M.F. MARTINS* e E.A. SCHMITZ**

## SUMÁRIO

Este artigo apresenta os algoritmos, a implementação e os resultados do programa roteador de canal que faz parte do conjunto de ferramentas do projeto de CI's do NCE/UFRJ.

## ABSTRACT

This paper describes the algorithms, implementation and results of the channel router program, which is included in the IC design tool set of the NCE/UFRJ.

* Engenheiro eletrônico (UFRJ, 1972), M.Sc. COPPE - UFRJ, 1975; Projeto Circuitos Digitais;

** Engenheiro eletrônico (UFRGS, 1971), M.Sc. COPPE - UFRJ, 1973; Ph.D. Imperial College 1980; Projeto de Circuitos Integrados.

Endereço de ambos os autores: Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro, CCMN, Cidade Universitária - Ilha do Fundão, Caixa Postal 2324 , CEP 20001, Rio de Janeiro, telefone (021) 290-3212, ramal 290 e 285.

O conjunto de ferramentas de projeto do NCE/UFRJ contem programas para executar diversas tarefas encontradas no decorrer do projeto de um circuito integrado LSI.

Como ferramentas típicas temos: simulador funcional, editor gráfico, gerador de módulos, verificador de regras de projeto, extrator, simulador e programas de plotagem. Estas ferramentas permitem definir, projetar e verificar um módulo ou um conjunto de módulos. Para completar este conjunto de ferramentas sentimos a necessidade de um programa de roteamento automático entre módulos.

Os atributos requeridos deste programa em ordem de importância são:
1. Integração com o conjunto existente de ferramentas. Isto implica que o programa seja compatível com o editor gráfico EDCI e que dê como resultado um arquivo no formato CIF;
2. Execute rapidamente de forma a poder oferecer ao projetista a opção de tentar vários arranjos de planta-baixa e escolher o que oferece a melhor forma de conexão.

Acreditamos que ambos os objetivos foram plenamente alcançados, como tentaremos mostrar no restante deste artigo.

## O PROBLEMA DO ROTEAMENTO DE CANAL

Consideremos um canal retangular com 2 linhas de terminais uma no topo e a outra na base. A cada terminal é assinalado um número inteiro de Ø a N sendo o Ø usado para representar ausência de conexão. Terminais com o mesmo número i ($1 < i < N$) são ligados entre si por meio da rede i.

Duas camadas são disponíveis para o roteamento: uma para as trilhas horizontais e a outra para as verticais estando dessa forma estas isoladas daquelas. A conexão entre uma trilha horizontal e uma vertical é feito então por meio de um furo. Cada rede possui no máximo uma trilha horizontal ligada aos terminais superiores e inferiores por meio de trilhas verticais. O problema é expresso por meio de uma lista de redes como mostra a figura 1 e uma solução aparece na figura 2.

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 4 | 5 | 1 | 6 | 7 | 0 | 4 | 9 | 10 | 10 |
| 2 | 3 | 5 | 3 | 5 | 2 | 6 | 8 | 9 | 8 | 7 | 9 |

Fig. 1

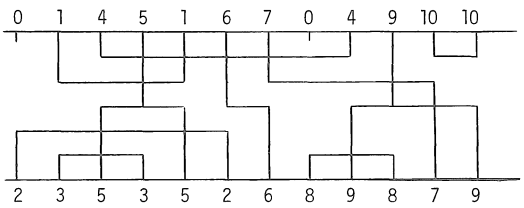
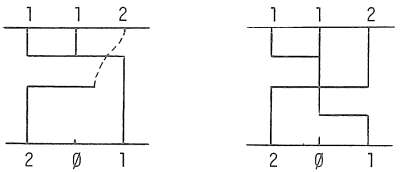



Fig. 2

Um problema que pode surgir durante o roteamento é denominado "conflito cíclico" exemplificado abaixo.

