J.A.S. BORGES*

SUMÁRIO

O projeto de circuitos integrados está intimamente ligado ao uso de um editor gráfico onde se possa fazer operações de criação e montagem das partes componentes. Este artigo mostra algumas características de dois editores gráficos para projeto VLSI que foram produzidos no NCE-UFRJ.

O primeiro editor é o EDMOS, que já vem sendo utilizado desde 1983 e utiliza um microcomputador nacional (SDE-42) com vídeo gráfico e disquete como estação do trabalho.

O segundo editor é o EDCI, um editor hierárquico que visa acompanhar o projeto desde a alocação inicial de áreas até a montagem completa.

ABSTRACT

The project of IC's always needs a graphic editor, where one can do operations of creating and assembling the lay-outs. This paper shows the two NCE-UFRJ's circuit editors.

The first of them is EDMOS, that is in use since 1983. This editor uses a brazilian microcomputer (EBC-SDE-42) with a videographic interface and diskette, very cheap, which is an important factor at nationwide context.

The second is EDCI, an hierarchical editor which supports the project since area allocation until complete assembly.

* Informático (1980 - UFRJ); Processamento Gráfico; Estações de CAD; Universidade Federal do Rio de Janeiro, Núcleo de Computação Eletrônica, Caixa Postal 2324, Rio de Janeiro, CEP 20001.

As fábricas de circuitos integrados recebem a especificação de um chip na forma de uma linguagem de descrição da geometria, na sua maioria a CIF (Caltech Intermediate Format) Assim, a maneira mais rudimentar de especificar um circuito integrado para a fábrica é desenhá-lo em papel milimetrado e depois traduzir manualmente cada elemento da figura para esta linguagem. O desenho traduzido é entrado através de um editor de textos e depois copiado para uma fita magnética, que é enviada ao fabricante.

Infelizmente isto é completamente inviável: a quantidade de elementos que compõem um circuito pequeno é da ordem de dezenas de milhares, o que implica num trabalho sobre-humano. Basta que um erro seja cometido no processo de codificação para que o chip não funcione (e que sejam perdidos dezenas de milhares de dólares, que é o que a fundição de silício cobra por um protótipo).

Mesmo no processo de alteração, o método manual apresenta muitos inconvenientes. Uma simples operação de ajuste, ou seja, puxar um trecho do desenho um pouco mais para a direita, envolve a alteração da posição de todos os elementos do desenho compreendidos naquele intervalo.

Uma das abordagens mais aceitas no projeto de Circuitos Integrados hoje, é o uso das chamadas "células padrão" (Standard cells, biblioteca de células) que já foram projetadas e não apresentam erro. A união destas células é trabalhosa pois envolve o problema da diagramação do espaço físico disponível no chip e da codificação das ligações entre os elementos (sempre muito numerosas) mesmo que muitas células se encaixem perfeitamente (por abutment).

Todo este processo, altamente sujeito a erros, cria a necessidade de uso de papel para verificação da montagem do layout, geralmente em plotter ou impressora gráfica. Porém até mesmo essa visualização é difícil, pois o desenho é sempre muito intrincado.

SOBRE TERMINAIS E ESTAÇÃO DE CAD

Embora seja certamente possível editar layouts (especialmente de células) em terminais alfanuméricos (Lewis-81) a prática demonstra que o uso de terminais gráficos na edição de circuitos integrados é muito melhor, principalmente pelo aspecto visual e pelo tamanho do circuito que é possivel se ver simultaneamente numa tela.

Entre as características desejáveis da tela do terminal, deveríamos a princípio especificar um terminal que tivesse uma tela de 1024 x 1024 pontos, cursor retangular, 256 cores simultâneas, suportado por um computador de médio porte.

Esses requisitos são difíceis de serem conseguidos no Brasil, especialmente porque a produção de terminais gráficos é ainda muito incipiente. É porém fácil de se conseguir um microcomputador com tela gráfica colorida de resolução 240 x 400 (no nosso caso, foi um EBC-SDE42 com interface videográfica), que no caso, fica transformado em uma estação local para edição de circuitos.

