In: Pesquisas Aplicadas em Modelagem Matemática - Volume I ISBN 0000000000
Editor: Dr. Paulo Sausen, pp. 1-16

Capítulo 1

MODELO MATEMÁTICO PARA TRANSFORMAÇÃO DE CIRCUITOS VLSI PLANARES EM CIRCUITOS VLSI 3D BASEADA EM PARTICIONAMENTO DE HIPERGRAFOS

Sandro Sawicki, Maira De Vlieger
E-mail addresses: {***sawicki,maira.vlieger***}***@unijui.edu.br***
Mestrado em Modelagem Matemática,
Universidade Regional do Noroeste do Estado do Rio Grande do Sul (UNIJUÍ),
Ijuí – RS – Brazil

Resumo

Circuitos 3D surgem como uma mudança no paradigma de projetos de circuitos integrados. São construídos através da integração de vários chips planares fabricados separadamente. Cada circuito é chamado na literatura de tier, e os fios que conectam tiers adjacentes formam interconexões verticais conhecidas como TSV *(Through Silicon Vias)* ou vias-3D. Este capítulo apresenta uma metodologia para geração de circuitos VLSI 3D equivalentes aos circuitos planares, para isso, modela matematicamente os passos dessa transformação. Além disso, estuda o impacto do particionamento de pinos de I/O na redução do número de interconexões verticais. Apresenta um modelo baseado na proximidade lógica dos pinos utilizando pesos entre pares de pinos de I/O, gerado a partir de um hipergrafo (circuito planar), para obter o mínimo corte entre as camadas (*tiers*). O modelo proposto calcula a área das tiers juntamente com o posicionamento dos pinos de I/O. O espaço em branco, a relação de aspecto, assim como a orientação dos pinos são preservados do circuitos inicial. Os resultados obtidos mantém a distribuição balanceada dos pinos entre as partições, minimizando o número de interconexões verticais quando comparados com dois outros métodos para particionamento de pinos.

1 Introdução

É de senso comum que a otimização das interconexões é uma questão de extrema importância para desempenho dos circuitos integrados. Atraso, potência, ruído e *crosstalk*

são algumas das questões relacionadas ao tamanho dos fios. Da mesma forma, a tecnologia envolvida no projeto de semicondutores avança rapidamente. Um reflexo dessa evolução pode ser percebido claramente através do crescimento do numero de elementos dentro de um chip. Assim, diversas questões elétricas que eram desprezadas em tecnologias anteriores começam a ser consideradas. Por exemplo, efeitos de uma conexão sobre outra, integridade do sinal, distribuição do consumo de potência, frequência de relógio como forte limitador, indutância, etc. Essas questões reforçam ainda mais a necessidade da criação de novos algoritmos e metodologias. Nesse sentido, o uso de circuitos VLSI 3D é uma possibilidade real para melhorar a qualidade das interconexões. Grandes empresas como IBM, Intel, Samsung, Micron, Cadence e Infineon estão investindo em soluções relacionadas a essa área [1]. Este trabalho modela matematicamente os passos para a conversão de circuitos planares em circuitos VLSI 3D e estuda o problema de particionamento de pinos de I/O juntamente com seu impacto no número de vias (TSVs) e tamanho dos fios *(wirelenght)*. Sabe-se que o posicionamento de pinos de I/O é mais eficiente se executado durante a etapa de *floorplanning*. Por outro lado, atualmente, quase todos os projetos e ferramentas são direcionadas às tecnologias 2D. Percebe-se, contudo, que uma técnica automática para migrar de uma tecnologia 2D para 3D pode reduzir significativamente o tempo de projeto. Embora alguns trabalhos [2][6] possam ter usado alguns critérios para fazer o particionamento de pinos de I/O, os detalhes de como foi implementado foram omitidos. Assume-se, então, que eles adotam soluções simplistas para o tratamento dos pinos de I/O. Esta abordagem assume que a fronteira de um bloco aleatório, em circuitos 2D, é delimitada por pinos de I/O. Também assume-se que os pinos de I/O podem ser movidos para qualquer tier. Muitos algoritmos de posicionamento de células são dirigidos pela localização fixa dos pinos. Algoritmos de posicionamento quadráticos [8], por exemplo, que são largamente usados na academia [9] e indústria [10] requerem que a posição dos pinos de I/O já esteja determinada para que seja computada a solução. Este trabalho é organizado da seguinte forma: A seção 2 apresenta a tecnologia VLSI 3D, suas estratégias e empilhamento e integração. A seção 3 relata os trabalhos relacionados e que envolvem o particionamento e posicionamento de circuitos VLSI 3D. Na seção 4 define-se o problema e o modelo matemático proposto para a transformação planar em 3D. Já nas seções 5 e 6 são discutidos, respectivamente, os resultados obtidos e as conclusões deste trabalho.

2 Circuitos VLSI 3D

As análises de Banerjee [7] e resultados práticos publicados por Davis [2] e Pavlidis [21] descrevem melhorias obtidas pelos circuitos VLSI através da metodologia 3D. Além disso, esse potencial pode ser explorado com propriedade pelas ferramentas de CAD.

