto da diagonal do retângulo.

PASSO 2 :

Levar o cursor até a posição desejada para o segundo ponto da diagonal do retângulo.

PASSO 3 :

Se esta estiver fora da janela de visualização,

então
- mover o desenho de meia tela.
- voltar ao PASSO 2.

senão
- ler a posição para o segundo ponto da diagonal.

PASSO 4 :

Se o segundo ponto # primeiro ponto,

então
- armazenar os pontos na estrutura de dados.
- desenhar o retângulo a partir destes dois pontos.

senão
- cancelar o retângulo que seria gerado.

3.2 PROCEDIMENTO DE GERAÇÃO DE POLÍGONO

PASSO 1:

Ler a posição para o primeiro vértice V1 do polígono.

PASSO 2 :

Levar o cursor até a posição desejada para o próximo vértice Vi.

PASSO 3 :

Se este estiver fora da janela de visualização,

então
- mover o desenho de meia tela.
- voltar ao PASSO 2.

senão
- ler a posição para o vértice Vi.

PASSO 4 :

Se Vi = Vi-1 e Vi # V1,

então
- terminar a geração de um polígono aberto.

Se Vi = Vi-2,

então
- cancelar a aresta Vi-1,Vi-2.
- voltar ao PASSO 2.

Se Vi = V1 e i > 2,

então
- desenhar aresta Vi,Vi-1.
- terminar a geração de um polígono fechado.

Se Vi = V1 e i = 2,

então
- cancelar o polígono que começava a ser gerado.

3.3 PROCEDIMENTO DE REMOÇÃO DE FIGURA

PASSO 1 :

Receber a posição do vértice que identifica a figura F a ser removida.

PASSO 2 :

Desenhar em linhas pontilhadas a figura F indicada para a remoção.

PASSO 3 :

Se não for confirmada pelo usuário a remoção da figura F, então redesenhar a figura com linhas cheias

senão
- apagar a figura da tela
- apagar a figura da estrutura de dados

Se a figura F for a última da lista de figuras,

então
- se houver figuras anteriores a ela

então (figura F é a última figura mas não a primeira)
- a figura que precede F passará a ser a última e F será apagada

senão (figura F é única na lista)
- a figura F será removida e a lista passará a ser uma lista vazia.

senão
- se a figura F for a primeira da lista

então (figura F é a primeira da lista mas não a última)
- a primeira figura da lista de figuras passará a ser aquela que sucede F e F será apagada da lista

senão (figura F se encontra no meio da lista)
- a figura que precede F será ligada com a figura que sucede F.

3.4 PROCEDIMENTO DE TRANSLAÇÃO DE FIGURA

PASSO 1 :

Receber a posição do vértice que identifica a figura a ser transladada.

PASSO 2 :

Receber a nova posição para a figura indicada.

PASSO 3 :

Desenhar a figura na posição estabelecida no PASSO 2.

PASSO 4 :

Se for confirmada pelo usuário a translação da figura,

então
- atualizar a estrutura de dados,trocando as posições de todos os vértices da figura original pelas posições dos vértices da figura transladada

senão
- redesenhar a figura na posição original.

3.5 PROCEDIMENTO DE REPETIÇÃO DE FIGURA

PASSO 1 :

Receber a posição do vértice que identifica a figura a ser repetida.

PASSO 2 :

Receber a posição para a repetição da figura indicada.

PASSO 3 :

Desenhar a cópia da figura na posição estabelecida no PASSO 2.

PASSO 4 :

Se for confirmada pelo usuário repetição da figura,

então
. atualizar a estrutura de dados, acrescentando, no final da lista de figuras, a nova figura obtida através da repetição

senão
. apagar da tela a figura repetida.

3.6 PROCEDIMENTO DE DEFORMAÇÃO DE FIGURA

PASSO 1 :

Receber o vértice da figura que será deformada.

PASSO 2 :

Receber a nova posição para este vértice.

PASSO 3 :

Desenhar a figura deformada.

PASSO 4 :

Se não for confirmada pelo usuário a deformação da figura,

então
. apagar a figura deformada e redesenhar a figura original

senão
. atualizar a estrutura de dados, trocando a posição original do vértice em questão com a posição atual (após a deformação).

3.7 PROCEDIMENTO DE ROTAÇÃO DA MACRO

PASSO 1 :

Armazenar, no vetor T, o tipo de transformação requisitada, no caso a rotação.

PASSO 2 :

Ler o ângulo A desejado para a rotação.

PASSO 3 :

Se o ângulo A = 270 graus então seno (A) = -1, cos (A) = 0;

Se o ângulo A = 90 graus então seno (A) = 1 , cos (A) = 0;

Se o ângulo A # 90 graus e ângulo A # 270 graus

então
- assumir ângulo A = 180 graus e sen (A) = 0, cos (A) = -1.

PASSO 4 :

Armazenar o valor do ângulo A no vetor T.

PASSO 5 :

Criar a matriz rotação OMAT, onde

OMAT(1,1) = cos (A) , OMAT(1,2) = sen (A)
OMAT(2,1) = -sen (A) , OMAT(2,2) = cos (A)

PASSO 6 :

Criar a matriz resultante, onde

matriz resultante = matriz acumulada * matriz rotação.

Se esta for a primeira transformação aplicada à célula,

então
. matriz acumulada = matriz identidade

senão
. matriz acumulada = matriz resultante das transformações anteriores

3.8 PROCEDIMENTO DE REFLEXÃO DA MACRO

PASSO 1 :

Ler qual o eixo desejado para reflexão.

PASSO 2 :

Armazenar, no vetor T, o tipo de transformação requisitada, no caso a reflexão em torno do eixo x ou do eixo y.

PASSO 3 :

Se eixo = x então ry = -1 , rx = 1

Se eixo = y então rx = -1 , ry = 1

PASSO 4 :

Criar a matriz reflexão FMAT, onde

FMAT(1,1) = rx
FMAT(2,2) = ry

PASSO 5 :

Criar a matriz resultante, onde

matriz resultante = matriz acumulada * matriz reflexão.

Se esta for a primeira transformação aplicada à célula,

então
. matriz acumulada = matriz identidade

senão
. matriz acumulada = matriz resultante das transformações resultantes

3.9 PROCEDIMENTO DE ALOCAÇMO DA MACRO

PASSO 1 :

Armazenar, no vetor T, o tipo de transformação requisitada, no caso a translação.

PASSO 2 :

Ler a posição (x,y) desejada para a translação, posição esta obtida através do teclado ou através do cursor.

PASSO 3 :

Criar a matriz translação TMAT, onde

TMAT(3,1) = x
TMAT(3,2) = y

PASSO 4 :

Criar a matriz resultante, onde

matriz resultante = matriz acumulada * matriz translação.

Se esta for a primeira transformação aplicada à célula,

então
. matriz acumulada = matriz identidade

senão
. matriz acumulada = matriz resultante das transformações resultantes

3.10 PROCEDIMENTO DE REPETIÇÃO DA MACRO

PASSO 1 :

Ler o número de repetições em x (repx) e o espaçamento em x.

Ler o número de repetições em y (repy) e o espaçamento em y.

Ler a posição inicial (px0,py0) via teclado ou através do cursor.

PASSO 2 :

Para cada y variando de 1 a repy, fazer :

PASSO 2a.

Se y = py0

então
. posy = py0

senão
. posy = posy + altura célula + espaçamento y

PASSO 2b.

Para cada x variando de 1 a repx, fazer :

. se x = px0

então
. posx = px0

senão
. posx = posx + comprimento célula + espaçamento x

. Criar matriz translação TMAT, onde

TMAT(3,1) = posx
TMAT(3,2) = posy

. Criar a matriz resultante, onde

matriz resultante = matriz acumulada * matriz translação.