O uso desta estação local é muito interessante, pois além do baixo custo, tem a vantagem da disponibilidade (em centros de computação universitários a disputa pelo uso do computador é enorme). Um micro é suficientemente barato para que possamos ter várias unidades, ao invés de um terminal muito caro, possibilitando assim que vários projetistas trabalhem ao mesmo tempo (o que é fundamental em turmas de cursos sobre projetos de VLSI.) Por outro lado, o armazenamento local de parte do circuito que está sendo editado e a possibilidade de processamentos locais (como verificação das regras de projeto) da célula que está sendo editada, faz com que só se precise mesmo usar o computador grande para os processos que envolvam maior necessidade de poder computacional (por exemplo, simulações do circuito).

A esta estação gráfica deve estar conectado um plotter de mesa e/ou uma impressora gráfica preto-e-branca, para hardcopy, pois a cópia em papel é necessária para que um projetista possa ir estudar em sua sala, liberando a estação para outra pessoa.

A estação local deve estar ligada ao computador principal através de uma linha de comunicações comum (9600 bauds), mas a frequência de transferência de informações é relativamente pequena (da ordem de 10 a 100 kbytes para arquivos CIF).

Assim, resumindo, na estação fazemos a edição das células e seu DRC. Usamos a estação como terminal gráfico simples para fazer a planta baixa, montagem do circuito e ligações entre as células, e usamos o computador principal para a verificação final e simulações, além da plotagem completa em plotter de alta velocidade.

VISÃO GERAL DO EDMOS

Mostraremos agora o EDMOS, que é o editor gráfico de células. O projeto deste editor foi influenciado por dois editores muito conhecidos: CIFSYM (Lewis-81) e CAESAR (Ousterhout-80). Em termos de processamento é semelhante ao primeiro; em termos de visualização, semelhante ao segundo.

O EDMOS é um editor que usa a memória local do video gráfico para armazenamento do layout (96kbytes). Não existem estruturas de dados intermediárias, ou seja, o layout é feito sobre a própria memória da tela, acendendo ou apagando os pontos convenientes como se estivéssemos colorindo um papel. Quando o desenho está completo, a memória de tela é varrida, extraindo-se o desenho.

Obs.: Foi feita posteriormente uma segunda versão do EDMOS que não mais utiliza estas características intrínsecas do equipamento utilizado. Esta nova versão incorpora para armazenamento das informações uma estrutura de dados armazenada através de um processo semelhante a memória virtual e que pode ser transportada facilmente para qualquer equipamento.

O tempo de projeto do EDMOS foi extremamente curto (um mês, um projetista), principalmente pela filosofia de uso direto da memória da tela, e pelo fato do processo de edição, em si, ser um processo puramente geométrico, com pouca necessidade de estruturas de dados sofisticadas.

Entre as principais facilidades do EDMOS, temos a criação e retirada de retângulos coloridos (box de CIF), ajuste de trechos, cópia, zoom, replicação de trechos, espelhamento, rotação, etc., que provêm a maioria das facilidades necessárias para a edição de células.

Entretanto o EDMOS não é capaz de editar layouts maiores que o tamanho da tela (240 x 400 lambda), embora com alguns macetes de projetista, isto tambem seja possível. Entretanto, poucas células são projetadas com mais do que este tamanho (não se contando com células que são compostas de outras células). Tanto as facilidades de montagem do layout quanto a hierarquia de células são providas pelo outro editor, EDCI, descrito mais adiante.

ALGUNS COMANDOS DO EDMOS

a) o cursor e sua movimentacao

O cursor é um retângulo branco, que pode ser movido, expandido e encolhido. Com este cursor delimitamos a área que nos interessa. O cursor é alterado usando exclusivamente o teclado. Este processo é mostrado em (Borges-83).

b) Modos de operação

Criamos dois modos de operação: rápido e lento. O primeiro é usado basicamente para mover o cursor e introduzir novos retângulos no desenho, teclando-se sempre uma tecla de cada vez. O modo lento é iniciado teclando-se RETURN e a seguir um mnemônico da função a executar. Neste modo, os comandos são interativos, ou seja, perguntam explicitamente todos os parâmetros envolvidos.

c) Comandos de movimentação de áreas

Existe uma série de comandos cuja finalidade é ajeitar o layout através da cópia ou movimentação de trechos. Entre estes comandos, temos espelhamento, rotação, cópia, repetição (matriciação), ajuste, zoom. Estes comandos são implementados muito facilmente através do manuseio direto da memória de tela.