2.1 Estratégias de Montagem de Circuitos VLSI 3D

Empilhamento de *Chips*: consiste na sobreposição vertical de *chips* pré-fabricados. Sua realizada através de conexões de I/O conhecidas como *wire bonding* [2]. Essas conexões passam pelo lado de fora do circuito, fazendo a ligação entre os diferentes chips (Figura 1). Tal metodologia não provê nenhuma vantagem à performance e à potência do circuito em comparação com outras técnicas de montagem, pois sua integração é fracamente acoplada

devido às conexões externas. Essa técnica reduz somente a área ocupada pelo chip na placa. Por esse motivo, é muito utilizada em dispositivos portáteis e telefones celulares.

Empilhamento de *Chip*-Sobre-*Wafer*: são circuitos pré-testados que são, posteriormente, empilhados sobre o wafer e posicionados individualmente por meio de equipamentos pick-and-place. Patti [17] relata que a precisão no alinhamento deste tipo de procedimento depende da via utilizada para comunicar os chips. Atualmente, o erro no alinhamento varia em torno de 10μ. Essa técnica provê melhor integração em comparação ao empilhamento de *chips*, pois a comunicação é fortemente acoplada, ou seja, as conexões cruzam pela parte interna do *chip*. Essa tecnologia de montagem 3D é adotada por companhias como ZyCube [18], Ziptronix [19] e Xanoptix [20].

Figura 1. Empilhamento de *chips* de tamanhos diferentes por meio de uma estrutura *wire bonded* de comunicação [16]

Empilhamento de *Wafers*: esta técnica empilha *wafers* inteiros, tendo a Tezzaron [1] como uma companhia que trabalha com esse tipo de montagem. Comparada com a técnica *chip*-sobre-*wafer*, a tecnologia utilizada aqui causa erros de alinhamento inferiores a 1μ e resulta numa comunicação mais integrada e de superfície planar maior [22]. A principal diferença entre a técnica de empilhamento de *wafers* e a de *chip*-sobre-*wafer* é o número de *chips* tratados ao mesmo tempo e o baixo nível de erros de alinhamento.

Empilhamento de Transistores: atualmente, ambas as técnicas de montagem (*chip*-sobre-*wafer* e empilhamento de *wafers*) são usadas comercialmente. A técnica de empilhamento de transistores integra camadas ativas fabricadas em um mesmo chip descartando qualquer tipo de alinhamento. Em outras palavras, a deposição e remoção de materiais formam os *chips*. Essa é a técnica ideal devido ao preciso alinhamento das vias e ao forte acoplamento que a caracterizam. Entretanto, as altas temperaturas resultantes desse procedimento inviabilizam sua aplicação. A tecnologia de fabricação de transistores de alta performance demanda temperaturas que destroem o cobre e o alumínio utilizado para as camadas de metal. Porém, é uma técnica muito promissora e com um grande campo para pesquisa [22].

2.2 Estratégias de Montagem de Circuitos VLSI 3D

Existem três estratégias de integração: *face-to-back*, *face-to-face* e *back-to-back*, como mostra a Figura 2. Na estratégia *face-to-back*, os *chips* são empilhados todos numa mesma

orientação. No topo da última camada de metal do *chip* 1 - como no exemplo da Figura 2 (a) - existe uma camada de isolante para que depois seja posicionado o substrato do chip 2. Para a fabricação das vias-3D, deve-se abrir uma fenda no isolante e no substrato. Posteriormente, essas fendas são preenchidas com metal. A via deve conectar a última camada de metal do *chip* 1 e a primeira camada de metal do *chip* 2. Na estratégia *face-to-face*, os *chips* são empilhados um de frente para o outro - como na Figura 2 (b). A última camada de metal do *chip* 1 é colocada com a frente voltada para a última camada de metal do *chip* 2 (separado somente por um isolante). A via que conecta o *chip* 1 com o *chip* 2 ocupa uma área menor do que a resultante da estratégia *face-to-back*. Por outro lado, na estratégia *face-to-face*, apenas duas camadas ativas podem ser empilhadas, pois as seguintes tendem a ser integradas conforme as estratégias *back-to-back* ou *back-to-face*. A fabricação que segue a estratégia *face-to-back* é mais fácil, pois são necessárias poucas mudanças nos processos tradicionais. A Figura 2 (c) mostra duas camadas ativas conectadas, ilustrando a estratégia de integração *back-to-back*. Nesse caso, para que a comunicação entre os chips seja realizada, ambas as camadas ativas devem ser perfuradas, o que cria o espaço para a via-3D. Percebe-se, que essa via é bastante grande se comparada a um transistor, o que é ruim devido às características elétricas incorporadas pelos tipos de materiais. As próximas seções destacam, com maiores detalhes, os tipos de vias e sua influência no projeto de circuitos 3D. Contudo, a cada integração que utiliza a estratégia *back-to-back*, duas novas integrações *face-to-face* são possíveis.

