PROJETO E IMPLEMENTAÇÃO DO BANCO DE DADOS DE UM SISTEMA PARA PAC DE MICROCIRCUITOS

Karla Parreiras Polanczyk
Nilton de Oliveira Junior
Alberto Henrique Frade Laender

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Caixa Postal 702
30161 - Belo Horizonte - MG
Brasil

SUMÁRIO

Este artigo descreve o projeto e a implementação do banco de dados de um sistema para PAC de microcircuitos. São apresentados os principais problemas de modelagem encontrados durante o projeto, as soluções adotadas, e a estratégia geral utilizada na implementação.

1. INTRODUÇÃO

A maioria dos sistemas para PAC (Projeto Assistido por Computador) consiste de um conjunto de ferramentas isoladas, cada uma exercendo uma função específica dentro do contexto da aplicação. Esforços têm sido feitos no sentido de integrar estas ferramentas em um único sistema. Uma das maneiras de se fazer esta integração é através de um banco de dados que atenda de forma centralizada às necessidades dos vários componentes do sistema. Desta forma, é possível manter a consistência entre as possíveis representações dos objetos de trabalho, bem como a integração de outras ferramentas disponíveis [5,6,7,9,20]. Sistemas para PAC, entretanto, trabalham com objetos de alta complexidade, diferentes daqueles existentes em aplicações convencionais. Isto requer técnicas especiais de modelagem de dados, o que torna os modelos de dados contemporâneos [18] e, consequentemente, a maioria dos sistemas de gerenciamento de banco de dados disponíveis comercialmente, inadequados para tais aplicações [6,7,14].

No caso de PAC de microcircuitos, a integração é fundamental para facilitar a manutenção da consistência dos diversos produtos resultantes das várias etapas de projeto. A utilização de um banco de dados como mecanismo de integração

permite o armazenamento consistente de todos os dados referentes ao circuito, desde aqueles que representam informações gráficas até os que são necessários na etapa de produção do "layout" (Veja a Figura 1).

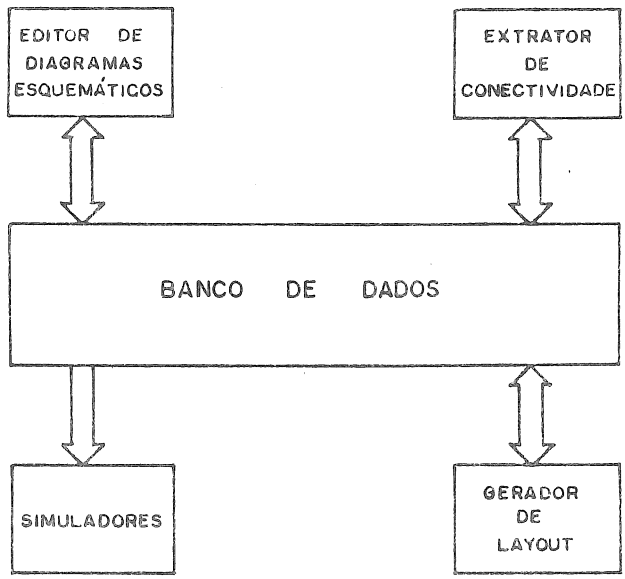

Figura 1

Este artigo descreve o projeto e a implementação do banco de dados utilizado pelo sistema AIPIM (Ambiente Integrado para Projeto Interativo de Microcircuitos) [2] em desenvolvimento no Departamento de Ciência da Computação da UFMG, através de convênio estabelecido com o Centro de Pesquisa e Desenvolvimento da TELEBRÁS. O AIPIM é um sistema para PAC de microcircuitos, utilizando células padrão, que provê, entre outras, facilidades para captura de diagramas esquemáticos [11], posicionamento de células [19], traçado automático de rotas [10] e desenho de "layouts" [21]. O banco de dados do AIPIM armazena todos os dados necessários aos seus vários componentes, possibilitando a total integração do sistema.

O restante do artigo se divide da seguinte maneira: na Seção 2 são analisados alguns aspectos referentes ao projeto de microcircuitos; na Seção 3 é descrito o projeto conceitual do banco de dados do sistema AIPIM; na seção 4 é apresentada uma breve descrição da implementação; finalmente, na Seção 5 são apresentadas algumas considerações finais sobre o trabalho.