Se a célula a ser repetida não tiver sofrido nenhuma transformação anteriormente,

então
. matriz acumulada = matriz identidade

senão
. matriz acumulada = matriz resultante das transformações anteriores.

. Chamar o procedimento de atualização de estrutura de dados e desenho.

3.11 PROCEDIMENTO DE ATUALIZAÇÃO DE ESTRUTURA DE DADOS E DESENHO

PASSO 1 :

Se a célula não estiver ainda armazenada na estrutura de dados,

então
. ler o arquivo que contém a descrição da célula .
. armazenar a célula na estrutura de dados.

PASSO 2 :

Atualizar a estrutura de dados, armazenando as informações relativas às transformações aplicadas à célula.

PASSO 3 :

Chamar o procedimento para desenhar a célula ou simplesmente o seu contorno, caso o programa não esteja no modo expansão.

PASSO 4 :

Zerar o vetor T que armazena as transformações aplicadas à célula.

3.12 PROCEDIMENTO DE CANCELAMENTO DA ÚLTIMA OPERAÇÃO

PASSO 1 :

Se for confirmado o cancelamento da última operação feita na célula,

então
- apagar a última macro-célula da tela.
- apagar a chamada da macro-célula da estrutura de dados.

3.13 PROCEDIMENTO DE RECORTE DE LINHAS

Como já foi visto, a janela de visualização exibe apenas uma pequena porção do "lay-out" todo. Assim, existem algumas linhas que compõem as figuras que se encontram parcialmente dentro da janela de visualização e parcialmente fora.

Existem outras, ainda, que se encontram totalmente fora da janela de visualização.

A exibição de cada uma destas linhas por inteiro pode causar não só uma demora significativa na exibição do "lay-out" como também pode gerar anomalias, caso ocorra rebatimento de linhas.

Também não seria solução deixar simplesmente de exibi-las, pois a imagem se tornaria incorreta.

Uma maneira de selecionar as partes visíveis das figuras para exibição é utilizar o algoritmo de recorte que divide cada linha em uma parte visível, dentro da janela de visualização, e uma parte invisível, fora da janela de visualização. As partes invisíveis devem ser descartadas.

O algoritmo de recorte de linha desenvolvido por Dan Cohen e Ivan Sutherland [03] foi projetado para descartar rapidamente todas as linhas invisíveis. Isto é de grande valia no caso de "lay-outs" bem maiores que a janela de visualização, onde um grande número de linhas já é colocado à parte.

Este algoritmo de recorte é composto de três testes. O primeiro verifica se a linha se encontra totalmente dentro da janela de visualização. Caso isto não ocorra, é feito o segundo teste que verifica se a linha se encontra, então, totalmente fora da janela. Se esta possibilidade também é descartada, fica a conclusão de que a linha intercepta pelo menos uma das bordas da janela. Resta, somente, calcular o(s) ponto(s) de intersecção da linha com a(s) borda(s) e remover a(s) porção(ões) da linha que se encontra(m) fora da janela.

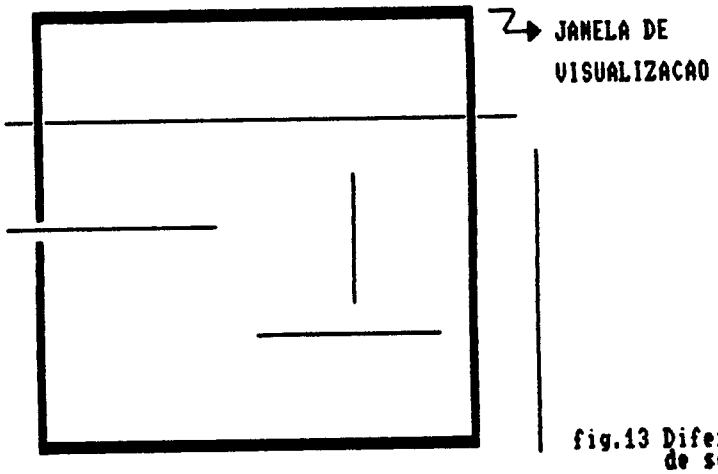


fig.13 Diferentes possibilidades de segmentos de linha

O algoritmo de recorte é dado a seguir:

PASSO 1 :

Verificar em que regiões os extremos da linha se encontram.

topo esq	topo	topo dir
esq	janela de visualiza-ção	dir
esq base	base	dir base

PASSO 2 :

Se os dois extremos da linha estiverem dentro da janela de visualização,

então
. ir para o PASSO 4

senão
. se os 2 extremos se encontram no topo da janela OU
os 2 extremos se encontram na base da janela OU
os 2 extremos se encontram à direita da janela OU
os 2 extremos se encontram à esquerda da janela,

então
. descartar a linha saindo do procedimento

senão
. para cada extremo fazer :

Se o extremo estiver à esquerda da janela,

então
. calcular o ponto da linha que intercepta a borda esquerda

senão
. se o extremo estiver à direita da janela,

então
. calcular o ponto da linha que intercepta a borda direita

senão
. se o extremo estiver no topo da janela,

então
. calcular o ponto da linha que intercepta a borda acima da janela

senão
. se o extremo estiver na base da janela,

então
. calcular o ponto da linha que intercepta a borda abaixo da janela.

PASSO 3 :

Se o novo ponto gerado permanece fora da janela de visualização,

então
. voltar ao PASSO 2

PASSO 4 :

Desenhar o segmento visível da linha.

3.14 PROCEDIMENTO DE REPRESENTAÇÃO DE ÁREAS

Uma vez que o EGIPCIS está adaptado para monitor de vídeo monocromático, todas as figuras são geradas de uma mesma cor independente do nível de máscara que representam.

A diferenciação entre os níveis se torna algo difícil, cabendo a identificação do "lay-out" àqueles que o geraram ou às pessoas experientes no assunto.

É certo que a opção de visualização seletiva dos níveis fornecida pelo programa é de grande auxílio ao usuário, já que os níveis podem ser visualizados separadamente, em suas dimensões reais ou mesmo normalizados.

Na tentativa de reparar a falta de cores, que seria o ideal na distinção dos níveis de máscara, o EGIPCIS trabalha com representação de áreas não apenas pelos seus contornos mas também pelos seus interiores.

O interior de uma área pode ser representada por:

1. Hachurados

   O interior da área é preenchido por linhas horizontais, verticais, diagonais ou cruzados.

2. Sólidos

   O interior da área é totalmente preenchido.

3. Texturados

   O interior da área é preenchido por padrões.

O segundo tipo de representação - sólidos, não tem sentido de ser utilizado, pois permanece a não distinção dos níveis.

O EGIPCIS faz o preenchimento de regiões através de hachurados e texturados, restringindo-se mais aos texturados uma vez que as hachuras, na maioria das vezes, dificultam a visualização do desenho. O programa associa cada nível de máscara a um tipo de preenchimento, conforme mostra a figura 14.

