ALLENDE: UMA LINGUAGEM DE ESPECIFICAÇAO DE LAYOUTS DE CIRCUITOS INTEGRADOS

José Monteiro da Mata*

Sumário

ALLENDE é uma linguagem simples e poderosa para a especificação de layouts de circuitos integrados. Em ALLENDE o layout é descrito hierarquicamente como uma composição de células; posições ou tamanhos absolutos nunca são especificados. A descrição do layout é traduzida para equações lineares, que expressam regras de desenho e posições relativas dos elementos do layout. A solução dessas equações determina o layout absoluto, que garantidamente não contem violações das regras de desenho.

ALLENDE consiste de 5 subrotinas que podem ser chamadas de um programa em Pascal ou C, e foi implementado para a tecnologia nMOS.

Abstract

ALLENDE is a simple and powerful procedural language for VLSI layout. In ALLENDE the layout is described hierarchically as a composition of cells; absolute sizes or positions are never specified. The layout description is translated into linear constraints, which express design rules and relative position of the layout elements. By solving these constraints we obtain the absolute layout, which is guaranteed to be free of design rule violations.

ALLENDE consists of five procedures to be called from a Pascal or C program, and has been implemented for the nMOS technology.

* Engenheiro eletricista (UFMG, 1974), Ph.D. em ciência da computação (Princeton, 1984); CAD, software básico; Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Caixa Postal 702, Belo Horizonte, MG 30000

Uma revisão das técnicas de projeto de circuito integrados torna-se necessária, devido ao custo associado ao projeto de circuitos complexos, à necessidade de tornar a tarefa de projeto de circuitos acessível a um maior número de pessoas, e à necessidade de ferramentas mais poderosas para lidar com modificações no projeto. São necessários métodos que aumentem a produtividade do projetista.

O desenho do layout é uma etapa crítica no projeto de circuitos integrados, porque ela envolve ferramentas dispendiosas e uma grande quantidade de intervenção humana, e também devido a seus efeitos nos custos de produção. Um dos objetivos das ferramentas de layout é produzir layouts corretos, mas ainda hoje há a proliferação de programas verificadores de layout, como programas verificadores de regras de desenho.

Neste trabalho nos concentramos no problema da produção de layouts. Temos dois objetivos principais em mente:

- possuir uma ferramenta **poderosa** que permita ao projetista obter **facilmente** um layout **correto** para seu projeto;
- possuir um componente que possa ser **expandido** e **integrado** com outros componentes de um sistema de projeto associado a uma **metodologia estruturada** de projetos.

Nossa abordagem para o problema de layouts de circuitos integrados é usar uma **linguagem** para descrever hierarquicamente a estrutura do circuito e a topologia do layout, e usar **equações lineares** para representar internamente o layout.

Nosso sistema de layout, ALLENDE[5], sucessor de ALI[3][7], possui características que o diferenciam das ferramentas de layout existentes. O uso de uma linguagem "procedural" e a representação do layout através de equações distingue ALLENDE de todos os editores gráficos de layout e da maioria das linguagens para descrição de layouts. A diferença com outras linguagens "procedurais" que usam equações para representar layouts consiste na forma de descrição do layout, no tipo de equações utilizadas, e também na forma de implementação do sistema. O resultado é a capacidade, flexibilidade e eficiência alcançados por ALLENDE.

2.1. Idéias básicas

Nossa abordagem para o problema de obtenção de layouts de circuitos integrados é através do uso de uma linguagem para descrição da estrutura do layout. A partir da representação textual do layout, equações lineares são geradas, resolvidas, e então o layout físico é obtido. As características da linguagem dependem, obviamente, da classe de objetos manipulados e do tipo de manipulação.

Em nosso sistema, o único objeto que temos são células. Uma célula corresponde a um retângulo com estrutura interna e terminais de comunicação.

