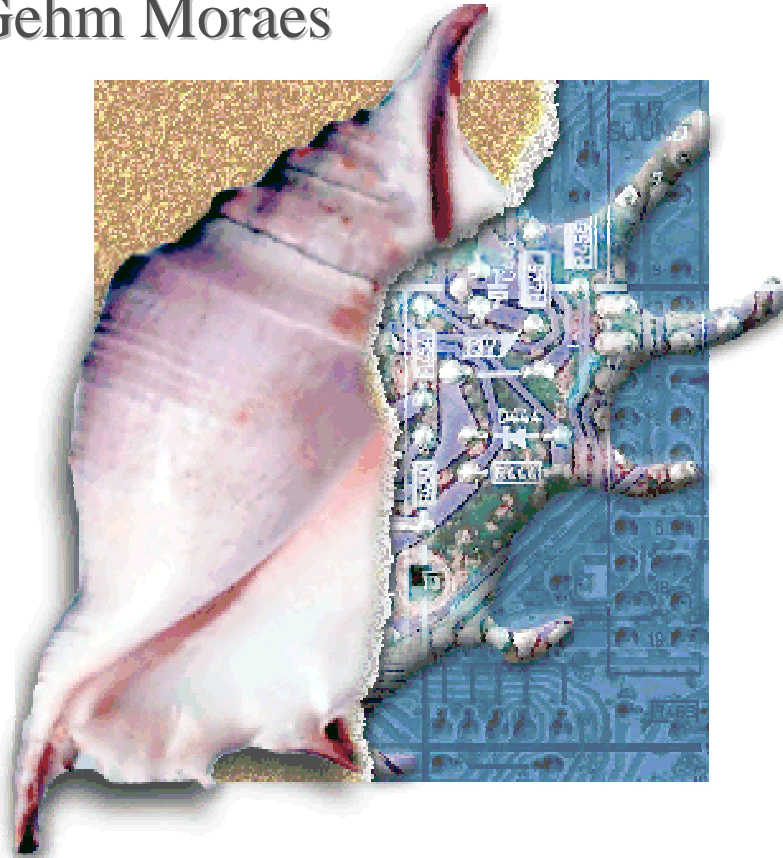


12 Anatomia de uma Ferramenta de Síntese Automática de Layout

Fernando Gehm Moraes



EMICRO2000



SOCIEDADE
BRASILEIRA DE
COMPUTAÇÃO



Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
3. TROPIC3 - Estilo de layout
4. Etapas da síntese da layout
5. Predição de Parasitas
6. Integração da Síntese Física à Síntese Lógica
7. Direções Futuras



1. Síntese Automática de Layout

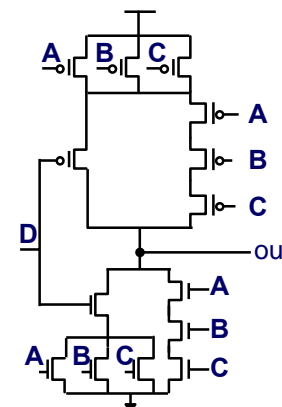


DEFINIÇÃO

- Síntese das máscaras de um módulo de um CI a partir de uma descrição lógica, **SEM** utilização de biblioteca de células pré-definidas

VANTAGENS

- número “ilimitado” de funções complexas
- transistores podem ser dimensionados livremente
- fácil migração tecnológica
- fluxo de projeto mais simples, pois não há gerência de complexas bibliotecas de células



Síntese Automática de Layout



PROBLEMAS

- ausência de pré-caracterização **impede** uma estimativa precisa do atraso no nível lógico (de entrada)
- topologia **fixa**, limitada a CMOS dual e portas de transmissão
- elevado **tempo** de CPU devido a compactação de layout (*simbólico em layout*)
- densidade de transistores por milímetro quadrado (**tr/mm²**) é menor que em standard-cells devido ao estilo regular de layout
- complexidade **limitada** a aproximadamente 10000 transistores

TECNOLOGIA SUBMICRÔNICA

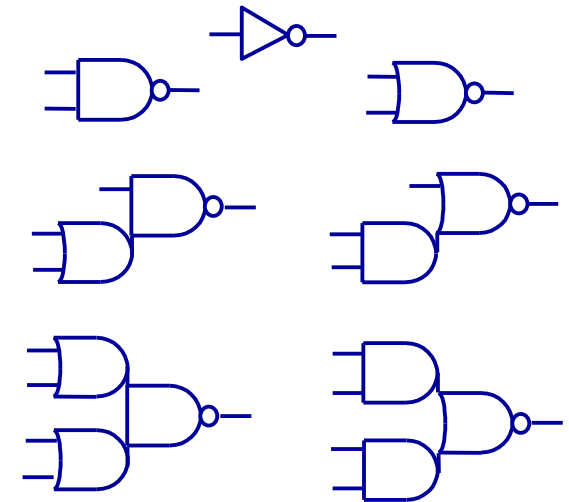
- atraso dominado pelas interconexões \Rightarrow pré-caracterização de pouca utilidade

Biblioteca Virtual de Células



- quantidade de funções disponíveis:
 - é **maior** que em bibliotecas pré-definidas
 - layout é gerado automaticamente
 - é função do número de transistores em série (desempenho elétrico)

		Número de Transistores PMOS em Série				
		1	2	3	4	5
Número de Transistores NMOS em série	1	1	2	3	4	5
	2	2	7	18	42	90
	3	3	18	87	396	1677
	4	4	42	396	3503	28435
	5	5	90	1677	28435	425803

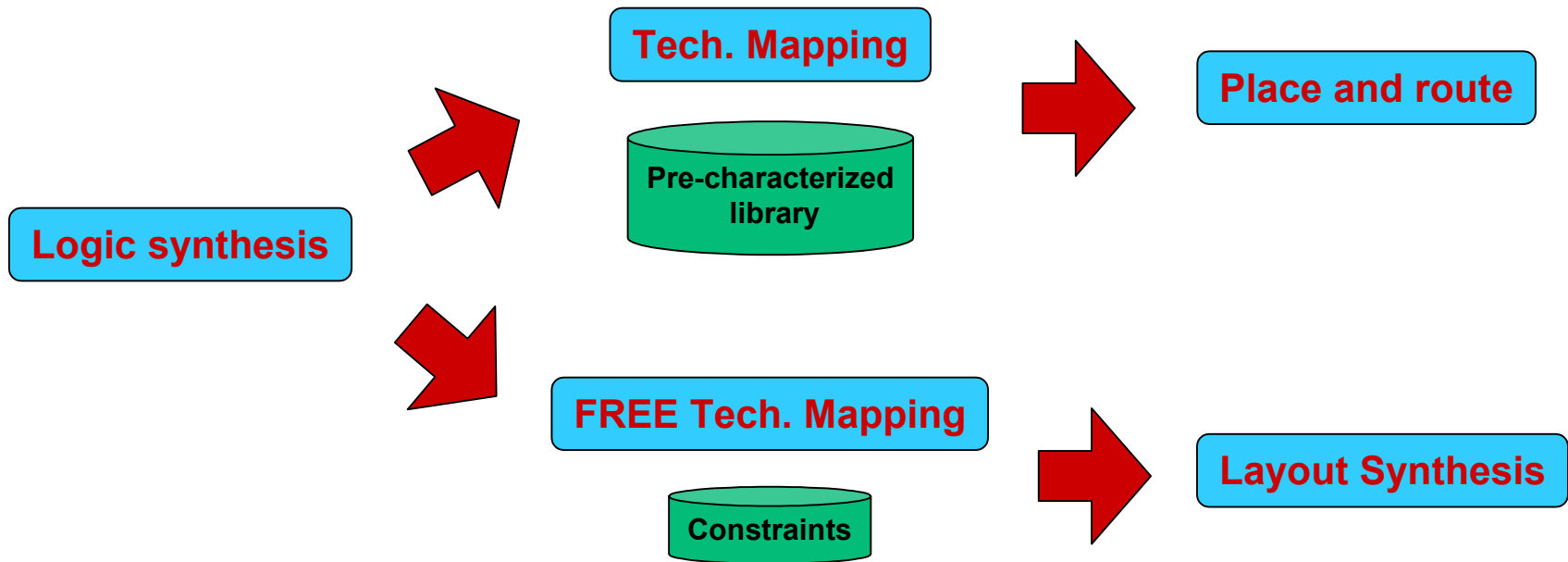


Biblioteca (2,2)

Síntese de Bibliotecas e Síntese de Macro-Células



- Fluxo de projeto simplificado



Síntese de Bibliotecas



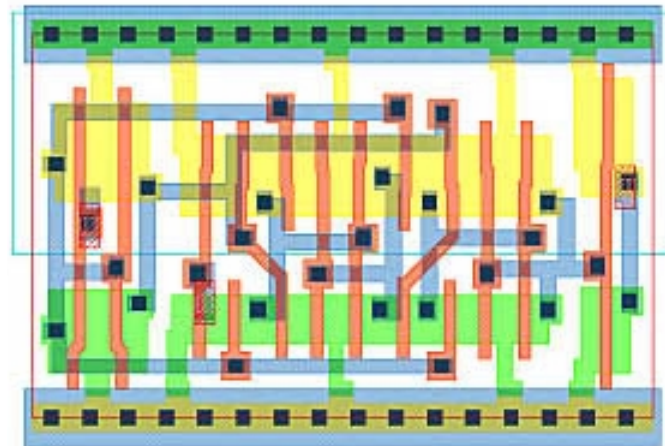
SÍNTESE DE BIBLIOTECAS

- gera-se um layout **único** para cada célula do netlist (*templates* simbólicos ou layout regular)
- pode-se utilizar layout não regular (layouts mais densos)
- resolve o problema de número limitado de funções e dependência às regras de desenho
- não permite dimensionamento individual das células

Exemplos:

CADABRA - <http://www.cadabratech.com>

MAGMA - <http://www.magma-da.com>



células de mesma
altura com posição
fixa para os pinos
de E/S

Síntese de Macro-Células



- gera-se **todas** as células do circuito
- **NÃO** há geração de biblioteca
- estilo de layout regular
- cada transistor pode ser dimensionado individualmente
- exige ferramentas de posicionamento e roteamento dedicadas
- se compactação é utilizada o tempo de CPU pode tornar-se proibitivo
- exemplo comercial: **CADENCE (LAS)**
- exemplo acadêmico: **TROPIC2 / TROPIC3**

Síntese de Macro-Células



BIBLIOTECAS VIRTUAIS

- para 4 transistores em série até 3503 funções diferentes
- menor número de transistores
 - $\pm 35\%$ - redução de área e atraso
- menor atividade de chaveamento
 - redução de potência dissipada

FÁCIL MIGRAÇÃO TECNOLÓGICA

- característica vital para suportar os rápidos avanços tecnológicos

COMPROMISSO POTÊNCIA-ATRASSO

- transistores podem ser individualmente dimensionados

REDUÇÃO DE CUSTOS

- bibliotecas pré-caracterizadas não são mais necessárias

Motivação para desenvolver uma ferramenta de síntese



- redução do tempo de CPU para a síntese de layout (suprimir compactação de layout)
- permitir a síntese de macro-células com pelo menos 25000 transistores
- reduzir a área de silício ocupada através da utilização de 3 níveis de metal e contatos empilhados
- prover uma rápida predição de capacitâncias e resistências parasitas
- integrar a síntese física à síntese alto nível (VHDL)

Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais

2. Estilos de layout

3. TROPIC3 - Estilo de layout

4. Etapas da síntese da layout

5. Predição de Parasitas

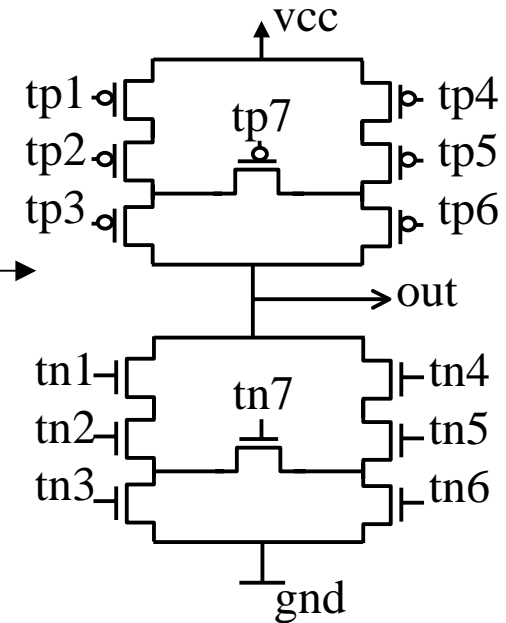
6. Integração da Síntese Física à Síntese Lógica

7. Direções Futuras

Estilo de layout



- define a orientação de cada camada no layout
- posicionamento dos transistores
- níveis de metal utilizados
- regras de utilização dos contatos
- regras de interconexão entre os níveis
- topologias permitidas ou não
- estratégia de roteamento
 - baseado em canais de roteamento
 - OTC - *over the cell*

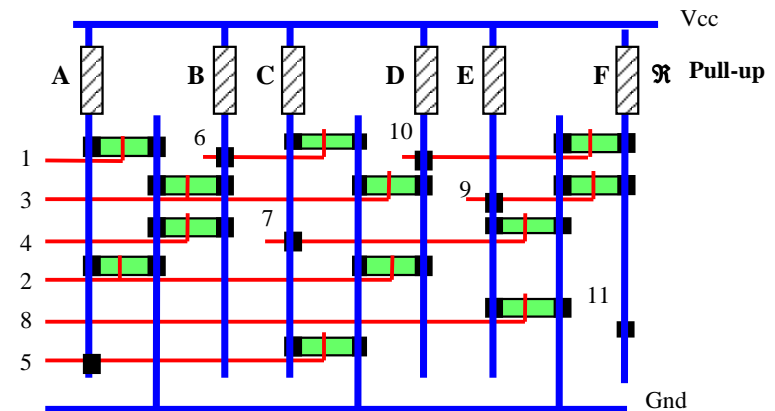
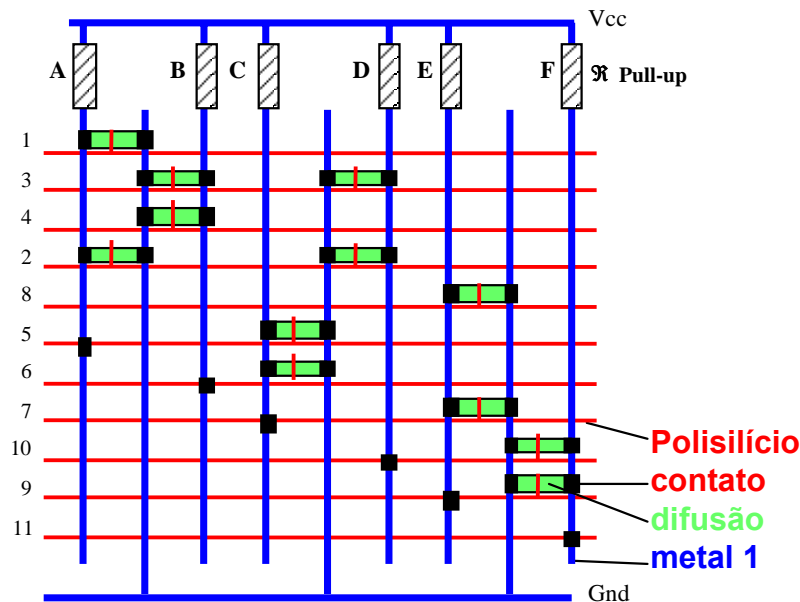
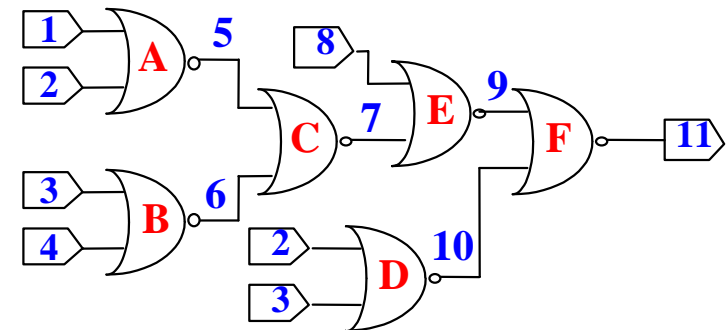


**Células planares normalmente não são sintetizadas
(impossível de reduzir em conexões série/paralelo)**

Weinberger array



- Primeira abordagem encontrada na literatura, 1967, para NMOS
- Layout matricial
 - apenas portas *nor*
 - linhas de poli e colunas de metal1



Weinberger array otimizado

Gate-Matrix



- **Estilo regular de layout (1980)**

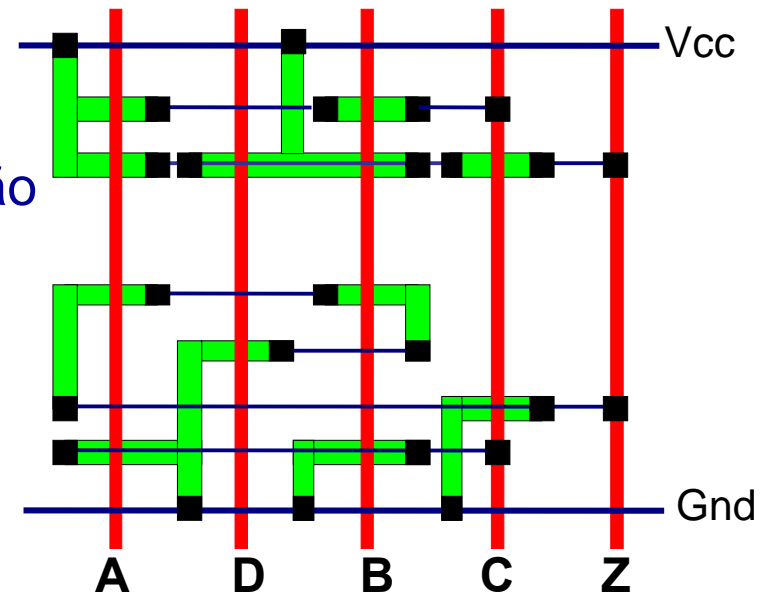
- Colunas: linhas de polisilício **compartilham** o mesmo sinal
- Linhas: difusão e metal1

- **Principal função custo**

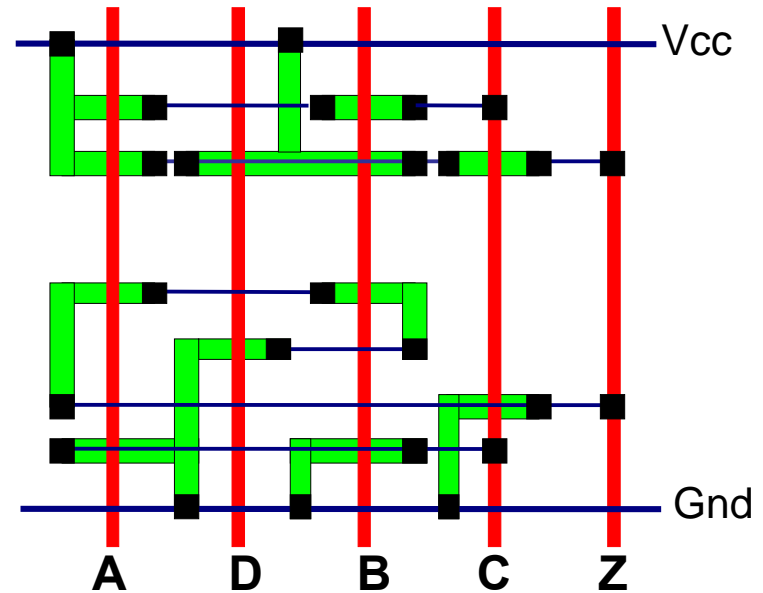
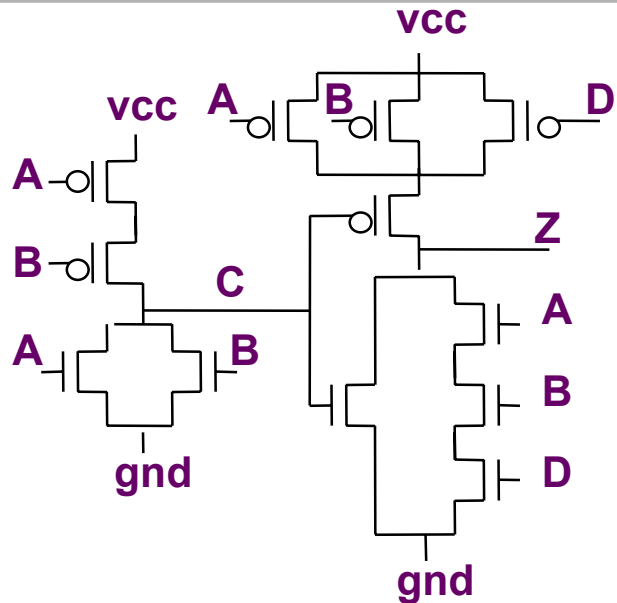
- Encontrar o posicionamento ótimo das colunas de polisilício afim de reduzir o roteamento

- **Limitações**

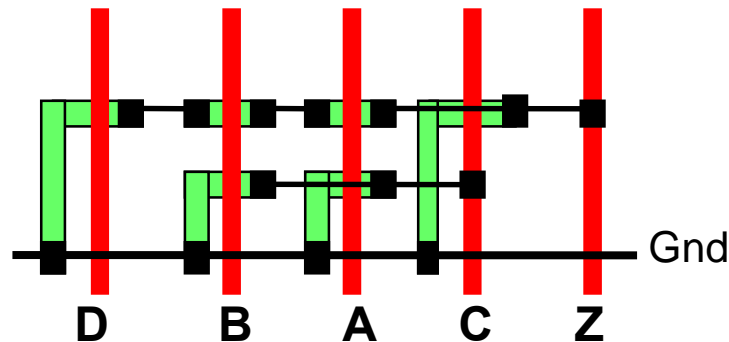
- Número elevado de quebras de difusão
 - Capacitâncias Parasitas (side-wall)
- Baixo desempenho para submicron
- Limitado a funções simples