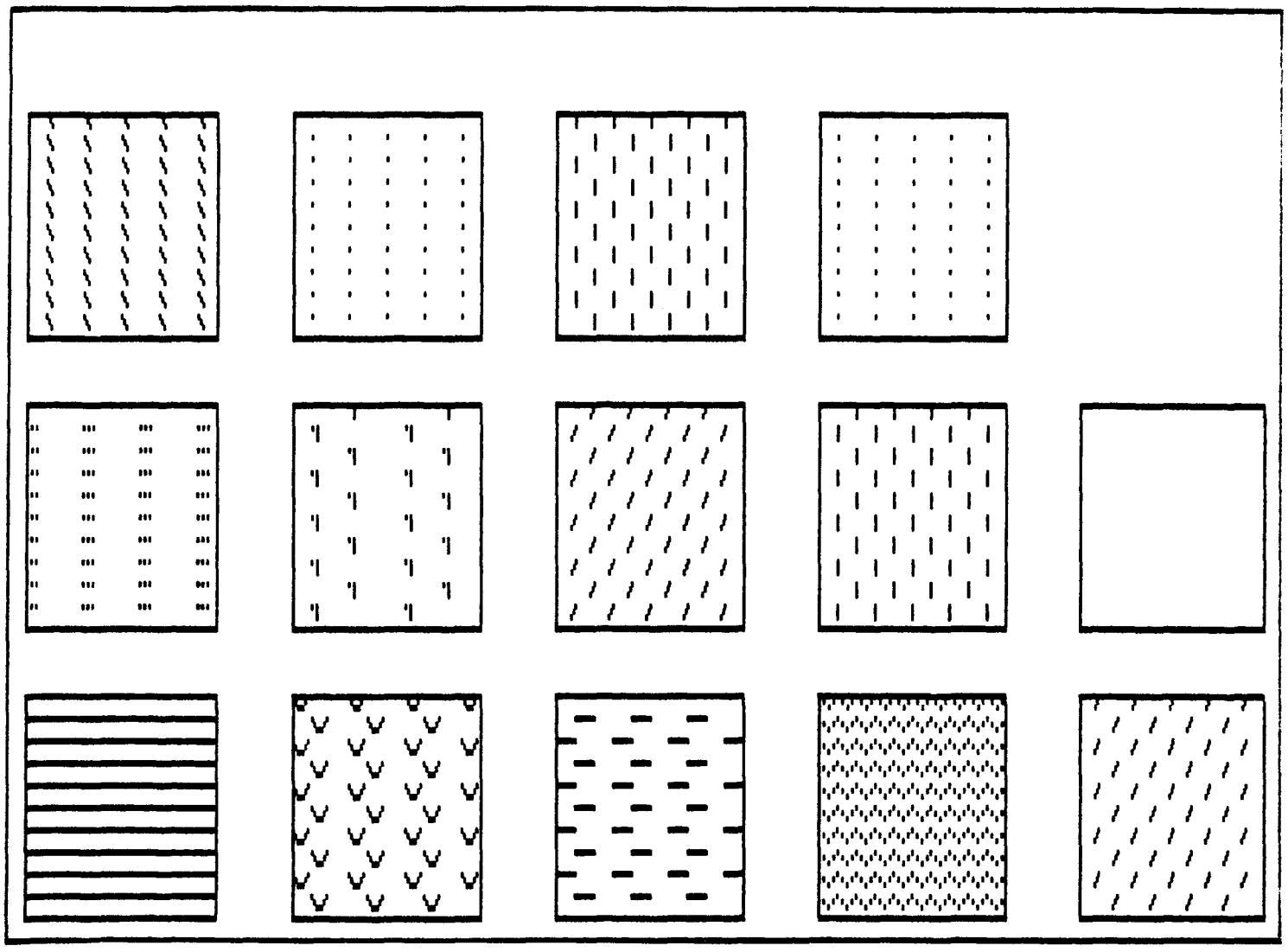


fig. 14 - Tipos de preenchimento

O algoritmo de preenchimento de áreas utilizado neste trabalho [17] parte do princípio de que cada linha horizontal (linha de hachurado) cruza a borda do polígono em um número par de pontos. Se tomarmos estes pontos de cruzamento aos pares, podemos afirmar que os segmentos que unem os pontos de cada par estão contidos no interior da área.

É dado a seguir o procedimento utilizado para esta tarefa :

PASSO 1 :

Computar os segmentos do hachurado em ordem decrescente, ou seja, da maior ordenada y para a menor ordenada y.

PASSO 2 :

Manter, a cada ordenada y de uma linha de hachurado, a lista de arestas do polígono que é interceptada pela linha de hachurado corrente.

PASSO 3 :

Classificar a lista em ordem crescente de abscissa. Esta lista fornece os pontos de intersecção da linha de hachurado com as arestas do polígono que, tratados aos pares, gerará os segmentos do hachurado.

PASSO 4 :

Traçar os segmentos do hachurado e passar à linha de hachurado imediatamente abaixo, atualizando a lista de arestas.

PASSO 5 :

Remover aquelas arestas que não mais interceptam a linha de hachurado corrente e inserir na lista as novas arestas.

PASSO 6 :

Se a lista de aresta se tornar vazia,

então
. a tarefa está terminada

senão
. voltar ao PASSO 3.

3.15 PROCEDIMENTO PARA DESENHAR O "LAY-OUT" TODO

PASSO 1 :

Se a estrutura de dados não estiver vazia

então
. se a raiz não estiver vazia

então
. se o programa estiver em modo expansão

então
. chamar procedimento para desenhar célula

senão
. chamar procedimento para desenhar contorno da célula

3.16 PROCEDIMENTO PARA DESENHAR CÉLULA

PASSO 1 :

Fazer para cada elemento da lista de conteúdo :

se o elemento representar uma macro

então
. entrar na macro em questão
. verificar as transformações a serem aplicadas
. chamar recursivamente o procedimento para desenhar célula, ou seja, voltar ao início do PASSO 1.

senão (o elemento representa, então, um nível de máscara)
. chamar procedimento para desenhar o nível

3.17 PROCEDIMENTO PARA DESENHAR NÍVEL

PASSO 1 :

Se o nível de máscara não estiver vazio

então
fazer para cada figura :
. aplicar as devidas transformações
. chamar procedimento de recorte para cada aresta de figura
. se o programa estiver em modo hachura

então
. chamar o procedimento de preenchimento de figuras

3.18 PROCEDIMENTO PARA DESENHAR CONTORNO DA CÉLULA

PASSO 1:

Fazer para cada elemento da lista de conteudo :

PASSO 1.a Se o elemento representar uma macro

então
. entrar na macro em questão
. verificar as transformações a serem aplicadas
. chamar recursivamente o procedimento para desenhar contorno da célula, ou seja, voltar ao início do PASSO 1.

senão (o elemento representa, então, um nível máscara)

se estivermos na raiz

então
. chamar procedimento para desenhar nível

senão
. calcular os extremos da macro utilizados para desenhar seu contorno

PASSO 1.b Em se tratando de uma macro-célula, exibir o seu contorno

3.19 PROCEDIMENTO PARA APAGAR UM NIVEL

PASSO 1 :

Se a estrutura de dados não estiver vazia,

então
- percorre-la a procura do nível desejado

PASSO 2 :

Se o nível for encontrado,

então
- zerar o apontador para a lista que contém as figuras daquele nível
- zerar o apontador para o final desta mesma lista
- apagar o nível da tela

3.20 PROCEDIMENTO PARA EXTRAÇÃO DE PARAMETROS

PASSO 1 :

Ler o nome do dispositivo dado pelo usuário.

PASSO 2 :

Exibir o "menu" contendo os possíveis cálculos a serem efetuados: comprimento e largura de canal, área de fonte e dreno, perímetro de fonte e dreno.

PASSO 3 :

Caso seja requisitado para cálculo:

- Comprimento ou largura de canal,

  então
  - chamar procedimento para medir distância entre 2 pontos

. Área de fonte ou área de dreno,

então
. chamar procedimento para medir área de um polígono

. Perímetro de fonte ou perímetro de dreno,

então
. chamar procedimento para medir perímetro de um polígono

3.21 PROCEDIMENTO PARA MEDIR DISTANCIA ENTRE 2 PONTOS

PASSO 1 :

distância = 0

PASSO 2 :

Ler as posições dos 2 pontos escolhidos (Xa, Ya) e (Xb, Yb).