Figura 2. Estratégias de integração: (a) *face-to-back*; (b) *face-to-face*; (c) *back-to-back*

As três estratégias citadas podem ser combinadas com tecnologias SOI *(Silicon on*

Insulator) ou *bulk*. A Figura 3 mostra a combinação de tecnologias e estratégias de integração. No uso da estratégia *face-to-back* com tecnologia *Bulk*, percebe-se que o custo de perfuração é maior se comparado com a tecnologia SOI. Consequentemente, o tamanho da via-3D também aumenta. Atualmente, a tecnologia SOI é a mais utilizada, mas é difícil definir com precisão a proporção de sua utilização, em razão dos diferentes tipos de processos empregados e dos benefícios específicos de cada tecnologia. São referidas como *"mixed integration"* as estratégias, comumente utilizadas, que combinam técnicas *back-to-back, face-to-face e face-to-back*. A Tezzaron, por exemplo, utiliza-a para alcançar o maior número de integrações face-to-face, nem que isso custe integrações *back-to-back*. A Figura 3 mostra as diferentes tecnologias e estratégicas empregadas nesse tipo de integração.

Figura 3. Estratégias de integração *face-to-back, face-to-face e mixed integration* com tecnologias *Bulk* e SOI.

3 Trabalhos Relacionados

A ideia de particionar um bloco de lógica aleatória em duas ou mais *tiers* de um circuito 3D já foi explorada em outros estudos [2][4][6]. Nessa abordagem, o estágio de posicionamento particiona e distribui as células em *tiers* separadas. Teórica [7] e empiricamente empiricamente [2][3][24] mostra-se que circuitos 3D podem reduzir o tamanho dos fios. Goplen e Sapatnekar apresentam em [4] um posicionador de células dirigido a forças voltado aos problemas térmicos dos circuitos 3D. Questões térmicas são uma das grandes preocupações dessa área. Nesse trabalho utiliza-se o algoritmo baseado em bipartições recursivas encontrado na ferramenta hMetis [14]. Sua função é dividir as células entre os vários *tiers* enquanto as vias 3D são minimizadas.

Como nos trabalhos de Sapatnekar [3][13], a abordagem de Obenaus [5] também apresenta um método de posicionamento direcionado a forças. Ele inicia com uma solução aleatória e melhora os resultados com base nas forças de atração. Ambos os trabalhos não

mencionam como foi realizada a migração dos pinos de I/O de um arranjo original 2D para um arranjo 3D. Para seus experimentos foram utilizados benchmarks padrão 2D. Assim, possivelmente, os pinos de I/O foram ignorados.

O trabalho de Deng [6] apresenta um posicionador 3D baseado na abordagem da ferramenta Capo. Seu fluxo de posicionamento é muito simples: a *netlist* é particionada em várias *tiers*. Após, cada *tier* é posicionada separadamente com informações das *tiers* já posicionadas. Essa metodologia tem como objetivo reduzir o tamanho das conexões 3D através das informações de *tiers* previamente posicionadas. Entretanto, este método assume que a borda de um bloco aleatório, em circuitos 2D, é delimitada por pinos de I/O. Também é assumido que esses pinos possam ser movidos para qualquer *tier*. Outros algoritmos de posicionamento de células são dirigidos pela localização fixa dos pinos. Os de posicionamento quadráticos [8], por exemplo, amplamente usados em pesquisas acadêmicas [9] e indústria [10], requerem que a posição dos pinos já esteja determinada para que seja computada a solução.

4 Definição do Problema

Antes do posicionamento, uma netlist 2D N_l é composta por um conjunto de células $G = \{g_l, g_2, g_3, ..., g_n\}$, um conjunto de pinos de I/O $P = \{p_l, p_2, p_3, ..., p_m\}$ e um conjunto de redes $N = \{n_l, n_2, n_3, ..., n_o\}$. Um hipergrafo H_g representa a *netlist*, onde $G \bigcup P$ é o conjunto de nodos, e N é o conjunto de hiperarestas. A posição fixa de cada pino de I/O p_i é representada por $X[i]$ e $Y[i]$ $(i \leqslant m)$, e sua orientação, por $O_r(p_i)$ ε {*north, south, east, west*}. A área *A* (altura *H*, largura *W*) a informação de seu canto inferior esquerdo pela coordenada (x_{ini}, y_{ini}).

Usualmente, os pinos de I/O cobrem toda a borda do circuito. A relação de espaços em branco *S* na área do posicionamento é alcançada pela subtração da área total de células (G_a) pela área disponível dentro dos pinos de I/O. A relação de aspecto A_r é computada pela divisão de *W* por *H*.