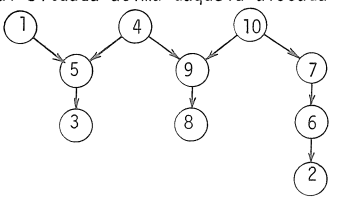



Neste caso uma solução seria alterar a ordem de sinais, o que é relativamente simples em se tratando de PLA's ou alocando mais de uma trilha horizontal para um ou mais sinais. O programa que desenvolvemos possui uma rotina para deteção de ciclos imprimindo ao primeiro ciclo detetato a sequência de redes que o compõe facilitando assim uma alteração na ordem dos sinais, abortando em seguida.

O objetivo do roteamento é minimizar o número de trilhas horizontais de modo que a largura e consequentemente a área do canal seja mínima.

RESTRIÇÕES VERTICAL E HORIZONTAL

Supondo que há somente um segmento horizontal para cada rede é claro que o segmento horizontal de uma rede conectada ao terminal superior de uma coluna qualquer deve estar situado acima do segmento correspondente a rede conectada ao terminal inferior daquela coluna. Esta relação pode ser representada por meio de um grafo orientado onde cada nó representa uma rede e cada seta dirigida de a para b significa que a rede a deve utilizar uma trilha horizontal situada acima daquela alocada para b.




Em geral quando há um caminho de cima para baixo de i para j dizemos que i é ancestral de j e j é descendente de i.

REPRESENTAÇÃO POR ZONAS

O segmento horizontal de uma rede é delimitado por suas conexões mais a esquerda e mais a direita. Sendo S(i) o conjunto de redes cujos segmentos interceptam a coluna i, desde que os segmentos de redes distintas não devem se superpor, para cada rede em S(i) devemos alocar uma trilha horizontal. Esta condição deve ser satisfeita para cada coluna mas, como é fácil ver, basta considerar apenas os conjuntos S(i) que não são sub-conjuntos um do outro. Para cada um deste conjunto corresponde um certo grupo de colunas ao

lingo do canal e que chamamos de zona indicada pela tabela abaixo. A direita aparece uma representação mais completa onde podemos visualizar que redes começam e/ou terminam em cada zona.

| Coluna | S(i) | Zona |
|---|---|---|
| 1 | 2 | |
| 2 | 1 2 3 | |
| 3 | 1 2 3 4 5 | 1 |
| 4 | 1 2 3 4 5 | |
| 5 | 1 2 4 5 | |
| 6 | 2 4 6 | 2 |
| 7 | 4 6 7 | 3 |
| 8 | 4 7 8 | |
| 9 | 4 7 8 9 | 4 |
| 10 | 7 8 9 | |
| 11 | 7 9 10 | 5 |
| 12 | 9 10 | |

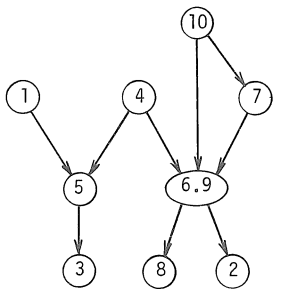
zona

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 1 | | 7 | | |
| 2 | | | 8 | |
| 3 | | | 9 | |
| 4 | | | | 10 |
| 5 | 6 | | | |

FUSÃO DE REDES

Antes de descrever o algoritmo definiremos a seguinte operação:
Sejam 2 redes i e j tais que:
1. Não há superposição horizontal na representação em zonas;
2. Não há um caminho orientado entre os nós i e j no grafo.

A fusão da rede j na rede i (dizemos que j foi "absorvido") consiste então em:
1. Modificar o grupo de restrição vertical "deslocando" o nó j até superpô-lo com o nó i. Consequentemente acrescentamos ao nó i os descendentes de j. Além disso todos os ancestrais de j passam a ser ancestrais de i;
2. Na representação em zonas a rede resultante passa a ocupar 2 zonas consecutivas (as zonas de i e de j).

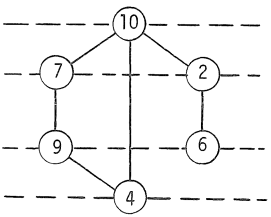



zona

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 1 | | 7 | | |
| 2 | | | 8 | |
| 3 | | | | |
| 4 | | | | 10 |
| 5 | 6.9 | | | |

O algoritmo a seguir funde redes sistematicamente de acordo com a representação em zonas.