PASSO 3 :

Se Xa # Xb,

então
. distância = distância + (Xb - Xa)

senão
. distância = distância + (Yb - Ya)

PASSO 4 :

Se mais trechos forem solicitados para cálculo,

então
. voltar ao PASSO 2.

(Isto ocorre quando a soma de comprimentos/larguras é desejada)

3.22 PROCEDIMENTO PARA MEDIR ÁREA DE UM POLÍGONO

PASSO 1 :

área = 0

PASSO 2 :

Ler as posições de 2 vértices opostos (Xa, Ya) e (Xb, Yb).

(Parte-se do princípio que a área de um polígono é igual a soma das áreas dos retângulos que o compõe).

PASSO 3 :

Hachurar a área a ser medida.

PASSO 4 :

área = área + (Xb - Xa) * (Yb - Ya)

PASSO 5 :

Se mais áreas forem solicitadas para cálculo,

então
. voltar ao PASSO 2

3.23 PROCEDIMENTO PARA MEDIR PERÍMETRO DE UM POLÍGONO

PASSO 1 :

perímetro = 0

PASSO 2 :

Ler a posição do primeiro vértice do polígono.

PASSO 3 :

Ler a posição do próximo vértice e calcular a distância (dist) entre ele a o vértice anterior.

PASSO 4 :

perímetro = perímetro + dist

PASSO 5 :

Se a posição do último vértice lido for igual a do primeiro vértice,

então
. fim de cálculo de perímetro

senão
. voltar ao PASSO 3

CAPÍTULO 5

TESTES DE USUÁRIO FINAL

V. TESTES DE USUARIO FINAL

1. INTRODUÇÃO

Os testes de campo do EGIPCIS foram realizados durante as II e III EBAI em 1987 e 1988, quando foi intensamente utilizado no decorrer do Laboratório de Microeletrônica. Esses testes contribuíam para a depuração do programa e otimização da relação usuário/ferramenta, como consequência da interação com os alunos.

2. RESULTADOS DO TESTE DA VERSÃO 1.0 - II EBAI

A principal deficiência apontada dizia respeito à ocorrência de rebatimento de linhas durante a exibição de um "lay-out" muito grande e que não tinha sido detetada nos primeiros testes, uma vez que estes foram realizados em células relativamente simples.

Vários testes foram realizados na tentativa de diagnosticar a causa desta anomalia, mas a mesma não foi esclarecida.

Existe a possibilidade do problema ter origem no procedimento DRAW embutido no compilador. Talvez o limite interno especificado para números inteiros, ao qual não se tem acesso, seja menor que aquele definido pelo Pascal Turbo.

O problema foi solucionado por meio de alteração do programa. Um algoritmo de recorte de linhas foi implementado (ver cap. IV, item 3.13), limitando a exibição do "lay-out" apenas ao trecho visível da tela.

Os alunos identificaram algumas dificuldades no uso da ferramenta para a manipulação do "lay-out", as quais puderam ser reduzidas através da implementação de novas opções como expansão/não expansão de macro, janela, preenchimento de figuras, translação de figuras e alocação de macro via teclado.

. Expansão/não expansão de macros

A necessidade de um comando de expansão/não expansão de macros evidenciou-se durante a fase de montagem da pastilha.

Com isto, permitiu-se ao usuário trabalhar ou não com todos os detalhes das macros. (Ver capítulo III, item 3.1.13).

O fato de o usuário poder optar apenas pela exibição do contorno das macros é uma vantagem, devido ao tempo consideravelmente menor se comparado com o da exibição detalhada das macros, o que aumenta a eficiência da ferramenta.

. Janela

A necessidade do comando JANELA decorre da ineficiência de operação causada pela aplicação de deslocamentos sucessivos em busca de um determinado trecho do "lay-out" (ver cap. III, item 3.1.5).

O comando JANELA os substitui por meio de uma operação única.

. Preenchimento de figuras

Em configurações mais complexas, os diversos níveis de máscara se tornaram indistinguíveis na tela.

Uma maneira de distinguí-los foi preencher as áreas com padrões específicos, já que o monitor de vídeo utilizado é monocromático (ver cap. III - item 3.1.3, cap. IV - item 3.1.14).

. Translação de figuras

A posição de uma determinada figura é modificada (ver cap. III - item 3.1.16.7).

. Alocação via teclado

A alocação de uma macro através do posicionamento do cursor acarretava uma demora muito grande, pois eram necessários deslocamentos sucessivos do "lay-out" até encontrar a posição desejada (nesta versão ainda não existia a opção JANELA).

Uma solução viável foi a de permitir a alocação através do teclado, ou seja, o usuário digita a posição desejada para a macro. Isto é válido tanto para a opção ALOCA como para a opção MATRIZ existentes no menu de montagem (ver cap. II - ítens 3.1.16.11.3 e 3.1.16.11.5).

Outras modificações foram efetuadas com a finalidade de melhorar o desempenho do programa :

. Cada nível de máscara é representado por um código e uma mudança no mesmo tornou-se possível durante a execução do programa.

Com a chamada da função "nivel", a tabela dos códigos dos níveis era apresentada na tela e com ela a opção para a mudança de código através da tecla "M".

Uma vez solicitada a mudança, o usuário recebia permissão para percorrer a tabela em busca do código (ou mesmo códigos) a sofrer alteração.

Desta maneira, o nível de contato representado pelo código CC, passava a ser representado, por exemplo, por CC1.

. A montagem do "lay-out" passou a ser realizada em uma única tela, permitindo um acompanhamento visual por parte do usuário.

. A estrutura de dados foi modificada para receber o "lay-out" em formato CIF.

Na primeira versão, o arquivo de descrição de "lay-out" era compatível com o arquivo de saida da LPG.

A LPG é uma linguagem de procedimentos gráficos [10] que permite ao usuário descrever o "lay-out" por meio de equações e que, através da mudança de determinados parâmetros, possibilita o redimensionamento das figuras.

A LPG, ao contrário do CIF, não permite hierarquia de células. Portanto, as células chamadas para compor o "lay-out" através do EGIPCIS eram salvas uma a uma no arquivo de descrição, mesmo que repetidas. Isto requeria um espaço muito grande em disco, além da demora que acarretava quando o "lay-out" era carregado na estrutura de dados. Além disto, o formato CIF é o adotado pelo Projeto Multiusuário Brasileiro.

3. RESULTADOS DO TESTE VERSÃO 2.0 - III EBAI

Durante a III EBAI o EGIPCIS foi submetido a nova bateria de testes de campo, que levaram a alterações adicionais no mesmo:

O EGIPCIS havia sido projetado para trabalhar no primeiro quadrante, apenas com coordenadas positivas.

Entretanto, vários alunos tiveram a necessidade de editar figuras e manipular macros em outros quadrantes, gerando coordenadas negativas.

Este fato acarretou erros na leitura do arquivo CIF, já que não se haviam previsto coordenadas negativas.

Com as alterações introduzidas não existem mais restrições quanto ao quadrante utilizado.

Além desta, outras modificações foram efetuadas para melhorar ainda mais o desempenho do programa:

. Os níveis de máscara passaram a ser discriminados em um arquivo à parte, podendo o usuário alterar os seus códigos conforme fosse conveniente.

  Com isto, tornou-se possível o uso deste editor qualquer que fosse a tecnologia adotada (ver cap. III, item 3.1.1).

. Foi criada a opção VISIB, atribuindo, assim, a visibilidade apenas para os níveis desejados (ver cap. III, item 3.1.16.9).

  Esta necessidade tornou-se evidente durante a utilização do EGIPCIS em um curso de pós-graduação realizado na Faculdade de Engenharia Elétrica da UNICAMP.