Se *Z* é o conjunto dos números que representam as *tiers* {*1, 2, ..., z* }, então, o problema pode ser definido por essa diretriz: dado uma *netlist 2D* N_l com pinos de I/O fixos, encontre o conjunto de *tiers* $T = \{t_1, t_2, ..., t_z\}$ e seus correspondentes $A_i, A_{ri}, G_{ai}, W_i, H_i, P_i, S_i, O_{ri}, X_i e Y_i (i \leq z)$ tal que

$$P_1 \cup P_2 \cup ... \cup P_t = P \tag{1}$$

$$\forall (a, b \in Z)(a \neq b \rightarrow P_a \cap P_b = \phi) \tag{2}$$

$$\forall (i \in Z) S_i \approx S \tag{3}$$

$$\forall (i \in Z) \forall (j \in Z) W_i = W_j \wedge H_i = H_j \tag{4}$$

$$\forall (i \in Z) A_{r_i} \approx A_r \tag{5}$$

$$\forall (i \in Z)(\forall a \in P_i(O_{r_i}(a) = O_r(a))) \tag{6}$$

$$\forall (t \in Z)(\forall a, b \in P_t(O_r(a) = O_r(b) \wedge X_i[a] < X_i[b] \rightarrow X[a] < X[b])) \tag{7}$$

$$\forall (t \in Z)(\forall a, b \in P_t(O_r(a) = O_r(b) \wedge Y_i[a] < Y_i[b] \rightarrow Y[a] < Y[b])) \tag{8}$$

Em outras palavras, cada *tier* terá seu próprio conjunto de pinos de I/O (equações 1 e 2), os espaços em branco *(whitespaces)* e a relação de aspecto *(aspect ratio)* preservados na mesma proporção do circuito 2D original (equações 3, 4 e 5), assim como a orientação e a ordem dos pinos (equações 6, 7, e 8).

Método de Migração Baseado no Menor Caminho Lógico

Seja $L_d(p_i, p_j)$ o tamanho do menor caminho em H_g de p_i para p_j (por exemplo, a distância lógica entre p_i e p_j), pode-se descrever o método de particionamento de pinos de I/O da forma detalhada abaixo:

1. Computar $L_d(i, j) \forall i, j \in P$
2. Criar um grafo completo P_g tal que *P* seja o conjunto de nodos e $L_d(i, j)(i, j \in P)$ seja o custo das arestas conectando os nodos *i* e *j*.
3. Executar o particionamento de P_g em $P_1, P_2, ..., P_z$ buscando a minimização do corte *(min-cut)* e o balanceamento de pinos entre as partições.
4. $\forall (i \in Z) G_{a_i} \approx \frac{G_a}{Z}$
5. $\forall (i \in Z) A_i = G_{a_i} \times (1 + S_i)$
6. $\forall (i \in Z) W_i = \sqrt{A_i} \times A_{r_i}; H_i = \frac{\sqrt{A_i}}{A_r}$
7. $\forall (i \in Z) \forall (p \in P_i) X_i[p] = x_{ini} + \frac{(X[p] - x_{ini}) \times W_i}{W}$
8. $\forall (i \in Z) \forall (p \in P_i) Y_i[p] = y_{ini} + \frac{(Y[p] - y_{ini}) \times H_i}{H}$
9. Legalizar pinos de I/O.

Considerando-se que, em um circuito real, os *fanouts* são limitados, uma simples busca BFS terá uma complexidade $O_{(n)}$. Portanto, usando-se um algoritmo de busca simples para computar o custo de um pino p_i para todos $p \in P$, a complexidade será $O_{(mn)}$.

A busca pelo menor caminho em um grafo *(shortest-path)* é um problema bem conhecido na ciência da computação. É utilizada em aplicações reais, tais como mapas rodoviários, jogos e roteamento de linhas telefônicas, de circuitos integrados e de redes, etc. [25][26].

Os valores de L_d são usados para criar um grafo completo P_g conectando todos os pares de pinos de I/O, como mostra a Figura 4. No terceiro passo, usa-se a ferramenta hMetis [14] para particionar os pinos de I/O (Figura 5). Essa ferramenta aceita a inserção de valores (pesos) para as células e arestas. Atribui-se o inverso do custo das arestas como peso. Além disso, é imposto um balanceamento rígido a fim de manter uma quantidade similar de pinos de I/O entre as *tier*.

O quarto passo é realizado por meio da divisão do número total da área de células pelo número de *tiers*. Os passos 5 e 6 computam a área destas, tal que a relação de aspecto *(aspect ratio)* e os espaços em branco *(whitespaces)* do circuito 2D original sejam preservados. Nesse ponto, a nova relação de aspecto ou espaços em branco pode ser usada.

Finalmente, os passos 7 e 8 computam as coordenadas *x e y* dos pinos de I/O para as *tiers* a que são destinados. A orientação original é preservada, de modo que os pinos

Figura 4. Ilustração do menor caminho entre dois pinos de I/O e seus pesos correspondentes no grafo completo.

Figura 5. *Netlist* inicial seguindo o fluxo de particionamento.

originais sejam mapeados para as *tiers* de tamanho menor. Por fim, a legalização (passo 9) é executada para garantir que não existam sobreposições *(overlaps)* entre os pinos de I/O.

Após o término do passo 9, é necessário descobrir qual a melhor sequência de *tiers* que auxilia a minimização de vias. Utilizou-se o algoritmo de *simulated annealing* para otimizar o número total de vias-3D, como mostra a Figura 6. A Figura 7 mostra duas ilustrações. A primeira, localizada à esquerda, representa a migração de um circuito 2D para um 3D. A segunda, à direita, apresenta uma *netlist*, sem definições, transformada em um circuito 3D.

