

Síntese de layout

Roteamento



Manoel Eusebio de Lima

Roteamento

- Roteamento é a fase na qual as redes do chip são finalmente conectadas e o circuito finalizado.
- O processo de roteamento é em geral dividido em duas fases:
 - Roteamento global (Global routing)
 - Roteamento detalhado (Detailed Routing)

Roteamento global

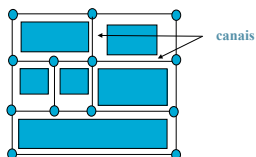
- Roteamento global, também conhecido como *"loose routing"* ou *"topological routing"* tem como objetivo decompor um grande problema de roteamento em pequenos e gerenciáveis problemas, facilitando o trabalho dos roteadores finais (roteamento detalhado).
- Este tipo de decomposição é feita pela sugestão de posicionamento de cada rede do circuito a um certo conjunto de regiões de roteamento (canais). Durante o estágio do roteamento global, o roteador conhece apenas a capacidade das áreas para roteamento e o número de fios que são selecionados para cruzar as respectivas áreas.

Roteamento global

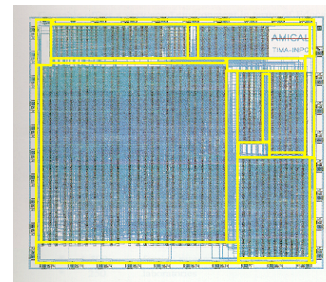
- Funções do roteador global:
 - planejar o roteamento globalmente;
 - reduzir o tamanho do chip;
 - reduzir o tamanho das interconexões e portanto o atraso;
 - garantir rotabilidade (100% de conexão);
 - balancear a congestão do layout;
- Antes do roteamento certos detalhes por onde serão feitos as conexões devem existir, tais como:
 - definição de canais em estruturas como standard-cells e gate arrays;
 - Definir canais para macrocélulas, os quais não são em geral bem definidos. Exceção ocorre, por exemplo, quando o posicionamento/floorplanning usa técnicas de particionamento.

Roteamento global - Modelos

- **Channel-graph**
 - Defina um grafo das interconexões dos canais, no qual cada vértice do grafo representa a interseção de canais e cada ramo do grafo representa um canal ou uma seção do canal. Este modelo adota pesos para cada canal para modelar o tamanho do fio e a capacidade do canal. Tal grafo pode ser usado para selecionar uma seqüência de canais para uma determinada rede.

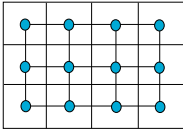


Roteamento global - Modelos



Roteamento global - Modelos

- Coarse grid**
 - É possível tirar vantagens da regularidade de estrutura como standard-cell e gate-arrays desde que o grafo representa um grid regular.

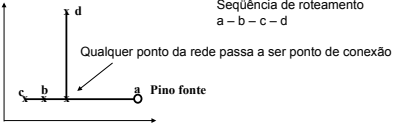


- A capacidade de fiação das células formada por esta estrutura é facilmente computável.
 - Um custo é associado a cada ramo no grafo baseado na capacidade de trilhas em cada região ao redor da célula.

Neste caso o tamanho dos fios pode ser modelado implicitamente desde que cada célula tem o mesmo tamanho das demais. Esta consistência simplifica a determinação dos tamanho dos caminhos para roteamento.

Roteamento global - Estimativa de tamanho do roteamento

- Steiner Tree**
 - Este modelo encontra o caminho mais curto para conexão de pinos. Ele permite que um fio da rede possa ser conectado em qualquer ponto ao longo da rede. Este processo é bastante complexo desde que se deve saber qual o melhor ponto de derivação da rede e a melhor rota a ser tomada a partir deste ponto aos pinos.

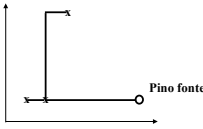


Seqüência de roteamento
 $a - b - c - d$

Qualquer ponto da rede passa a ser ponto de conexão

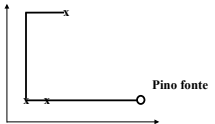
Roteamento global - Estimativa de tamanho do roteamento

- Minimal Spanning tree**
 - Este método permite derivação apenas na posição dos pinos.
 - Diferentemente do Steiner Tree a conexão com a rede é permitida apenas nos nós da rede.
 - Para uma rede de n pinos, a árvore pode ser construída determinando-se a distância entre todos os possíveis pares de pinos e conectando ao menores $(n-1)$ arcos.
 - Minimum spanning tree descreve a rede mais barata para conectar todos os nós de um dado conjunto de vértices.



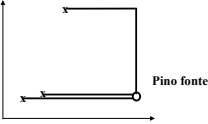
Roteamento global - Estimativa de tamanho do roteamento

- Chain connections**
 - Este método não permite qualquer tipo de derivação. Cada pino é simplesmente conectado ao próximo pino na forma de uma corrente. Estas conexões são mais simples de serem implementadas que "spanning tree", mas o resultado final apresenta interconexões de tamanho maior.