  O projeto desenvolvido era de tamanho razoavelmente grande (mais ou menos 4.000μ x 4.000μ) e muitas vezes o usuário desejava acompanhar apenas um ou dois níveis.

  Uma vez implementada esta opção, a exibição na tela somente dos níveis requeridos fez-se de maneira muito mais rápida.

VI. CONCLUSÕES GERAIS

1. INTRODUÇÃO

Os testes realizados com o editor gráfico EGIPCIS mostram que o mesmo satisfaz as especificações apresentadas no capítulo III.

Todas as funções do programa foram implementadas de acordo com os requisitos estabelecidos inicialmente, visando sempre um melhor aproveitamento do "hardware" e "software" disponíveis.

Estas funções atingiram as expectativas dos projetistas de circuitos integrados e estudantes, que direta ou indiretamente contribuiram para o planejamento de cada uma delas.

A seguir são feitas comparações entre o editor gráfico EGIPCIS e os quatro editores gráficos descritos no capítulo II - ME, TEDMOS, CAESAR e KIC.

São listadas, também, algumas alterações adicionais desejáveis no programa, que se propõem para uma eventual terceira revisão do mesmo.

2. COMPARAÇÃO ENTRE OS EDITORES GRAFICOS

Em relação aos editores gráficos desenvolvidos em microcomputador - ME e TEDMOS (ver cap. II), o EGIPCIS permite a hierarquização de células. Nele, as células previamente projetadas e testadas são alocadas de maneira conveniente pelo usuário, após sofrerem as devidas transformações(rotação e/ou reflexão).

Já em relação ao CAESAR e KIC (ver cap. II), podemos apresentar um conjunto de vantagens e desvantagens.

Vantagens do EGIPCIS :

. Os recursos mínimos de "hardware" exigidos pelo EGIPCIS são de menor custo, permitindo que um número maior de pessoas possam ter acesso a ele. O uso de tela de maior resolução e a cores permitiria, entretanto, um desempenho melhor em relação ao usuário.

CAPÍTULO 6

CONCLUSÕES GERAIS

. O tempo de redesenho na tela de uma célula é menor no EGIPCIS em relação ao KIC e CAESAR. Esta característica é devido ao fato de , no caso dos microcomputadores, a informação ser transmitida diretamente para a memória de vídeo e, no caso dos computadores ligados a terminais , a informação ser feita através de linhas de comunicação (por ex: RS 232C) com velocidade limitada.

. O fato de o EGIPCIS ser executado em microcomputador permite uma maior portabilidade tanto do próprio programa como das células por ele criadas. Esta característica atende à especificação de um maior intercâmbio principalmente a nível universitário.

Desvantagens do EGIPCIS :

. O número de funções disponíveis para o usuário é menor em relação ao KIC e CAESAR.

. Nestes dois editores são implementados alguns mecanismos para acelerar o redesenho como, por exemplo, a reconstituição parcial do desenho exibido, limitando-se à transmissão somente das regiões onde ocorreram alterações.

Outras vantagens e desvantagens destes dois editores em relação ao EGIPCIS estão mais ligados ao "hardware" a que eles se propõem do que propriamente ao programa. Como, por exemplo, podemos citar que a precisão de um desenho na tela está ligada à precisão do terminal usado. A presença ou não de cor depende do tipo de terminal. Também, a velocidade para se processar certas operações depende do computador onde tal "software" está sendo executado.

Para finalizar, deve-se ressaltar que já existem versões disponíveis do KIC para microcomputador e, futuramente, novas versões do EGIPCIS podem ser implantadas em computadores de maior porte, adaptando seu uso para terminais coloridos como TEKTRONIX, AED, DATANAV etc.

A tabela abaixo mostra as principais funções disponíveis em cada um dos editores até então mencionados :

FUNÇÃO\EDITOR	TEDMOS	ME	EGIPCIS	CAESAR	KIC
grade	-	x	x	x	x
escala	x	x	x	x	x
desloca	-	x	x	x	x
janela	x	x	x	x	x
restaura	-	x	x	x	x
hachura	x	x	x	x	x
salva cif	x	x	x	x	x
visibilidade	x	-	x	x	x
retângulo	x	x	x	x	x
fio	-	-	-	-	x
polígono	-	-	x	-	x
arco	-	-	-	-	x
texto	-	-	-	x	x
chama macro	-	-	x	x	x
expande macro	-	-	x	x	x

FUNÇÃO\EDITOR	TEDMOS	ME	EGIPCIS	CAESAR	KIC
rotação macro	-	-	x	x	x
reflexão macro	-	-	x	x	x
matriz	x	-	x	x	x
cancela oper.	-	-	x	x	x
seleciona nível	x	-	-	x	x
seleciona área	x	-	-	x	x
seleciona macro	-	-	-	x	x
seleciona figura	x	x	x	x	x
extrai parâme-tros para SPICE	-	-	x	-	-

As funções que não estão implementadas no EGIPCIS estão descritas a seguir:

FIO :

Criar um caminho cuja largura não deve ser menor que a dimensão mínima para aquele nível. Esta função é muito útil na criação de ligações entre células (ver item 3.4).

ARCO :

Criar um arco cuja largura não deve ser menor que a dimensão mínima para aquele nível.

TEXTO :

Escrever dentro do "lay-out", se necessário.

SELECIONA ÁREA :

Selecionar uma determinada área a sofrer modificação (ver ítem 3.1).

SELECIONAR NÍVEL :

Selecionar apenas um nível a sofrer modificação dentro de uma área já selecionada (ver ítem 3.1).

SELECIONAR MACRO :

Selecionar uma macro a sofrer modificação (ver ítem 3.2).

3. OTIMIZAÇÕES FUTURAS

O EGIPCIS está aberto a modificações e otimizações. Algumas sugestões são dadas nos itens que se seguem.

3.1 IMPLEMENTAÇÃO DE UM ALGORITMO DE SELEÇÃO DE PARTE DO "LAY-OUT" PARA TRANSLAÇÃO, COPIA OU ELIMINAÇÃO.

Modificar uma parte do "lay-out". Não apenas um polígono por vez como ocorre atualmente, mas um conjunto de polígonos, sejam eles de um mesmo nível ou de níveis diferentes.

Na parte do "lay-out" selecionada não deve constar nenhuma macro ou parte delas. Macros não podem ser modificadas.

O usuário deve fornecer os pontos da diagonal do retângulo que engloba a região desejada para modificação, seja ela translação, cópia ou eliminação.

O programa poderá abordar o problema de duas maneiras :

a) a modificação atingindo somente os polígonos contidos inteiramente no retângulo;

b) a modificação atingindo os polígonos contidos inteiramente no retângulo e também aqueles que interceptam as bordas do retângulo.

Cabe à pessoa que venha a introduzir essa alteração, juntamente com um projetista, decidir qual a melhor opção.

O procedimento a ser implementado segue os passos abaixo:

PASSO 1 :

Receber a modificação desejada : translação, cópia ou eliminação.

PASSO 2 :

Receber os pontos da diagonal do retângulo que engloba o trecho do "lay-out" a ser modificado.

PASSO 3:

Receber um ponto de referência, por exemplo, um dos vértices do retângulo.

PASSO 4 :

Percorrer a estrutura de dados e verificar, nível por nível, quais os polígonos que estão inteiramente contidos no retângulo e, caso a opção b) seja implementada, verificar também quais os polígonos que interceptam as bordas do retângulo.

PASSO 5 :

Se for realizada uma translação,

então

- ler a nova posição para o trecho do "lay-out".
- apagar o trecho do "lay-out" da posição original.
- redesenhá-lo na nova posição indicada, guardando a modificação na estrutura de dados.

Se for realizada uma cópia,

então
- ler a posição para receber esta cópia.
- repetir o trecho do "lay-out" na posição indicada.
- guardar a modificação na estrutura de dados.