Interpretação e Formato de Arquivos Utilizados no Modelo Proposto

O processo inicia-se com a leitura do circuito *2D* no formato *bookshelf (nodes, net, pl, scl, aux)*, e as informações de número e tamanho de células, quantidade de redes e pinos de I/O são carregadas e organizadas na estrutura de dados da ferramenta.

O formato do arquivo segue a seguinte definição: um hipergrafo *H = (V, E)*, com *V* vértices e *E* hiper-redes, é armazenado em um arquivo texto contendo $|E|$ + *1* linhas, caso não existam pesos nos vértices, e $|E| + |V|$ + *1* linhas caso existam.

A primeira linha pode conter dois ou três números inteiros. O primeiro é o número de hiper-redes ($|E|$), o segundo é o número de vértices ($|V|$), e o terceiro *(f)* informa o tipo do hipergrafo. Dependendo do valor de *f*, o hipergrafo *H* pode ter peso tanto nas hiper-redes quanto nos vértices, ou em ambos. Caso *H* não tenha nenhum peso atribuído, todas as hiper-redes e vértices têm o mesmo peso e, com isso, *f* é omitido.

Depois da primeira linha, as restantes $|E|$ linhas armazenam os vértices contidos em uma hiper-rede (uma linha por hiper-rede). Segundo o fluxo proposto, o primeiro parti-

Figura 6. Fluxo que ilustra os custos das arestas e a execução do *simulated annealing* para minimizar o número de vias-3D.

Figura 7. Migração de circuitos planares para circuitos 3D.

cionamento necessita da informação do peso atribuído a cada uma das arestas. Nesse caso, o arquivo de entrada gerado segue o formato ilustrado na Figura 8 (a), sendo todo o peso atribuído à aresta armazenada sempre na primeira posição da linha.

Após a execução da busca pelo menor caminho lógico entre pares de pinos de I/O, um grafo completo é criado e os pesos das distâncias são adicionados a cada uma das arestas. Na primeira linha, o primeiro número inteiro corresponde à quantidade de hiper-redes existentes, o segundo, ao número de vértices, e o terceiro indica que somente pesos nas arestas serão considerados. As próximas linhas representam as hiper-redes, sendo os números inteiros correspondentes aos vértices do hipergrafo.

Terminado o particionamento dos pinos de I/O, um novo arquivo é gerado contendo duas informações em cada uma das n linhas, sendo $n = |V|$ (nesse caso, somente pinos). A primeira informação é do pino de I/O, e a segunda, a do número da partição. Esse arquivo é convertido como uma nova entrada (com vértices fixos) para que os pinos sejam fixados

Figura 8. (a) Arquivo gerado após o primeiro particionamento, fixando os pinos de I/O; (b) com base na posição dos pinos, as células são particionadas e fixadas nas partições.

antes do particionamento das células.

No arquivo de vértices fixos, *t* é o valor inteiro correspondente ao número da partição. Assim, quando $t \geq 0$, fixa-se o vértice na partição corrente, e quando $t < 0$, os vértices ficam livres para migrar para qualquer partição (esse caso acontece na segunda fase, em que os pinos estão fixos, e as células, livres para o próximo particionamento), como ilustra a Figura 8 (b). Cada linha do arquivo corresponde a um vértice particionado (Figura 9 e Figura 10).

Na etapa de particionamento de células (segundo passo), o critério a ser considerado é o balanceamento entre as partições, o que pode envolver o número de células em cada partição ou considerar a área de cada célula. Como, em circuitos *standard cells*, as células têm tamanhos variados (mesma altura e largura diferentes), considerou-se a área destas em cada partição, como ilustra a Figura 11 (a) e (b). No momento da movimentação de uma célula para outra partição, a área da célula é subtraída da área da partição atual e incrementada na área da partição-destino. Utilizou-se um balanceamento rígido de células para manter as áreas das partições equilibradas.

Figura 9. Hipergrafo contendo pesos nas hiper-redes e seu formato de arquivo correspondente.

Figura 10. Hipergrafo contendo pesos nos vértices e seu formato de arquivo correspondente.

Figura 11. (a) Balanceamento de *tiers* baseado no número de células; (b) balanceamento de *tiers* baseado na área de células contidas nas *tiers*.

5 Resultados Experimentais

O modelo resultou em uma solução algoritmica para o particionamento de pinos de I/O e foi comparado com outros dois modelos que seguem a mesma formulação do problema descrito na Seção 4. O primeiro método é executado por meio do algoritmo hMetis *(state-of-the-art)* e chamado *unlocked pins*. Neste, permite-se que a partição dos pinos seja realizada livremente junto com as células do circuito. O segundo modelo é chamado de *alternate pins*. Trata-se de um particionamento pseudoaleatório que divide os pinos alternadamente em cada uma das *tiers* com o objetivo de preservar o balanceamento inicial dos pinos de I/O. Ambos os modelos são ilustrados na Figura 12 e substituem os passos 1, 2, 3 e 4, fazendo-se necessários apenas os passos 5, 6, 7, 8 e 9.