Gate-Matrix



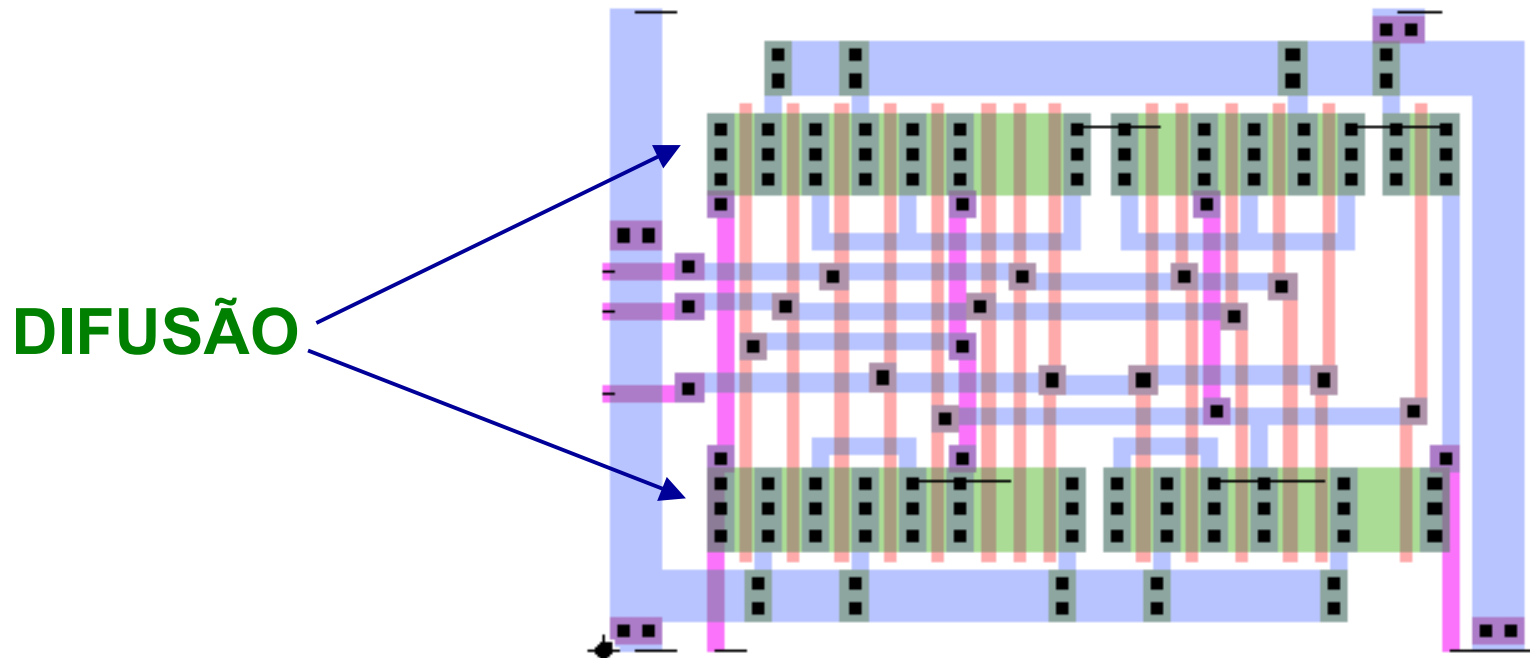
Gate Matrix otimizado



Linear-matrix



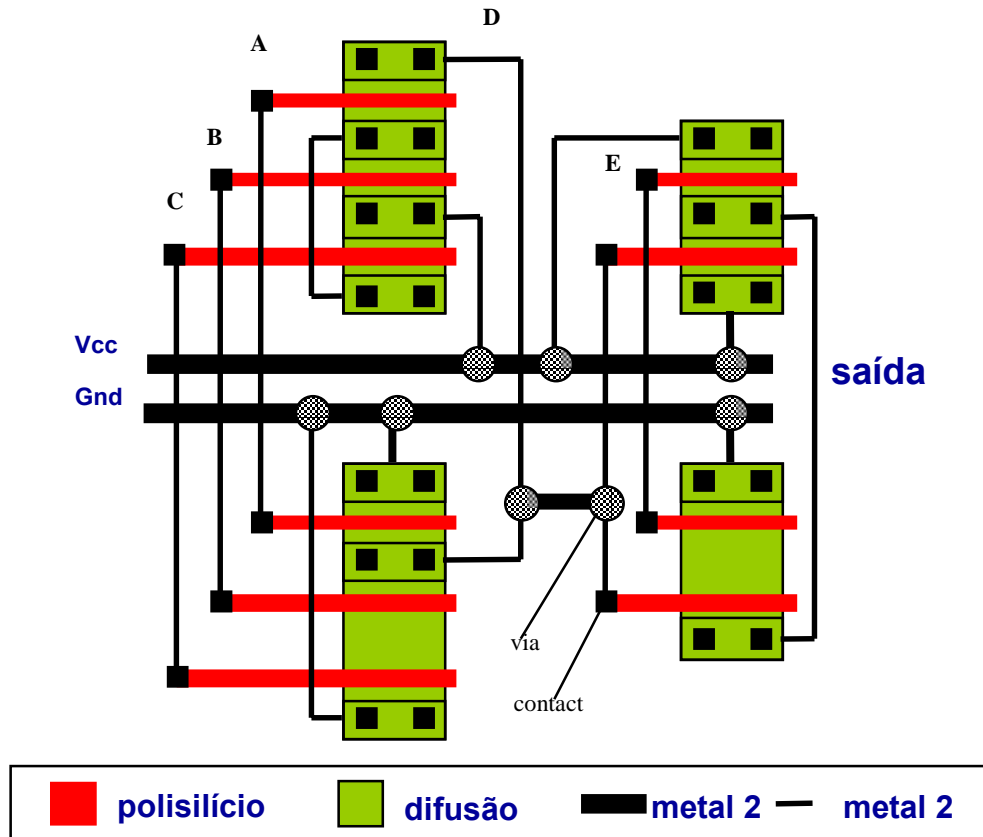
- Duas linhas horizontais de transistores (1983)
- Apenas dois transistores por coluna de polisilício
- Reduzido número de quebras de difusão
- Exemplos: LAS e TROPIC



Linear-matrix vertical



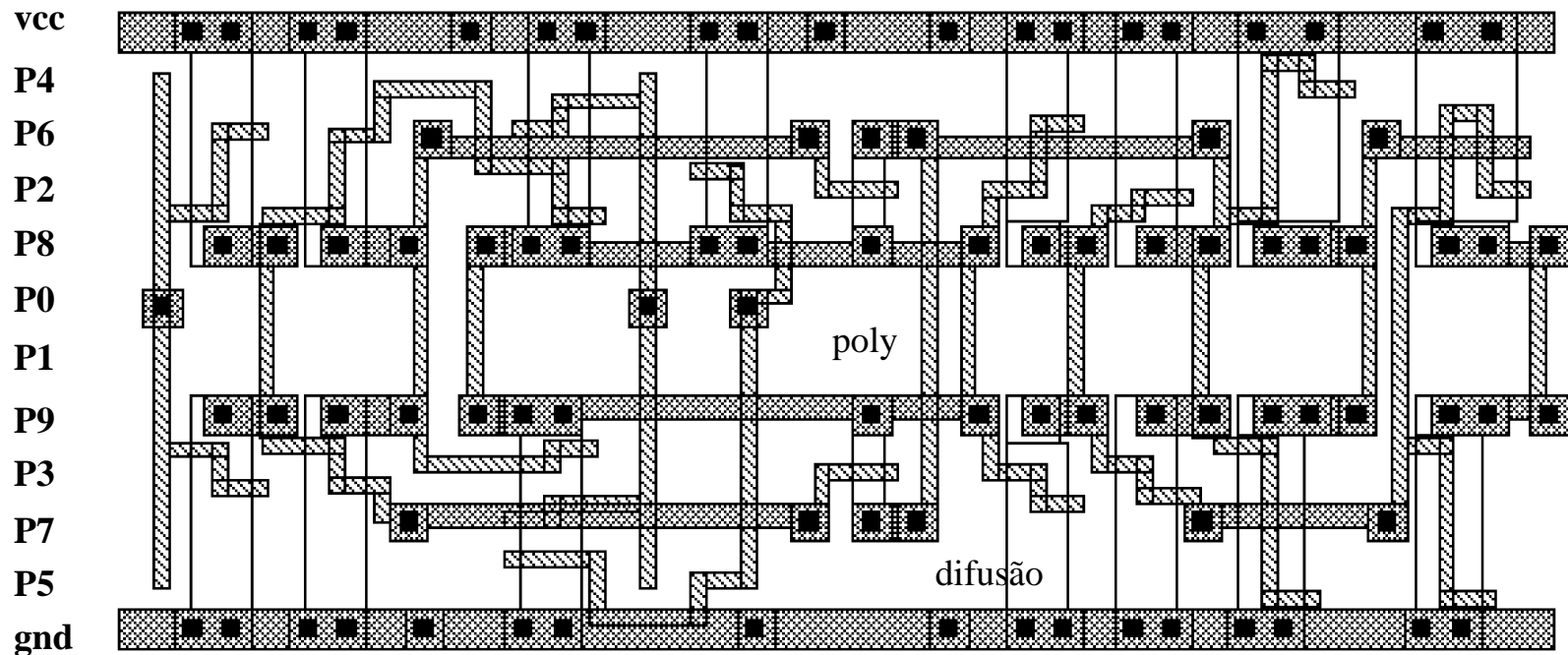
- objetivo: reduzir o uso de polisilício
- muitos contatos e desperdício de área



TRANCA



- Roteamento sobre a área ativa, com ausência de canais de roteamento
- 10 trilhas internas, INSUFICIENTE para grandes circuitos



Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
- 3. TROPIC3 - Estilo de layout**
4. Etapas da síntese da layout
5. Predição de Parasitas
6. Integração da Síntese Física à Síntese Lógica
7. Direções Futuras

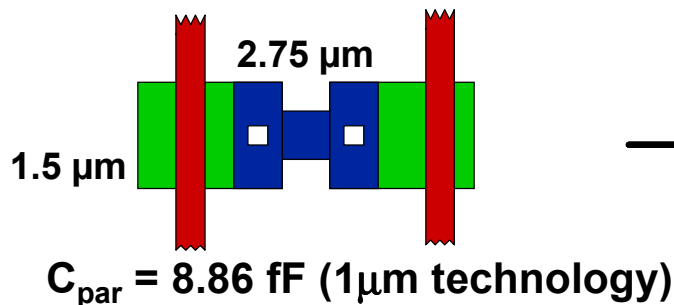
Estilo de layout para TROPIC3



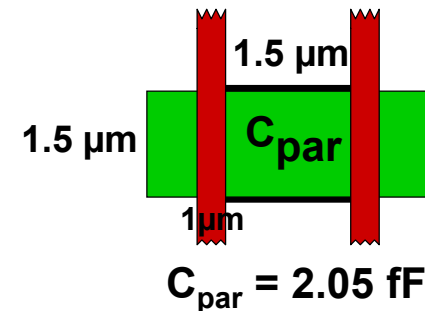
- Tropic3: ferramenta de síntese automática de layout para 3 níveis de metal
- Sem compactação de layout
- Estilo linear-matrix

Capacitância parasita para 2 transistores conectados em série:

conexão em metal1:
(diffusion gap, gate-matrix)



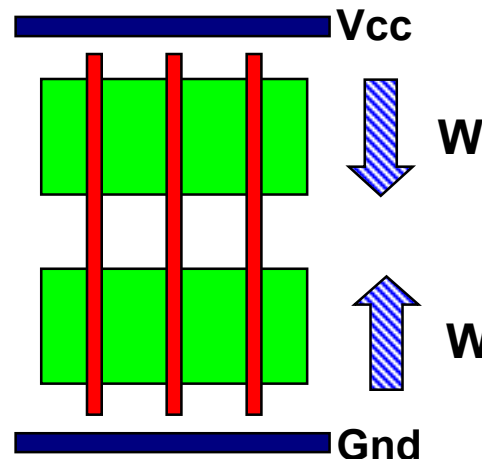
conexão por justaposição
(linear matrix)



Layout a nível de célula - transistores (1/3)



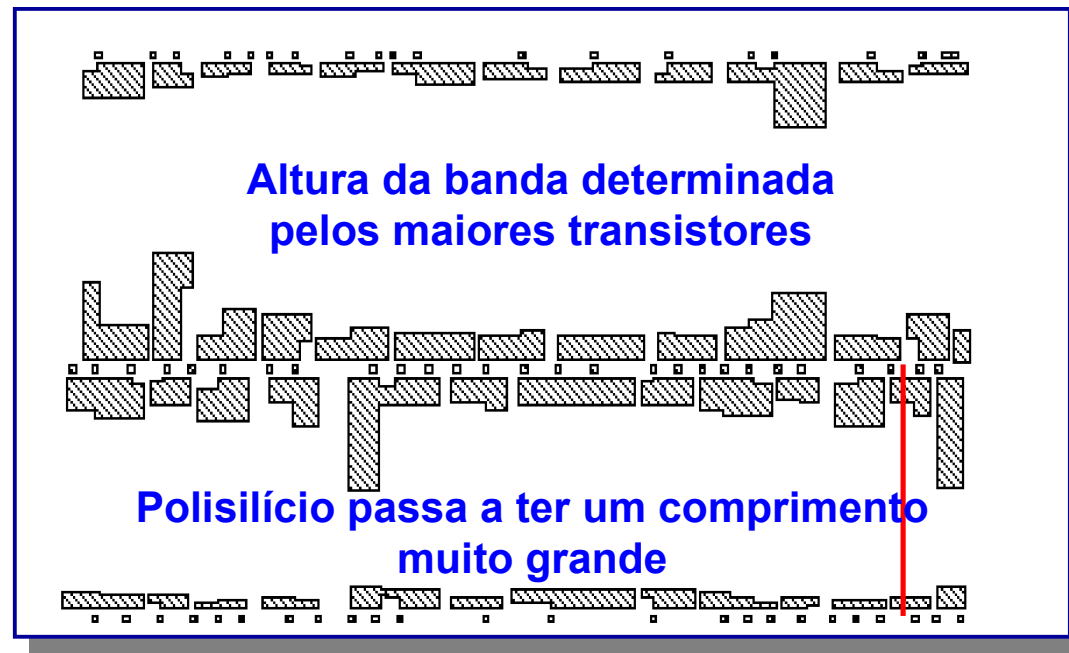
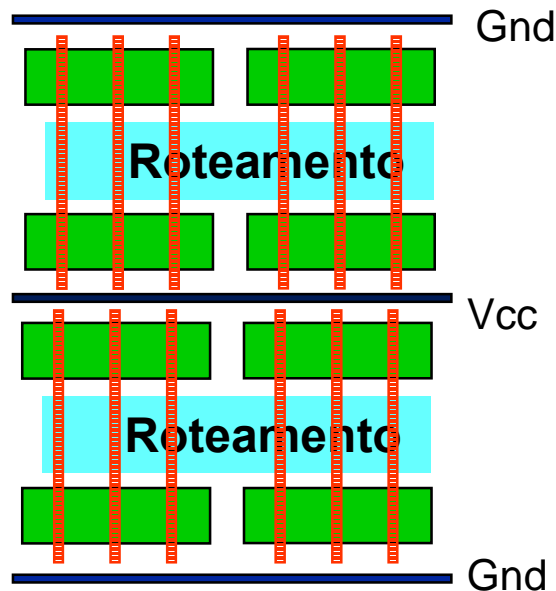
- Linhas de alimentação **externa** aos transistores
 - procedimento semelhante a std-cells
 - separação entre planos N/P não é mínima
 - se roteamento entre transistores:
 - aumento da altura do polisilício
 - dimensionamento dos transistores separa os planos N/P
 - altura da banda proporcional ao maior transistor



Layout a nível de célula - transistores (2/3)



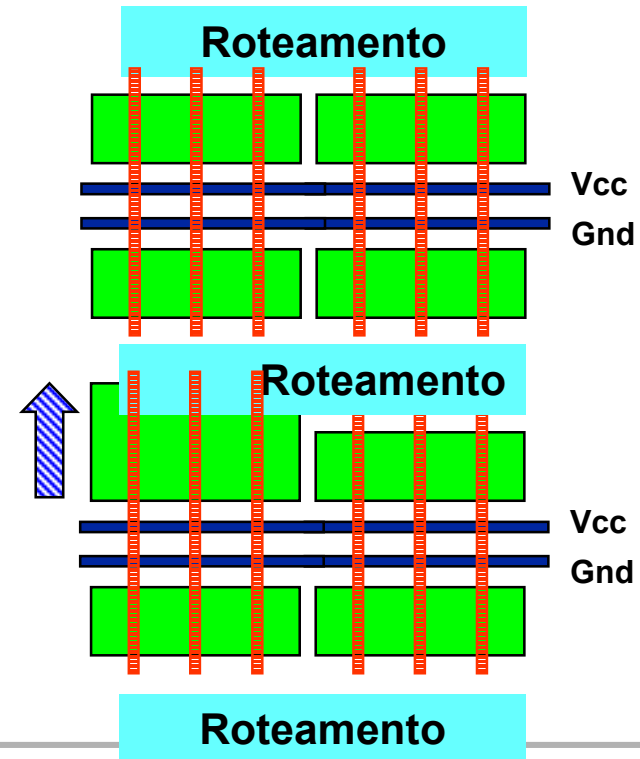
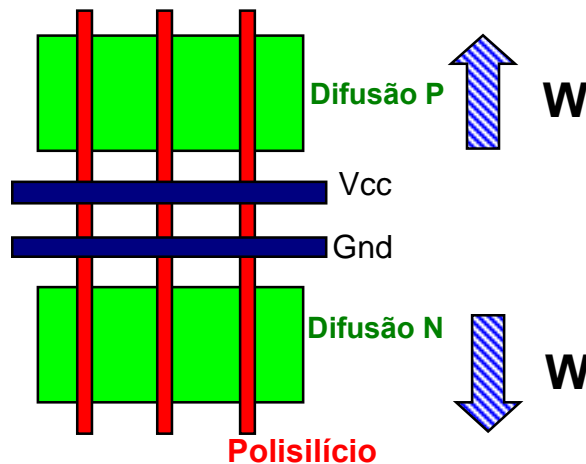
- Linhas de alimentação **externa** aos transistores
 - mais de uma linha de células - banda: justaposição e roteamento entre transistores
 - estilo de TROPIC2



Layout a nível de célula - transistores (3/3)



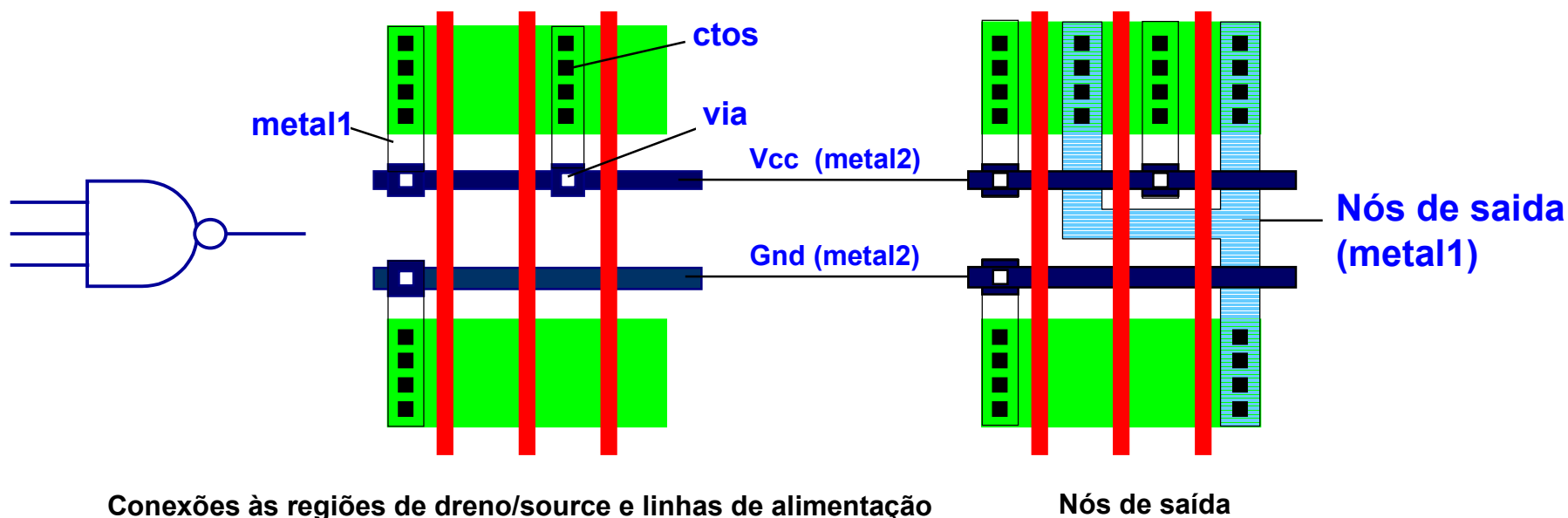
- Linhas de alimentação **entre** os transistores
 - estilo de TROPIC3
 - canais externos/sobre os transistores
 - polisilício com altura mínima



Conexão à alimentação e nós de saída



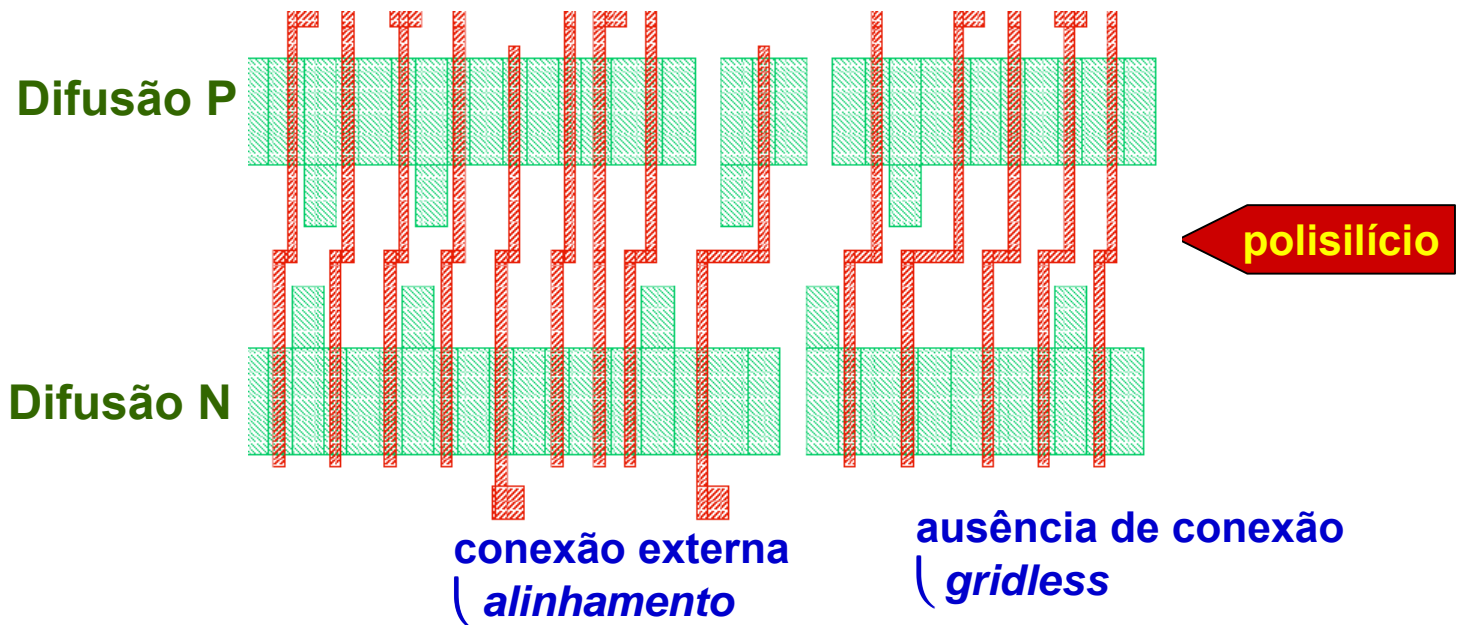
- alimentação em metal2
- drenos/source conectados em metal1/múltiplos contatos
- body-ties sobre a via de alimentação (contato empilhado)
- saídas: em metal1, sem interrupções (contatos ou pontes)