Se for realizada uma eliminação,

então
- apagar o trecho do "lay-out" da tela.
- apagar o trecho do "lay-out" da estrutura de dados.

As figuras 1 e 2 mostram, conforme a abordagem adotada, quais figuras do "lay-out" seriam selecionadas para modificação.

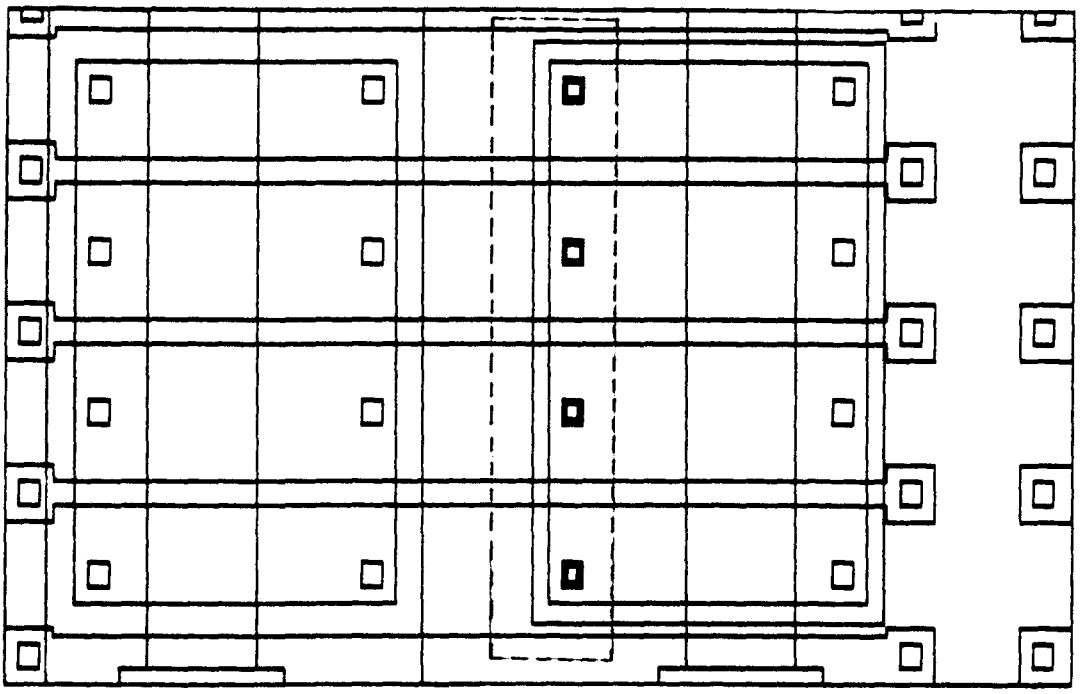


fig. 1 - Modificação atingindo somente os polígonos contidos inteiramente no retângulo.

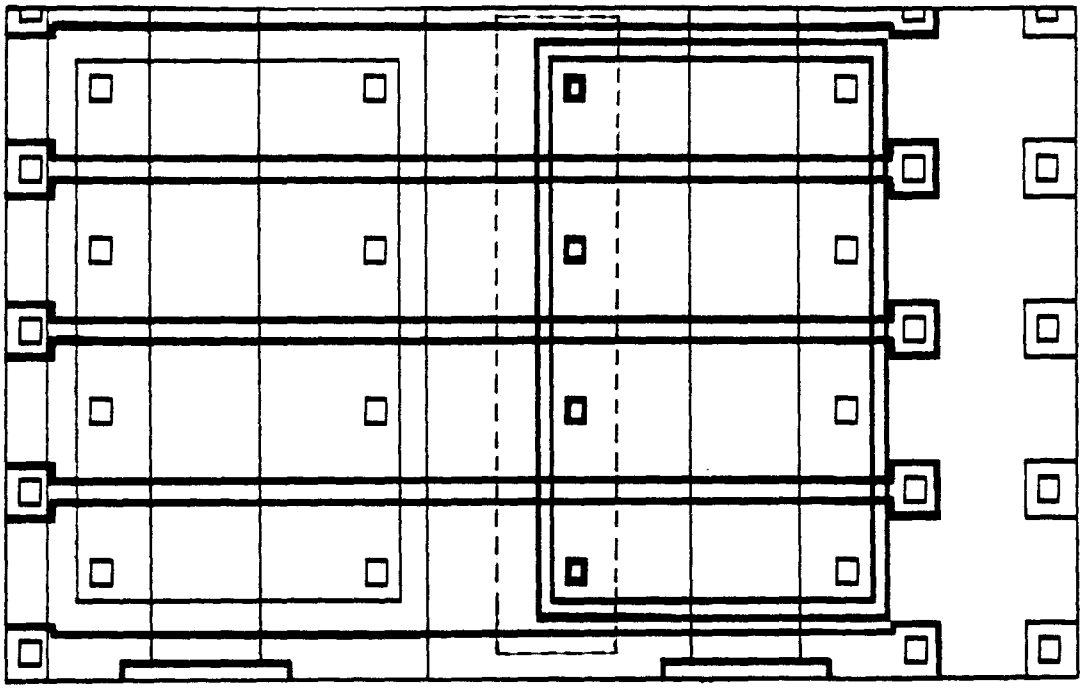


fig. 2 - Modificação atingindo os polígonos contidos inteiramente no retângulo e aqueles que interceptam as bordas do retângulo.

3.2 IMPLEMENTAÇÃO DE UM ALGORITMO DE SELEÇÃO DE MACRO-CÉLULAS PARA TRANSLAÇÃO, CÓPIA OU ELIMINAÇÃO.

Modificar uma macro-célula que foi incluida no "lay-out".

A modificação deve atingir toda a macro, ou seja todos os poligonos que fazem parte da macro.

O usuário deve fornecer a macro desejada para modicação (translação, cópia ou eliminação).

O procedimento a ser implementado segue os passos abaixo :

PASSO 1 :

Receber a modificação desejada : translação, cópia ou eliminação.

PASSO 2 :

Apontar com o cursor a macro a sofrer modificação.

PASSO 3 :

Percorrer a estrutura de dados em busca desta macro.

PASSO 4 :

Se for realizada uma translação,

então
- ler a nova posição para a macro.
- apagar a macro da posição original.
- redesenhá-la na nova posição indicada, guardando esta nova posição na estrutura de dados.

Se for realizada uma cópia,

então
- ler a posição para esta cópia.
- repetir a macro na posição indicada.
- guardar esta cópia na estrutura de dados.

Se for realizada uma eliminação,

então
- apagar a macro da tela.
- apagar a chamada da tela da estrutura de dados.

3.3 IMPLEMENTAÇÃO DE UM ALGORITMO DE RECORTE DE POLÍGONO CÔNCAVO.

Torna-se inconveniente, algumas vezes, dividir polígonos em segmentos de linha quando se deseja fazer recorte ("clipping").

Isto acontece no caso de polígonos preenchidos por padrões ou mesmo por uma coloração qualquer, onde a forma do polígono de saida (polígono recortado) deve ser a mesma do polígono de entrada.

Em se tratando destes polígonos, é necessário assegurar que não foi introduzida nenhuma alteração no polígono recortado. Isto porque, quando se faz o recorte de um polígono, é criada pelo menos uma borda que não está fechada. É preciso fechar esta borda de maneira apropriada para que o polígono possa ser preenchido posteriormente (ver fig. 3).

No caso de polígono côncavo, o recorte é feito de tal maneira que vários polígonos menores são obtidos e todos estes polígonos devem ser fechados corretamente (ver fig. 4).