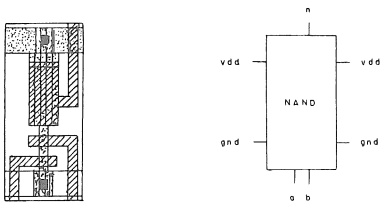


Fig. 1 - Uma célula nand

A composição de duas células é feita especificando sua posição relativa (esquerda, direita, acima, abaixo), e o resultado é outra célula. Uma célula individual pode também sofrer rotação ou espelhamento.

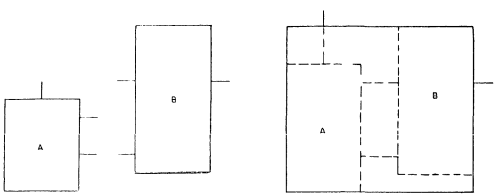
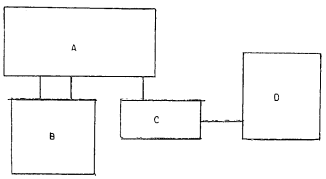


Fig. 2 - Composição de células

Na composição de células não é necessário se preocupar com o "alinhamento" dos fios de comunicação entre células; as condições para alinhamento são expressas nas equações geradas, e então o tamanho de uma célula depende do contexto no qual ela é instanciada.

A primeira idéia básica é descrever o layout hierarquicamente como a composição de células. No último nível da hierarquia (folhas da árvore) estarão **células do sistema** (contatos, transistores, etc.) ou células **rígidas** (partes do layout previamente definidos).



(A above (B left C)) left (rotated90 D)

Fig. 3 - Descrição estruturada do layout

Ao usar uma célula do sistema devemos especificar os fios em cada lado da célula. Quando é feita a composição de duas células, os fios a serem conectados, e os fios de comunicação da célula resultante, são determinados por contexto.

A segunda idéia básica é usar uma linguagem intermediária para descrever a estrutura do layout. Esta estrutura representa a posição relativa das células, em uma forma estruturada como na figura 3. O programa do usuário gera esta forma intermediária, a partir da qual equações lineares são extraídas, e então obtido o layout final. O objetivo é separar aspectos da linguagem do usuário (representação e processamento) de aspectos do layout (geração de equações e produção do layout); a forma intermediária lida somente com a estrutura do layout, enquanto a linguagem do usuário pode ter todo o poderio de uma linguagem "procedural".

Nosso sistema de layout é baseado nestas idéias. Existe uma linguagem do usuário, ALLENDE, que é nada mais do que Pascal ou C com a adição de algumas subrotinas e funções. A saída do programa do usuário é a estrutura do layout em linguagem intermediária (PIF), da qual são extraídas as equações lineares, que são resolvidas, produzindo o layout absoluto em CIF (Caltech Intermediate Form).

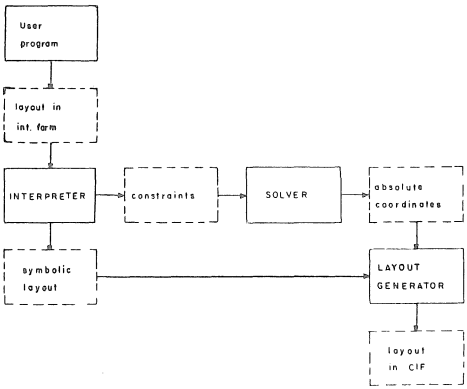


Fig. 4 - Processo de geração do layout

Erros podem ocorrer durante compilação do programa do usuário, execução, interpretação do programa PIF, e solução do sistema de equações. Erros de compilação e execução são os usuais, detetados pelo compilador Pascal ou C ou durante execução. No caso de erro durante interpretação da forma intermediária que descreve o layout, o interpretador PIF identifica a célula onde ocorreu o erro. O único erro possível durante o processo de solução das equações é uma célula rígida não cabendo no contexto onde ela é instanciada; nesse caso o programa que resolve as equações aponta a célula que causou o erro.