d) Visibilidade

Uma das operações mais importantes é a visibilidade, ou seja, ver-se somente algumas camadas do chip. Isto é obtido através de um mascaramento feito por hardware no gerador de cores, permitindo um esforço de software mínimo.

e) pegar e guardar um layout no disquete

Os layouts são gravados em CIF e podem ser trazidos para a posição do cursor, rodados ou espelhados. Da mesma forma, também podem ser jogados em disco em CIF.

f) DRC online

O EDMOS está conectado diretamente ao verificador de regras de projeto (DRC) projetado por (Teles-83), o que permite a verificação imediata dos layouts (ou parte deles) durante o processo de edição. Neste caso são mostrados piscando na tela os trechos os quais apresentam algum tipo de violação geométrica, como descrito em (Mead & Conway - 80).

HARDCOPY

O EDMOS prevê saídas para impressora alfanumérica e gráfica. No caso de impressora alfanumérica, cada pixel é representado pela sobreimpressão de 5 caracteres, representando cada uma das camadas NMOS. Os seguintes caracteres foram utilizados por darem uma boa sobreimpressão: O (difusão) / (polissilício) V (metal) . (implantação iônica) W (contato).

A impressão gráfica pode ser feita em uma impressora de pontos barata (Globus-M100) , que possui facilidade de se especificar diretamente as configurações das agulhas na linha (modo gráfico). A impressora tem uma pequena diferença entre as distâncias X e Y, mas no caso de checkplot isso não faz muita diferença.

A codificação para a impressora gráfica é feita através de uma matriz de 4 x 4, utilizando uma padronização utilizada na XEROX PARC. Isto permite a impressão numa página de um trecho com largura 198 lambda. A impressão de uma área com 198 x 198 lambda demora cerca de 5 minutos.

A vantagem do uso de impressora gráfica é grande. A rapidez e a relativa qualidade do desenho obtido permitem que se evite tirar saídas em plotter, que apresenta uma certa dificuldade de visualização (Borges-83) e um tempo enorme de desenho. O ideal, seria, entretanto, uma impressora gráfica colorida, mas esse é um equipamento difícil de obter e muito caro.

FUNCIONAMENTO E PRINCIPAIS ALGORITMOS DO EDMOS

O EDMOS trabalha preenchendo áreas da tela com cores, e consultando o conteúdo da tela. Isso, na tecnologia utilizada (raster scan) é extremamente simples, pois basta consultar diretamente a memória da tela. Para aumentar a portabilidade do software, criamos um conjunto de funçoes que implementam essas primitivas de acesso:

ESCVID (x,y,cor) : colore uma posição da tela
LEVID (x,y,cor) : informa a cor de um ponto da tela
RETANG (x,y,dx,dy,cor,opção) : desenha um retângulo colorido, superpondo (fazendo "ou" dos bits) ou reescrevendo os pontos
MASCOR (masc) : seleciona a máscara de visibilidade (video lookup table).

Os algoritmos mais interessantes são o que desenha o cursor e o que extrai o desenho da tela, transformando-o para CIF.

O cursor é um retângulo branco, que pode ser movido na tela. Não existe no nosso equipamento cursor gerado por hardware. Para implementar esta função, ao colocar o cursor na tela, copiamos previamente o conteúdo das posições da tela onde ele será colocado para uma área auxiliar e substituímos estes pontos pela cor branca. Para apagar o cursor recolocamos os pontos a partir da área auxiliar. A movimentação do cursor é composta de um apagamento seguido de um redesenho do cursor. Sempre que alguma função vai ser executada, o cursor é previamente apagado, e terminada a função, é recolocado.

A extração do desenho é feita varrendo-se a área delimitada por cursor, linha a linha, de baixo para cima, localizando-se as sequências de pontos com a mesma cor. É mantida uma lista com as sequências da linha anterior. As novas sequências são comparadas com as da lista, procurando-se achar sequências idênticas. Vai-se acumulando o tamanho(no sentido vertical) das sequências idênticas, ou seja, vai-se obtendo o tamanho dos re - tângulos. As sequências da linha atual sem equivalência na lista, são inseridas (pois são na verdade novos retângulos). As sequências da lista, sem equivalência na sequência da linha atual, são consideradas como fim de retângulo, e retiradas (ou seja, é da da sua saída em CIF).