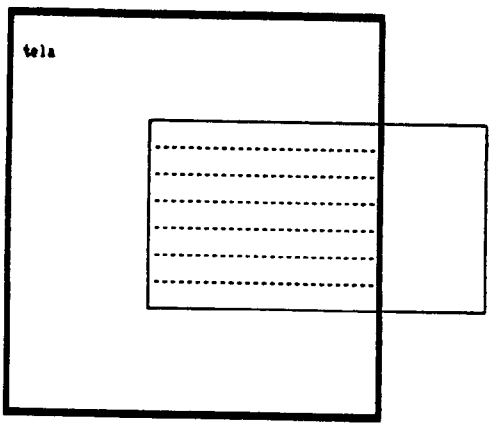


fig. 3 Recorte de um polígono convexo

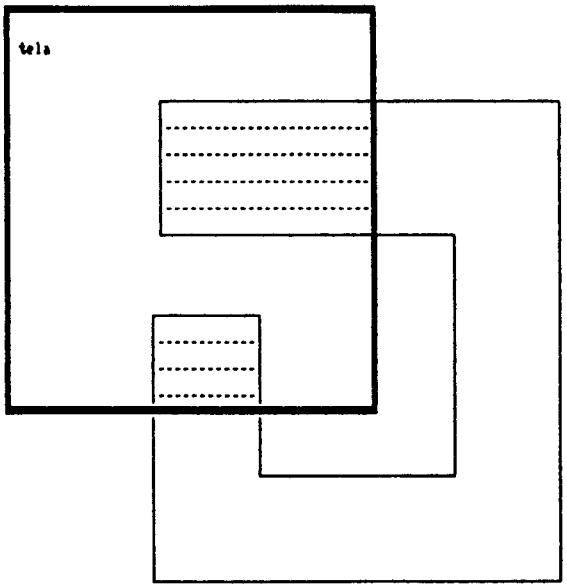


fig. 4 Recorte de um polígono concavo

Para resolver este problema, deve ser implementado um algoritmo de recorte de polígono côncavo, que recorta o polígono contra cada uma das bordas da tela, produzindo ao final um ou vários polígonos fechados.

Atualmente, o EGIPCIS realiza apenas o recorte de linhas, o que não acarreta nenhum problema no caso do programa não se encontrar em modo HACH, ou seja, quando não estão sendo atribuidos padrões aos níveis de máscara (ver fig. 5).

Porém, quando o comando HACH (cap. III, item 3.1.3) é requisitado, pode-se verificar a ocorrência de falhas nos padrões gerados no interior dos polígonos (ver fig. 6).

Isto ocorre devido ao fato de o algoritmo de recorte de linhas produzir segmentos de linha como saida, sem se importar com o fato do polígono não permanecer fechado. E, para que o preenchimento possa ser realizado, as figuras devem estar fechadas.

Um algoritmo eficiente de recorte de polígono côncavo pode ser encontrado em [23].

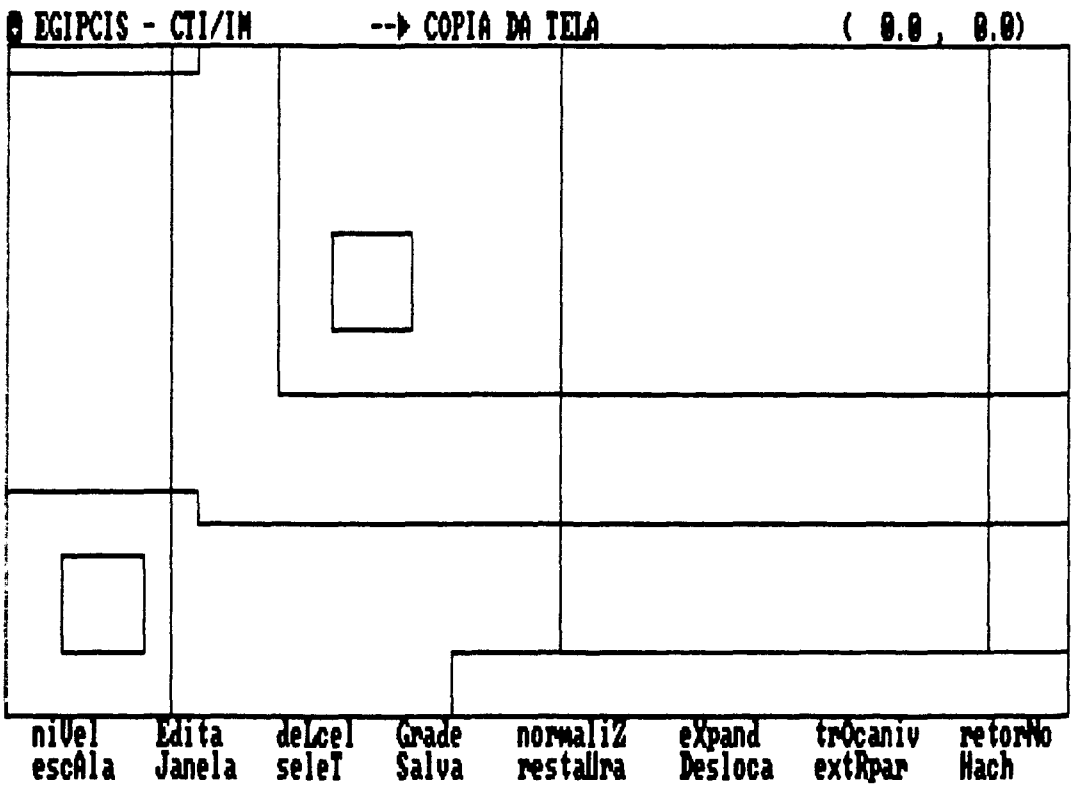


fig. 5 - Programa fora do modo HACH

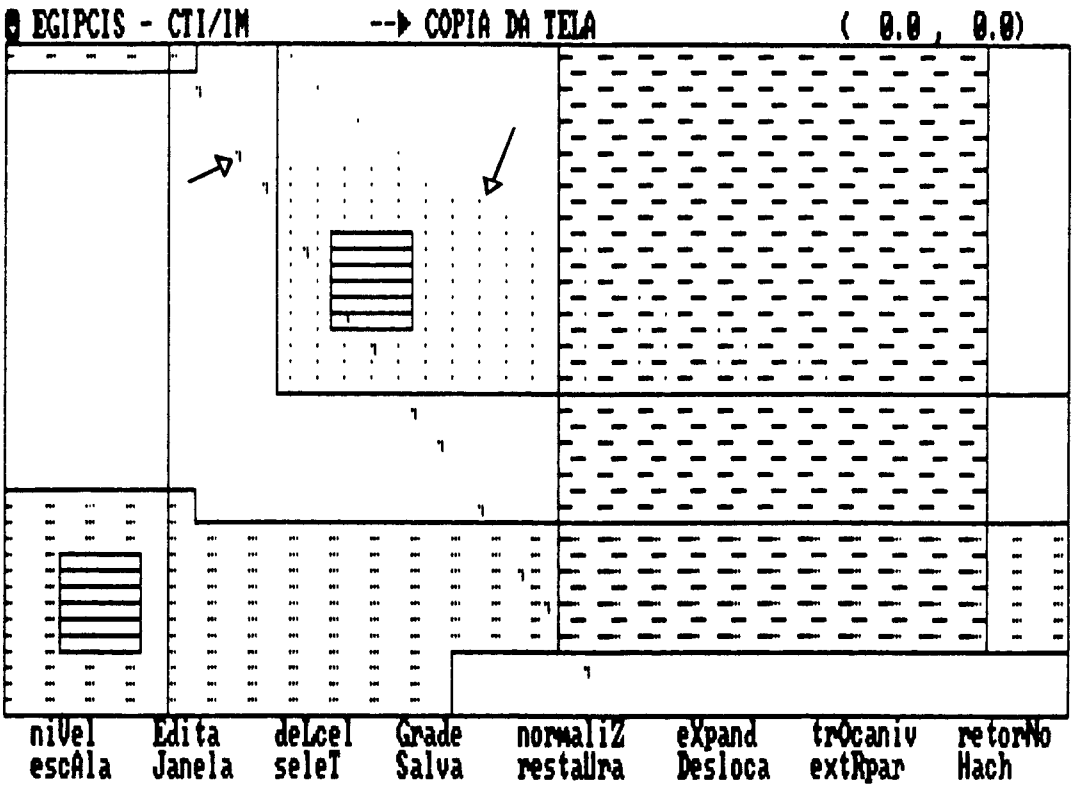


fig. 6 - Programa em modo HACH

3.4 IMPLEMENTAÇÃO DE UM ALGORITMO DE ROTEAMENTO ENTRE CÉLULAS.

Fazer ligações entre as células de modo interativo.