2.2. A linguagem ALLENDE

ALLENDE (A Layout Language Effective for nMOS Design) é um conjunto de subrotinas e funções que podem ser chamadas de um programa em Pascal ou C, permitindo ao usuário descrever o layout de um circuito VLSI. Basicamente, o usuário descreve células e seu posicionamento relativo. A hierarquização de células aparece naturalmente através do uso de subrotinas para descrever células.

O usuário pode fazer uso de todo o poderio do Pascal ou C. As subrotinas e funções para descrever o layout permitem uma enorme quantidade de parametrização, o que permite obter layouts completamente diferentes somente alterando um parâmetro no programa.

A saída do programa do usuário é o layout na forma intermediária (PIF), a partir da qual equações lineares são obtidas, resolvidas, e produzido o layout absoluto em CIF. O layout obtido garantidamente não contém violações das regras de desenho.

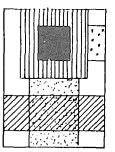
As seguintes subrotinas permitem ao usuário descrever um layout:

```
syscell(kind,wire1,wire2,wire3,wire4,ratio)
extcell(filename)
place(operator)
begincell(cellname)
endcell(wirenames)
```

syscell especifica uma célula do systema. **extcell** especifica uma célula externa. **begincell** e **endcell** são usadas para delimitar uma célula composta. **place** especifica o operador a ser aplicado para uma composição de células.

Como a ação destas subrotinas é gerar algum código intermediário a ser interpretado mais tarde, no programa do usuário podem existir comandos em Pascal ou C misturados com chamadas a estas subrotinas. O usuário pode também definir novas subrotinas em termos destas subrotinas básicas.

A figura 5 mostra um programa ALLENDE, em C, com o layout correspondente. Neste programa o uso da variável "clock" representa um exemplo de parametrização.



```
#include "/va/allende/usr/def.h"
main()
{
  wiretype clock;
  clock = poly(2);
  begincell("example");
    syscell(CONTACT,nowire,nowire,diff(2),metal(3),0);
    place(ABOVE);
    syscell(CROSSING,clock,metal(3),clock,metal(3),0);
  endcell("  ");
}
```

Fig. 5 - Um programa ALLENDE

A subrotina syscell permite a especificação de uma célula do sistema. CONTACT, TRANSISTOR e CROSSING são algumas das células do sistema. O único lugar onde o usuário deve especificar fios (wires) é nas células do sistema, onde devem ser especificados os fios nos lados esquerdo, superior, direito e inferior da célula. As funções metal(width), poly(width), diff(width), e a função mais geral wire(layer,width), permitem a especificação de um fio. Aqui é onde pode ser feita muita parametrização. "layer" e "width" podem ser parametrizados; além disso, um "wire" pode ser uma variável. E também possível ter mais de um "wire" em cada lado de uma célula, permitindo superposição de fios ou células mais complexas.

Os operadores a serem aplicados às células podem ser: LEFT, RIGHT, ABOVE, BELOW, ROTATED0, ROTATED90, ROTATED180, ROTATED270, FLIPPED0, FLIPPED90, FLIPPED45, FLIPPED135.

A subrotina extcell especifica um arquivo contendo uma célula externa. A célula externa pode estar em forma intermediária, em cujo caso a chamamos "flexível", ou pode estar em CIF, quando então a chamamos "rígida". Um tipo especial de células externas são "pads", que são células rígidas. Existe uma biblioteca de "pads".

As subrotinas begincell e endcell são usadas para delimitar uma célula composta. O parâmetro de begincell é uma sequência de caracteres contendo o nome da célula; o nome pode conter somente brancos, caso não se queira atribuir nome à célula. O nome é usado para localizar erros.

O parâmetro de endcell é uma sequência de caracteres contendo os nomes dos fios que constituem a interface da célula. Caso esses nomes sejam especificados, eles aparecerão no código CIF, e são úteis para simulação.