DA NECESSIDADE DE UM EDITOR HIERÁRQUICO

À primeira vista, o projeto de um circuito integrado pode ser comparado ao projeto top down de software. Tem-se, a partir da idéia geral do problema, a divisão do projeto em blocos, a confecção da planta baixa (que é a divisão em partes do espaço físico), e o projeto de cada célula, que pode conter subcélulas, e assim por diante. Podem existir células pré-programadas (standard cells, bibliotecas de células), com entradas e saídas bem definidas que são utilizadas por todos os projetistas. O projeto das células é feito às vezes como caixas vazias em que existem apenas definidos os pontos de conexão,em analogia com subrotinas "dummy", que possuam especificados apenas seus parâmetros formais.

Infelizmente, uma abordagem puramente top-down é utópica neste caso. A alocação do espaço físico é uma outra dimensão que ultrapassa a noção algorítmica do projeto. Existem neste caso problemas geométricos que tornam muito complexo o projeto: o encaixe das células, sua interligação, minimização do layout, alimentação elétrica, etc...

Daí, dividimos o problema em duas partes: projeto das células e montagem das células . Essas duas partes se interrelacionam e criam um círculo vicioso: o projeto depende das especificações para a montagem e a montagem depende do projeto.

Essa situação é sempre minimizada quando se utilizam células projetadas (por exemplo , obtidas em bibliotecas de células) ou geradas por geradores automáticos como descritos em (Teles & Schmitz-85). O que acaba contando, é a experiência anterior do projetista, que se torna capaz de estimar com relativa precisão o tamanho das células e arbitrar as disposições de montagem que dêem como resultado um layout mais compacto. Entretanto, este é um problema que permanece, e para o qual não existe ainda uma solução definitiva.

Para auxiliar a parte de montagem, é fundamental um editor que consiga ajudar no pro - cesso da alocação do espaço e ligação das partes componentes do circuito. Este editor deve ter a noção de hierarquia, ou seja, ser capaz de tratar da colocação de células em disposição umas dentro das outras, recursivamente, para que a montagem e projeto tenham uma interface menos complicada.

VISÃO GERAL DO EDCI

O EDCI foi projetado para poder rodar tanto no computador principal,utilizando a esta-

ção de trabalho apenas como unidade de saída, ou executar mesmo na própria estação de trabalho, de maneira stand-alone. Essa independência é obtida pela utilização de rotinas gráficas modulares e pelo fato do editor ser escrito em "C" bastante padrão.

Da mesma forma que o EDMOS, ele usa uma tela alfanumérica e uma tela gráfica. Na tela alfanumérica se mantém todo o diálogo com o operador, e são mostrados todos os status da edição (posição do cursor, nome do símbolo e do layout, etc...). Na tela gráfica , aparece o layout que se está projetando, numa escala arbitrária. Essa separação de telas possibilitou o uso de terminais gráficos com resolução bastante modesta (240 x 400 pontos).

No EDCI, o chip é tratado como um caixote visto do alto. Dentro deste caixote podem existir três tipos de coisas: retângulos coloridos (box de CIF), textos ou outros caixotes (símbolos mais internos, células de CIF). Dentro dos caixotes internos podem existir outros retângulos, textos e caixotes, e assim sucessivamente.

Foi minimizada a quantidade de coisas que o editor desenha automaticamente. Como agora estamos tratando de layouts completos, o tempo de desenho pode ser muito grande. É melhor, então,que seja opção do operador, por exemplo, mostrar um símbolo apenas através de seu contorno e conexões, do que todo o interior expandido. Além disso, é permitido o uso de escala menor que 1, ou seja, pode-se ter uma visão macroscópica do chip e nesse caso o numero de elementos desenhados na tela é enorme, caso se deseje ver o conteúdo de todos os subsímbolos.

OPERAÇÃO DO EDCI

Inicialmente é mostrado na tela o contorno de um retângulo, representando o chip, e dentro dele, outros retângulos vazios, representando o nível imediatamente inferior de células (ou seja as células chamadas pelo nível mais global), e os retângulos (Boxes CIF) coloridos deste nível.