Linhas de Polissilício (1/3)



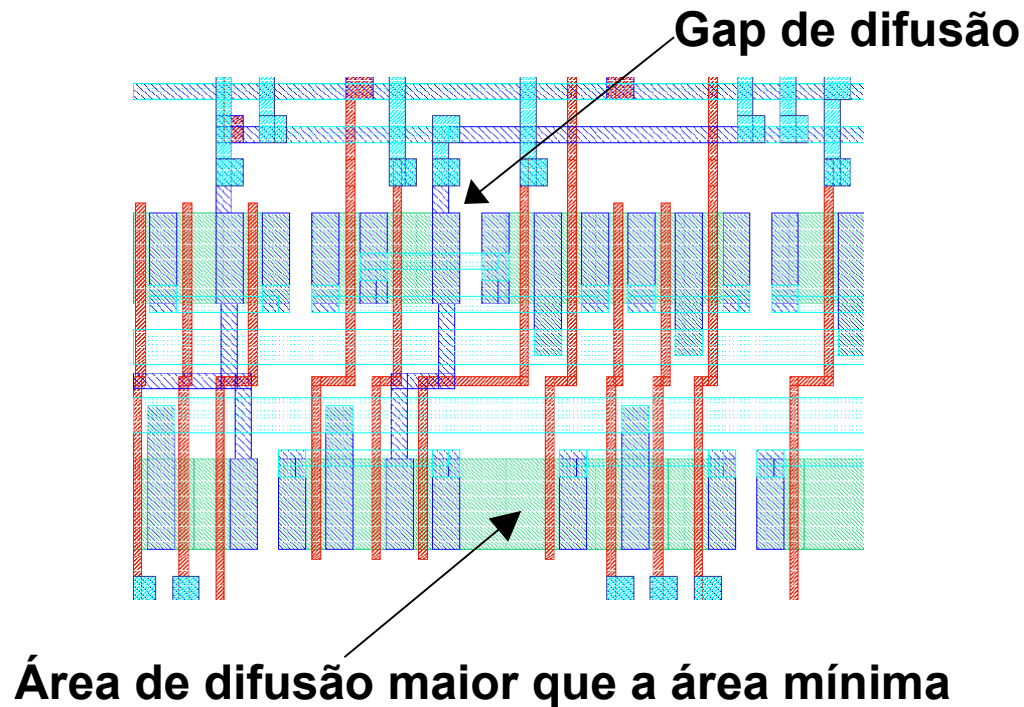
- inserção automática de *jogs* (quebras)
- distância entre linhas de poli em função do roteamento externo: alinhamento ou não a uma grade



Linhas de Polissilício (2/3)



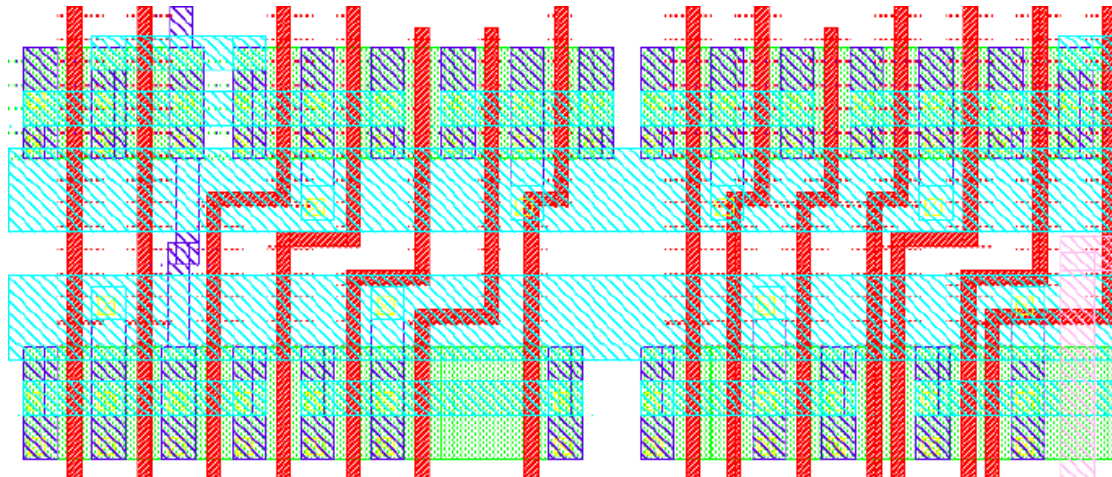
- pode ocorrer aumento excessivo nas áreas de dreno/source



Linhas de Polissilício (3/3)



- **otimização: inserção de múltiplos jogs**
 - objetivo: reduzir área de difusão nos drenos/source
- **problemas:**
 - definição do número máximo de jogs (para evitar efeito *escada*)
 - onde inserir body-ties?
 - mesmo com múltiplos jogs continuam havendo áreas de difusão não mínima
 - baixo impacto na área final do circuito

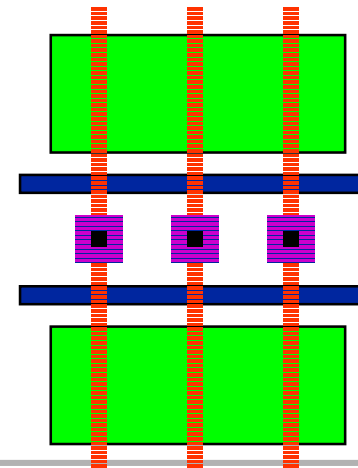
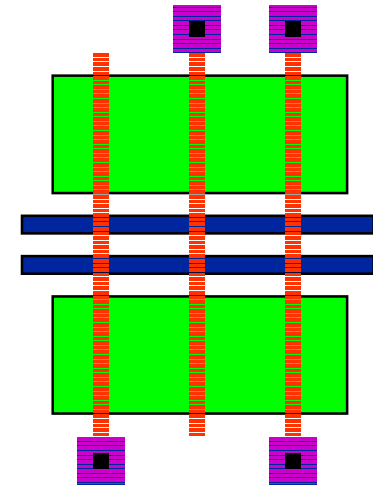


Nós de entrada saída (1/2)

U



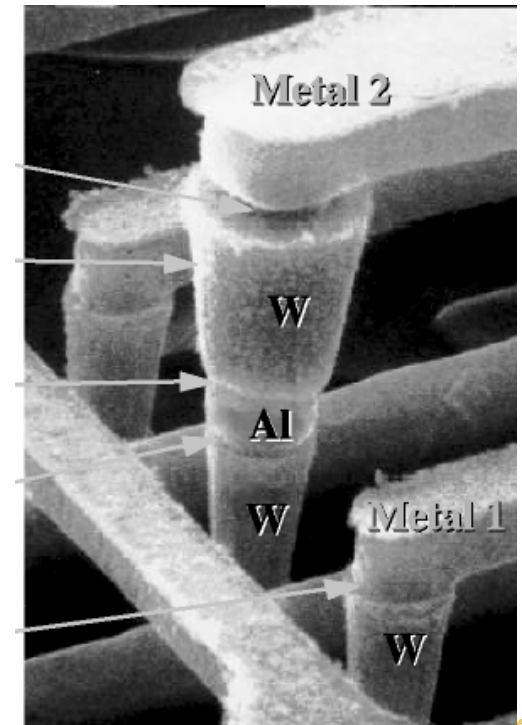
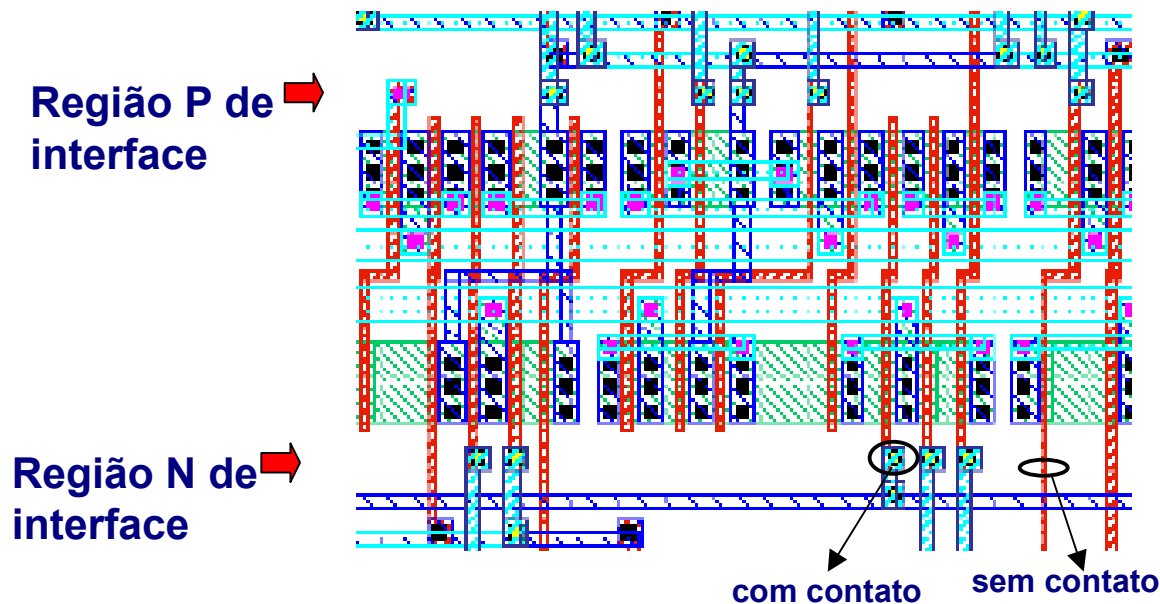
- Entradas e saídas na periferia das células
 - necessário ter uma trilha para contatos - *região de interface*
 - estes contatos representam obstáculos para o roteamento
 - TROPIC3
- Entradas e saídas no centro
 - facilita o dimensionamento dos transistores
 - roteamento é muito difícil
 - pouca transparência



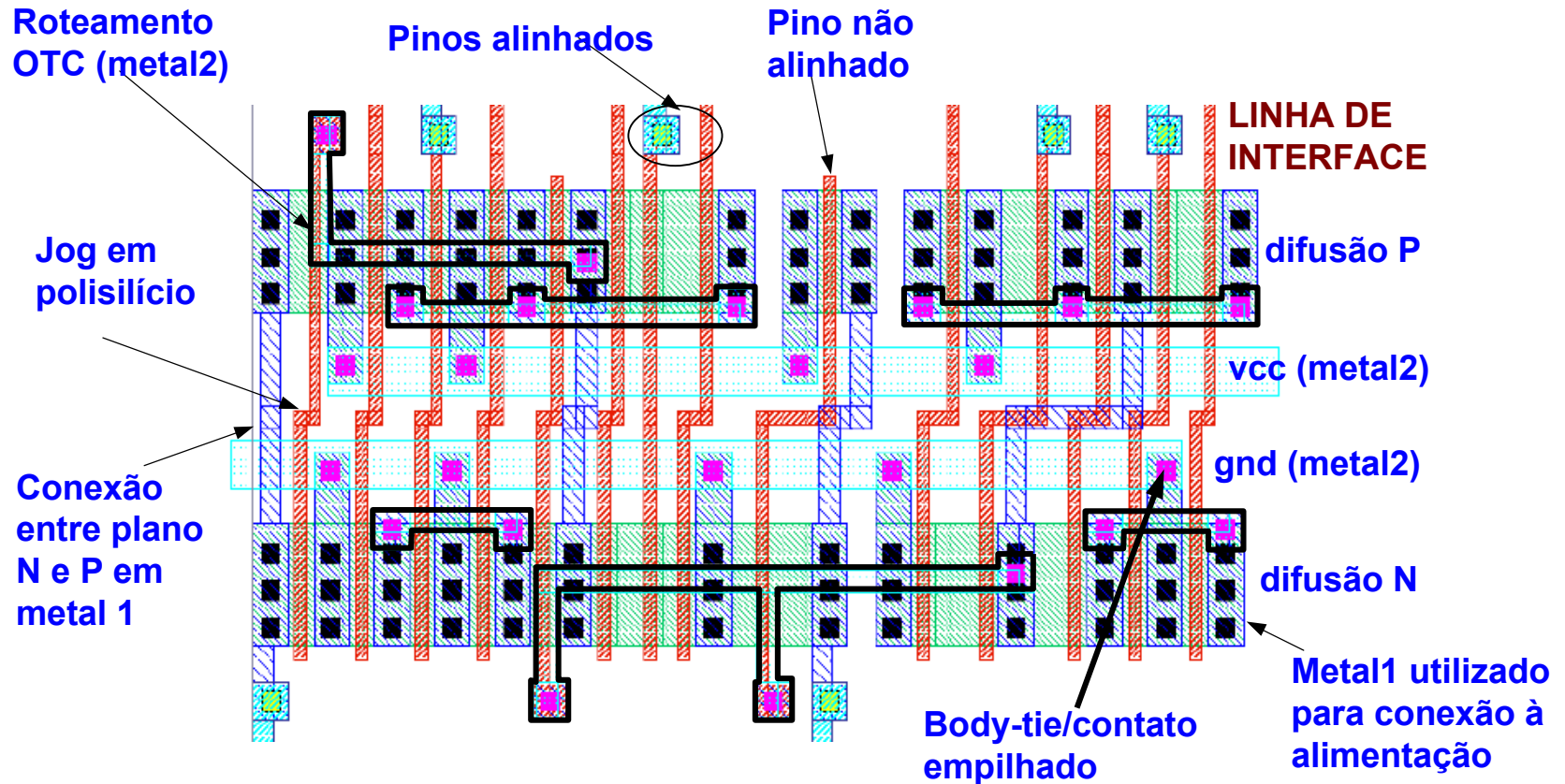
Nós de entrada saída (2/2)



- **Região de interface:**
 - utilizada para conectar níveis não adjacentes, exemplo, poli-metal3, com inserção de contatos empilhados
 - representa uma perda de área



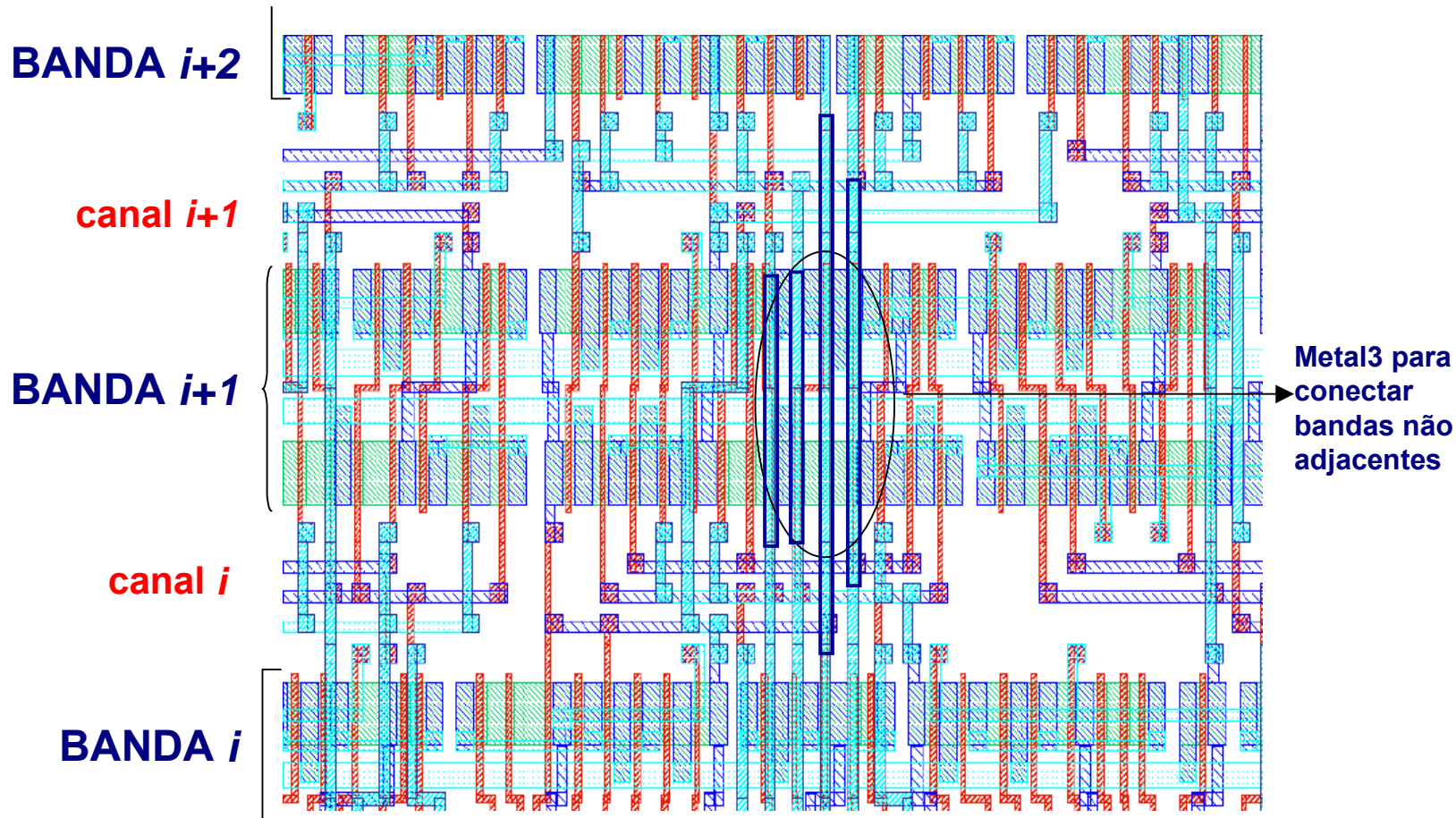
Layout completo de uma banda



Conexão entre bandas não adjacentes



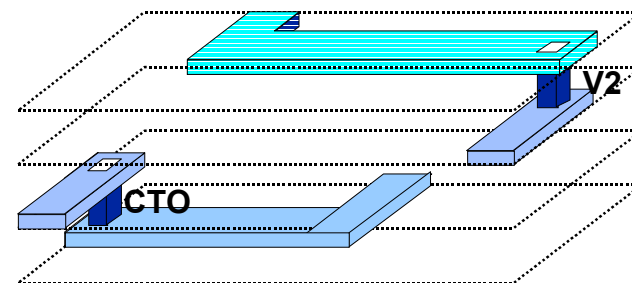
- Bandas transparentes ao metal3



Utilização dos níveis



- utilização de 4 níveis de roteamento
- escolha dos níveis na região de roteamento é arbitrária



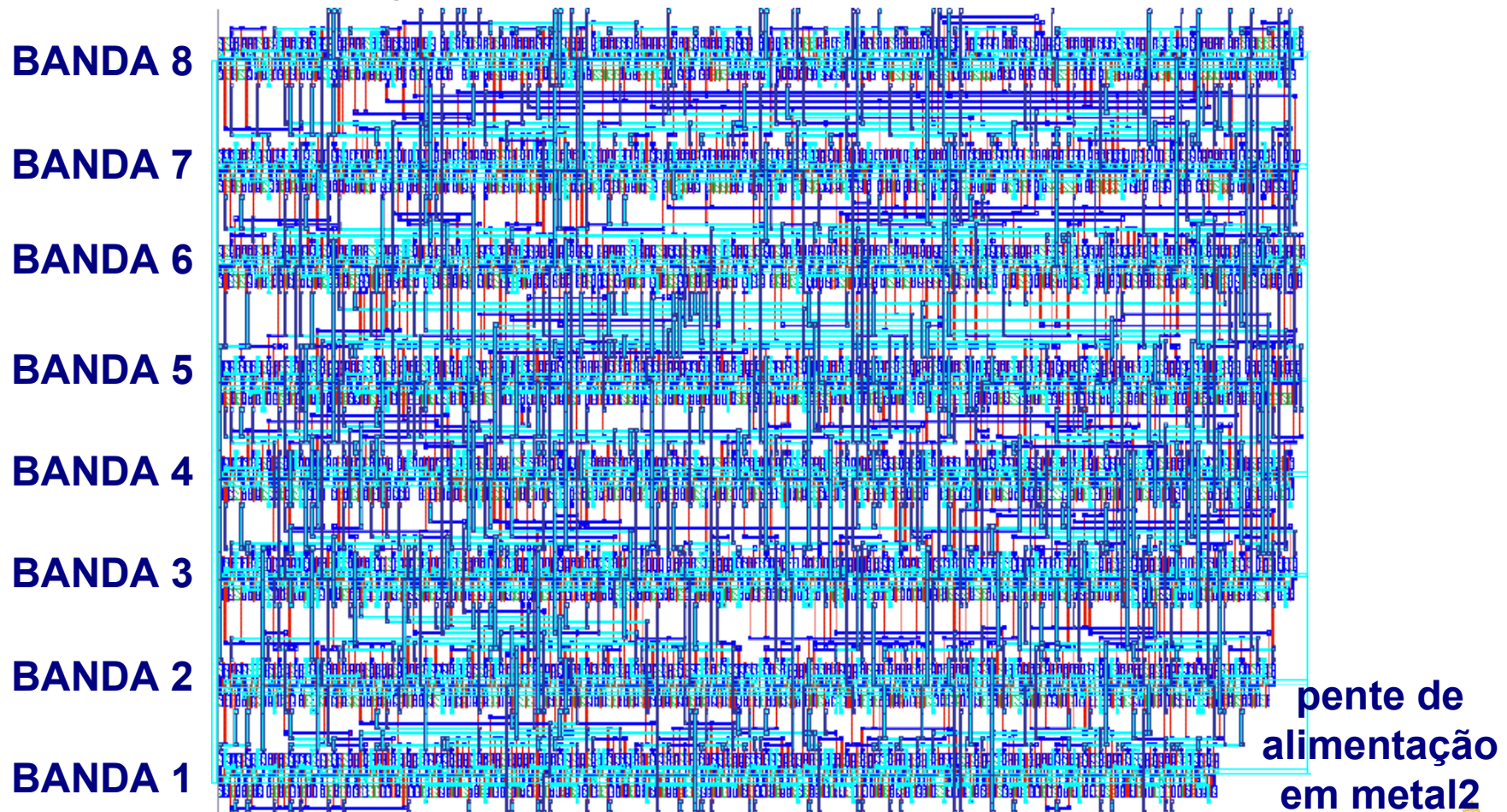
Região	Horizontal	Vertical
Banda (célula)	Difusão Metal2	Polisilício Metal1 Metal3
Roteamento	Metal1 Metal2	Polisilício Metal3