A figura 6 mostra um programa que descreve uma árvore binária de células "nand" e o layout correspondente. Assume-se que a célula flexível "nand", como mostrada na figura 1, foi gerada anteriormente.

```
program binarytree(output);
const
#include "/va/allende/usr/const.h"
type
#include "/va/allende/usr/type.h"
#include "/va/allende/usr/proc.h"
procedure root;

begin
  extcell('nand.pif');
end;

procedure btree(n: integer);
begin
  begincell('btree');
    if n = 1 then root
    else begin
      root;
      place(ABOVE);
      begincell('  ');
        btree(n-1);
        place(LEFT);
        btree(n-1);
      endcell('  ');
    end;
  endcell('  ');
end;

begin
  btree(3);
end.
```

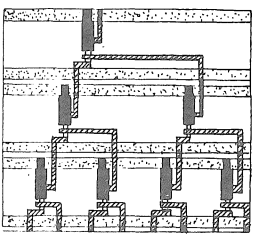


Fig. 6 - Programa para uma árvore binária

2.3. A forma intermediária PIF

PIF consiste em uma forma compacta de representar a estrutura do layout, como na figura 3. Os objetos no layout são células, e os operadores especificam posição ou orientação. Nossa representação do layout é exatamente como uma expressão aritmética; operandos são

células, operadores binários especificam posição relativa (esquerda, direita, acima, abaixo) e operadores unários especificam orientação (rotação ou espelhamento). A precedência de operadores é a usual, e parênteses podem ser usados para mudar precedência.

Um layout em PIF é uma combinação estruturada de células, ao passo que um layout em CIF é uma combinação de retângulos e outros elementos em qualquer ordem. O resultado da interpretação de um programa PIF é um conjunto de equações lineares, o layout em forma simbólica (somente coordenadas relativas) e o circuito (a nível de chaves) para simulação.

A figura 7 mostra um programa PIF, gerado pelo programa ALLENDE da figura 5. O código "C [. . d2 m3]" representa uma célula que é o contato de dois fios: um em "diffusion" com largura 2 lambda vindo da direita e outro em metal com largura 3 lambda vindo do lado inferior; os dois "."s indicam nenhum fio nos lados esquerdo e superior. "A" significa "above", "+" significa "crossing", os parênteses delimitam uma célula, e "$example" especifica o nome "example" para a célula.

```
$example
(
C [ . . d2 m3 ]
A
+ [ p2 m3 p2 m3 ]
)
```

Fig. 7 - Um programa PIF

Em PIF, a construção do layout é como avaliação de expressão. Se usarmos uma gramática para descrever a linguagem PIF, a construção do layout pode ser feita em paralelo ao "parsing", de uma forma "bottom up". Em realidade, isto é similar à forma de concepção e implementação do sistema de formatação de textos matemáticos EQN[2]. Em EQN equações são vistas como um conjunto de "caixas" combinadas de diversas formas.

2.4. Layouts através de equações lineares

Como mostrado na figura 4, em nosso sistema de layout existem diferentes representações para o layout: a representação do usuário (em ALLENDE), a forma intermediária (em PIF), a forma simbólica (em termos de coordenadas relativas) e a forma absoluta (em CIF).

Nossa representação simbólica do layout é em termos das coordenadas relativas dos elementos do layout; a relação entre estas coordenadas é expressa por um conjunto de equações lineares. As variáveis nessas equações são as coordenadas X e Y dos objetos no layout. As equações descrevem a interação entre os objetos, e podem vir das regras de desenho, conectividade e hierarquia na descrição do layout. Resolvendo o sistema de equações e substituindo os valores obtidos para as coordenadas no layout simbólico obtemos o layout absoluto.

O conjunto de equações é resolvido de forma a minimizar a área total. Devido ao grande número de elementos no layout, a forma das equações deve ser tão simples quanto possível, para reduzir a complexidade do algoritmo de solução. Assume-se que equações em X e Y são desacopladas: nenhuma equação envolve ambas as coordenadas X e Y, e equações envolvendo X e Y são independentes. Não se permite relacionar equações através do operador or, por exemplo. Com o desacoplamento das equações em X e Y o problema de compactação é equivalente a resolver dois sistemas de equações independentes.