O grupamento dos nós por geração não é tão trivial como pode parecer a primeira vista. O nó 11 e o nó 4, embora sejam ambos filhos do nó 3, não pertencem a mesma geração uma vez que existe um caminho do nó 11 até o nó 4. Um outro exemplo, ainda mais significativo, aparece abaixo. Embora haja um caminho direto do nó 10 (no topo) até o nó 4 este encontra-se hierarquicamente abaixo do nó 6 onde não há tal caminho. O critério para definição das gerações consiste então em considerar sempre os caminhos mais longos.




No algoritmo para alocação de trilha que aparece a seguir foi empregado o termo descendente para indicar que existe algum caminho entre 2 redes e o termo filho no sentido restrito em que a distância é igual a 1.

```
t = 1;
P = [redes do topo do grafo];
Enquanto P ≠ [] faça
   Início
      Q =[ ];
      Para cada rede r em P faça
         Início
            Atribuir trilha t para todas as redes fundidas com r;
            t = t + 1;
            Q = Q + [filhos de r]
         Fim.
      Para cada rede r em P faça
      P = P - [descendentes de r]
   Fim.
```

OTIMIZAÇÃO DAS ROTAS

O programa de roteamento tinha como objetivo inicial apenas o de minimizar o número de trilhas horizontais e com isso a área do canal. Observando, no entanto, diversos resultados verificou-se que embora esta condição fosse satisfeita ocorria que algumas redes se apresentavam desnecessariamente longas como mostra o exemplo a seguir.

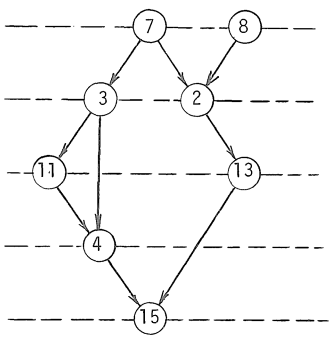
```
L = [ ];
para z de zi até zf faça
  Início
     L = L + [redes que terminam na zona z],
     R = [redes que começam na zona z + 1];
     Fundir redes de L com redes de R de modo a minimizar o aumento do mais longo cami
     nho no grafo;
     L = L - [redes fundidas no passo anterior]
  Fim.
```

Antes de prosseguir devemos observar pelas definições de L e R que nunca haverá restrição horizontal entre uma rede qualquer de L e uma qualquer de R. O passo chave do algoritmo é o que realiza a função entre as redes de L e R. Se há um caminho $n_1$ - $n_2$ - ... $n_k$ no grafo de restrição vertical então 2 quaisquer dessas redes não podem ocupar a mesma trilha horizontal, ou seja, se no mais longo caminho o número de nós é k pelo menos k trilhas serão necessárias para fazer as interconexões. A fusão de redes deve portanto ser feita de modo que o mais longo caminho após a fusão seja tão pequeno quanto possível.

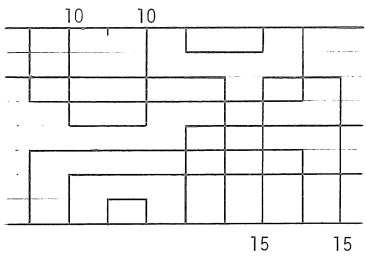
Para minimizar o mais longo caminho usamos uma série de considerações heurísticas descritas no artigo de Takeshi Yoshimura e Ernest S. Khu: "Efficiente Algorithms for Channel Routing". (IEEE TRANS on CAD, V. 1, No. 1, JAN 82, pp. 25-35).

ALOCAÇÃO DE TRILHAS

Uma vez chegando ao grafo na sua forma mais simples, ou seja, com o menor número possível de nós resta então fazer a alocação de trilhas. Para cada nó - que na verdade corresponde a um grupo de redes que se fundiram - devemos alocar uma trilha. Essa alocação deve levar em conta a hieraquia entre os nós estabelecida pelo grafo. Devemos portanto começar pelo topo do grafo identificando a partir daí cada uma das gerações até chegar à base como indicado abaixo.