Um layout completo

Circuito: C1355
2244 transistores



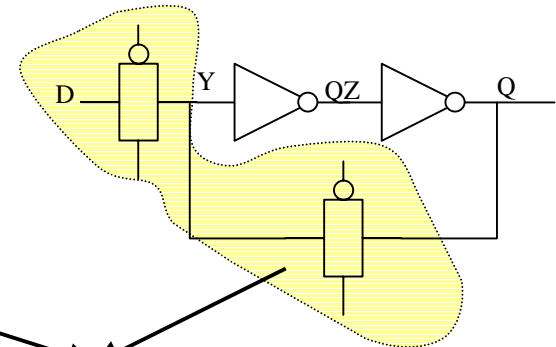
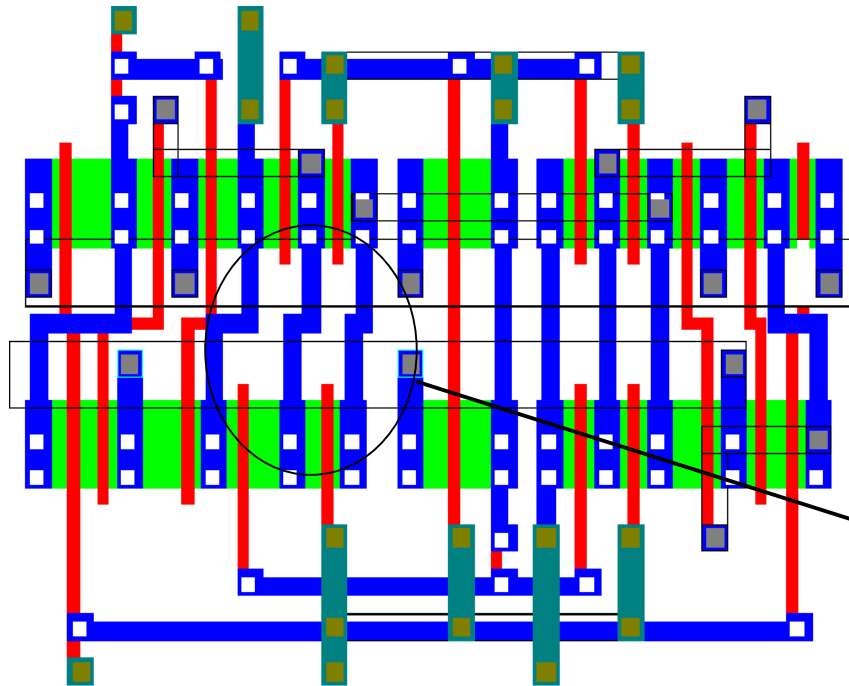
- observar: distribuição uniforme do roteamento



Célula não dual



- Porta de transmissão - TG
- Agrupamento de TGs em oposição de fase para redução de área e otimização elétrica

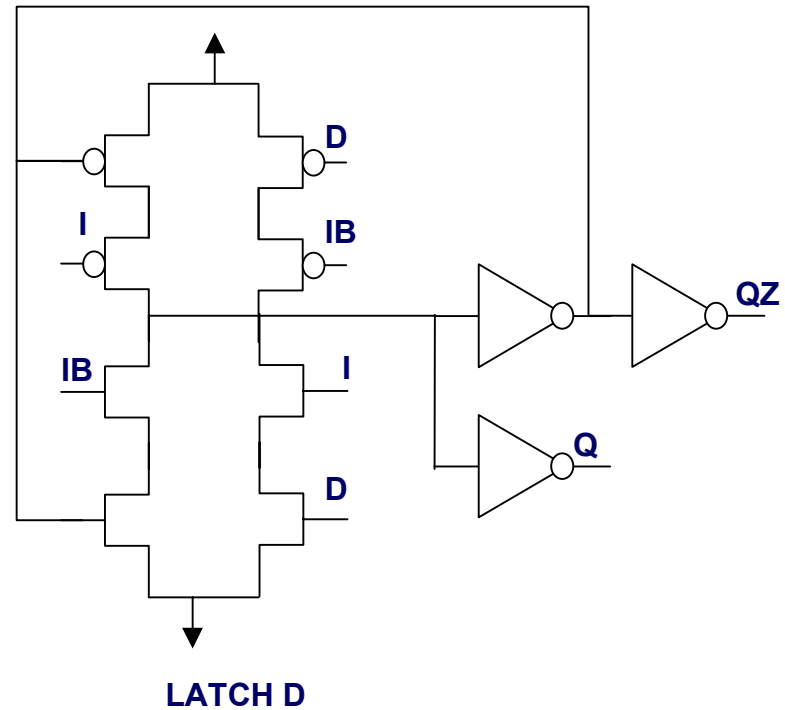
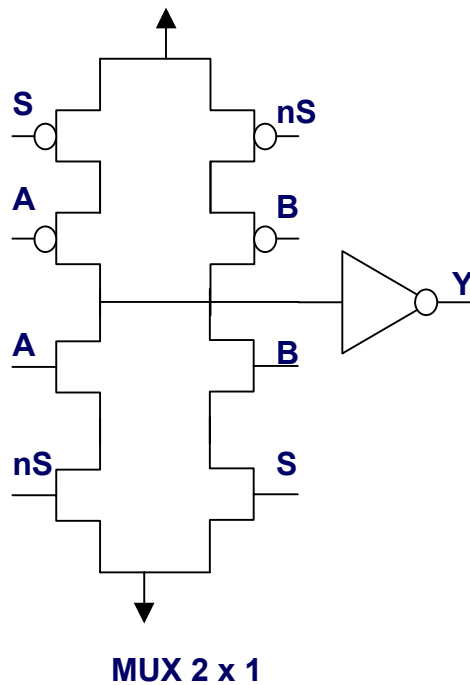


Par de portas de transmissão com transistores não duais

Célula pseudo-dual



- Casos de células não duais, mas que possuem mesmo número de entradas
- Sintetizadas como portas complexas



Codificação da tecnologia

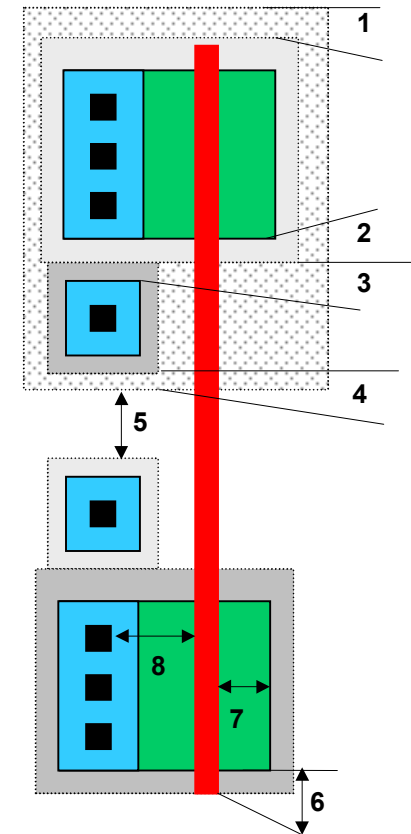


- simples, visando fácil migração tecnológica
- regras geométricas
 - 28 regras: separações, larguras e transistores



- regras CIF
- regras elétricas
- grade de roteamento:

$$\text{grid} = 2 * \max(MCTO, MVIA1, MVIA2) + \max(LCTO, LVIA1, LVIA2) + \max(DM1, DM2, DM3)$$



Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
3. TROPIC3 - Estilo de layout
- 4. Etapas da síntese da layout**
5. Predição de Parasitas
6. Integração da Síntese Física à Síntese Lógica
7. Direções Futuras

Etapas da síntese física



- 1 Leitura do arquivo Spice
 - 2 Planificação do netlist Spice
 - 3 Extração de células folha
 - 4 Posicionamento de células folha
 - 5 Geração de células folha
 - 6 Assinalamento de pinos
-
- 7 Geração de banda
 - 8 Fixação das coordenadas dos pontos de passagem sobre as bandas
 - 9 Roteamento de canal
 - 10 Geração do arquivo de layout em formato CIF

Independente das
regras de
desenho



Etapas dependentes
das regras de
desenho



Extração de células folha (1/4)



- **Terceira etapa da síntese**

- entrada: duas listas de transistores, N/P
- saída: conjunto de células folha (nand, nor, AOI, inversor)
- células como flip-flops, ands são quebradas em células folha

- **Princípio básico**

- redução do netlist de entrada em cadeias série/paralelo

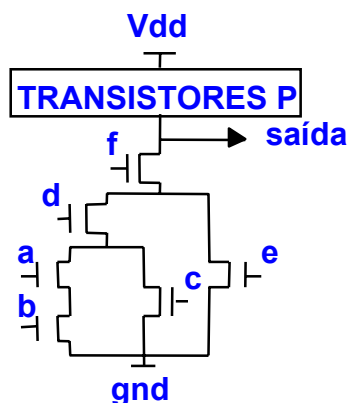
- **Algoritmo**

```
função reduz_netlist( lista de transistores )  
{  
    enquanto ( busca_transistores_em_paralelo(lista de transistores) ∨  
               busca_transistores_em_série (lista de transistores)  
            );  
}
```

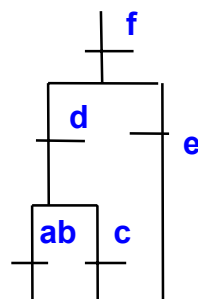
Extração de células folha (2/4)



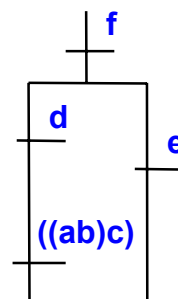
• Redução série/paralelo, um exemplo



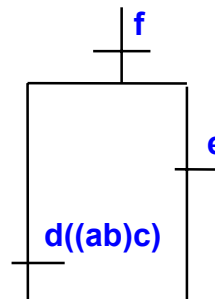
célula



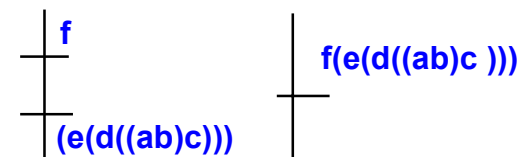
primeira iteração



segunda iteração



terceira iteração



	1ª iteração	2ª iteração	3ª iteração
Transistores em paralelo	-	(ab)c	(e(d((ab)c)))
Transistores em série	ab	d((ab)c)	f(e(d((ab)c)))

Extração de células folha (3/4)

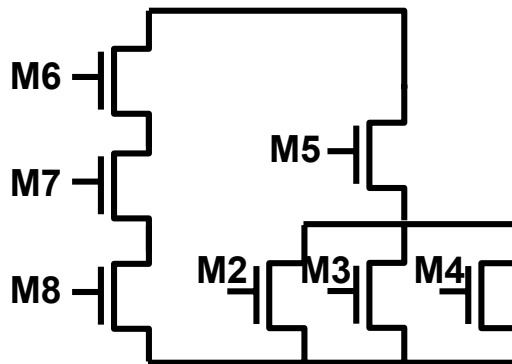


- cada célula folha é determinada pela **união** de um grafo N contendo um nó comum a um grafo P - etapa de **pareamento**
- porta de transmissão: transistor N em paralelo a um transistor P, não pertencentes a nenhum grafo
- ao final do pareamento temos:
 - cadeias N e P: permitem recuperar os nós dreno/gate/source e as dimensões W/L de cada transistor de cada célula
 - rede de saída das células
 - número de entradas, o que define a **largura** das células
 - se é porta dual (AOI) ou porta de transmissão

Extração de células folha (4/4)



- a extração de células folha pode ser uma ferramenta independente, para EXTRAção LÓGica - **EXTRALO**
- pode-se obter a partir de um netlist contendo transistores (**SPICE**) o **VHDL** estrutural do mesmo - *spi2vhdl*



[([M2M3M4] M5) (M6 M7 M8)]

paralelo

série

paralelo

série

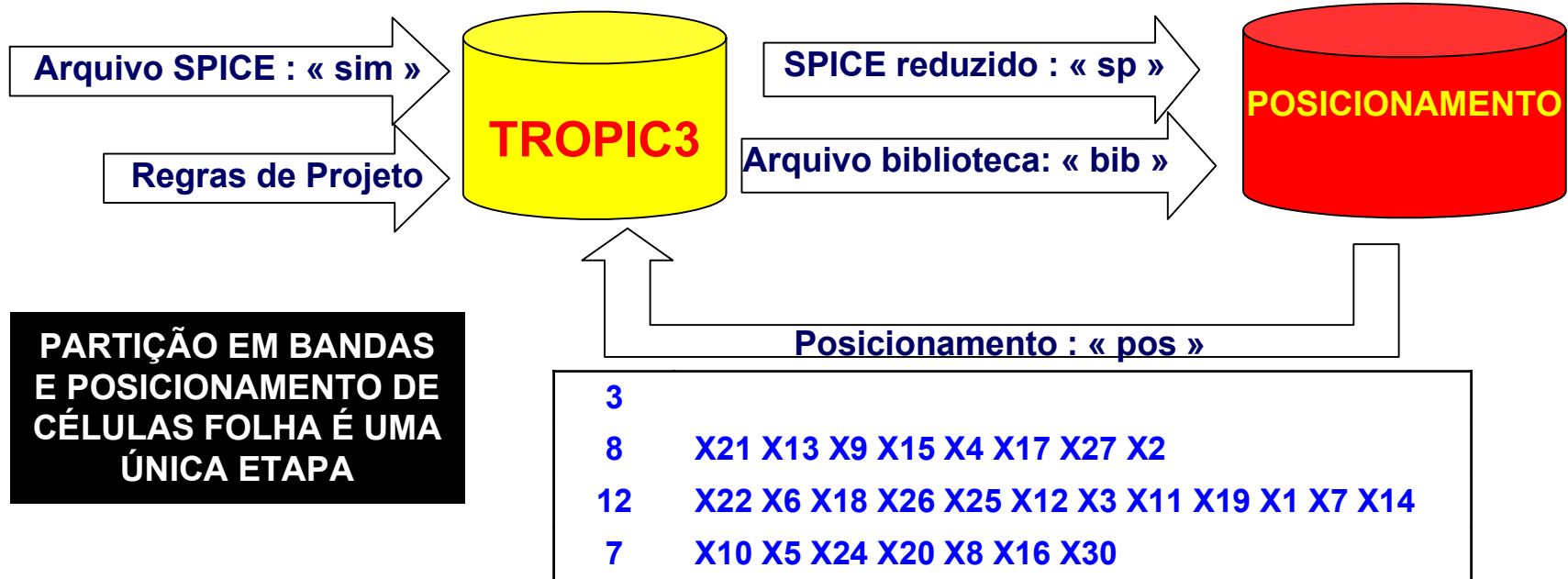
F <= not(((M2 or M3 or M4) and M5) or (M6 and M7 and M8))

Posicionamento de células folha

(1/4)



- **programa externo a TROPIC**
 - entrada: lista de células básicas (sp) e largura das células (bib)
 - saída: células posicionadas

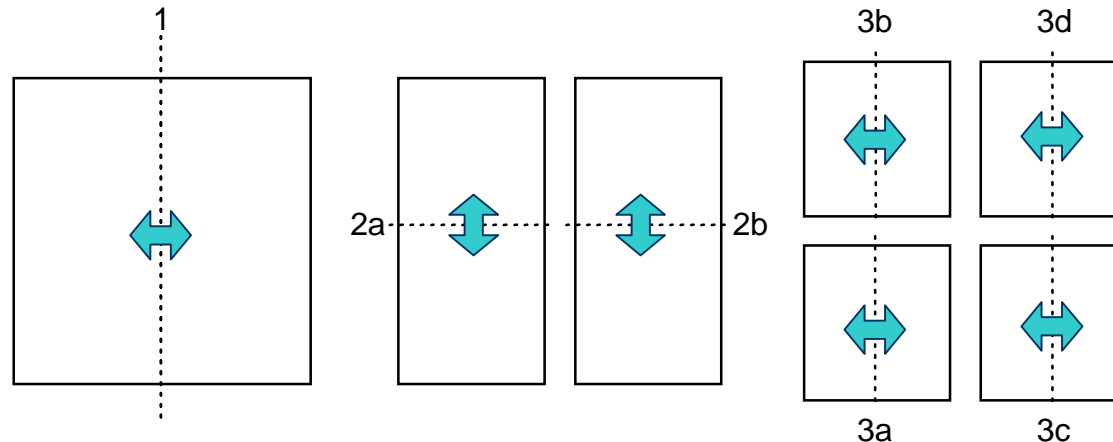


Posicionamento de células folha

(2/4)



- Algoritmo: *min-cut*, ou corte mínimo
 - função custo: menor número de redes cruzando a interface entre os quadrantes
 - algoritmo determinístico
 - razões para adoção:
 - redução do comprimento total das conexões
 - distribuição homogênea do roteamento (horizontal/vertical)
 - divisão sucessiva do circuito em quadrantes

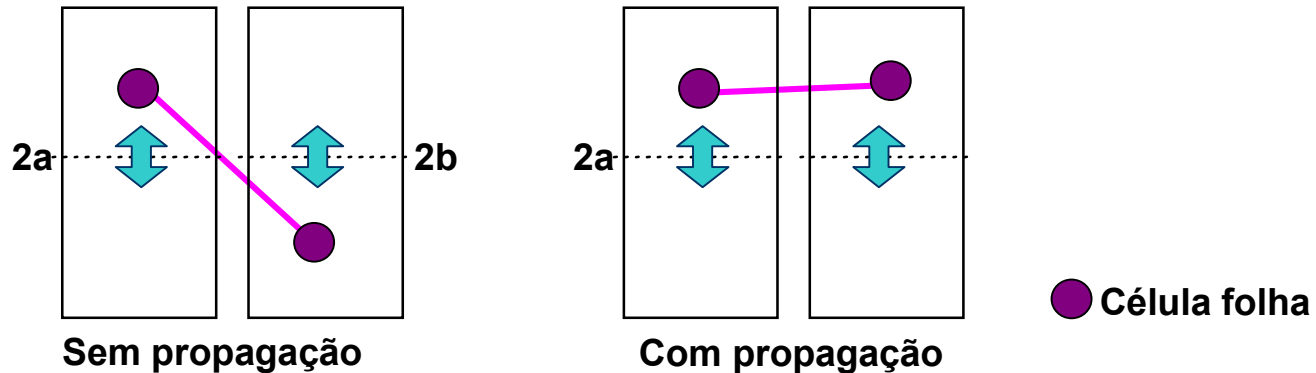


Posicionamento de células folha

(3/4)



- existem dependências entre os quadrantes
- otimização do algoritmo: propagação de pinos



- a partição de cada quadrante leva em consideração os quadrantes já posicionados
- os pinos atuam como “forças”, atraindo as células
 - problema: milhares de pinos para cada quadrante

Posicionamento de células folha

(4/4)



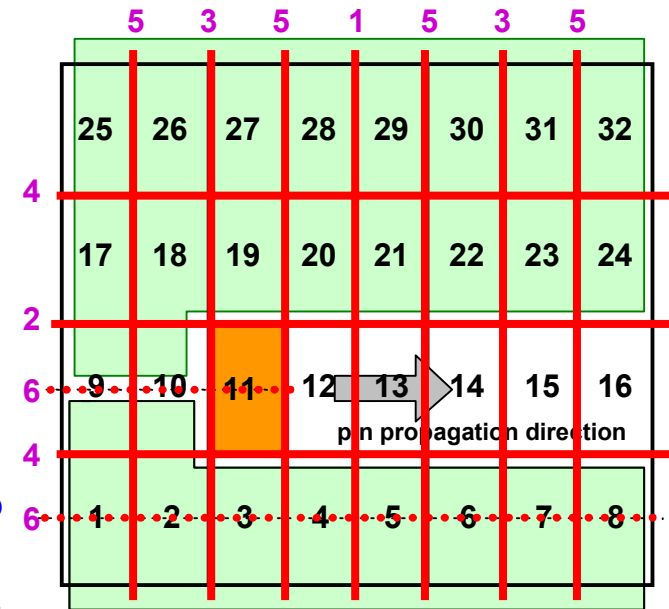
Sinais de interface para a partição horizontal do quadrante 11:

Pinos norte:

- entrada/saídas **norte** da macro-célula presentes no quadrante 11
- sinais **comuns** entre os quadrantes 11 com os quadrantes 17 a 32
- sinais **comuns** entre os quadrantes 11 e os sinais presentes na partição superior dos quadrantes 9 e 10 (já particionados)

Pinos sul:

- entrada/saídas **sul** da macro-célula presentes no quadrante 11
- sinais **comuns** entre os quadrantes 11 com os quadrantes 1 a 8
- sinais **comuns** entre os quadrantes 11 e os sinais presentes na partição inferior dos quadrantes 9 e 10 (já particionados)