2. CONSIDERAÇÕES ACERCA DO PROJETO DE MICROCIRCUITOS

As aplicações de PAC envolvem geralmente dados de natureza complexa e que variam de aplicação para aplicação. É imprescindível, portanto, que ao se desenvolver um sistema para PAC, se compreenda claramente o seu ambiente de aplicação. Assim, analisaremos aqui algumas características inerentes ao projeto de microcircuitos. Esta análise não pretende ser exaustiva e tem como objetivo caracterizar as necessidades básicas de um ambiente de projeto de microcircuitos, tendo como base a visão do sistema AIPIM.

Um microcircuito é constituído basicamente de componentes, interligados através de sinais, cujo projeto lógico é expresso através de um diagrama esquemático. Neste diagrama, os componentes são representados graficamente por figuras geométricas e correspodem a instâncias de macros, que são as funções pré-definidas disponíveis para a elaboração do circuito. As macros podem ser utilizadas em um diagrama tantas vezes quantas necessárias, sendo que as instâncias correspondentes não precisam ser necessariamente iguais, uma vez que o projetista pode alterar algumas de suas características básicas (por exemplo, os parâmetros elétricos).

As macros podem ser de dois tipos: aquelas que constituem as células mais básicas de um circuito, as macros padrão, e aquelas criadas pelo projetista a partir de outras, as macros de projeto. As macros de projeto possuem um projeto interno que representa estruturalmente a função lógica ou elétrica que elas sintetizam. Estas macros compreendem, portanto, o projeto hierárquico de um circuito. Este é um aspecto muito importante, uma vez que possibilita ao projetista produzir o seu circuito hierarquicamente e por etapas. Uma macro de projeto pode ainda apresentar diversos "layouts" e ser considerada como uma célula básica para a geração de outros "layouts". Desta forma, o projetista pode acoplar ao seu circuito blocos já testados. Estes blocos constituem as macros de layout.

Existe um outro aspecto importante a se considerar no que diz respeito às macros. Uma macro de projeto pode ter vários projetos internos associados, sendo que qualquer um deles pode ser utilizado em qualquer uma de suas instâncias presentes em um diagrama esquemático. Isso caracteriza a existência de múltiplas versões de uma macro de projeto, possibilitando ao projetista várias opções para elaboração do circuito.

Na seção seguinte veremos como o banco de dados do sistema AIPIM foi projetado para atender a estas necessidades. Em particular, mostraremos como foram modelados os aspectos referentes ao projeto hierárquico de circuitos e à existência de múltiplas versões de um circuito ou macro de projeto.

3. O BANCO DE DADOS DO SISTEMA AIPIM

O banco de dados do sistena AIPIM foi projetado para atender aos seguintes objetivos:

- suportar o editor de diagramas esquemáticos, armazenando todos os dados que representam informações gráficas e de contexto necessárias ao seu funcionamento.

- permitir o projeto hierárquico de circuitos, através do armazenamento das definições de macros de projeto e macros de layout.

- suportar a existência de diferentes versões de um mesmo circuito, macro de projeto ou macro de layout.

- possibilitar uma pesquisa espacial eficiente, viabilizando a exibição pelo editor gráfico dos objetos situados numa determinada área ou janela de visualização.

- suportar o posicionamento de células e o traçado de rotas na etapa de elaboração do "layout".

- permitir que outras ferramentas já existentes (simuladores, por exemplo) possam ser facilmente integradas ao sistema AIPIM quando necessário.

O projeto conceitual [3] do banco de dados envolveu basicamente duas etapas: (1) o desenvolvimento do modelo conceitual, no qual são descritos os objetos envolvidos e os relacionamentos entre eles, e (2) a especificação das funções primitivas de manipulação dos dados (criação, remoção, atualização e recuperação). A relação destas funções é apresentada no Apêndice 1. Uma descrição mais detalhada pode ser encontrada em [12,14].

A Figura 2 apresenta o modelo conceitual produzido com base no modelo de entidades e relacionamentos estendido (MER+), uma simplificação do modelo proposto em [3]. Antes de apresentarmos os detalhes do modelo conceitual, faremos uma breve descrição do MER+ e de sua simbologia.