As tabelas 1, 2 e 3 mostram os resultados de experimentos realizados com duas *tiers* e com o uso dos *benchmarks* ISPD 2004 [15]. A coluna que apresenta a área da *tier* é calculada antes da partição atual de células. A *tier* com maior área é usada como padrão para as demais. A área total é simplesmente *n* vezes a área da maior *tier*. O *wirelenght* total é a soma do *wirelenght* encontrado pelo posicionador em cada *tier* separadamente. O número de vias e pinos de I/O também são relatados nas tabelas. Estas mostram o desvio padrão do número de pinos a fim de avaliar os balanceamentos. Analisando-se as tabelas 1, 2 e 3, pode-se fazer as seguintes observações:

- O número de pinos de I/O é muito bem balanceado com o modelo *alternate pins*, tendo uma média de desvio padrão menor do que 1. A média atingida com o algoritmo proposto por este trabalho é de 5 pinos. Contudo, o método *unlocked pins*

(hMetis) apresenta um desbalanceamento no número de pinos que o invalida completamente (média do desvio padrão de 150 pinos).

- O número de vias encontradas pelos modelos *unlocked pins* e *alternate pins* é sempre pior do que o modelo proposto, mostrando que um particionamento de pinos de I/O simplista pode piorar o algoritmo de minimização de cortes *(min-cut)*.
- O número de vias do modelo proposto é sempre menor do que o dos outros. Este é um resultado importante, pois mostra que este método auxilia o particionamento das células, de modo que se encontre o melhor corte e um balanceamento eficiente entre as *tiers*. O resultado do tamanho dos fios obtido pelo algoritmo proposto é menor, em média, do que o encontrado com o método *alternate pins*. O método *unlocked pins* leva ao melhor *wirelength*. Contudo, não foram computadas as conexões o tamanho das conexões verticais.

Tabela 1. Resultados experimentais encontrados com o uso do modelo proposto, com duas *tiers*.

Benchmarks	IOs	Área 2D	Área *Tier*	Total WL	IOs *Tier* 0	IOs *Tier* 1	σ # IOs	Vias
ibm01	246	2.380,800	1.209,856	2.11E+06	120	126	4	374
ibm02	259	3.064,208	1.547,568	4.50E+06	126	133	5	396
ibm03	283	3.751,968	1.896,128	6.48E+06	138	145	5	1.064
ibm04	287	4.782,848	2.417,664	7.02E+06	147	140	5	735
ibm06	166	4.106,592	2.078,784	7.51E+06	81	85	3	1.059
ibm07	287	7.136,672	3.612,960	1.23E+07	140	147	5	992
ibm08	286	7.403,840	3.799,840	1.07E+07	140	146	4	1.298
ibm09	285	8.617,104	4.328,208	1.41E+07	139	146	5	699
Média	264	5.155,504	2.590,126	8.08E+06	129	134	5	826

Tabela 2. Resultados experimentais encontrados com o uso do modelo unlocked pins, com duas *tiers*.

Benchmarks	IOs	Área 2D	Área *Tier*	Total WL	IOs *Tier* 0	IOs *Tier* 1	σ # IOs	Vias
ibm01	246	2.380,800	1.209,856	2.14E+06	0	246	174	441
ibm02	359	3.064,208	1.578,784	4.39E+06	259	0	183	547
ibm03	283	3.751,968	1.897,280	6.22E+06	283	0	200	1.146
ibm04	287	4.782,848	2.414,592	7.30E+06	287	0	203	738
ibm06	166	4.106,592	2.078,784	7.73E+06	75	91	11	1061
ibm07	287	7.136,672	3.596,112	1.13E+07	0	287	203	994
ibm08	286	7.403,840	3.737,920	1.03E+07	127	159	23	1.324
ibm09	285	8.617,104	4.326,144	1.40E+07	0	285	202	806
Média	264	5.155,504	2.583,684	7.91E+06	129	134	150	882

É importante ressaltar que o *wirelenght* medido é impreciso, pois as conexões entre as *tiers* foram desconsideradas. Uma das vantagens obtidas pelo método proposto é a de melhorar o resultado das conexões verticais (vias-3D).

A Tabela 4 apresenta a média dos resultados do desvio padrão do número de pinos de I/O em cada uma das cinco partições. O algoritmo *alternate pins* apresenta um resultado

Tabela 3. Resultados experimentais encontrados com o uso do modelo alternate pins, com duas *tiers*.