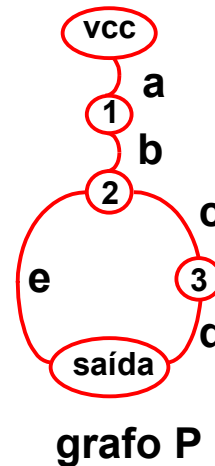
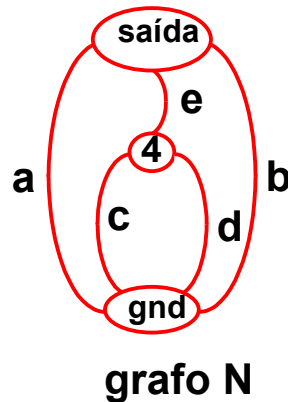
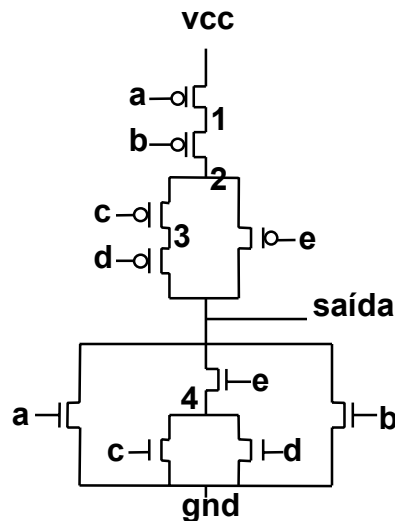


PROPAGAÇÃO HORIZONTAL DE PINOS

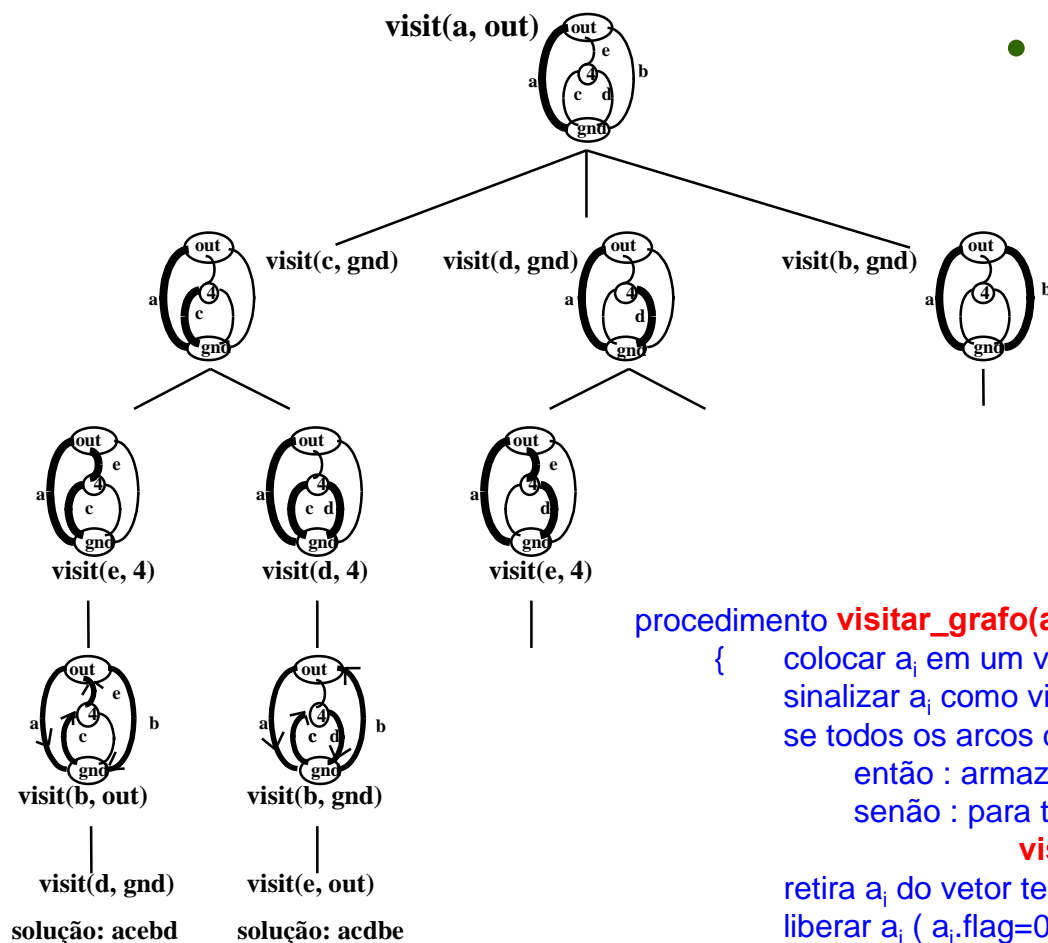
Geração de células (1/4)



- geração de células determina a **ordem** dos transistores dentro de cada célula e **não** o layout da célula
 - entrada: lista de células básicas
 - saída: ordem dos transistores em cada célula folha com número **mínimo** de quebras de difusão
- modelagem de cada grafo através de listas de arestas/vértices



Geração de células (2/4)



- Algoritmo: determinação do caminho de Euler, ou seja, passar por todos os vértices do grafo, apenas uma vez em cada vértice

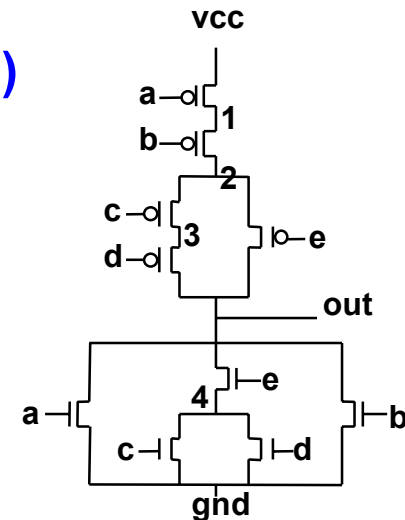
procedimento **visitar_grafo**(a_i, v_k)

```
{
  colocar  $a_i$  em um vetor temporário
  sinalizar  $a_i$  como visitado (  $a_i.flag=1$  )
  se todos os arcos do grafo já foram visitados
    então : armazenar o vetor temporário como uma solução temporária
    senão : para todos os arcos  $a_s$  conectados ao vértice vizinho de  $s_k$  :
      visitar_grafo( $a_s, v_{vizinho}$  )
  retira  $a_i$  do vetor temporário
  liberar  $a_i$  (  $a_i.flag=0$  )
}
```

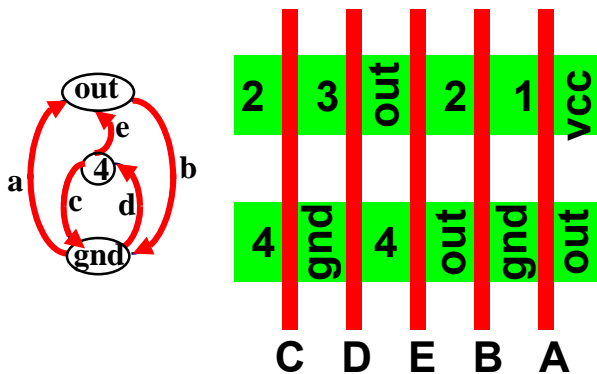

Geração de células (3/4)



- **Ordenamentos encontrados (no exemplo ao lado)**
 - 32 caminhos no plano N
 - 4 caminhos no plano P
 - caminhos comuns: CDEBA e ABEDC (simétricos)
 - 1 linha de roteamento interno em cada plano (nós 4 e 2)

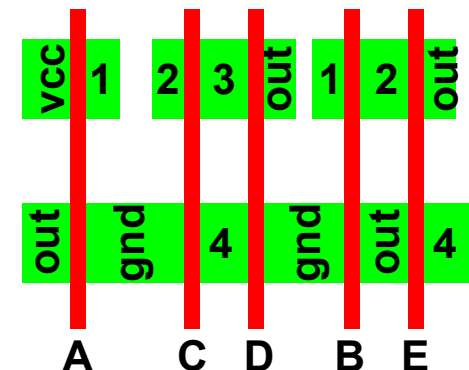


Layout equivalente para ordem CDEBA:



Se ordem for ACDBE:

- 2 gaps no plano P
- área dif. não mínima





Geração de células (4/4)

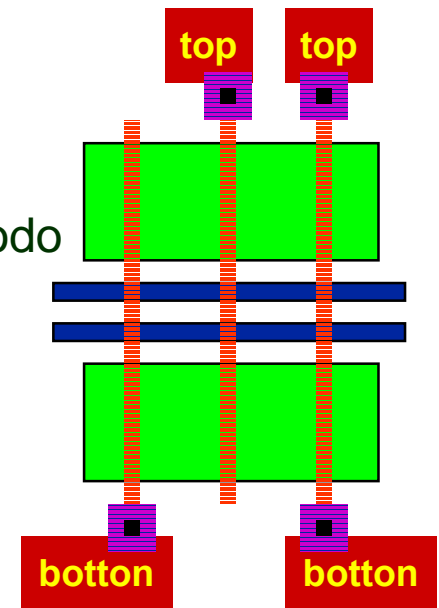


- **Célula sem caminho de Euler**
 - em apenas um plano
 - seleciona-se o ordenamento no plano que tem caminho de Euler de tal forma a **minimizar** o número de quebras no plano sem caminho
 - em ambos planos
 - **situação rara** (contornável por reordenamento de transistores)
 - **solução não ótima, obtida por concatenação de soluções parciais**
 - **TROPIC3 não implementa otimização neste caso, nem reordenamento de transistores**
- **Desempate quando há mais de uma solução comum:**
 - menor número de quebras nas linhas de difusão
 - menor número de linhas de roteamento interno / menor comprimento
 - maior número de conexões à alimentação

Assinalamento de pinos (1/6)



- já definimos: 
 - ordem relativa dos transistores no interior de cada célula
 - posição relativa de cada célula em cada banda
- as células de cada banda podem ter conexões tanto pelo lado superior (*top*) quanto pelo lado inferior (*bottom*) 
- objetivo da etapa de assinalamento de pinos :
 - definir o lado da cada entrada/saída nas células
 - corresponde a criar uma imagem simbólica de todo o circuito a ser gerado
- **DEFINE DE FORMA INDIRETA A LARGURA DO CIRCUITO, pois os pinos t/b serão alinhados à grade de roteamento**



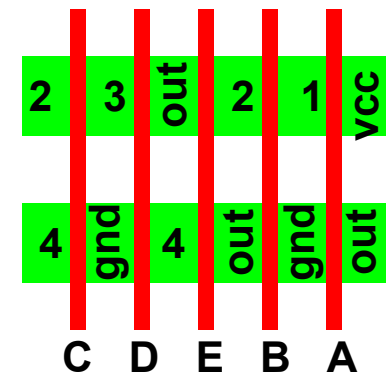
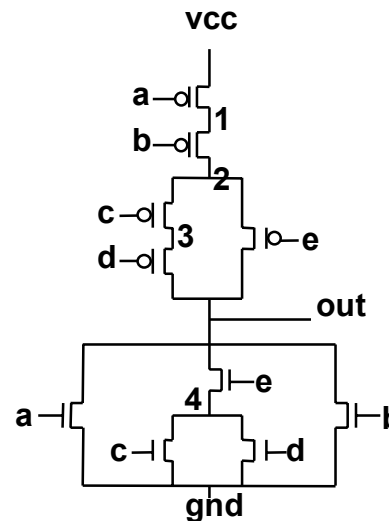
Assinalamento de pinos (2/6)



- Antes do procedimento de assinalamento:
 - filtra-se as redes que não precisam ser roteadas
 - fixa-se quais redes serão roteadas sobre os transistores - OTC:

Exemplo:

- redes filtradas: '1' e '3'
- redes internas: '2' e '4'



Assinalamento de pinos (3/6)



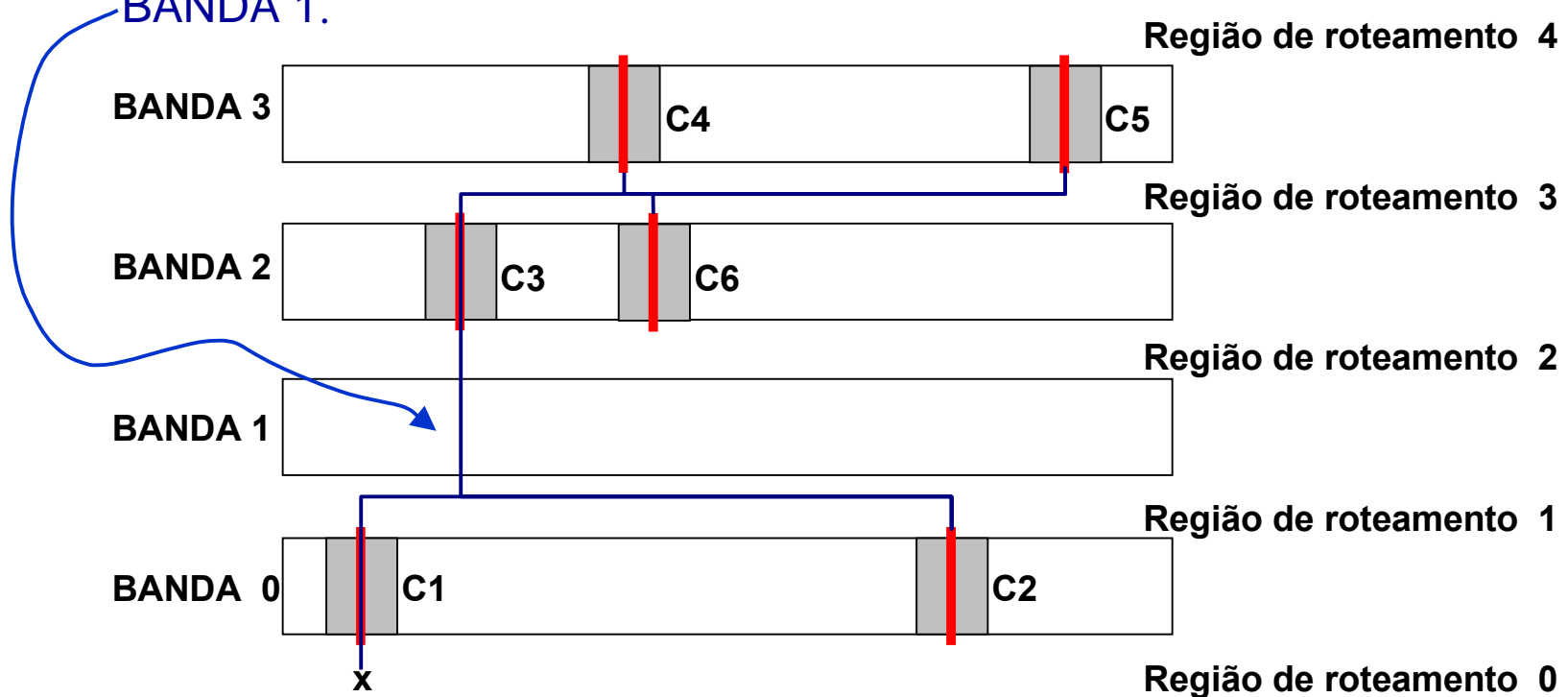
- São redes OTC
 - TODAS as redes internas às portas complexas
 - algumas redes presentes em apenas um canal de roteamento
 - assim como não há praticamente células sem caminho de Euler, o número máximo de trilhas em redes OTC é 4, o que não acarreta em aumento de área
- Vantagem do uso de redes OTC
 - redes OTC não são alinhadas à grade de roteamento, reduzindo-se a área do circuito

Assinalamento de pinos (4/6)



Exemplo de assinalamento correto de pinos

- rede “x”, interface sul, conectada às células C1 a C6
- assinalamento ótimo: t/b C1, t C2, t/b C3, b C4, b C5, t C6
- observar que será necessária a passagem de uma linha em metal3 sobre a BANDA 1.

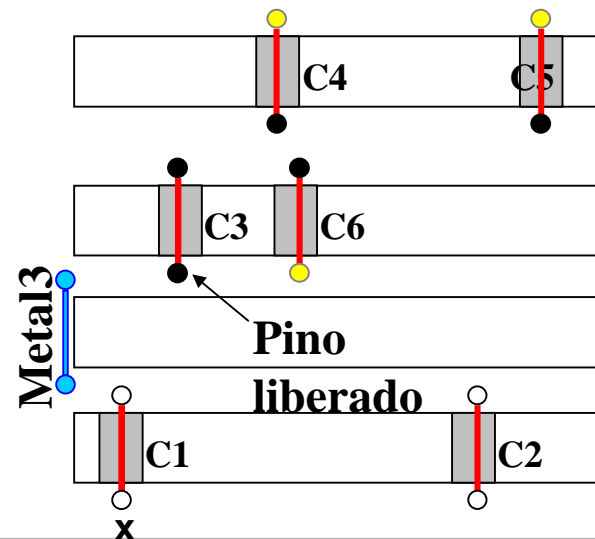
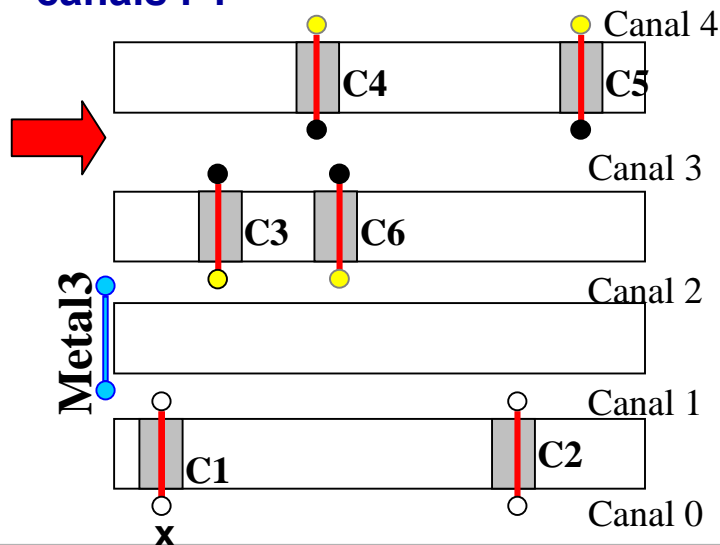


Assinalamento de pinos (5/6)



Procedimento (ver algoritmo no texto)

- escolhe-se o canal 3 (4 pinos livres), marca-se estes 4 pinos como *usados* (em preto), e os demais (pinos superiores do canal $i+1$ e inferiores do canal $i-1$) como *mortos* (em amarelo)
- como nos canais superiores ao canal $i+1$ não há pinos livres, não há necessidade de liberar nenhum pino
- porém como nos canais $i-1$ há pinos livres, *libera-se* um pino cuja coordenada esteja dentro do intervalo das coordenadas da rede nos canais $i-1$

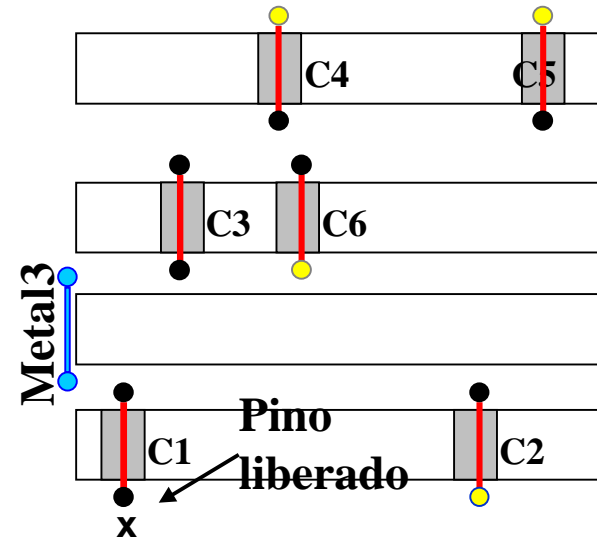
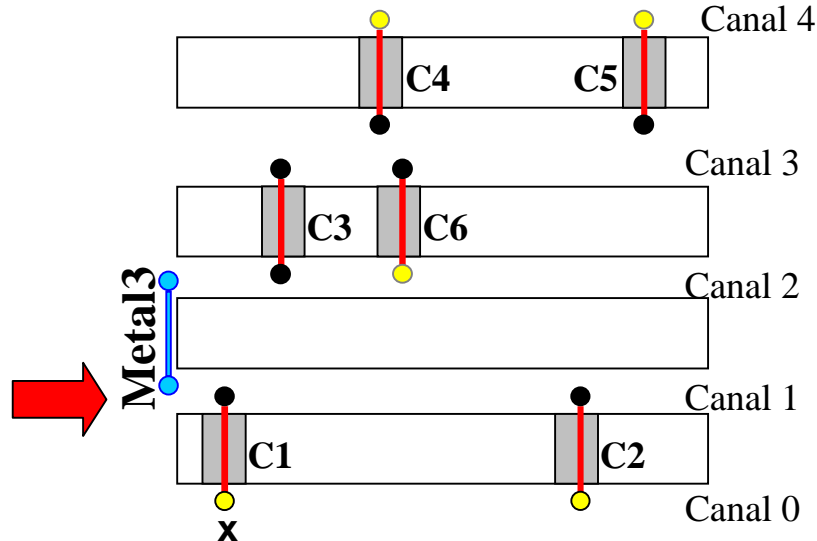


Assinalamento de pinos (6/6)



Procedimento (continuação)

- escolhe-se agora o canal 1, pois ele tem 3 pinos livres (células C1 e C2 e ponte em metal3 a ser inserida sobre a banda 1). Matar os pinos no canal 0.
- o pino escolhido para liberação é aquele conectado na interface sul
- como não há mais nenhum canal com pino livre, o algoritmo termina



Geração de banda (1/3)



- dependente das regras de desenho
- determina a coordenada real de cada dreno/gate/source
- principal função custo: redução das áreas de difusão, visando reduzir capacidades parasitas
- **observação:** ao final desta etapa permite-se mover células entre bandas adjacentes, visando uniformizar a largura das bandas e assim reduzir a área ocupada

Geração de banda (2/3)



- **algoritmo guloso**

- varre cada banda da esquerda para a direita, coluna a coluna

para todas bandas r do circuito

para todas as células c da banda r

para todos os transistores t da célula c

para os planos n e p de t {

- determinar a coordenada em microns do dreno t referente ao plano n ou p , levando-se em conta se o dreno deve ser alinhado ou não à grade virtual.
 - Se for alinhado, fixar a coordenada geométrica com a primeira posição livre da grade
 - Se não for alinhado, utilizar regra de distância mínima em relação ao source anterior.
- determinar a coordenada em microns do gate t referente ao plano n ou p . Mesmas restrições que o dreno, tomado-se o cuidado de não se colocar linhas de polisilício em curto-circuito;
- determinar a coordenada em microns do source t referente ao plano n ou p , segundo as mesmas restrições.