3.1. O Modelo de Entidades e Relacionamentos Estendido (MER+)

O MER+ é uma extensão do modelo de entidades e relacionamentos proposto por Chen [4]. No modelo de entidades e relacionamentos, as primitivas básicas de modelagem são entidades, atributos e relacionamentos. As entidades correspondem aos objetos do mundo real inerentes à aplicação em questão. Os atributos descrevem as propriedades destes objetos e os relacionamentos representam associações significativas entre entidades. Os relacionamentos podem ser dos tipos "um-para-um" (1:1), "um-para-muitos" (1:N), "muitos-para-um" (N:1) e "muitos-para-muitos" (M:N) [18].

O MER+ estende o modelo de Chen, adicionando outras construções que permitem expressar, por exemplo, hierarquias de generalização [16,18] e restrições de existência (totalidade) nos relacionamentos [18]. A Figura 3 apresenta a simbologia básica adotada pelo MER+.

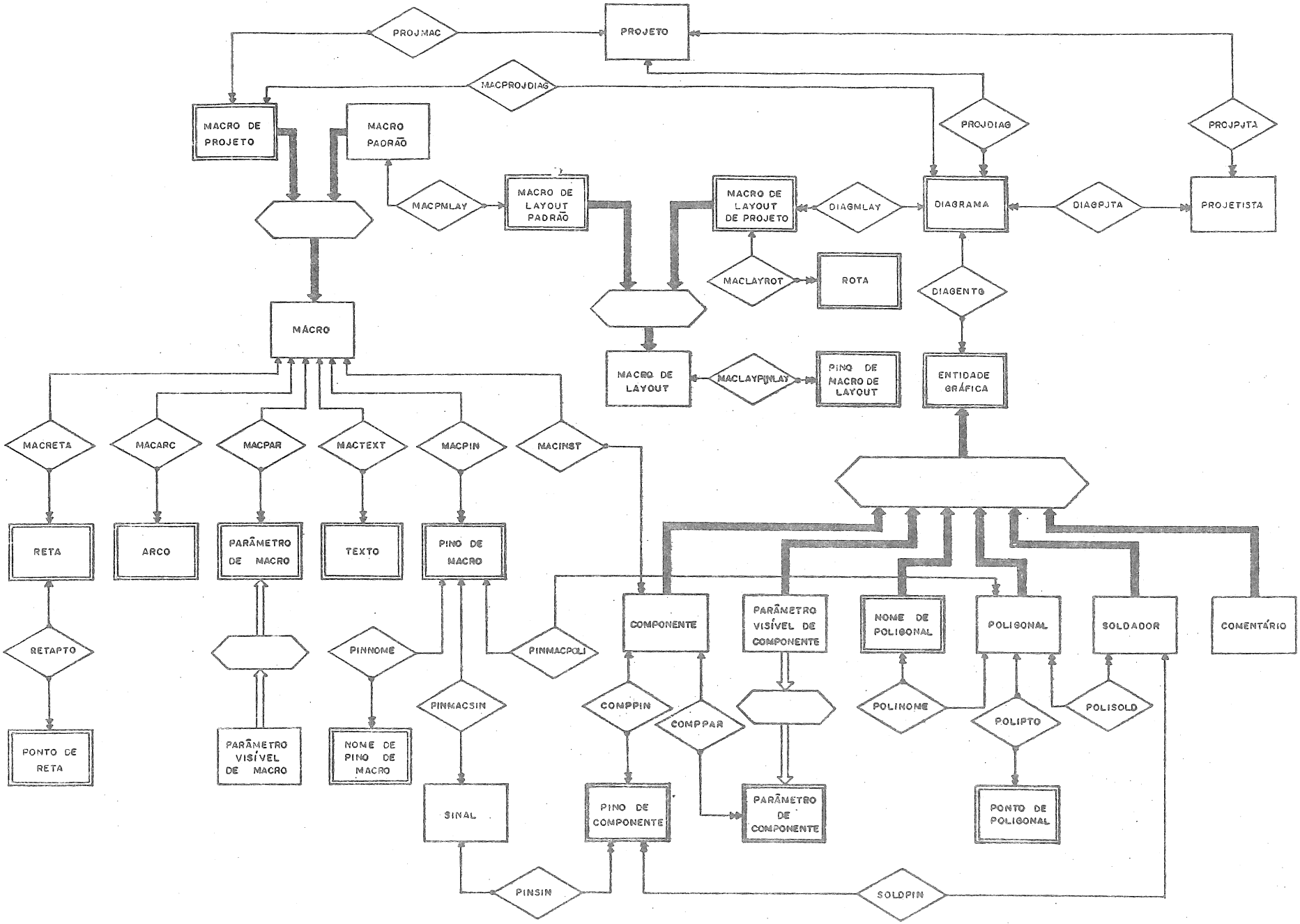
PROJMAC
PROJETO
MACPROJDIAG
MACRO DE PROJETO
MACRO PADRÃO
PROJDIAG
PROJPJTA
MACPMLAY
MACRO DE LAYOUT PADRÃO
MACRO DE LAYOUT DE PROJETO
DIAGMLAY
DIAGRAMA
DIAGPJTA
PROJETISTA
MACLAYROT
ROTA
DIAGENTG
MÁCRO
MACRO DE LAYOUT
MACLAYPINLAY
PINO DE MACRO DE LAYOUT
ENTIDADE GRÁFICA
MACRETA
MACARC
MACPAR
MACTEXT
MACPIN
MACINST
RETA
ARCO
PARÂMETRO DE MACRO
TEXTO
PINO DE MACRO
RETAPTO
PINNOME
PINMACPOLI
COMPONENTE
PARÂMETRO VISÍVEL DE COMPONENTE
NOME DE POLIGONAL
POLIGONAL
SOLDADOR
COMENTÁRIO
PONTO DE RETA
PARÂMETRO VISÍVEL DE MACRO
NOME DE PINO DE MACRO
PINMACSIN
COMPPIN
COMPPAR
POLINOME
POLIPTO
POLISOLD
SINAL
PINO DE COMPONENTE
PARÂMETRO DE COMPONENTE
PONTO DE POLIGONAL
PINSIN
SOLDPIN