Benchmarks	IOs	Área 2D	Área *Tier*	Total WL	IOs *Tier* 0	IOs *Tier* 1	σ # IOs	Vias
ibm01	246	2.380,80	1.209,86	2.35E+06	123	123	0	428
ibm02	259	3.064,21	1.548,88	4.30E+06	130	129	1	503
ibm03	283	3.751,97	1.877,28	6.13E+06	142	141	1	1.099
ibm04	287	4.782,85	2.414,59	7.54E+06	144	143	1	750
ibm06	166	4.106,59	2.098,78	7.37E+06	83	83	0	1.075
ibm07	287	7.136,67	3.596,11	1.18E+07	144	143	1	1.049
ibm08	286	7.403,84	3.797,92	1.11E+07	143	143	0	1.307
ibm09	285	8.617,10	4.326,14	1.43E+07	143	142	1	780
Média	264	5.155,504	2.583,684	8.10E+06	132	131	0,63	874

Figura 12. Algoritmo *Alternate Pins* e algoritmo *Unlocked Pins* (hMetis).

excelente, obviamente porque a métrica utilizada teve essa finalidade. O método *unlocked pins* teve um desvio padrão enorme; em muitos casos, várias *tiers* ficaram sem nenhum pino de I/O. Esse desbalanceamento invalida o método *unlocked pins*. O algoritmo proposto por este trabalho se aproxima do balanceamento ótimo.

Tabela 4. Desvio padrão do número de pinos de I/O.

Num Tiers	unclocked pins	alternate pins	proposto
Média σI/O 2 tiers	233	0,4	7
Média σI/O 3 tiers	252	0,4	6
Média σI/O 4 tiers	177	0,4	5
Média σI/O 5 tiers	189	0,4	6
Média Total	213	0,4	6

A Tabela 5 considera a relação do número máximo e da área ocupada com o tamanho

das vias em tecnologia SOI e *Bulk*. Analisando pode-se fazer as seguintes considerações:

- As tecnologias baseadas em *Bulk* provocam desperdício de área ativa, a cada inserção de vias-3D. Isso é devido ao tamanho da área das vias (cerca de $50\mu m$). Existem casos em que a área ocupada pelas vias é maior do que a área da *tier*. Assim, uma importante conclusão é a necessidade de reduzir o número de vias-3D em tecnologia CMOS *Bulk*.
- Na tecnologia baseada em SOI, o número de vias-3D alcança cerca de 3% da área do circuito.

Tabela 5. Comparação dos três modelos considerando-se a área das *tiers* em relação ao número máximo de vias-3D, em tecnologias SOI e *Bulk*

# *Tiers*	Métodos	Área *Tier*	Max #Vias	*Bulk* 50μ	*Bulk* 50μ	SOI 5μ	SOI 5μ
2	proposto	3.833.189	1.375	3.437.500	90%	34.375	1%
3	proposto	2.595.794	1.466	3.665.000	141%	36.650	1%
4	proposto	1.934.213	1.567	3.917.500	203%	39.175	2%
5	proposto	1.559.209	1.744	4.360.000	280%	43.600	3%
2	*unclocked pins*	3.844.129	1.496	3.740.000	97%	37.400	1%
3	*unclocked pins*	2.587.478	1.852	4.630.000	179%	46.300	1%
4	*unclocked pins*	1.917.220	1.667	4.167.500	217%	41.675	2%
5	*unclocked pins*	1.553.191	1.936	4.840.000	312%	48.400	3%
2	*alternate pins*	3.835.153	1.520	3.800.000	99%	38.000	1%
3	*alternate pins*	2.583.077	1.978	4.945.000	191%	49.450	2%
4	*alternate pins*	1.930.053	1.991	4.977.500	258%	49.775	3%
5	*alternate pins*	1.555.691	2.284	5.710.000	367%	57.100	3%

6 Conclusões

Este trabalho apresentou um modelo matemático para a transformação de circuitos planares em circuitos VLSI 3D o qual resultou em uma solução algoritmica. O modelo extraiu de um hipergrafo informações para se obter o melhor corte entre as partições e estudou o seu impacto na área do circuito, balanceamento de pinos e tamanho das conexões. Baseou-se na idéia de manter os pinos logicamente próximos em uma mesma *tier*. Além disso, foi desenvolvida uma solução algorítmica para validar o modelo que permitiu mostrar que a distribuição balanceada de pinos melhora o tamanho dos fios e o número de vias 3D se comparado com as outras abordagens. Mantém um bom balanceamento de pinos de I/O entre as partições enquanto minimiza o número de vias através da heurística do menor caminho lógico. De acordo com os resultados experimentais, um particionamento simplista de pinos de I/O pode aumentar o número de vias 3D. Além disso, percebe-se que usando um particionamento regular (células + pinos de I/O juntos) obteve-se um desbalanceamento enorme dos pinos de I/O. Conclui-se, então, que as informações do menor caminho lógico entre os pinos de I/O pode ajudar a heurística de particionamento de células a minimizar o número de vias 3D.

Referências

[1] 3D ICs Industry Sumary at Tezzaron homepage: http://www.tezzaron.com/ technology/3D20IC20Summary.htm. Access on Jan 2012.

[2] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer and P. D. Franzon; et. al. Demystifying 3D ICs: The Pros and Cons of Going Vertical. IEEE Design and Test of Computers - special issue on 3D integration; pp 498-510, Nov.-Dec. 2005.

