Implementação de Um Método de Posicionamento de Transistores Baseado em Satisfatibilidade Booleana na Ferramenta ASTRAN

Andrei A. O. Bubolz[1]; Gustavo H. Smaniotto[1], Leomar S. da Rosa Jr.[1], Maicon S. Cardoso[1], Felipe de S. Marques[1]

[1]Universidade Federal de Pelotas – {aaobubolz, ghsmaniotto, leomarjr, mscardoso, felipem} @inf.ufpel.edu.br

1. INTRODUÇÃO

Otimizações nas etapas de síntese lógica e física são fatores cruciais para o aprimoramento de circuitos VLSI. Recentemente, pesquisas apontaram que o *design* de portas complexas em tempo de execução fornece melhorias em diferentes aspectos do projeto digital, tais como área total, potência dissipada e atraso de sinal.

Um dos passos mais importantes para a obtenção dessas otimizações, em termos de síntese física de células, é o de posicionamento de transistores (REIS, 2011). Através deste processo, é possível obter ganhos em diferentes atributos do leiaute final (área, comprimento de fio e número de contatos, por exemplo).

O primeiro método de posicionamento foi proposto em (UEHARA, 1981), pela qual a minimização na largura total em redes duais é obtida através de uma abordagem baseada em grafos, fazendo a busca da solução com menor número de quebras de difusão. Além disso, fora proposto o estilo de leiaute 1D, no qual os transistores são colocados exclusivamente em uma única direção ao longo de duas linhas de difusão P/N. Usando esse formato, diferentes algoritmos foram propostos para resolverem o problema de posicionamento de transistores.

Embora obtenham resultados otimizados, a maioria dessas metodologias não garantem o posicionamento dos transistores na menor largura possível, ou seja, a solução ótima em termos de área. Em (IIZUKA, 2004) e (IIZUKA, 2005), foram introduzidos métodos de posicionamento ótimos para redes de transistores duais e não-duais. Essa técnica consiste em descrever o posicionamento através de um conjunto de expressões Booleanas (cláusulas) e determinar se há pelo menos uma atribuição de variáveis que satisfaça todas cláusulas.

ASTRAN (ZIESEMER, 2007), ferramenta de EDA estado da arte para design de portas complexas sob demanda, usa uma técnica baseada em *Threshold Accepting* para determinar uma solução que maximize o compartilhamento de difusões e minimize o comprimento de interconexões. Essa metodologia baseia-se em uma busca local probabilística que inicia por um posicionamento possível aleatório dentro de um espectro de soluções, e então, explora sua vizinhança em busca de otimizações. No entanto, esse método eventualmente fica preso em mínimos locais, não alcançando a solução ótima, e consequentemente, o melhor posicionamento.

O método de posicionamento proposto neste trabalho visa evitar este problema usando a abordagem de Satisfatibilidade Booleana, originalmente introduzida por (IIZUKA, 2004). Além disso, o método implementado é capaz de tratar redes de transistores CMOS com qualquer tipo de estrutura, garantindo o menor número de quebras de difusão e a solução ótima em termos de área em tempo computacional aceitável.

2. METODOLOGIA

A entrada do método proposto é uma descrição de uma porta complexa em nível de transistores, ou seja, um arquivo em formato *spice* contendo os componentes *pull-up* e *pull-down* da rede lógica.

No estilo de leiaute 1D adotado (Fig. 1), os transistores são divididos em duas linhas contendo os componentes PMOS e NMOS. Cada transistor é representado por um conjunto de variáveis, as quais representam todas possibilidades de posicionamento na célula, como mostrado na Tabela 1. O tamanho inicial desse conjunto depende do número de transistores inicial, computado a partir da entrada.

Para gerar o conjunto de cláusulas SAT, cinco passos são executados. O primeiro é um passo de simplificação, enquanto o restante implementa as restrições de posicionamento. No processo, variáveis auxiliares são criadas com objetivo de representar resultados intermediários das cláusulas SAT, sem nenhum significado físico para o posicionamento. A seguir estão descritos alguns conceitos necessários para o entendimento do trabalho, bem como os principais estágios do algoritmo implementado.

Transistor variável: é um componente usado dentro do problema para representar possíveis *gaps.* Ele funciona como um componente polivalente, podendo assumir valores dinâmicos em seus terminais, e assim, ser colocado em qualquer posição. Esse componente pode aparecer em duas situações distintas: (1) no caso de o número inicial de transistores diferir, transistores variáveis são adicionados para completarem a igualdade; (2) após cada iteração em que não é encontrado um posicionamento satisfatório, sendo, então, adicionado um transistor variável em cada linha.