Figura 2

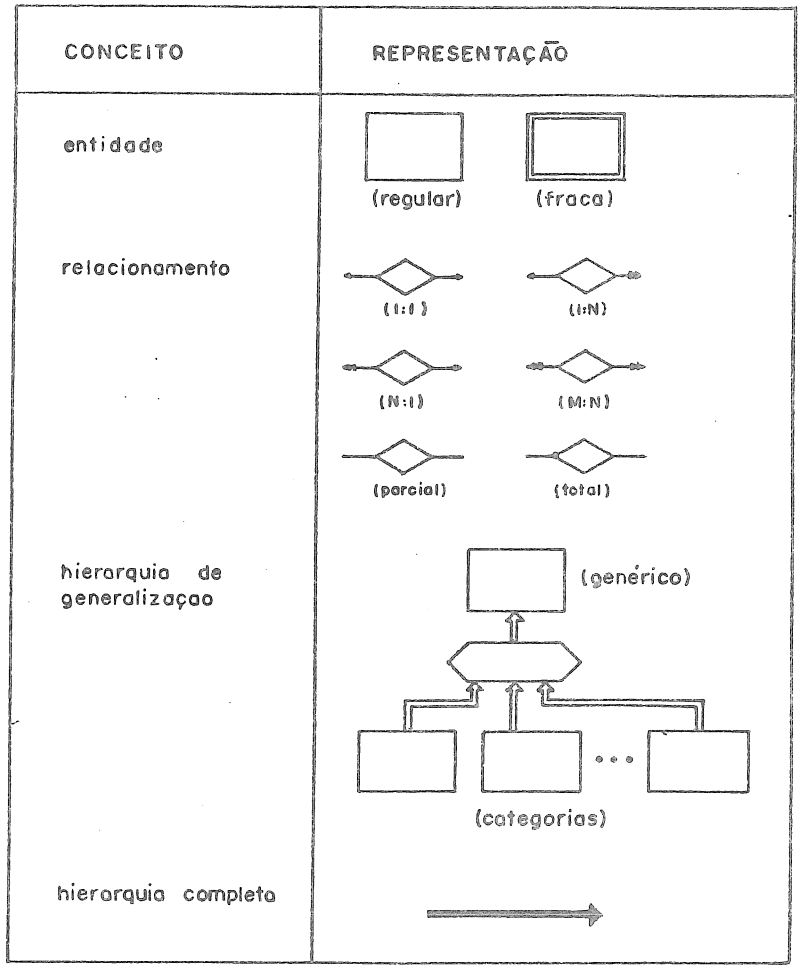

Figura 3

3.2. Descrição do Modelo Conceitual

O modelo conceitual apresentado na Figura 2 descreve a estrutura lógica do banco de dados do sistema AIPIM, levando-se em conta os objetivos citados anteriormente. Discutiremos aqui como os problemas inerentes à aplicação foram