Fixação dos feedthroughs (1/5)

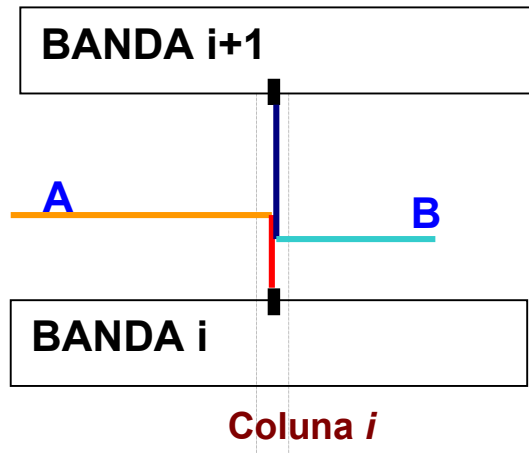


- fixação das coordenadas de metal3 que conectam bandas não adjacentes
- 4 regras restringem o uso do metal3:
 1. os *feedthroughs* são alinhados à grade vertical de roteamento
 2. o número de redes diferentes por coluna no canal de roteamento é no máximo 2
 3. não são permitidos *feedthroughs* sobre gates de transistores OTC
 4. não são permitidos *feedthroughs* pertencentes a diferentes redes em bandas adjacentes com mesma coordenada

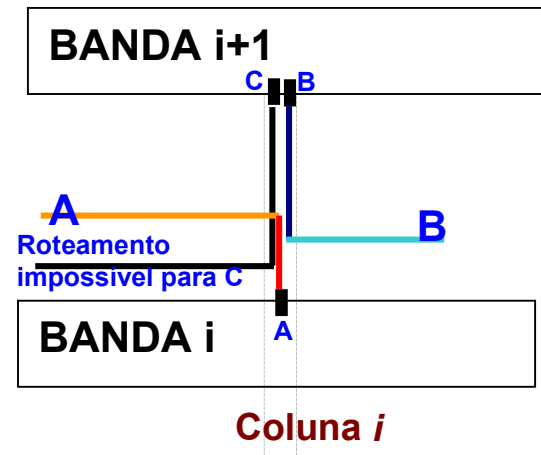
Fixação dos feedthroughs (2/5)



- o número de redes diferentes por coluna no canal de roteamento é no máximo 2 (podem haver até 4 pinos na mesma coluna, desde que pertencentes a 2 redes)



duas redes na mesma colunas são sempre roteáveis com 4 níveis de roteamento, sem jogs

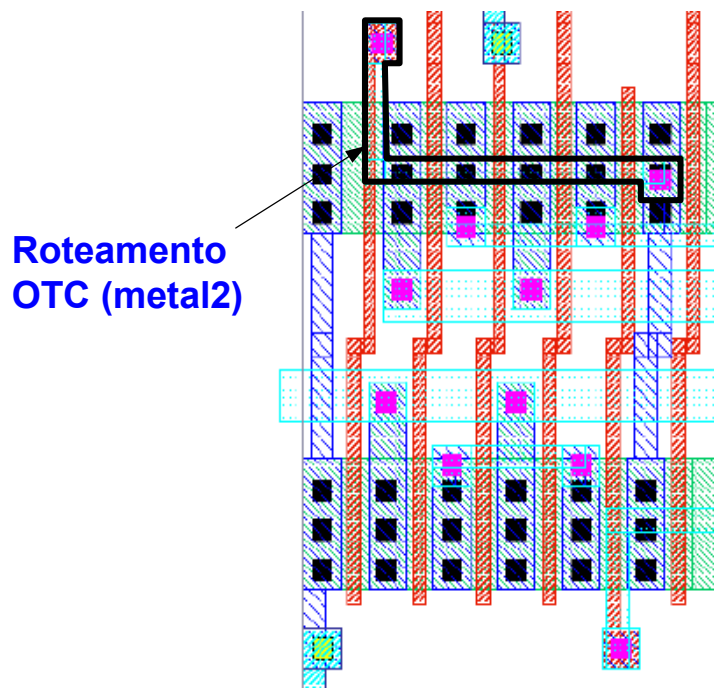


Caso tenhamos 3 redes na mesma coluna, o roteamento pode até ser realizável em alguns caso, mas como o apresentado é impossível.

Fixação dos feedthroughs (3/5)



- não são permitidos *feedthroughs* sobre gates de transistores OTC



- o metal2 usado para a conexão OTC, **pode vir** a impedir uma conexão de um metal3 ao canal de roteamento
- logo, há um certo **compromisso** em espaço para feedthroughs e número de redes OTC

Fixação dos feedthroughs (4/5)

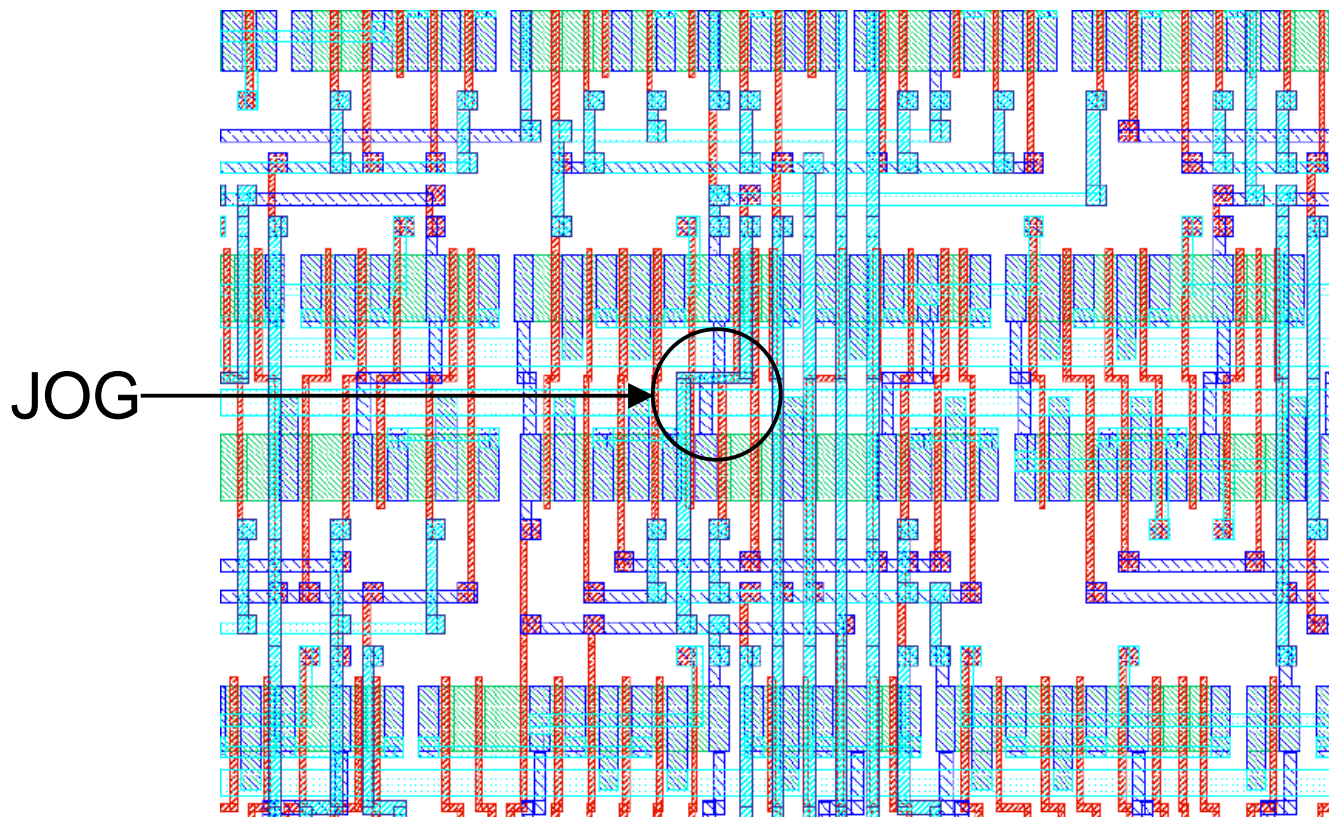


- **Procedimento:**
 - ordena-se as redes conforme o número de feedthroughs necessários (inicia-se pelas redes com o maior número de feedthroughs)
 - o procedimento é baseado no cálculo dos intervalos nas redes em bandas superiores e inferiores para cada feedthrough a ser inserido
 - tenta-se posicionar o feedthrough na coordenada calculada, caso não dê, tenta-se a primeira posição à esquerda, a primeira à direita e assim sucessivamente, até encontrar posição livre
- **Há inserção automática de jogs nos feedthroughs**
- **É hoje a única etapa de TROPIC3 que pode abortar a síntese devido à falta de espaço para inserção de feedthroughs**
- **Como contornar o problema: menor número de bandas ou utilizar interfaces leste/oeste ao invés de norte/sul**

Fixação dos feedthroughs (5/5)



- Exemplo:



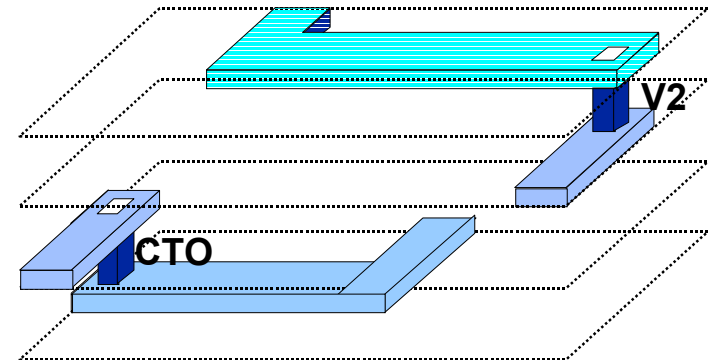
Roteamento de canal (1/4)



- algoritmo de base: left-edge
- roteamento é simbólico, sobre uma grade (passo de contato)
- utilização de 4 camadas de roteamento, com dois canais **superpostos e independentes**
- em determinadas situações, de ciclos verticais, utiliza-se um só nível horizontal e dois verticais

Porque dois canais superpostos?

- redução da área de roteamento
- problema a avaliar: capacitâncias de acoplamento



Roteamento de canal (2/4)

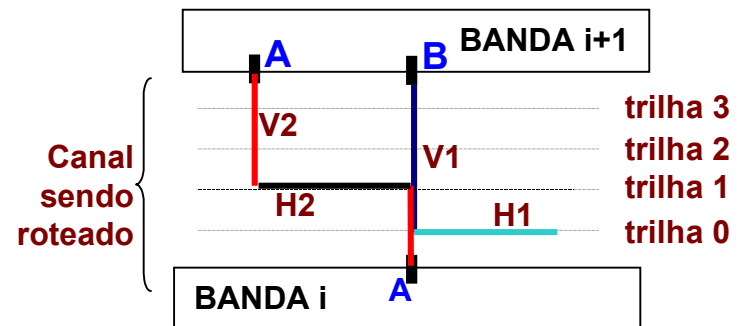


Para todos os canais de roteamento:

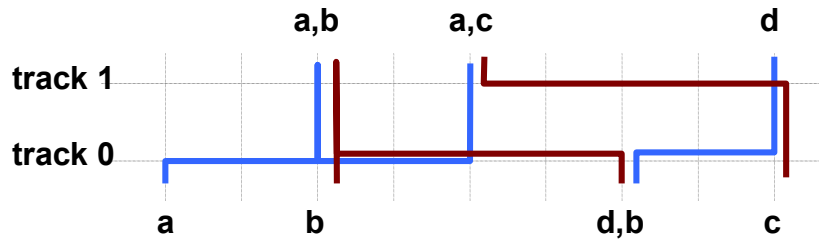
- (a) **Trilha = 0;**
- (b) **Repetir enquanto houverem redes a serem roteadas no canal:**
 - (b.1) Inserir na trilha corrente o maior número possível de redes, iniciando na coordenada esquerda do canal indo até a extremidade direita, utilizando o nível de roteamento horizontal1. Para que uma rede seja inserida, não devem haver restrições verticais.
 - (b.2) Repetir 'b.1', para o nível horizontal2. Estamos neste caso sobrepondo 2 canais de roteamento.
 - (b.3) Se 'b.1' e 'b.2' falharem (restrições verticais em todos os fios), insere-se redes utilizando ambos níveis verticais para o mesmo nível horizontal. Neste caso não é permitido sobreposição de roteamento.
 - (b.4) $\text{trilha} = \text{trilha} + 1$;

Exemplo de restrição vertical:

- supor que estamos roteando a trilha 1
- rede 'A' não pode ser roteada com H1, devido à rede 'B' estar utilizando V1/H1



Roteamento de canal (3/4)



Roteamento:

Net 'a', with H1-V1, in track 0

Net 'b', with H2-V2, superposed to net 'a', in track 0

Net 'd', with H1-V1, superposed to net 'd', in track 0

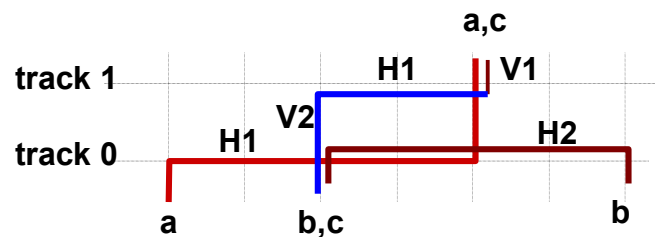
Net 'c', with H2-V2, in track 1.

Observar:

- superposição vertical das camadas
- podem haver mais de dois pinos por coluna, desde que respeitada a restrição de duas redes diferentes

.....

Exemplo de ciclo vertical:



Considere as redes 'a' e 'b' já roteadas.

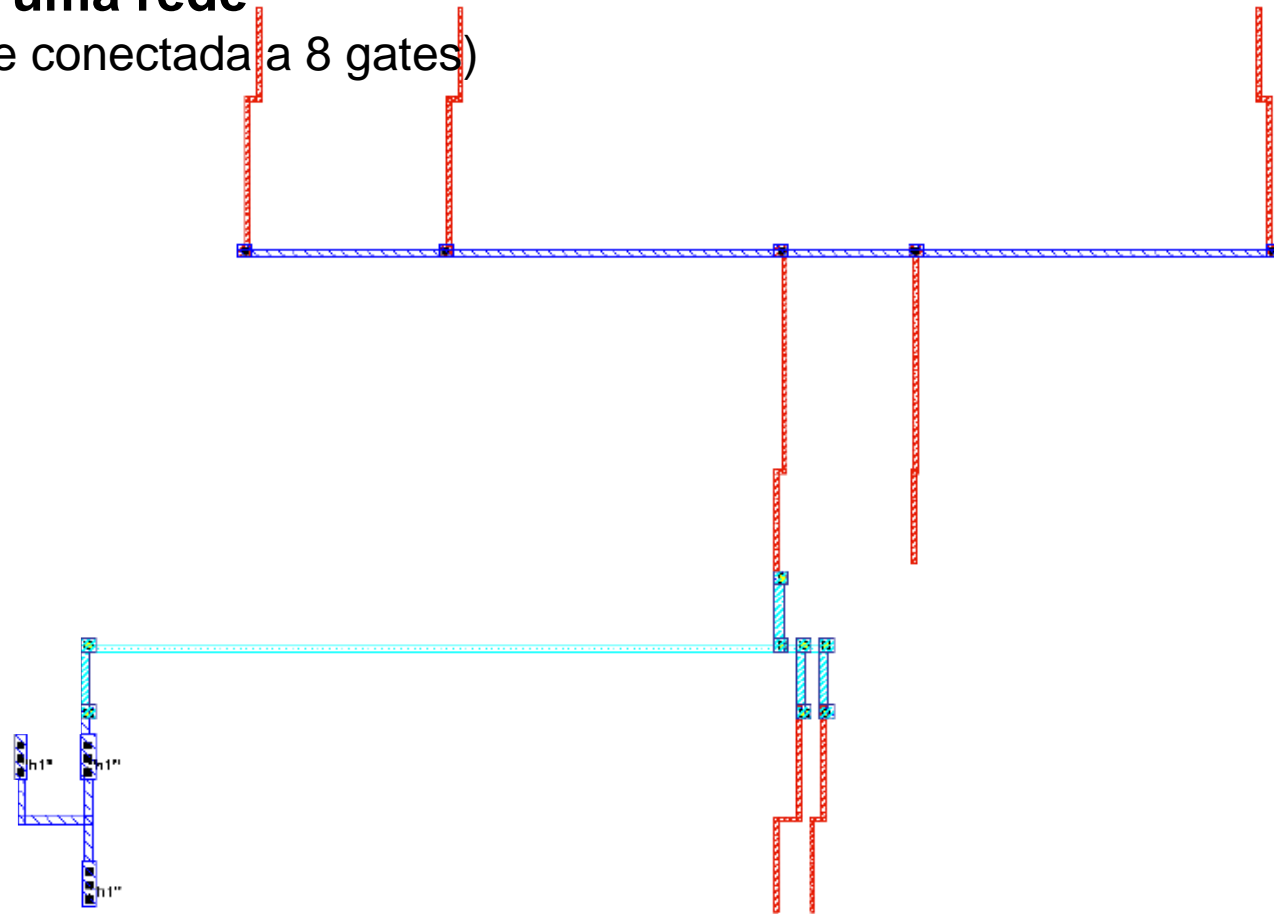
Rede 'c':

- não pode usar H1/V1
- não pode usar H2/V2
- SOLUÇÃO: usar H1/V1/V2

Roteamento de canal (4/4)



Roteamento típico de uma rede
(na figura temos uma rede conectada a 8 gates)



Geração do arquivo CIF (1/2)



- última etapa do procedimento de síntese
- simplesmente escreve-se a base de dados no formato CIF
 - outros formatos GDS2 ou RS (não implementados)
- procedimento
 - gera-se os polígonos do canal 0, banda 1, canal 1, banda 2, etc.
- complexidade típica da base de dados:
250.000 polígonos

Geração do arquivo CIF (2/2)



- Arquivo contendo apenas retângulos, camadas e textos
- Não há hierarquia neste arquivo

DS 0 1 10;
9 adder;
L metal1;
B 600 600 12300 2300;
L metal2;
B 600 600 12300 2300;
L metal3;
B 600 600 12300 2300;
L v2;
B 400 400 12300 2300;
L v1;
B 400 400 12300 2300;
L poly;
B 600 600 12300 300;
L contact;
B 300 300 12250 250;
L metal1;
B 600 600 12300 300;
L metal2;
B 600 600 12300 300;

L metal3;
B 600 600 12300 300;
L v2;
B 400 400 12300 300;
L v1;
B 400 400 12300 300;
L text;
94 carry 12250 250;
L metal3;
B 600 2000 12300 1000;
L metal1;
B 600 600 300 2300;
...
...
...
L text;
94 c 5250 14850;
L poly;
B 250 1000 5125 14100;
L metal1;
B 13400 400 11700 13800;
DF;

Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
3. TROPIC3 - Estilo de layout
4. Etapas da síntese da layout
- 5. Predição de Parasitas**
6. Integração da Síntese Física à Síntese Lógica
7. Direções Futuras

Extração de capacitâncias parasitas



- **Objetivo**

- Prover ao usuário uma estimativa **rápida** das capacitâncias/ resistências parasitas
- Uso: ferramentas de timing, simuladores elétricos/lógicos, síntese lógica

- **Difusão**

- áreas de dreno e source calculadas por TROPIC (relatório)

- **Roteamento**

- ferramenta dedica a *extração de fios* - **LASCA**
- cálculo da área e perímetro de cada conexão
- cálculo das capacitâncias de acoplamento
 - mesmo nível e entre níveis adjacentes

- **Warning em nós fortemente carregados** ($C_{load}/C_{in} > C_{limit}$)

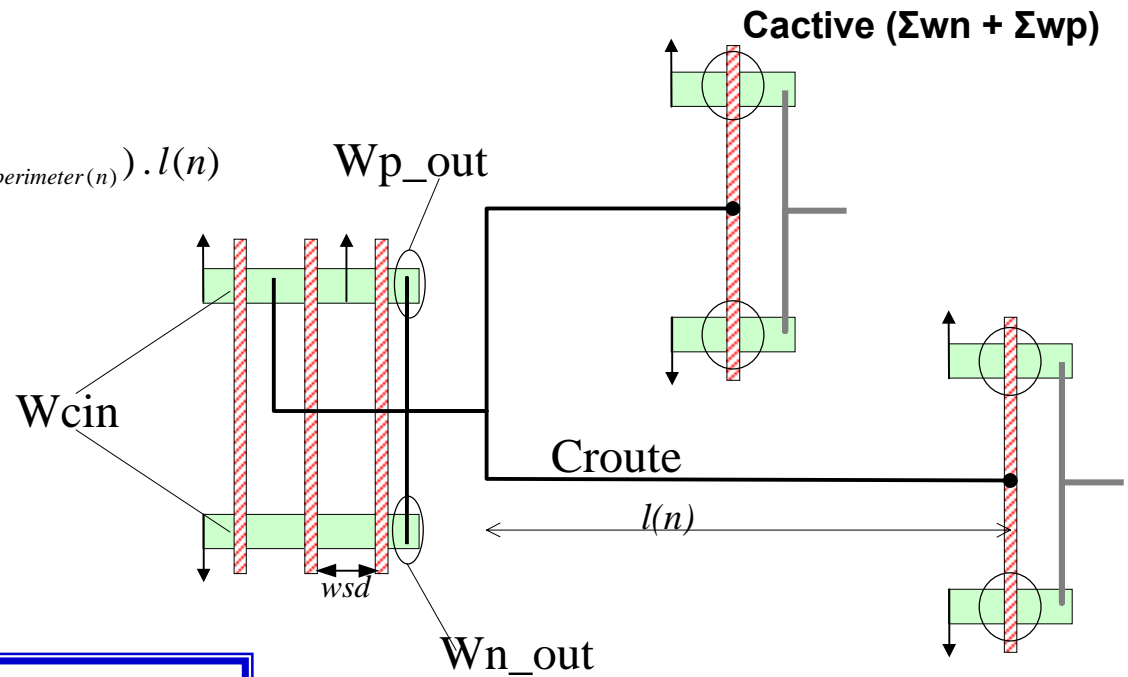
- útil para indicar onde inserir buffers ou dimensionar transistores