O layout completo é representado usando equações da forma:

$$x_i = x_j$$
$$x_i - x_j > d \qquad (d > \emptyset, \text{ inteiro})$$
$$x_i - x_j = e \qquad (e > \emptyset, \text{ inteiro})$$

Temos um algoritmo eficiente para resolver tais equações, descrito em [4]. O algoritmo é baseado na classificação topológica.

Cada equação $x_i - x_j = e$ corresponde a uma célula rígida. O usuário pode controlar o número de equações construindo o layout em diversas etapas: obtendo células rígidas e usando-as no próximo nível da hierarquia de células. Se não existem equações da forma $x_i - x_j = e$ no conjunto de equações geradas, sempre existe uma solução para as equações, visto que as equações são geradas de forma a não criar "ciclos". A única situação em que não há solução para o conjunto de equações ocorre quando uma célula rígida não cabe no contexto em que ela é instanciada. Por exemplo, alguma condição pode forçar uma separação maior entre dois fios na interface de uma célula rígida.

O sistema de layout ALLENDE compreende atualmente os seguintes programas:

- ALLENDE
  Ele toma um programa em Pascal ou C e gera o layout (célula rigida), o circuito a nivel de chaves, ou uma célula flexivel para ser usada posteriormente.

- SIMULATE
  Este programa é um simulador a nivel de chaves, baseado em [6].

- CIF2CELL
  A idéia de CIF2CELL é tornar possivel o uso de código CIF gerado por outras ferramentas. Ele basicamente determina a interface da célula. O código CIF é então usado como uma célula rigida.

- CIF2CIRCUIT
  Quando células rigidas são usadas, não é possivel extrair o circuito da descrição em "alto nivel" do layout. Nesse caso, o circuito é extraido do layout em CIF.

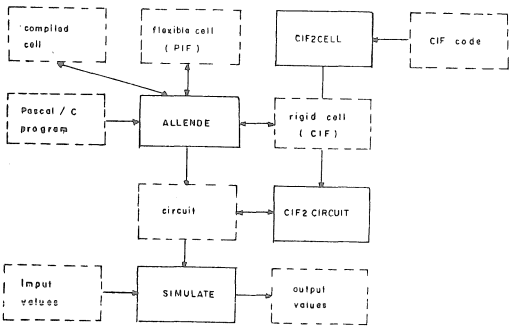


Fig. 8 - O sistema ALLENDE

Este sistema de layout funciona para a tecnologia nMOS, e extensões para outras tecnologias estão sendo estudadas. Os programas que compõem o sistema foram escritos em C e Pascal. O sistema roda sob o sistema operacional UNIX, em um computador VAX. Para fins de utilização deste sistema de layout no Brasil, está sendo providenciado o transporte desse sistema para o sistema operacional MVS, no mesmo VAX, e possivelmente para microcomputadores.

3. Vantagens de nosso método

- concepção hierárquica
  A linguagem ALLENDE força uma concepção hierárquica, e a representação textual do layout reflete esta hierarquia.

- poder expressivo
  Todo o poderio de uma linguagem "procedural" está disponível ao projetista.

- parametrização
  Diferentes layouts podem ser obtidos simplesmente mudando um parâmetro no programa.

- documentação
  O programa é uma documentação do projeto.

- ferramenta aberta
  Editores gráficos tendem a ser ferramentas fechadas no sentido de que é difícil automatizar o processo de layout além do que o projeto original do sistema permite. Linguagens "procedurais" são melhores neste aspecto. A entrada para um compilador é texto que pode ser gerado por seres humanos ou por um programa, enquanto um editor gráfico tem uma natureza interativa, sendo projetado basicamente para aceitar comandos gerados por seres humanos.

- desnecessários equipamentos caros
  Não há necessidade de sofisticados terminais gráficos.

- células flexíveis
  Posições e tamanhos absolutos das células são determinados somente depois de especificadas as posições de todas as células, eliminando problemas de compatibilidade de interface entre células.