Primeiramente, era necessário possibilitar uma pesquisa espacial eficiente, uma vez que o editor de diagramas esquemáticos precisa ter acesso a todos os objetos graficamente presentes em uma determinada janela de visualização (a janela de visualização é uma área retangular, determinada pelas coordenadas de seus cantos inferior esquerdo e superior direito) [11]. Note que estes objetos podem corresponder a entidades diferentes no banco de dados, como por exemplo, componentes, poligonais, soldadores, etc. Para solucionar este problema, foi criada uma entidade especial, a ENTIDADE GRÁFICA, que é uma generalização de todos os objetos que podem aparecer na tela do editor. A hierarquia de

generalização, neste caso, é uma hierarquia completa [3], uma vez que cada ocorrência da ENTIDADE GRÁFICA pertence a uma de suas categorias. Desta forma, é possível identificar todos aqueles objetos presentes em uma determinada área do circuito, independentemente de seus tipos.

Um outro aspecto importante modelado refere-se às macros de projeto. Durante uma sessão de trabalho com o editor, o projetista pode definir macros de projeto, criando o seu circuito de forma hierárquica. Para representar isto conceitualmente, foi criada a entidade MACRO DE PROJETO. Através do relacionamento MACPROJDIAG, podemos obter o(s) diagrama(s) esquemático(s) correspondente(s) ao(s) projeto(s) interno(s) de uma determinada macro de projeto. Desta forma, foi também resolvido o problema de se armazenar diferentes versões de projeto interno para uma mesma macro de projeto.

Assim como as macros de projeto, as macros de layout também precisavam ser modeladas. Note, no entanto, que uma macro de layout pode estar associada ou ao projeto interno de uma macro de projeto ou ao projeto interno de uma macro padrão. Por isto, foi criada a entidade MACRO DE LAYOUT como uma generalização das entidades MACRO DE LAYOUT PADRÃO e MACRO DE LAYOUT DE PROJETO. Assim, através do relacionamento DIAMGLAY, pode-se obter as possíveis macros de layout de um determinado diagrama esquemático. Com isto podemos obter também as possíveis versões de um diagrama esquemático.

Finalmente, cabe ressaltar que várias das entidades definidas no modelo conceitual são entidades fracas [4,18], ou seja, entidades cuja existência é dependente de outra. Esta é uma característica inerente à própria aplicação. Assim, por exemplo, toda ocorrência da ENTIDADE GRÁFICA (por exemplo, um componente) é dependente da existência do diagrama ao qual está associada, que por sua vez é dependente de um projeto.

4. IMPLEMENTAÇÃO

O sistema AIPIM está sendo implementado em Pascal Microsoft, em um microcomputador do tipo PC-AT com 2 MBytes de memória principal, duas unidades de disco flexível, um disco rígico com capacidade de 20 MBytes, e facilidades para entrada e saída gráficas (monitor gráfico, tablete, "plotter" de mesa e impressora gráfica). Para suporte ao banco de dados está sendo utilizado o sistema Btrieve [16], que é um sistema de gerenciamento de arquivos que possui algumas características relacionais. O acesso ao banco de dados é feito através de uma série de procedimentos que implementam as funções de manipulação especificadas na fase de projeto conceitual e relacionadas no Apêndice 1. Estes procedimentos provêm uma interface padrão, através da qual são mantidas as restrições de integridade definidas para o banco de dados.

O esquema relacional, gerado a partir do modelo conceitual como sugerido em [3], se encontra detalhado no Apêndice 2. Os nomes das relações e dos atributos são auto-explicativos e dão uma idéia da semântica envolvida. Note que, para cada entidade, um identificador gerado internamente foi definido como sendo a chave primária da relação correspondente. Desta forma,

estabeleceu-se um mecanismo de identificação simples e eficiente (em geral um dos principais problemas neste tipo de aplicação [5, 20]), evitando-se também a propagação de chaves compostas para representação dos relacionamentos.