Existe um cursor retangular que o operador pode mover de forma semelhante ao EDMOS para se posicionar dentro do layout. Eventualmente, o operador vai querer entrar num subsímbolo, alterar seu conteudo, entrar num subsímbolo ainda mais interno, sair dele, e assim por diante, passeando de uma forma relativamente confortável dentro do chip. É possível isolar um trecho do layout ou um símbolo para edição em escala maior, alteração de tamanho dos símbolos ou seu posicionamento, inclusão, criação e retirada de elementos no circuito, etc... O controle da posição de navegação dentro do chip é feita completamente de maneira visual, ou seja, através do cursor. Para manter-se a noção de hierarquia, ao sair de um símbolo (opcionalmente) pode-se mandar recobri-lo, e assim tornar a visualização mais simples.

Os comandos do EDCI implementados em sua primeira versão são os seguintes:

- movimenta cursor
- cria, remove, entra e sai de símbolo
- cria e remove box

. ajusta tamanhos e posições de símbolos
. expande chamadas de simbolos até qualquer nível
. isola, amplia e reintegra símbolos
. cria matrizes de símbolos
. salva e recupera o conteúdo do cursor
. cria e remove hierarquias
. traz simbolo em CIF para o banco de dados ou dele extrai um arquivo em CIF
. mostra, altera, manuseia tabela de símbolos
. seleciona janela de edição, visibilidade, escala
. comando de ajuda online para todos os comandos
. possui interface para conexão com o DRC e o gerador de rotas automático.

ALGORITMOS

Os algoritmos deste editor e suas principais estruturas de dados são baseados em uma árvore que define as relações de hierarquia e o conteúdo das células. No manuseio desta árvore, são utilizados algoritmos conhecidos de busca, manuseio, inclusão e exclusão em árvores aplicadas a processamento gráfico, como descritas em (Newman & Sproull), bem como toda a parte de manuseio de transformações matriciais descritas naquela referência.

Os principais algoritmos podem ser achados em (Pires & Borges-85).

CONCLUSÃO

O EDMOS e o EDCI estão em funcionamento no NCE-UFRJ e são utilizados em cursos de graduação e pós-graduação de projeto VLSI. São utilizados ainda pela equipe de pesquisa em projeto VLSI do NCE (Silva Fº, Schmitz, Faller -83), e foram importantes ferramentas no projeto de um dos primeiros chips que foram fabricados no Brasil.

A versão atual destes editores é destinada à tecnologia NMOS, mas eles foram construídos de forma paramétrica, para permitir um mínimo esforço de conversão para outras tecnologias (por exemplo, CMOS ou HMOS). São relativamente portáteis, e rodam com um mínimo de necessidades computacionais.

A parte gráfica foi projetada especialmente para poder trabalhar com qualquer tamanho de tela e o protocolo gráfico está centralizado em uma única rotina para possibilitar seu transporte para virtualmente qualquer terminal gráfico Raster scan colorido.

1 . BORGES, J.A.S. Ferramentas gráficas para o Projeto de Circuitos Integrados
Anais do Congresso Nacional de Informática - 1983

2 - LEWIS, D. Layout design on an alphanumeric terminal
VLSI Design, vol. II, no. 4 (1981)

3 - FAIRBAIRN, D et ROWSON, J. ICARUS: an interactive integrated circuit layout program anais da 15a. Design Automation Conference, 1978

4 - NEWMAN, W. et SPROULL Principles of interactive computer graphics
2nd. edition - MacGraw Hill - 1979

5 - PIRES, O.V.S. et BORGES, J.A.S. Estruturas de dados e algoritmos de um editor hierárquico para projeto VLSI - submetido à comissão de seleção do II SBCCI - 1985

6 - OUSTERHOUT, J. CAESAR: an interactive editor for VLSI layouts
VLSI Design, vol. II, no. 3 (1981)

7 - TELLES, A.A.S. Verificação de Projeto de Circuitos Integrados
Tese de mestrado - COPPE-UFRJ - 1983

8 - SILVA FQ, Y., SCHMITZ, E., et FALLER, N. É possivel projetar circuitos integrados no Brasil ? Anais do Congresso Nacional de Informática - 1983

9 - TELLES, A.A.S et SCHMITZ MAKEROM - um gerador automático de ROM
Submetido à comissão de seleção do II SBCCI - 1985