[3] C. Ababei, Y. Feng, B. Goplen, H. Mogal, T. Zhang, K. Bazargan and S. Sapatnekar. Placement and Routing in 3D Integrated Circuits IEEE Design and Test of Computers - special issue on 3D integration; pp 520-531, Nov.-Dec. 2005.

[4] B. Goplen; Sachin Sapatnekar; Efficient Thermal Placement of Standard Cells in 3D Ics usisng Forced Directed Approach. In: Internation Conference on Computer Aided Design, ICCAD'03, November, San Jose, California, USA, 2003.

[5] S. Obenaus, T. Szymanski. Gravity: Fast Placement for 3D VLSI. ACM Transacions on Design Automation of Electronic Systems, New York, v.8, p.69-79, March 1999.

[6] Y. Deng; W. Maly. Interconnect Characteristics of 2.5-D System Integration Scheme. In: Proc. of the International Symposium on Physical Design, ISPD 2001, New York, NY, USA. Anais. . . ACM Press, 2001. p.171- 175.

[7] K. Banerjee and S. Souri and P. Kapur and K. Saraswat. 3D-ICs: A Novel Chip Design for Improving Deep Submicrometer Interconnect Performance and Systems on-Chip Integration. Proceedings of IEEE, vol 89, issue 5, 2001.

[8] C. J Alpert; T. Chan; D. J. Huang.; I. Markov; K. Yan. Quadratic placement revisited. In: Proc. of the 34th Annual Conference on Design Automation, DAC 1997, New York, NY, USA. Anais. . . ACM Press, 1997. p.752-757.

[9] N. Viswanathan; C.C.-N Chu. FastPlace: Efficient Analytical Placement Using Cell Shifting, Iterative Local Refinement,and a Hybrid Net Model. IEEE Transactions on CAD, Volume 24, Issue 5, pp 722-733, May 2005.

[10] P. Villarrubia, CPLACE: A standard cell placement program. IBM Tech. Dis. Bull.,vol32 no. 10A, pp. 341-342, Mar. 1990.

[11] P. Benkart, A. Heittmann, H. Huebner, U. Ramacher, A. Kaiser, A. Munding, M. Bschorr, H-J Pfleiderer, E. Kohn. 3D Chip Stack Technology Using Through-Chip Interconnects. IEEE Design and Test of Computers - special issue on 3D integration; pp 512-517, Nov.-Dec. 2005.

[12] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. Multilevel hypergraph partitioning: Application in VLSI domain. In Proceedings of 34th Annual Conference on. Design Automation, DAC 1997, pages 526-529, 1997.

[13] S. Spatnekar and K. Nowka; New Dimensions in 3D Integration; In: IEEE Design & Test of Computers; - special issue on 3D integration; pp 496-497, Nov.-Dec. 2005.

[14] Hypergraph and Circuit Partitioning at hMetis Home Page, http://glaros.dtc.umn.edu/gkhome/views/metis/hmetis/. Access on Mar 2012.

[15] ISPD04 - IBM Standard Cell Benchmarks with Pads. http://www.public.iastate.edu/nataraj/ISPD04Bench.htmlBenchmarkDescription. Access on Mar 2012.

[16] E. Beyne; 3D Interconnection and Packaging; 2nd Electronics System-Integration Technology Conference, Londres, 2006.

[17] R. Patti. Three-dimensional integrated circuits and the future of system-on-chip designs. Proceedings of IEEE, [S.l.], v.94, p.1214-1224, 2006.

[18] Zycube Technology: Homepage. Disponível via WWW em ¡ http://www.zycube.com/e/index.html¿. Acesso em Janeiro de 2012.

[19] Ziptronix Technology: Homepage. Disponível via WWW em ¡http://www.ziptronix.com/¿. Acesso em Janeiro de 2012.

[20] Cubic Wafer Technology: Homepage. Disponível via WWW em ¡ http://www. xanoptix.com¿. Acesso em Fevereiro de 2012.

[21] V. Pavlidis; E. G. Friedman; Interconnect-Based Design Methodologies for Three-Dimensional Integrated Circuits, In: Proceedings of the IEEE, Volume: 97, Issue: 1, On page(s): 123-140, 2009.

[22] S. Gupta; M. Hilbert; S. Hong; R. Patti, R. Techniques for Producing 3D ICs with High-Density Interconnect. Available at: ¡http://www.tezzaron.com/¿. Access on: Novembro. 2011.

[23] S. Das; A. Fan; K. Chen; C. S. Tan; N. Checka; R. Reif. Technology, performance, and computer-aided design of three-dimensional integrated circuits. In: ISPD '04: PROCEEDINGS OF THE 2004 INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN, 2004, New York, NY, USA. Anais... ACM Press, 2004. p.108-115.

[24] S. Sawicki; Particionamento de Pinos e Pads de I/O em Circuitos VLSI 3D. Tese de Doutorado, PPGC-UFRGS, 2009.

[25] E. W. Dijkstra; A note on two problems in connexion with graphs. In: Numerische Mathematik. 1 (1959), S. 269-271

[26] N. A. Sherwani; Algorithms for VLSI Physical Design Automation. Norwell, MA, USA: Kluwer Academic Publishers, 1998.