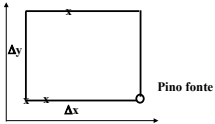
Roteamento global - Estimativa de tamanho do roteamento

- Source-to-sink connections**
 - Neste método a saída de um módulo é conectado a todas as entradas por meio de fios separados. Este é o método mais simples de implementação.
 - Este modelo é pouco usado. Ele gera um tamanho excessivo de interconexão em áreas de baixa congestão, embora apresente uma boa aproximação para regiões congestionadas.



Roteamento global - Estimativa de tamanho do roteamento

- Semi-perimeter method**
 - Este método é bastante eficiente e o mais utilizado na estimativa de tamanho de uma rede. O tamanho do fio é dado pela metade do perímetro do menor retângulo que engloba todos os pinos da rede.
 - Consegue bons resultados para rede de dois a três pinos.
 - Para regiões altamente congestionadas este método sempre subestima o tamanho das redes.



Roteamento global - Exemplo de heurística

Algoritmo Steiner_Tree

O problema de "Steiner Tree" é um problema NP-hard. Portanto, em vez de conseguirmos um resultado mínimo, ou mínima distância em uma rede, através de heurísticas, identificamos rapidamente possíveis soluções de bom tamanho.

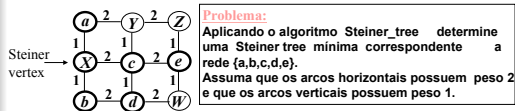
Em geral heurísticas de "Steiner tree" usam modificações de outros algoritmos para o cálculo do caminho mais curto entre dois pinos como o algoritmo de Dijkstra ou variação do algoritmo de Lee.

Usualmente as heurísticas são processadas no modelo "greedy".

Algoritmo Steiner_Tree;

```

Begin M <- Conjuntos de nós marcados (*nós nos quais a rede tem pinos*)
s <- seleccione um nó de M;
M <- M - {s};
Aplica-se Algoritmo_Dijkstra para encontrar a menor distância de
s para algum nó e de M,  $\pi_{s,e}$ ;
(* O Algoritmo_Dijkstra encontra o menor caminho entre dois pontos
em um grafo com pesos*);
M <- M - {e};
V <- {s,e}; (* nós da Steiner tree* *)
While M ≠ ∅ Do
Begin
e <- next(M); (* apanha outro nó de M*)
Aplica Algoritmo_Dijkstra para encontrar o caminho mais curto
de e para algum node x de V;
V( $\pi_{e,x}$ ) <- nós cobertos por  $\pi_{e,x}$ ;
(*remove nós marcados cobertos pelo caminho  $\pi_{e,x}$ 
M <- M - M ∩ V( $\pi_{e,x}$ );
EndWhile
End.
    
```



Problema:
Aplicando o algoritmo Steiner_tree determine uma Steiner tree mínima correspondente a rede {a,b,c,d,e}.
Assuma que os arcos horizontais possuem peso 2 e que os arcos verticais possuem peso 1.

Solução:

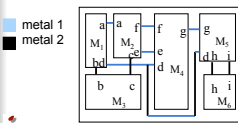
- Os vértices do grafo são {a,b,c,d,e,X,Y,Z,W}
- Inicialmente $V = \Phi$ e o conjunto de vértices marcados $M = \{a,b,c,d,e\}$
- Selecione o nó a como nó inicial.
- Encontre o menor caminho de a para o mais próximo nó marcado.
- $\pi_{a,b} = \{a,X,b\}$. Assim $V = \{a,X,b\}$ e $M = \{c,d,e\}$
- Assuma c como novo nó para cálculo da menor distância.
- O menor caminho de c para qualquer dos vértice de V é $\pi_{c,x} = \{c,X\}$.
- Portanto $V = \{a,X,b,c\}$ e $M = \{d,e\}$.
- Assume d o próximo nó a ser selecionado.
- O menor caminho de d para qualquer dos vértice de V é $\pi_{d,c} = \{d,c\}$.
- Assim os conjuntos V e M passam a ser $V = \{a,X,b,c,d\}$ e $M = \{e\}$.
- Finalmente o vértice e é selecionado e conectado ao nó c.

Roteamento detalhado

Roteamento detalhado

Dada uma região com pinos em seus lados e possivelmente no centro, roteamento detalhado é o processo de implementação da geometria das interconexões entre os pinos especificados pela netlist de um determinado circuito.

A região onde os pinos estão localizados podem ter formas retangulares ou não, e irregulares com obstruções internas.



O roteamento detalhado pode ser feito em vários níveis de metalização, em geral, com preferencial de nível de metal por direção no roteamento. No exemplo ao lado metal 1 tem preferência na horizontal e metal 2 na vertical.