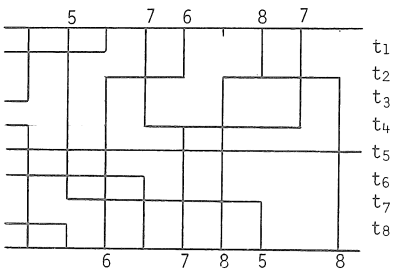



| GERAÇÃO | NÓS | TRILHAS |
|---|---|---|
| 1 | 7,8 | 1,2 |
| 2 | 2,3 | 3,4 |
| 3 | 11,13 | 5,6 |
| 4 | 4 | 7 |
| 5 | 15 | 8 |




A rede 10 poderia ocupar a trilha 1 em vez da 3 e a rede 15 ocupar a trilha 7 em vez da 2. Visualmente é simples fazer o deslocamento para cima ou para baixo mas como programar este tipo de tarefa? Antes de mais nada o sentido do deslocamento depende das conexões da rede ao longo do canal; uma rede deve ser deslocada para cima ou para baixo conforme o número de conexões para cima ou para baixo seja maior respectivamente. Toda rede com igual número de conexões não necessitaria se deslocar. Foi criado então um vetor (CONEXÕES: ARRAY [0 ... MAX REDES] OF INTEGER) inicializado com tudo zero. A rotina LE DADOS ao ler o arquivo de entrada "fios.dat" vai incrementando ou decrementando cada elemento do vetor e no final teremos então, associado a cada rede, um inteiro positivo, zero ou negativo indicando qual o deslocamento a ser feito.

Uma vez identificadas as redes a serem deslocadas e qual o sentido de deslocamento o passo seguinte consiste em determinar, para cada uma delas quais as redes que limitam seus deslocamentos. Este tipo de problema aparece ilustrado abaixo.




A rede 7 deve ser deslocada para cima o máximo possível. Acima dela estão livres as trilhas 1 e 3. A trilha 1, no entanto, não pode ser aproveitada por causa da rede 6 que funciona como um "bastante" impedindo que a rede 7 atinja aquela trilha. Da mesma forma a rede 8 não poderia ocupar a trilha mais baixa por causa da rede 5. Deveria ocupar então a trilha 6.

Este segundo passo sugere o uso de um vetor de vetores: a cada rede é associado um conjunto de redes "batentes". Na prática mostrou-se mais conveniente o uso de um vetor de records tendo por componentes um vetor e uma escala indicando o número de redes "batentes" encontradas.

Quando uma rede qualquer tem maior número de conexões para cima ela deve ser deslocada para cima e neste caso devem ser listadas todas as redes ao longo do canal que apareçam acima dela. Se o número maior de conexões for para baixo devem ser listadas aquelas que pareçam abaixo. Em um trecho da procedure CRIAGRAFO todo o canal é percorrido e para cada rede encontrada (in ou sup) um dos testes é feito: conexões [inf] > 0 ou conexões [sup] < 0. No primeiro, se verdadeiro, a rede batente será sup e no segundo será inf.

O próximo passo consiste em reunir em um vetor todas as redes a serem deslocadas no mesmo sentido (para cima ou para baixo). Estas redes serão então deslocadas uma a uma a partir do primeiro. Antes disso porém fazemos uma primeira ordenação das redes em ordem crescente de largura e a seguir uma ordenação em ordem decrescente de saldo de conexões. Teremos, após estas 2 ordenações, situadas em primeiro lugar na lista as redes com maior saldo de conexões e portanto com maior redução no comprimento devido ao deslocamento. Se 2 ou mais redes tem mesmo saldo serão deslocadas primeiro aquelas com menos largura e portando maior probabilidade de encontrar trilhas disponíveis.

IMPLEMENTAÇÃO

O programa foi escrito em Pascal, tendo aproximadamente 1300 linhas. Apresenta a saída em dois formatos: a primeira em impressora e a segunda na forma de um arquivo CIF contendo a descrição do lay-out na tecnologia NMOS.

As figuras seguintes mostram exemplos de dois canais gerados pelo programa, com tempos de execução iguais a 1,5 minutos para 17 trilhas e 2,5 minutos para 45 trilhas (num microcomputador com processador 68000 de 8MHZ).