O usuário fornece o posicionamento dos pinos de entrada e de saida de cada célula, o tipo de ligação a ser realizada (poli ou metal) e a largura em micra desta ligação.

As posições dos pinos podem estar, por exemplo, guardadas em um arquivo a parte e, quando este for requisitado, o "lay-out" passa a conter, nestas posições, símbolos indicando ali existirem entradas ou saidas.

Com a utilização do cursor que pode se movimentar através de teclado ou "tablet" (ver ítem 3.5), as ligações vão sendo desenhadas.

Esta tarefa pode ser realizada com o "lay-out" normalizado na tela ou em suas dimensões reais. Devido à baixa resolução de tela, talvez seja necessário algum tipo de correção caso a primeira opção seja escolhida.

O procedimento a ser implementado segue os passos abaixo :

PASSO 1 :

Receber do usuário o posicionamento dos pinos, o tipo de ligação e a largura da ligação.

PASSO 2 :

Receber do usuário o caminho a ser traçado entre um pino de entrada e um pino de saida.

O algorítmo utilizado para receber os pontos é similar àquele utilizado na geração de polígonos (cap. IV, ítem 3.2).

3.5 IMPLEMENTAÇÃO DA OPÇÃO DE ENTRADA DE DADOS VIA "TABLET".

Possibilitar a entrada de dados não só via teclado, mas também via "tablet".

Esta opção traz pelo menos dois benefícios :

- rapidez na entrada dos dados;
- possibilidade de digitalizar um desenho, no caso de um usuário menos experiente.

3.6 COMPILAÇÃO EM TURBO PASCAL VERSÃO 4.0.

A versão 4.0 do PASCAL TURBO apresenta uma série de diferenças em relação à versão 3.0, entre elas :

- velocidade de compilação 2 a 3 vezes mais rápida;
- capacidade de realização de compilações separadas (quebra do programa em vários módulos e compilação de cada um destes módulos separadamente);

- na versão 4.0 cada módulo do programa está limitado em 64K de memória, enquanto na versão 3.0 todos os módulos juntos devem somar 64K;

- maior número de procedimentos e funções disponíveis;

- mais tipos de dados (longint, shortint,...).

Estes recursos implicam em maior facilidade para desenvolver um programa, além de possibilitar a geração de programas com um maior número de linhas de código.

Deve haver, porém, um compromisso entre memória de programa e memória de dados dinâmicos. Um programa demasiado grande acarreta uma diminuição na memória disponível para dados dinâmicos, não permitindo a edição de projetos mais complexos.

3.7 IMPLEMENTAÇÃO DA FUNÇÃO "CONTROLE ESTATISTICO DE PROJETO"

Apresentar, ao usuário, uma tabela contendo informações sobre a sessão de trabalho corrente: data, horário, nome do projeto, número de vezes que cada comando foi requisitado, nome das macro-células incluídas no "lay-out", dimensão do "lay-out", níveis de máscara editados naquela sessão.

Saída em tela e impressora.

REFERÊNCIAS BIBLIOGRÁFICAS

[01] Billingsley G. and Keller, K., "KIC II - User's Manual" - University of California, California - 1985.

[02] Costa, E. M., "Projeto de Circuitos Integrados"- Edição Preliminar, II EBAI, 1986.

[03] Foley, J. D. and Dam, A. Van, "Fundamental of Interactive Computer Graphics". Addison- Wesley Publishing Co., 1983.

[04] Gonnet, G.H., "Handbook of Algorithms and Data Structures", Addison-Wesley Publishing Co., - 1984.

[05] Grupo de Microeletrônica - PGCC, "MICROEDITOR - Editor Gráfico para Microeletrônica" v2.3. Universidade Federal do Rio Grande do Sul, RS, 1986.

[06] Harrington S., "Computer Graphics - A Programming Approach" - Mc Graw-Hill Inc, 1983.

[07] Liesenberg, H. K. E., "Projetos VLSI e seu Suporte Computacional" - Editora Gráfica Formato Ltda - V Escola de Computação, UFMG, Belo Horizonte - 1986.

[08] Maciel, T., "INTERAC - Manipulador de figuras" - Documento Interno DSC/IM/CTI, 1985.

[09] Magalhães, L. P., "Computação Gráfica". Editora da UNICAMP , 1986. I Escola Brasileira e Argentina de Informática - 1986.

[10] Mammana, C. e Dimov, J., "Manual do Usuário da LPG", CODEX - RT035, LED/FEC/UNICAMP, 1983.

[11] Mammana, C.I.Z. e Machado, S.H., "Sistema Didático de Projeto", II Escola Brasileira e Argentina de Informática, 1987.

[12] Mammana, C.I.Z., e Mammana, A.P., "Introdução ao Projeto de Circuitos Integrados", II Escola Brasileira e Argentina de Informática, 1987; III Escola Brasileira e Argentina de Informática, 1988.

[13] "Manual de usuário do sistema SDP" - Laboratório 3. III Escola Brasileira e Argentina de Informática - 1988.

[14] Mead, C. and Conway, L., "Introduction to VLSI System" - Addison-Wesley Publishing Co., 1980.

[15] Newman, W. M. and Sproull, R. F., "Principles of Interactive Computer Graphics". Mc Graw - Hill Inc., 1984.

[16] Ousterhout, J., "CAESAR - VLSI Circuit Editor". Report N. UCB/CSD 83/115. University of California, California - 1983.

[17] Persiano, R. C. M. e Oliveira, A. A. F., "Introdução à Computação Gráfica". Editora Gráfica Formato Ltda. V Escola de Computação, UFMG, Belo Horizonte - 1986.

[18] Pressman, R. S., "Software Engineering: A Practitioner's Approach". Mc Graw-Hill Inc. 1984.

[19] Richards, J.E., "Filling Complex Polygons by Region - Fill Methods on Raster Graphics Terminals". Computer Graphics Forum 6, 49-54, 1987.

[20] Schmitz, E., Assis, J.S. e Borges, J.A., "Turbo Editor para Circuitos Integrados MOS - TEDMOS II", NCE, Universidade do Rio de Janeiro, 1987.

[21] Software Engineering Technical Committee of IEEE Computer Society, "IEEE Guide to Software Requirements Specifications" - julho 1984.

[22] Tozzi, C. L., "PAC - Projeto Auxiliado por Computador". Editora da UNICAMP. I Escola Brasileira e Argentina de Informática - 1986.

[23] Weiler, K. and Atherton, "Hidden Surface Removal using Polygon Area Sorting" - Computer Graphics 11 (2), 214, 1977.

[24] Weste, N. and Eshraghian, K., "Principles of CMOS VLSI Design - A Systems Perspective", Addison-Wesley Publishing Co., 1985.