Considerações com relação ao tamanho de rede no roteamento

As redes são consideradas para análise de roteamento em função de seus tamanhos:

Procedimentos:

- Rotear as redes mais simples primeiro. Isto significa o roteamento daquelas redes que possuem apenas dois ou três pinos e as quais ocupam uma simples linha. Isto representa aproximadamente 75% das redes em um projeto normal.
- Use um algoritmo de "Steiner tree" para rotear redes de tamanho intermediário.
- As redes restantes (grandes redes) com muitos terminais devem ser roteadas usando "maze routers".

Obs: o fato de 75% das redes caírem dentro da categoria de redes simples é importante. Uma vez estabelecido o roteamento, nós temos um modelo mais apurado da congestão antes de começarmos as conexões grandes.

Roteamento detalhado - exemplo

Algoritmo de Lee

- Considere um plano para roteamento para facilitar a estratégia, embora esta técnica possa ser usado para situações multi-planos.
- Denominemos um ponto como fonte (source) e um outro como destino (sink)
- Definamos de nó cada possível posição no plano por onde pode haver passagem de uma conexão elétrica (um caminho para roteamento).
- O algoritmo faz uso de duas listas de nós
 - NEW - guarda os nós vizinhos de um nó quando não foram ainda visitados na pesquisa e não são bloqueados para roteamento
 - OLD - guarda os nós já visitados.

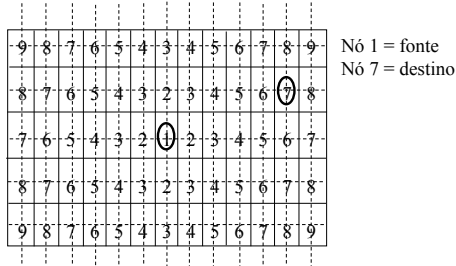
Roteamento detalhado

Algoritmo de Lee

- LM1 - inicialização
 - Marque o nó fonte como visitado
 - OLD ← nó fonte
- LM2 - Propagação: comece com uma lista NEW vazia
 - Para cada nó em OLD
 - Crie conjunto NEW ← os nós vizinhos do corrente nó que não foram visitados e não estão bloqueados
 - Marque nós em NEW como visitados e a direção que foram visitados.
 - IF sink pertence a NEW, go to LM3
 - Else coloque NEW em OLD, OLD ← NEW
 - IF NEW é vazio, o caminho não existe, "Erro – não é possível rotear"; end.
- go to LM2
- LM3 - **Backtrace**: começando do nó destino(sink), siga as direções anotadas dos nós para o nó fonte.

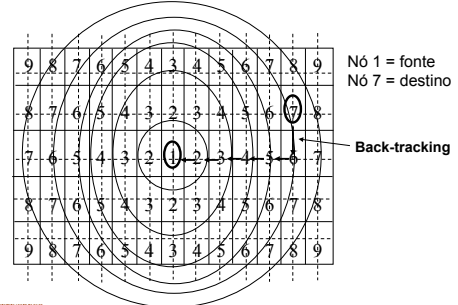
Roteamento detalhado

Algoritmo de Lee-Moore



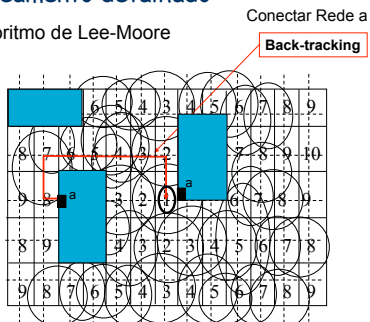
Roteamento detalhado

Algoritmo de Lee-Moore



Roteamento detalhado

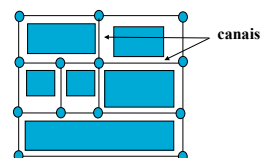
Algoritmo de Lee-Moore



Roteamento detalhado

Channel routing (roteamento de canal)

- Este tipo de roteamento normalmente se prevalece da técnica de particionamento chamada "divider and conquer" (dividir e conquistar), na qual o problema do layout é dividido em vários problemas de roteamento de canais. Estes canais são então resolvidos separadamente.



Roteamento detalhado

Caracterização de um canal (Hasimoto e Stevens - 1971)

1. A região de roteamento é retangular sem obstruções e com pinos fixos em seus lados opostos.
2. A existência de pinos flutuantes é possível nas outras duas extremidades do canal. Estes pinos indicam a necessidade de estender alguns das redes para regiões fora do canal.
3. Existem apenas dois níveis permitidos para interconexões:
 - Todos os segmentos horizontais da rede são roteados num nível (em geral).
 - Todos os segmentos verticais preferencialmente são roteados no outro nível de metal.
4. O canal é sub-dividido em trilhas cujo espaçamento não viola as regras do projeto.
5. Uma vez roteado o canal, apenas a distância entre os seus dois lados com pinos fixos pode ser modificada, e esta distância pode ser apenas incrementada.