- obediência a regras de desenho
  As regras de desenho estão implícitas no processo de geração de equações, garantindo que o layout obtido não contém violações a essas regras.

4. Conclusões

O sistema de layout ALLENDE tem sido usado por um grande número de pessoas, em universidades e centros de pesquisa dos Estados Unidos. Diversos circuitos foram projetados e fabricados com sucesso. ALLENDE foi usado também em experimentos que requerem a produção de diversos layouts diferentes para um mesmo tipo de circuito [1]. Algumas ferramentas de layout, como um gerador de PLA (programmable logic array) a partir de equações booleanas e um roteador de "pads", foram escritas em ALLENDE com esforço mínimo.

Um aspecto importante de um sistema, visto somente quando o sistema é usado, é a deteção e localização de erros. Em ALLENDE os layouts produzidos são corretos por definição, em termos de conectividade e regras de desenho. Em caso de erros na especificação do layout pelo usuário, o sistema aponta as células envolvidas no erro.

Quanto a compactação, os layouts produzidos pelo sistema são relativamente densos. É difícil fazer uma comparação de densidade para layouts produzidos por ALLENDE e layouts produzidos manualmente, porque isto depende da regularidade do layout e da engenhosidade do projetista. Baseado nos layouts já produzidos, encontramos que para estruturas regulares a densidade é praticamente a mesma, enquanto que para estruturas irregulares gastamos aproximadamente 20% mais área do que layouts compactados manualmente.

A representação estruturada do layout e o uso de uma linguagem intermediária (PIF) levaram a um sistema bastante eficiente, em termos de gasto de memória e tempo de execução, e uma implementação simples.

Em um programa ALLENDE a estrutura do circuito e a estrutura do layout se sobrepõem, isto é, o usuário descreve ao mesmo tempo a estrutura do circuito e a estrutura do layout. A estrutura do circuito vai até o nível de transistores e contatos. Poderíamos ter uma linguagem permitindo a especificação do circuito em um nível mais alto (a nível de porta ou a nível funcional, por exemplo). Desta

especificação o layout em PIF seria gerado. Generalizando, PIF (ou ALLENDE) poderia ser a linguagem objeto para uma ferramenta de projeto de circuitos integrados, até mesmo um compilador de silício.

Além de ser uma ferramenta poderosa, ALLENDE está associada a uma metodologia estruturada de projeto de circuitos integrados. Uma ferramenta que força hierarquia e o uso de estruturas regulares vai melhorar a forma como são projetados circuitos integrados. Este é um passo na direção do domínio da complexidade do projeto de circuitos integrados.

5. Referências

[1] K. Iwano and K. Steiglitz
"Some Experiments in VLSI Leaf-cell Optimization."
1984 IEEE Workshop on VLSI Signal Processing, Los Angeles, CA, Nov. 1984.

[2] B. Kernighan and L. Cherry
"A System for Typesetting Mathematics"
Communications of the ACM, Vol. 18, No. 3, March 1975.

[3] R.J.Lipton, J.Valdes,G.Vijayan,S.C.North, and R.Sedgewick
"VLSI Layout as Programming"
ACM Transactions on Programming Languages and Systems, Vol. 5, No. 3, July 1983.

[4] J. M. Mata
"Solving Systems of Linear Equalities and Inequalities Efficiently"
15th Southeastern Conference on Combinatorics, Graph Theory and Computing, Baton Rouge, LA, March 1984.

[5] J. M. Mata
"A Methodology for VLSI Design and a Constraint-Based Layout Language"
Ph.D. Thesis, Princeton University, Oct. 1984.

[6] V. Ramachandran
"An Improved Switch-Level Simulator for MOS Circuits"
20th Design Automation Conference, Miami Beach, FL, June 1983.

[7] J. Valdes and R. L. Kalin
"ALI2 Documentation and Implementation Guide: Language Overview"
VLSI Memo #8, Princeton University, Feb. 1983.