Redução de variáveis: Esse passo é executado apenas na primeira iteração, onde pelo menos uma linha não terá transistores variáveis. O objetivo desse procedimento é processar as impossibilidades de alocação de acordo com os terminais dos transistores, podendo, dessa forma, reduzir o número de variáveis e, consequentemente, a exigência computacional do problema.

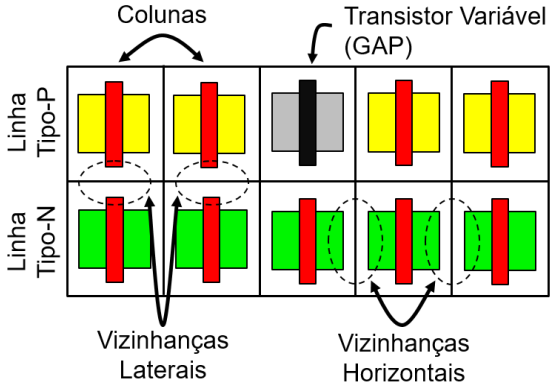

Fig. 1. Estilo de leiaute adotado (1D).

TABELA I.
Conjunto de Variáveis usada na formulação.

Nome	Condição que torna a variável verdadeira	Tamanho
$T_p(i,c,f)$	PMOS *i* posicionado na coluna *c* com orientação *f*	$2P^2$
$T_n(j,c,f)$	NMOS *j* posicionado na coluna *c* com orientação *f*	$2N^2$

1ª Restrição: alocação de transistores: cada componente precisa ser alocado exclusivamente em uma coluna. As cláusulas que implementam essa restrição são apresentadas na fórmula abaixo.

$$\bigwedge_{i=1}^{P}\left\{\bigwedge_{f\in(f,nf)}\left\{\bigvee_{c=0}^{P-1}T_p(i,c,f)\ \wedge\ \left\{\bigwedge_{j=0}^{P-2}\bigwedge_{k=j+1}^{P-1}\left[\neg T_p(i,j,f)\ \vee\ \neg T_p(i,k,f)\right]\right\}\right\}\right\}$$

2ª Restrição: colunas vazias: uma coluna não pode estar vazia e nem conter sobreposição de transistores. As cláusulas que implementam essa restrição são apresentadas na fórmula abaixo.

$$\bigwedge_{c=1}^{P}\left\{\bigwedge_{f\in(f,nf)}\left\{\bigvee_{i=1}^{P}T_p(i,c,f)\right\}\right\}$$

3ª Restrição: vizinhanças horizontais: todos os transistores devem possuir vizinhos com terminais equivalentes com os seus, com exceção daqueles posicionados nas extremidades, pelos os quais necessitam de apenas um valor. Primeiramente, todos os possíveis pares são organizados em uma lista de acordo com seus valores de dreno e fonte. Considerando essa lista, é necessário alocar exclusivamente um desses pares em cada vizinhança. As cláusulas que implementam essa restrição são apresentadas na fórmula abaixo.

$$\bigwedge_{N=1}^{P-1}\left\{\bigwedge_{f\in(f,nf)}\left\{\bigvee_{i=0}^{m-1}U_i(T_x,T_y,N)\ \wedge\ \left\{\bigwedge_{j=0}^{m-2}\bigwedge_{k=j+1}^{m-1}\left[\neg U_i(T_x,T_y,N)\ \vee\ \neg U_i(T_x,T_y,N)\right]\right\}\right\}\right\}$$

4ª Restrição: vizinhanças verticais: todos transistores do tipo P e N colocados na mesma coluna necessitam possuir o mesmo valor de porta. O conjunto de possibilidades é feito da mesma forma que o anterior. As cláusulas que implementam essa restrição são apresentadas na fórmula abaixo.

$$\bigwedge_{C=1}^{P}\left\{\bigwedge_{f\in(f,nf)}\left\{\bigvee_{i=0}^{m-1}U_i(T_{px},T_{ny},C)\ \wedge\ \left\{\bigwedge_{j=0}^{m-2}\bigwedge_{k=j+1}^{m-1}\left[\neg U_i(T_{px},T_{ny},C)\ \vee\ \neg U_i(T_{px},T_{py},C)\right]\right\}\right\}\right\}$$

Após processar todos os passos apresentados acima, o arquivo contendo as cláusulas é escrito a fim de ser computado por um resolvedor SAT. O resolvedor retorna, em caso de satisfatibilidade, a primeira atribuição das variáveis que tornam todas as expressões verdadeiras, tendo assim, um posicionamento de transistores na menor largura possível. No caso de não haver uma atribuição que satisfaça as cláusulas, são adicionados 2 transistores variáveis (um em cada linha) e o processo é reiniciado.