Capacitâncias parasitas



$$C_{active} = (\sum W_n + \sum W_p) \cdot l_{min} \cdot Cox$$

$$C_{route} = \sum_{n=poly,m1,m2,m3} (C_{area(n)} \cdot w(n) + 2 \cdot C_{perimeter(n)}) \cdot l(n)$$



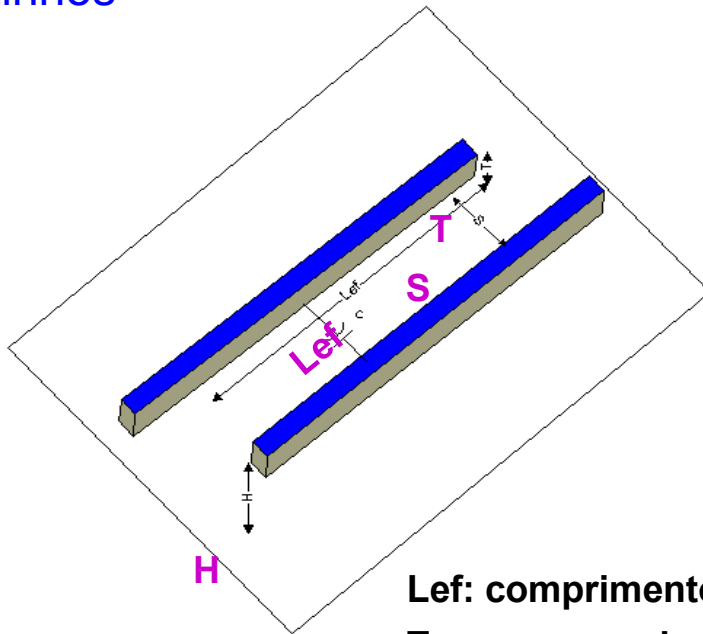
$$C_{load} = C_{active} + C_{route} + C_{diffusion}$$

$$C_{in} = (avg(W_{n_cin}) + avg(W_{p_cin})) \cdot l_{min} \cdot Cox \cdot drive$$

Capacitâncias parasitas de acoplamento



Capacitâncias entre fios no mesmo nível:
calculada a partir da área comum entre fios vizinhos



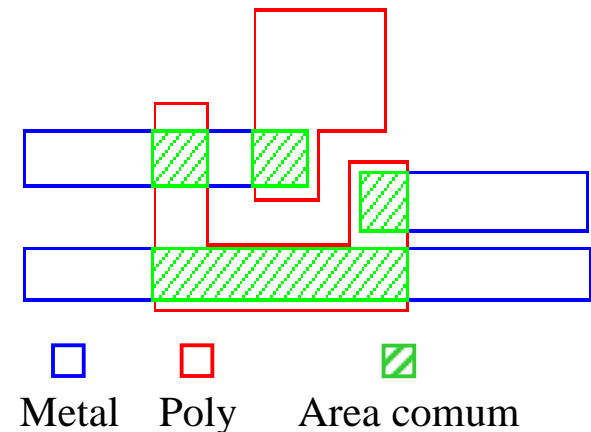
Lef: comprimento comum entre os fios

T: espessura do nível

S: distância entre os fios

H: distância ao plano terra

Capacitância de crossover:
calculada pela área comum entre os níveis adjacentes



Avaliação das Capacitâncias

1/4



- Comprimento das conexões
 - histograma gerado pelo LASCA

POSICIONAMENTO: DISTRIBUIÇÃO HOMOGÊNEA DO COMPRIMENTO DAS CONEXÕES

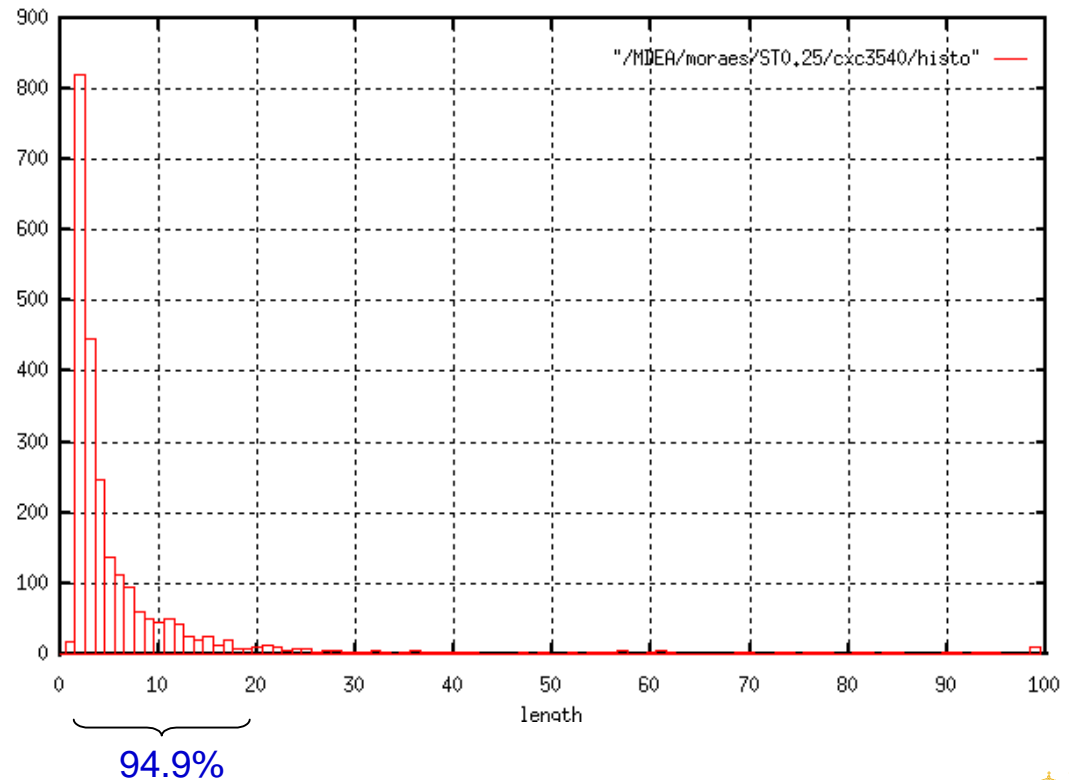
Exemplo: C3540

- $0.25\mu\text{m}$ ($w=2\mu\text{m}$)
- 7154 transistores
- 2349 redes
- **94.9%** < 200 μm
- 121 redes com Cload/Cin > Clim

$C_{100\mu\text{m}} = 7$ (M3) to 11.4 (poly) fF
(área e perímetro)

$C_{IN(w=2\mu\text{m})} = 6.9$ fF

P	0-100 μm	85.9 %	[2027]
P	101-200 μm	9.0 %	[213]
P	201-300 μm	2.2 %	[51]
P	301-400 μm	0.7 %	[17]
P	401-500 μm	0.3 %	[6]
		98.1 %	[2314 / 2359]



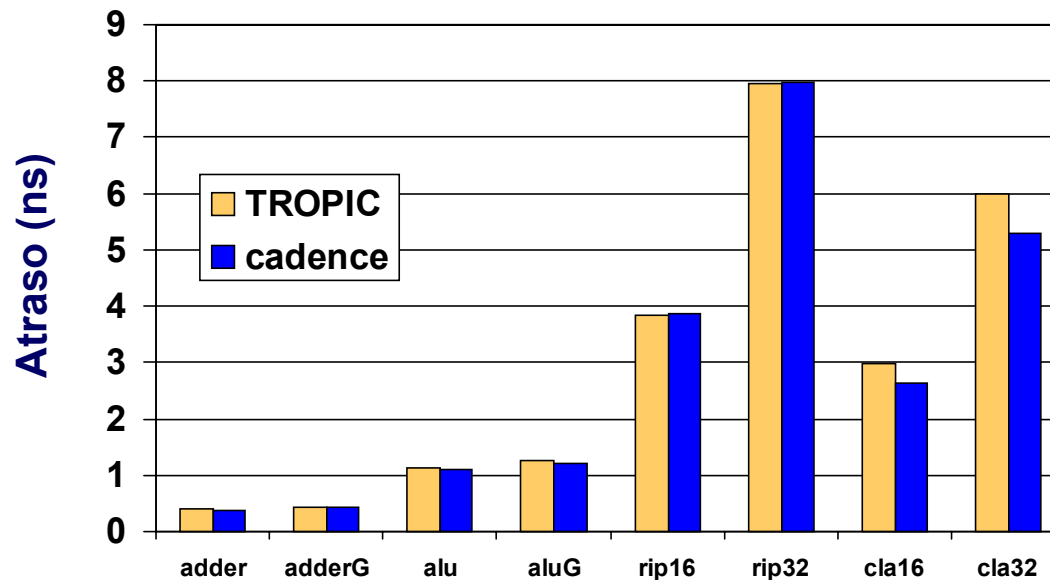
Avaliação das Capacitâncias

comparação de atraso

2/4



- **Capacitâncias extraídas:**
 1. LASCA: área, perímetro e acoplamento
 2. CADENCE: área, perímetro, acoplamento e dispositivos (extrator elétrico)
- **Vantagens em usar TROPIC3 + LASCA:**
 - velocidade e menor número de ferramentas no fluxo de projeto
 - menos de 5% de diferença entre as simulações obtidas simulando-se com as capacitâncias calculadas (LASCA) e extraídas



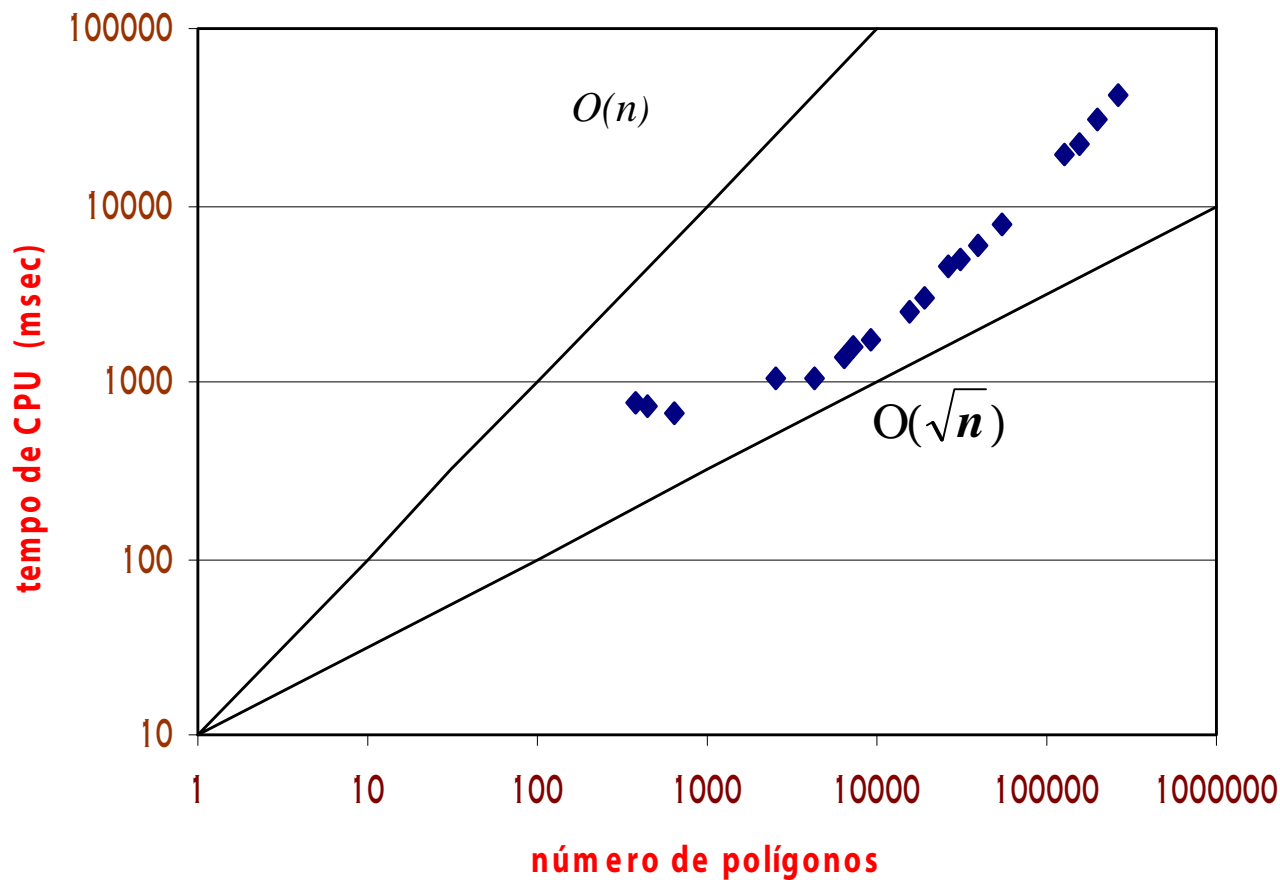
Dados de simulação:
W=2um L=0.25um
Load = 35f

Benchmark

Avaliação das Capacitâncias

tempo de CPU

3/4



Avaliação das Capacitâncias

largura *versus* n# de buffers

4/4



- Área é proporcional à largura dos transistores (*óbvio, mas nem tanto...*)
- Solução inicial de dimensionamento é aquela que minimize o número de buffers, sem um acréscimo muito grande na área

Circuito: mult2 (1239 redes)

W μm	Número estimado de buffers	Densidade (tr/mm^2)
0.5	342	62004
1.0	174	62004
1.5	94	62004
1.8	82	62004
2.0	79	60767
2.5	74	57882
3.0	72	55258

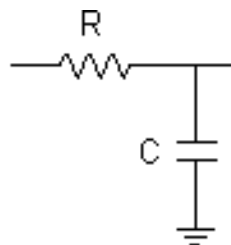
mesma área

Technology: 0,25 μm

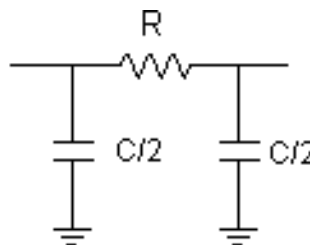
Modelos de Extração com Resistências



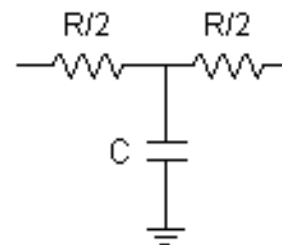
- definido pelo usuário



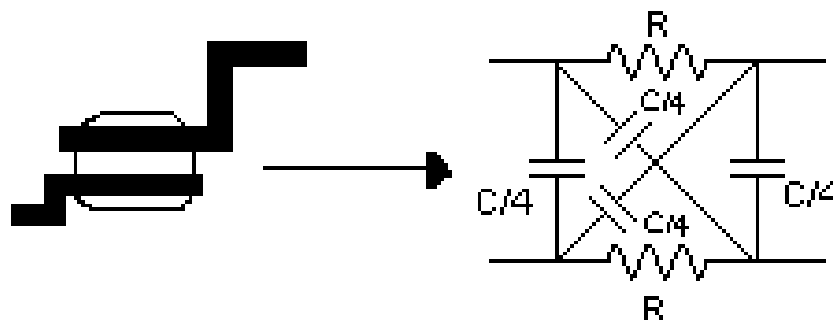
L lumped



π lumped



T lumped



Cruzamento de duas conexões e a respectiva extração.

Relatório gerado pelo LASCA, além do arquivo “extraído”



- **Parte 1: capacitâncias relativas à capacitância de entrada**
 - C_{act}/C_{in} , C_{rot}/C_{in} , C_{dif}/C_{in} e C_{load}/C_{in} ($C_{load} = C_{act} + C_{rot} + C_{dif}$)
 - se $C_{load}/C_{in} > F_{LOAD}$ imprimir-se warning
- **Parte 2: valor absoluto das capacitâncias**
 - C_{act} , C_{rot} , C_{dif} , C_{load} e C_{in}
- **Parte 3:**
 - número de transistores conectados a cada rede
 - comprimento total de cada rede (por nível)
 - número total de contatos/vias em cada rede
- **Parte 4: parâmetros topológicos de cada célula**
 - fanin / fanout, número de transistores em série, W médio da célula
 - número máximo de transistores conectados ao caminho serial mais longo

Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout



1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
3. TROPIC3 - Estilo de layout
4. Etapas da síntese da layout
5. Predição de Parasitas
- 6. Integração da Síntese Física à Síntese Lógica**
7. Direções Futuras

Integração da Síntese Física à Síntese Lógica



- **Primeira alternativa**

- utilizar ferramentas dedicadas para mapeamento tecnológico (TABA, SYNTHETIC)

- fluxo:

síntese lógica \Rightarrow portas simples \Rightarrow mapeamento \Rightarrow AOIs \Rightarrow TROPIC

- **Alternativa inadequada, pois:**

- as restrições temporal e/ou de área devem ser aplicadas sobre as células que serão utilizadas de fato, e não portas simples

- estas ferramentas para mapeamento não trabalham com lógica síncrona, o que impede de gerar circuitos reais

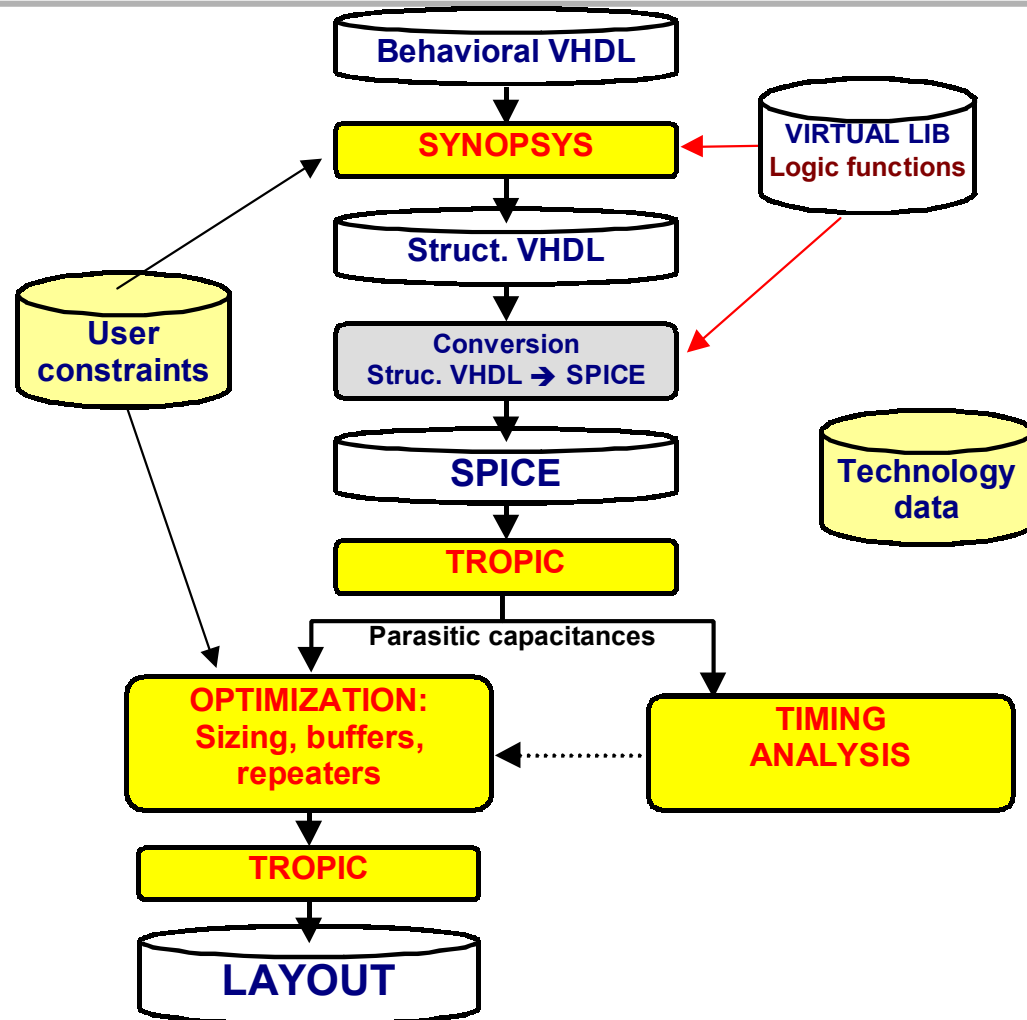
- uso de formatos intermediários que não são padrões

Integração da Síntese Física à Síntese Lógica

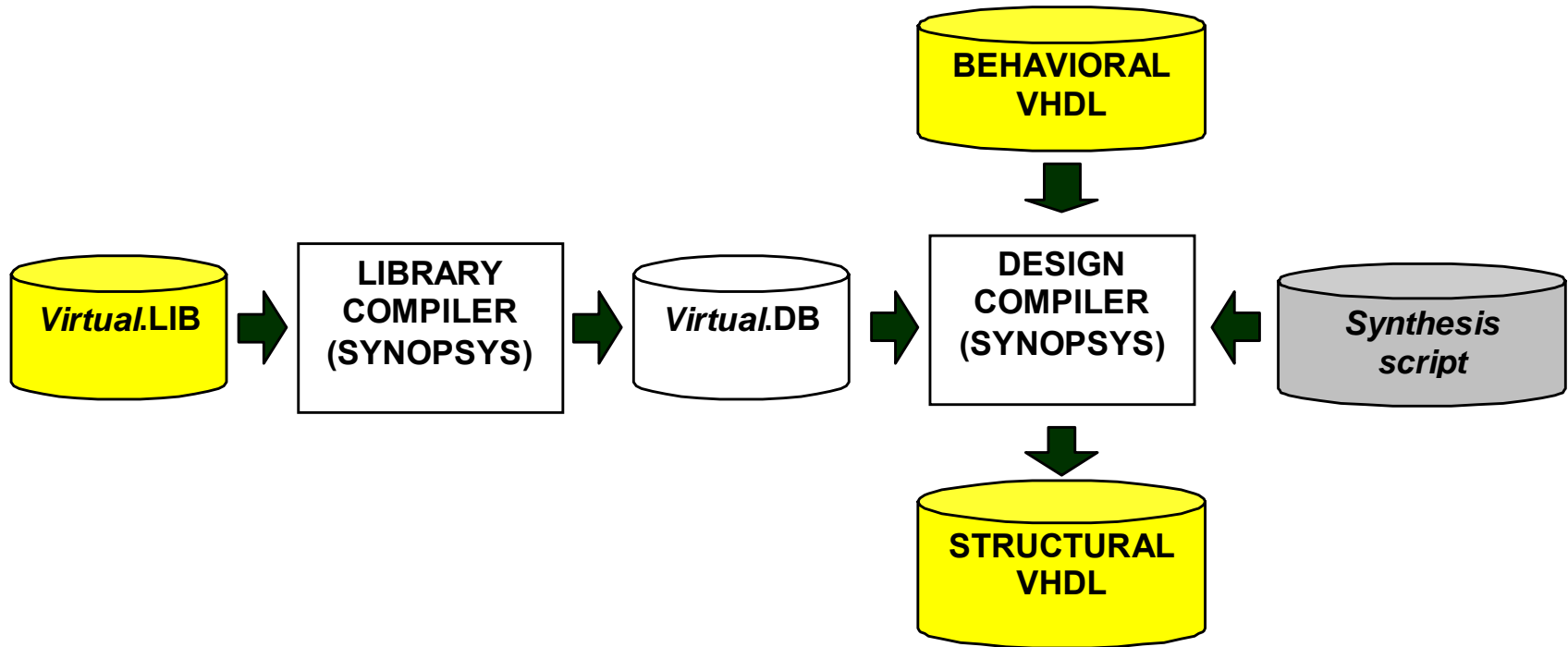


- Segunda alternativa
 - uma ferramenta que realize ambas tarefas, síntese lógica e mapeamento tecnológico
- Alternativa adequada, pois:
 - é possível modelar a área e o atraso para cada célula da biblioteca virtual
 - as restrições de área/atraso serão aplicadas sobre as células que realmente serão utilizadas na síntese física
 - pode-se sintetizar circuitos com lógica síncrona e tri-states
 - entrada a partir de VHDL