Roteamento detalhado - exemplo

Algoritmo Left-Edge

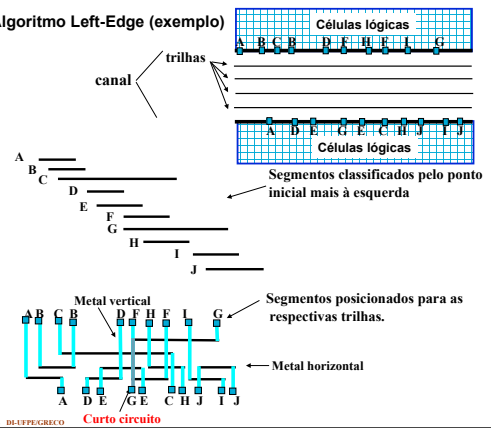
- Este algoritmo estabelece que cada rede consiste de um simples trunk (conexão horizontal-trilha horizontal) sobre um determinado nível de metal e derivações (branches) no outro nível de metal.
- Sujeita a tais condições, o algoritmo produz a solução com o mínimo número possível de trilhas. Este número é denominado densidade do canal e é encontrado determinando-se o número máximo de redes que usam trilhas horizontais ao longo do canal.

Roteamento detalhado

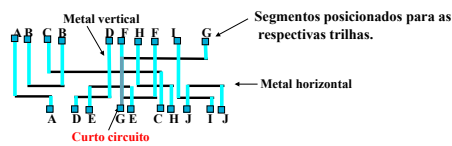
Algoritmo Left-Edge (Hasimoto e Stevens - 1971)

1. Todas as conexões horizontais necessárias são seleccionadas em ordem crescente a partir dos pontos das redes mais a esquerda do canal.
2. O algoritmo processa uma trilha por vez, começando no lado esquerdo.
3. O primeiro segmento encontrado é colocado na primeira trilha.
4. Então com o intuito de otimizar o uso de trilhas ao máximo, o algoritmo continua a varredura para o próximo segmento que poderá ser colocado na mesma trilha.
5. A varredura continua até o fim da trilha.
6. A algoritmo repete o processo para o número de trilhas que se faça necessário para o roteamento das redes no canal.

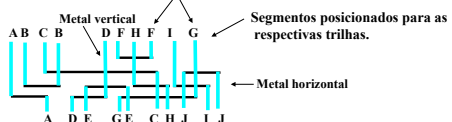
Algoritmo Left-Edge (exemplo)



Algoritmo Left-Edge (exemplo)

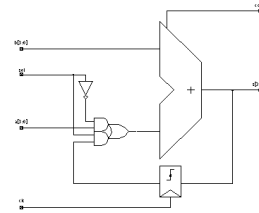


Modificação das redes G, L e F para retirtada do curto circuito



Somador com acumulador (Standard-Cells - Alliance)

Exemplo de layout:
- Roteamento em canal



**Addaccu
Comportamental
VHDL**

```

ENTITY addaccu IS
    PORT(
        a      : in BIT_VECTOR(0 TO 3);
        b      : in BIT_VECTOR(0 TO 3);
        clk     : in BIT;
        rst     : out BIT_VECTOR(0 TO 3);
        add, sub, vadd, vsub : in BIT;
    );
END addaccu;

ARCHITECTURE functional OF addaccu IS
    SIGNAL outsum, outreg :out BIT_VECTOR(0 TO 3);
    SIGNAL carry         :out BIT_VECTOR(0 TO 2);
    SIGNAL reg           :out BIT_VECTOR(0 TO 3) REGISTER;

    BEGIN

        ASSERT (add = '1' and sub = '0')
            REPORT "Wrong power supplies"
            SEVERITY WARNING;

        ASSERT (add = '0' and sub = '1')
            REPORT "Wrong power external supplies"
            SEVERITY WARNING;

        WITH sel SELECT
            outsum <= a WHEN '0', outreg WHEN '1';
            carry(0) <= outsum(0) and b(0);
            carry(1) <= outsum(1) and b(1) xor carry(0);
            carry(2) <= outsum(2) and b(2) or (outsum(1) and carry(0) or (a(1) and carry(0)));
            carry(3) <= outsum(3) and b(3) xor carry(2);
            a <= sum;

        L0 : BLOCK (ck = '0' and not d<STABLE)
            BEGIN
                reg <= CEARDED not sum;
            END BLOCK;
            outreg <= not reg;
        END
    
```

DI-IPPEGRECO

**Layout do circuito
addaccu(standard-cell)**

DI-IPPEGRECO

Silicon Wafer

DI-IPPEGRECO