3. RESULTADOS E DISCUSSÃO

Análises nos leiautes gerados nos permitiram avaliar a solução proposta, comparando-a com a solução presente na ferramenta ASTRAN atualmente.

Através dessas análises torna-se possível caracterizar importantes aspectos geométricos, tais como área, comprimento de fio e número de contatos.

Os experimentos foram feitos utilizando um *benchmark* contendo 53 portas complexas criadas a mão (LOGICS, 2012). Os resultados obtidos são

apresentados na Fig. 2, e representam a porcentagem de otimização comparando o método proposto e o método atualmente presente na ferramenta ASTRAN.

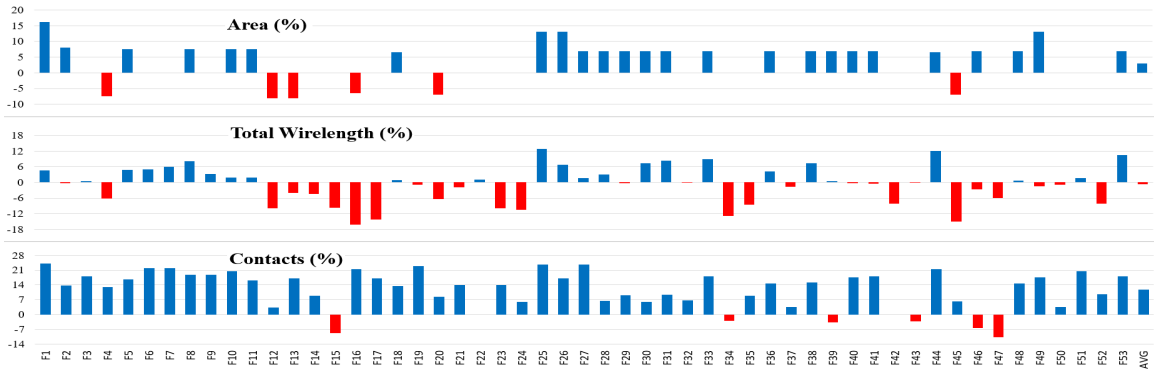

Fig. 2. Comparação entre o método proposto e o método atual.

O método proposto nesse trabalho obteve, em média, leiautes com 3,02% menos área, 0,82% mais comprimento de fio e 12,05% menos contatos.

Apesar de garantir o melhor posicionamento, em alguns casos houve perda em área em relação ao método atual. Isso se dá ao fato do nosso algoritmo não considerar os processos de roteamento e compactação, etapas seguintes ao processo de posicionamento no fluxo de síntese do ASTRAN.

4. CONCLUSÕES

Nesse trabalho foi proposto um método de posicionamento de transistores aplicando Satisfatibilidade Booleana para a ferramenta ASTRAN, que atualmente usa um método baseado em busca local probabilística. Com esse novo método, torna-se possível obter invariavelmente o melhor posicionamento de transistores possível, melhorando o leiaute final gerado pela ferramenta, estado da arte na tecnologia CMOS de 65nm.

5. REFERÊNCIAS BIBLIOGRÁFICAS

REIS, A. Design Automation of Transistor Networks, a new Challenge. **IEEE International Symposium on Circuits and Systems (ISCAS)**, p.2485–2488, 2011.

UEHARA, T. et. al. Optimal Layout of CMOS Functional Arrays. **IEEE Transactions on Computers**, p.305–312, 1979.

T. IIZUKA et. al. High speed layout synthesis for minimum-width CMOS logic cells via Boolean satisfiability**. ASP-DAC '04: Proceedings of the 2004 conference on Asia South Pacific design automation**, p.149–154, 2004.

T. IIZUKA, et al. Exact Minimum-Width Transistor Placement for Dual and Nondual CMOS cells. **IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences**, p.3485–3491, 2005.

A. ZIESEMER et. al. Transistor level automatic layout generator for noncomplementary CMOS cells. **IFIP International Conference on VLSI - SoC 2007**, p.116–121, 2007.

LOGICS. 53 NSP Handmade Networks. 2012. **Logic Circuit Synthesis Labs,** Universidade Federal do Rio Grande do Sul. Acesso em: 05 de setembro de 2018, DisponÍvel em http://www.inf.ufrgs.br/logics/docman/53 NSP Catalog.pdf.