Fluxo de Projeto (*ideal*)



Biblioteca Virtual - Synopsys (1/3)

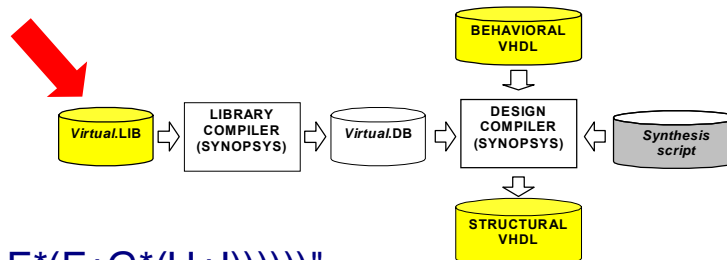


Biblioteca Virtual - Synopsys (2/3)



- Exemplo de célula complexa

```
cell(AOI1125) {  
  area : 18;  
  pin(z) {  
    direction : output;  
    function : "! (A*(B+(C*(D+E*(F+G*(H+I))))))";  
    max_capacitance : 0.250000;  
    timing() {  
      intrinsic_rise : 0.25 ;  
      intrinsic_fall : 0.30 ;  
      rise_resistance : 0.15 ;  
      fall_resistance : 0.10 ;  
      related_pin : "A B C D E F G H I" ;  
    }  
  }  
  pin(A B C D E F G H I) {  
    direction : input;  
    capacitance : 0.250000;  
  }  
}
```



Biblioteca Virtual - Synopsys (3/3)



- flip-flops, latches e tri-states estão presentes em um arquivo padrão devido à complexidade destes elementos
- são configurados para cada tecnologia por simulação
- 9 elementos nesta biblioteca:
 - 4 FFDs (D, set, reset, set/reset)
 - 4 LATCHs (D, set, reset, set/reset)
 - 1 buffer tri-state
- documentação Synopsys indica como conjunto mínimo 3 elementos:
 - FFD set/reset
 - LATCH D set/rest
 - buffer tri-state

Script de síntese



```
link_library = {sccg33.db libdff.db}  
target_library = {sccg33.db libdff.db}
```

// Bibliotecas de portas complexas e de elementos
// sequenciais

```
cell_list = { sklansky }
```

```
foreach (cell, cell_list) {
```

```
    read -f vhdl cell + ".vhdl"  
    current_design cell
```

```
    set_input_delay 0.0 all_inputs()  
    set_max_delay 3.5 to all_outputs()  
    out_load = 0.25  
    set_load out_load all_outputs()  
    set_max_area 300  
    set_max_fanout 6 cell
```

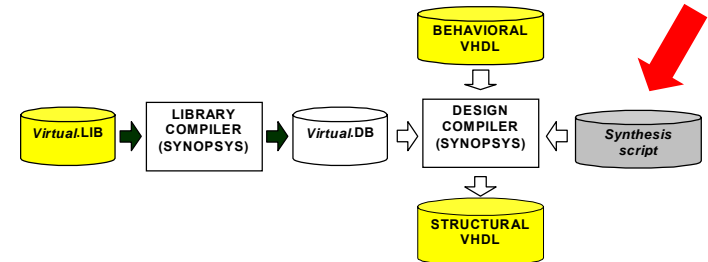
```
    compile -map_effort medium
```

```
    /* Removes multiply-instantiated hierarchy */  
    uniquify -force  
    ungroup -flatten -all
```

```
    vhdlout_write_components = TRUE  
    vhdlout_single_bit = "TRUE"  
    vhdlout_dont_create_dummy_nets = "TRUE"
```

```
    write -format vhdl -hierarchy -output cell + "_S.vhdl"
```

```
    }  
quit
```



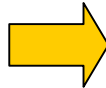
VHDL em SPICE



- a saída da síntese lógica conterá uma lista de instâncias de portas virtuais

U31 : AOI13 port map (PGij(1), PGij(0), PGjk(1), PGjk(0), PGkl(1), s2);

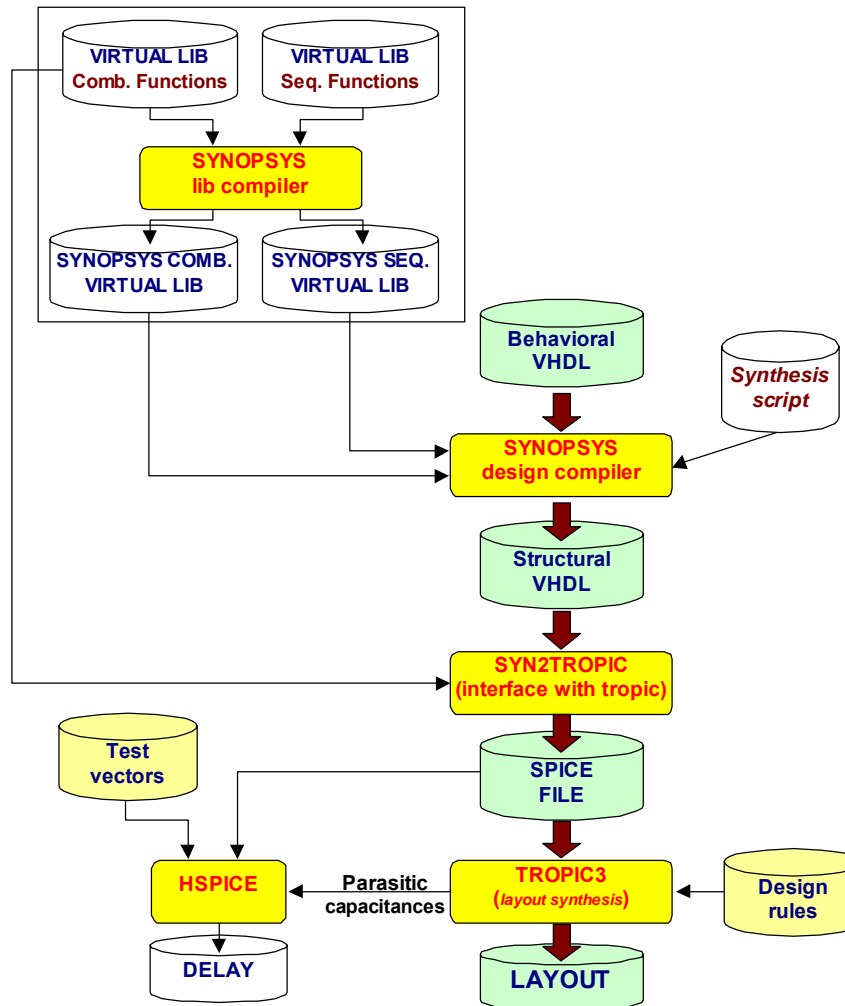
Da linha “function” na
descrição da célula virtual:
 $!((A*B*C)+(D*E*F)+(G*H*I))$



```

SUBCKT AOI1 out A B C D E F G H vcc
MN1      gnd      I      6      NMOS    w=2.0um  l=0.25um
MN2      6        H      3      NMOS    w=2.0um  l=0.25um
MN3      3        G      out    NMOS    w=2.0um  l=0.25um
MN4      gnd      F      5      NMOS    w=2.0um  l=0.25um
MN5      5        E      2      NMOS    w=2.0um  l=0.25um
MN6      2        D      out    NMOS    w=2.0um  l=0.25um
MN7      gnd      C      4      NMOS    w=2.0um  l=0.25um
MN8      4        B      1      NMOS    w=2.0um  l=0.25um
MN9      1        A      out    NMOS    w=2.0um  l=0.25um
MP10     vcc      I      16     PMOS    w=2.0um  l=0.25um
MP11     vcc      H      16     PMOS    w=2.0um  l=0.25um
MP12     vcc      G      16     PMOS    w=2.0um  l=0.25um
MP13     16       F      15     PMOS    w=2.0um  l=0.25um
MP14     16       E      15     PMOS    w=2.0um  l=0.25um
MP15     16       D      15     PMOS    w=2.0um  l=0.25um
MP16     15       C      out    PMOS    w=2.0um  l=0.25um
MP17     15       B      out    PMOS    w=2.0um  l=0.25um
MP18     15       A      out    PMOS    w=2.0um  l=0.25um
.ends AOI1
    
```

FLUXO DE PROJETO HOJE



Exemplo Completo



```

Entity div4 is
  Port(
    clk,rst: in std_logic;
    div, div4, div8: out std_logic );
end div4;
Architecture arch of div4 is
  component DIV2
    Port( clk,rst: in std_logic; d2_clk: out std_logic );
  end component;
  signal carry : std_logic_vector(2 downto 0);
begin
  st0 : DIV2 Port Map ( clk => clk, rst=> rst, d2_clk => carry(0) );
  st1 : DIV2 Port Map ( clk => carry(0),rst=> rst, d2_clk => carry(1) );
  st2 : DIV2 Port Map ( clk => carry(1),rst=> rst, d2_clk => carry(2) );
  div <= carry(0);
  div4 <= carry(1);
  div8 <= carry(2);
end;

```

```

Entity DIV2 is
  Port(clk,rst: in std_logic; d2_clk: out std_logic );
end DIV2;
Architecture arch of DIV2 is
  signal temp : std_logic;
begin
  d2_clk <= temp;
  process (clk,rst)
  begin
    if (rst='0') then
      temp <= '0';
    elsif (clk'event and clk='1') then
      temp <= not temp;
    end if;
  end process;
end;

```

```

**
**   SUBCIRCUITS   (generated from the lib file)
**
.SUBCKT DFFR D rst clk QN Q vcc
MN1      H clk gnd gnd NMOS  l=0.25u  w=2u
MP2      H clk vcc vcc PMOS  l=0.25u  w=2u
MN3      NH  H gnd gnd NMOS  l=0.25u  w=2u
MP4      NH  H vcc vcc PMOS  l=0.25u  w=2u
...
...
.ends DFFR

.SUBCKT GAT1_1 out A vcc
MN1      gnd A out gnd NMOS  l=0.25u  w=2u
MP2      vcc A out vcc PMOS  l=0.25u  w=2u
.ends GAT1_1

**
**   COMPONENTS   (generated from the VHDL file)
**
X1  net21 n3 clk net21 div vcc DFFR
X2  net20 n3 n1 net20 div8 vcc DFFR
X3  net19 n3 n2 net19 div4 vcc DFFR
X4  n3 rst vcc GAT1_1
X5  n2 net21 vcc GAT1_1
X6  n1 net19 vcc GAT1_1

**
**   INPUTS AND OUTPUTS
**   (generated from the ENTITY in the VHDL file)
**
*interface: div8      orient n output
*interface: div4      orient n output
*interface: div       orient n output
*interface: rst       orient s input
*interface: clk       orient s input

```

Sumário

Anatomia de uma Ferramenta de Síntese Automática de Layout

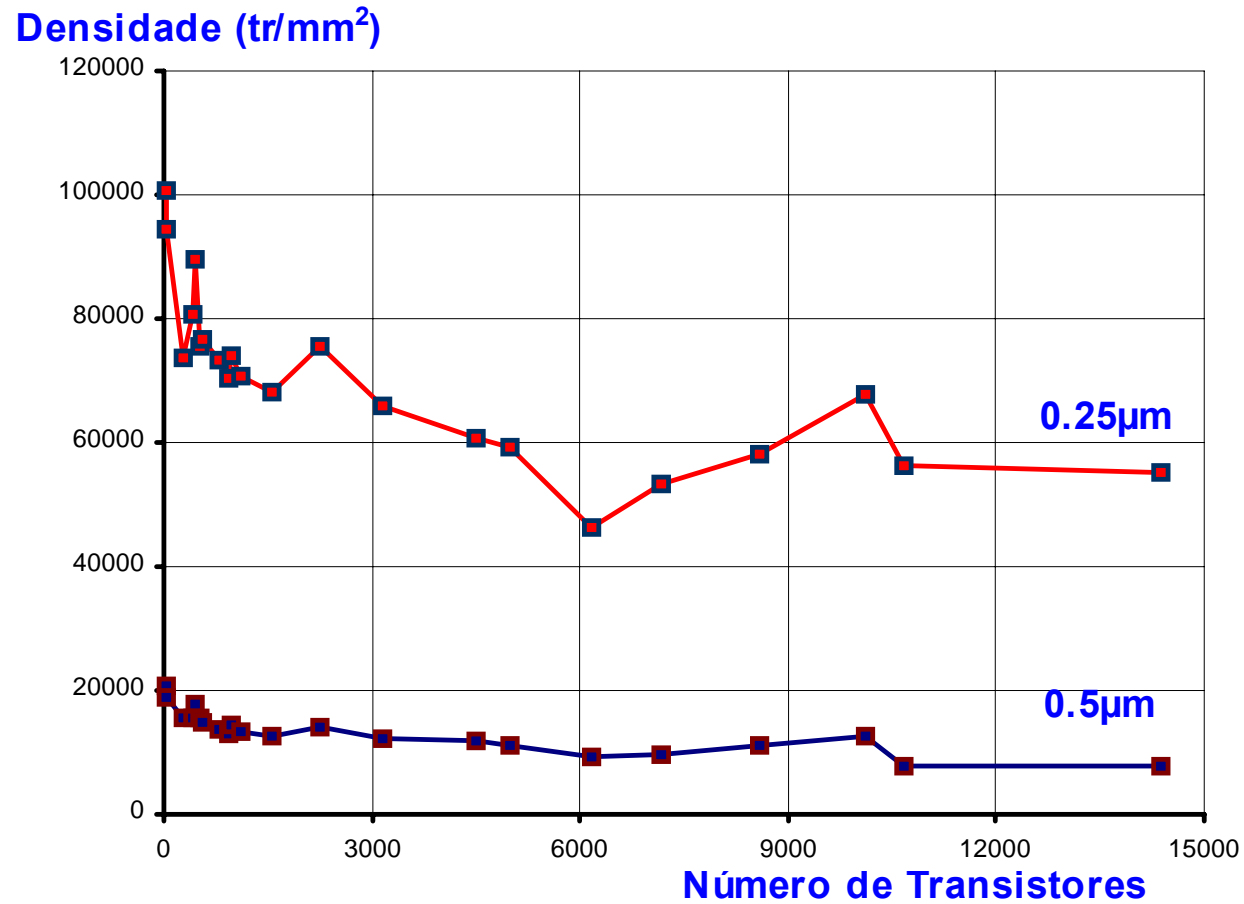


1. Síntese Automática e Bibliotecas Virtuais
2. Estilos de layout
3. TROPIC3 - Estilo de layout
4. Etapas da síntese da layout
5. Predição de Parasitas
6. Integração da Síntese Física à Síntese Lógica
7. Direções Futuras

Resultados (1/3)



- Área



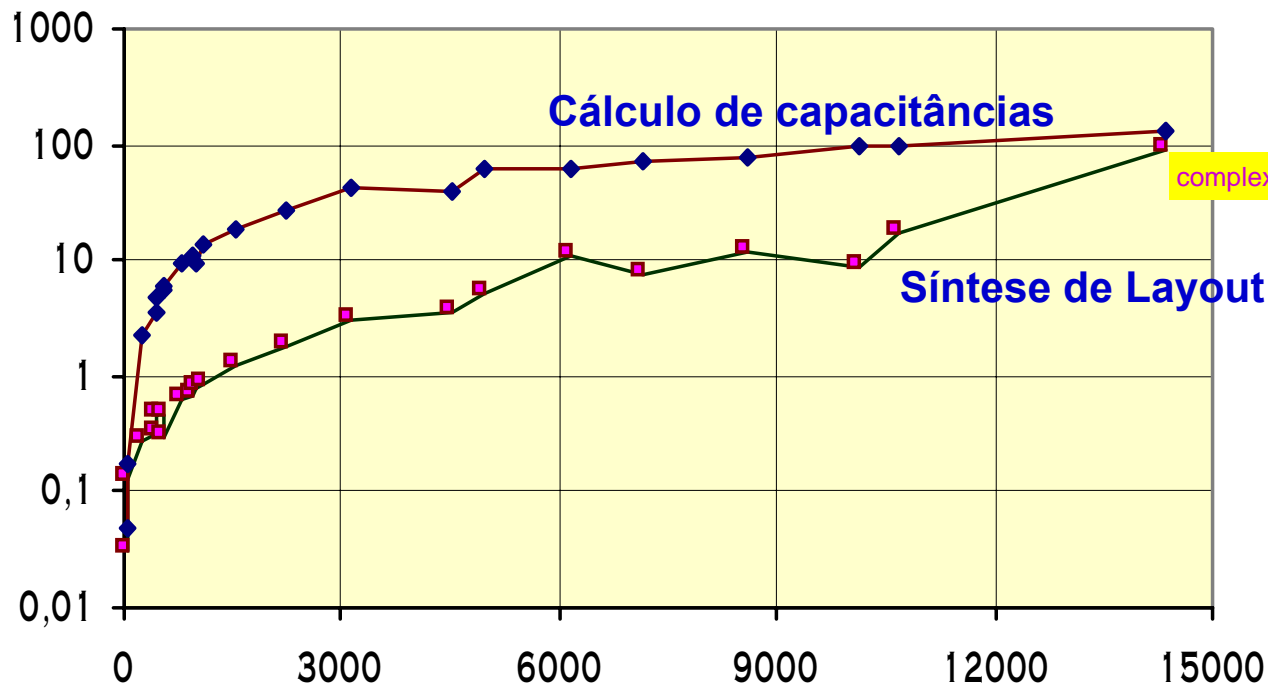
Resultados (2/3)



- **Tempo de CPU**

- menos de 2 minutos para sintetizar 15000 transistores !

Tempo de CPU (s) [log]



22 benchmarks
Ultra-sparc 10

Número de
transistores

Resultados (3/3)



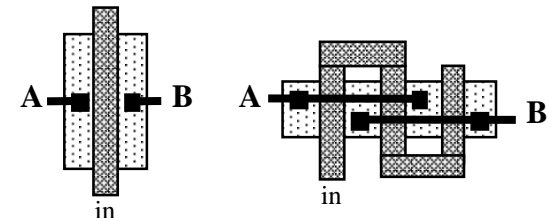
- Comparação com LAS e TROPIC2 (tec. 0.7 microns)
- LAS e TROPIC2 usam compactação de layout

Circuit	Tr#	Rows#	LAS dml	TROPIC2	TROPIC3
Adder	28	2	4780	5942	10136
Addergate	40	2	5598	6020	9542
Alu	260	4	5812	5294	7606
Alugate	432	4	6050	5405	7494
Rip	448	5	5961	5610	8726
Cla	528	5	5614	5498	8133
Hdb3	570	4	4903	6516	7892
Mult6	972	7	5950	5779	7152
Mult2	4512	16	4879	4080	5924

Trabalho a continuar (1/3) - física



- 1) Efetivar o trabalho de melhora no estilo de layout
 - múltiplos jogs no polisilício
 - escolha da inserção do local dos body-ties
 - tornar opcional múltiplos contatos nos drenos/sources (silicetos)
- 2) Mudar a estratégia de roteamento - considerar uma única área de roteamento ao invés de canais+OTC
- 3) Compactar os canais de roteamento, como filtragem de contatos/vias
- 4) Síntese de lógica não dual
- 5) Estratégia de paralelização de transistores grandes
- 6) Tornar homogênea as áreas de difusão



Trabalho a continuar (2/3) - otimização



- 7) **Implementar procedimentos de otimização**
 - dimensionamento de transistores
 - inserção de buffers
- 8) **Validar os modelos de cálculo de parasitas, principalmente os modelos π/T**
- 8) **Integrar a síntese física a um ferramenta de estimação de atraso que considere capacitâncias parasitas e portas complexas**
- 10) **Integrar a síntese física a um ferramenta de estimação de potência dissipada**

Trabalho a continuar (3/3) - lógica



- 11) Melhorar a modelagem da temporização das células virtuais (uso de look-up tables)
- 12) Automatizar o procedimento da geração da biblioteca virtual
- 13) Definir um procedimento de calibração dos flip-flops, latches e tri-state
- 14) Realimentar a síntese lógica com as capacitâncias parasistas calculadas
- 15) Manter o mesmo placement entre várias sínteses físicas
- 16) Sintetizar (verdadeiros) benchmarks a partir de VHDL com o TROPIC3 ⇒ **síntese de soft cores**

CONCLUSÃO (1/2)



- **Ferramenta de síntese física operacional**
 - sem compactação de layout
 - fácil migração tecnológica
 - permite partir de SPICE e chegar diretamente nas máscaras
 - iniciando integração com VHDL
- **Há muito trabalho a ser feito**
- **MOTOROLA: chance para tornar síntese física automática uma realidade**

CONCLUSÃO (2/2)



Referências Adicionais

- **Artigos e texto completo da EMICRO**

<http://www.inf.pucrs.br/~moraes/FMpapers.html>

- **Software**

<http://www.inf.pucrs.br/~moraes/tropic2000.tar.gz>

- **Contato**

moraes@inf.pucrs.br