Como ilustração da interface do banco de dados, é apresentada abaixo a definição dos procedimentos que implementam as principais funções primitivas para manipulação de uma poligonal. A definição inclui a lista de relações às quais cada um dos procedimentos tem acesso. RemPolig, por exemplo, tem acesso às relações POLIGONAL, ENTIDADE_GRÁFICA, NOME_DE_POLIGONAL, PONTO_DE_POLIGONAL e POLISOLD. Estas relações correspondem às entidades de mesmo nome e ao relacionamento POLISOLD representados no modelo conceitual (Figura 2). Isto significa que ao removermos uma poligonal do banco de dados, este procedimento não só remove a tupla correspondente da relação POLIGONAL, como também remove as tuplas associadas das relações ENTIDADE_GRÁFICA, PONTO_DE_POLIGONAL, NOME_DE_POLIGONAL e POLISOLD.

Procedimento CriaPolig
descrição: insere uma nova poligonal no banco de dados;
parâmetros de entrada: iddiag, xmin, ymin, xmax, ymax, nomepolig, estilolinha, cor, marcaexcon;
parâmetros de saída: idpolig, resultado;
relações manipuladas: ENTIDADE_GRÁFICA, POLIGONAL;

Procedimento AchaPolig
descrição: recupera os valores dos atributos de uma poligonal;
parâmetros de entrada: idpolig;
parâmetros de saída: nomepolig, xmin, ymin, xmax, ymax, estilolinha, cor, marcaexcon, resultado;
relações manipuladas: ENTIDADE_GRÁFICA, POLIGONAL;

Procedimento RemPolig
descrição: remove uma poligonal do banco de dados;
parâmetros de entrada: idpolig;
parâmetros de saída: resultado;
relações manipuladas: ENTIDADE_GRÁFICA, POLIGONAL, NOME_DE_POLIGONAL, PONTO_DE_POLIGONAL, POLISOLD;

Procedimento AtPolig
descrição: atualiza os valores dos atributos de uma poligonal;
parâmetros de entrada: idpolig, seleção, nomepolig, estilolinha, cor, marcaexcon, xmin, ymin, xmax, ymax;
parâmetros de saída: resultado;
relações manipuladas: ENTIDADE_GRÁFICA, POLIGONAL, NOME_DE_POLIGONAL;

5. CONCLUSÕES

Um banco de dados é uma forma bastante eficiente de se integrar diferentes ferramentas e assegurar a consistência dos dados de projeto em um sistema para PAC de microcircuitos [5,6,9,20]. O projeto de um banco de dados para tal aplicação não é, entretanto, uma tarefa fácil. As características da aplicação e a falta de ferramentas adequadas para modelagem dos dados impõem uma série de dificuldades ao projetista do banco de dados.

No caso particular do banco de dados do sistema AIPIM, estas dificuldades foram parcialmente superadas através da utilização do MER+. Apesar de não possuir facilidades para a modelagem de objetos complexos, o MER+ possibilitou uma abordagem simples através da qual a semântica da aplicação foi capturada de forma satisfatória. Desta forma, foi possível gerar um esquema relacional consistente e especificar uma interface que assegura as restrições de integridade definidas, possibilitando a utilização do sistema Btrieve de forma sistemática e eficiente.

AGRADECIMENTOS

Os autores gostariam de agradecer a todos o membros do projeto AIPIM pelas sugestões e comentários. Este trabalho foi desenvolvido com apoio financeiro do convênio UFMG/TELEBRÁS.

REFERÊNCIAS

[1] BATORY, D. S. & KIM, W. Modeling concepts for VLSI CAD objects. ACM Transactions on Database Systems 10, 3 (September 1985), 322-346.

[2] CAMPOS. I. M. AIPIM - Ambiente Integrado para Projeto Interativo de Microcircuitos. Anais do I Congresso da Sociedade Brasileira de Microeletrônica, Campinas, Julho 1986, pp. 351-360.

[3] CERI, S. (ed.) Methodology and Tools for Database Design. North-Holland, Amsterdam, 1983.

[4] CHEN, P. P. S. The entity-relationship model - Toward a unified view of data. ACM Transactions on Database Systems 1, 1 (March 1976), 9-36.

[5] CHU, K. et al. Vdd - A VLSI design database system. Databases for Engineering Aplications, ACM Database Week (May 1983), 39-50.

[6] KATZ, R. H. A database approach for managing VLSI design data. Proc. ACM/IEEE Design Automation Conference, Las Vegas, Nevada, June 1982, pp. 274-282.

[7] McLELLAN, P. Effective data management for VLSI design. Proc. ACM/IEEE

Design Automation Conference, Las Vegas, Nevada, 1985, pp. 652-657.

[8] McLEOD, D. et al. An approach to information management for CAD/VLSI applications. Databases for Engineering Applications, ACM Database Week (May 1983), pp. 39-50.

[9] NEUMANN, T. On representing the design information in a common database. Databases for Engineering Applications, ACM Database Week (May 1983), pp. 81-87.

[10] OLIVEIRA, E. Um sistema para o traçado automático de rotas em circuitos integrados baseados em "standard-cells". Anais do I Congresso da Sociedade Brasileira de Microeletrônica, Campinas, Julho 1986, pp. 310-319.

[11] OLIVEIRA, N. Jr. & POLANCZYK, K. P. EDDIE - Um Editor Gráfico de Circuitos a Nível Lógico. Anais do I Congresso da Sociedade Brasileira de Microeletrônica, Campinas, Julho 1986, pp. 419-429.

[12] POLANCZYK, K. P. Banco de Dados do AIPIM - Descrição Funcional da Visão do Subsistema EDDIE. Relatório Técnico, DCC-ICEX-UFMG, 1986.

[13] POLANCZYK, K. P. Descrição do Esquema Relacional do Banco de Dados do AIPIM. Relatório Técnico, DCC-ICEX-UFMG, 1986.

[14] POLANCZYK, K. P. Projeto e implementação de um banco de dados para suporte a PAC de microcircuitos. Dissertação de mestrado, DCC-ICEX-UFMG, 1987 (em elaboração).

[15] SIDLE, T. W., Weakness of commercial database management systems in engineering applications. Proc. ACM/IEEE Design Automation Conference, Minneapolis, Minesotta, June 1980.

[16] SMITH, J. M. & SMITH, D. Database abstractions: aggregations and generalizations, ACM Transactions on Database Systems 1, 2 (June 1977), 105-133.

[17] SOFTCRAFT, INC. Btrieve Version 3.15. Austin, Texas, 1984.

[18] TSICHRTZIS, D. & LOCHOVSKY, F. Data Models, Prentice-Hall, Englewood Cliffs, N.J., 1982.

[19] TRILLO, J. L. R. Um posicionador de células padrão em "layouts" de circuitos integrados. Anais do I Congresso da Sociedade Brasileira de Microeletrônica, Campinas, Julho 1986, pp. 320-328.

[20] WONG, S. & BRISTOL, W. A. A Computer-aided design data base. Proc. ACM/IEEE Design Automation Conference, San Diego, California, June 1979, pp. 398-402.

[21] ZOLA, W. M. N. CIFDRAW - Um programa para "check-plot" de circuitos

integrados. Anais do I Congresso da Sociedade Brasileira de Microeletrônica, Campinas, Julho 1986, pp. 463.

APÊNDICE 1

FUNÇÕES DE MANIPULAÇÃO DO BANCO DE DADOS DO SISTEMA AIPIM

1) Funções para manipulação da entidade Projeto:
CriaProj, CriaProjPjta, AchaProjPjta, AchaProj, RemProj, RemProjPjta, AtProj;

2) Funções para manipulação da entidade Diagrama:
CriaDiag, AchaDiagPjta, AchaDiagProj, AchaDiag, RemDiag, AtDiag;

3) Funções para manipulação da entidade Projetista:
CriaPjta, CriaPjtaDiag, AchaPjtaAloc, AchaPjtaDiag, RemPjta, RemPjtaDiag;

4) Funções para manipulação das entidades Macro Padrão e Macro de Projeto:
CriaMacroPadrão, CriaMacroProjeto, CriaReta, CriaPtoReta, CriaArco, CriaTexto, CriaPinoMacro, CriaNomePinoMacro, CriaParamMacro, CriaParVisMacro, AchaMacroPadrão, AchaMacroProjeto, AchaRetaMacro, AchaPontoReta, AchaArcoMacro, AchaParamMacro, AchaParVisMacro, AchaPinoMacro, AchaNomePinoMacro, AchaTextoMacro, RemMacroProj, AtMacProj, AtReta, AtPtoReta, AtArco, AtTexto;

5) Função para manipulação da entidade Entidade Gráfica:
AchaEntGraf;

6) Funções para manipulação da entidade Componente:
CriaComp, CriaParamComp, CriaParVisComp, CriaPinoComp, AchaParamComp, AchaParVisComp, AchaPinoComp, AchaComp, RemComp, RemParVisComp, RemParamComp, AtComp, AtParComp, AtParVisComp;

7) Funções para manipulação da entidade Poligonal:
CriaPolig, CriaNomePolig, CriaPtoPolig, AchaPolig, AchalNomePolig, AchaNomesPolig, AchaPtoPolig, RemPolig, RemNomePolig, AtPolig, AtNomePolig, AtPtoPolig;

8) Funções para manipulação da entidade Soldador:
CriaSold, CriaSoldPin, CriaSoldPoli, AchaSold, AchaSoldPoli, AchaSoldPoli, AchaSoldPin, AchaPinSold, RemSoldPin, RemSoldPoli, RemSold, AtSold;

9) Funções para manipulação da entidade Comentário:
CriaComent, AchaComent, RemComent, AtComent;

APÊNDICE 2

ESQUEMA RELACIONAL DO BANCO DE DADOS DO SISTEMA AIPIM

PROJETO (idproj, nomeproj, nomeresp, dcriação)

PROJETISTA (idpjta, nomepjta)

PROJPJTA (idproj, idpjta)

DIAGRAMA (iddiag, nomediag, dcriação, dultatu, respultatu, xmin, ymin, xmax, ymax, nmens, netlist, refdiagx, refdiagy, raiox, raioy, idproj, idmacroproj)

DIAGPJTA (iddiag, idpjta)

ENTIDADE_GRÁFICA (identgraf, xmin, ymin, xmax, ymax, categoria, iddiag)

COMENTÁRIO (idcoment, texto, cor, orientação, tamtexto)

COMPONENTE (idcomp, nomecomp, cor, teta, escala, translacao, posicaox, espelhamento, idmacro)

NOME_DE_POLIGONAL (idnomepolig, nomepolig, cor, orientação, tamtexto, idpolig)

PARÂMETRO_VISÍVEL_DE_COMPONENTE (idparviscomp, completo, orientação, tamtexto, cor)

POLIGONAL (idpolig, nomepolig, estilolinha, cor, npontos, idpinomacro)

SOLDADOR (idsold, cor, translacao, escala, teta)

PARÂMETRO_DE_COMPONENTE (idparamcomp, nomepar, valor, visibilidade, idcomp)

PINO_DE_COMPONENTE (idpinocomp, idpinomacro, idsinal, idsold, idcomp)

PONTO_DE_POLIGONAL (idptopolig, pontox, pontoy, idpoli)

POLISOLD (idpolig, idsold)

MACRO (idmacro, nomeresp, dcriação, refx, refy, xmin, ymin, xmax, ymax, categoria)

MACRO_PADRÃO (idmacropadrao, nomemacro)

MACRO_DE_PROJETO (idmacroproj, nomemacro, idproj)

RETA (idreta, npontos, idmacro)

PONTO_DE_RETA (idptoreta, pontox, pontoy, idreta)

ARCO (idarco, arc1x, arc1y, arc2x, arc2y, arc3x, arc3y, idmacro)

TEXTO (idtexto, texto, txtxrel, txtyrel, cor, orientação, tamtexto, idmacro)

PINO_DE_MACRO (idpinomacro, xrel, yrel, nomepino, tipo, idmacro)

NOME_DE_PINO_DE_MACRO (idnomepinmacr, xrel, yrel, cor, orientação, nomepino, tamtexto, idpinomacro)

PARAMETRO_DE_MACRO (idparammacro, nomeparam, visibilidade, valor, idmacro)

PARAMETRO_VISÍVEL_DE_MACRO (idparvismacro, xrel, yrel, completo, orientação, nomepar, tamtexto, cor)

SINAL (idsinal, nomesinal, crítico, capacitância, idpinomacro)

MACRO_DE_LAYOUT (idmacrolayout, altura, largura, pad, categoria)

PINO_DE_MACRO_DE_LAYOUT (idpinmaclay, xinf, xsup, nomepino, intercamb, idmacrolayout)

MACRO_DE_LAYOUT_PADRÃO (idmaclaypadrao, nomemacrolayout, idmacropadrao)

MACRO_DE_LAYOUT_DE_PROJETO (idmaclayproj, nomemacrolayout, iddiag)

ROTA (idrota, canal, máscara, x, y, comprimento, idmacrolayproj)