Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 76 (1985)

Heft: 21

Artikel: Systèmes CAO pour circuits intégrés VLSI

Autor: Piguet, C. / Zinszner, R. / Dijkstra, E.

DOI: https://doi.org/10.5169/seals-904703

Systèmes CAO pour circuits intégrés VLSI

C. Piguet, R. Zinszner, E. Dijkstra, G. Berweiler

Cet article se propose d'analyser quatre types de systèmes CAO automatiques pour circuits intégrés. La classification proposée est essentiellement basée sur la manière dont sont créées les cellules de la librairie du système CAO, à savoir:

- *dessins de cellules en layout géométrique,*
- *procédures de génération de cellules en layout géométrique,*
- *dessins de cellules en layout symbolique,*
- *procédures de génération de cellules en layout symbolique.*

Les avantages et inconvénients des quatre systèmes automatiques peuvent être comparés, notamment les caractéristiques du logiciel d'assemblage des cellules. La classification proposée peut s'appliquer à des systèmes CAO existants ou futurs de manière à pouvoir juger de leurs performances.

In diesem Aufsatz werden vier CAD-Systeme untersucht, deren Konzepte auf der automatischen Synthese von integrierten Schaltungen beruhen. Die vorgeschlagene Einteilung basiert auf der Art und Weise, wie die in der Zellenbibliothek enthaltenen Layout-Zellen beschrieben sind, d.h. als:

- *Maskenpläne in geometrischem Layout,*
- *Erzeugungsverfahren von Zellen in geometrischem Layout,*
- *Maskenpläne in symbolischem Layout,*
- *Erzeugungsverfahren von Zellen in symbolischem Layout.*

Diese Klassifizierung macht einen Vergleich der Vor- und Nachteile der verschiedenen Systeme möglich; die Merkmale der Programme für den Layout der Zellen werden dargelegt. Die vorgeschlagene Einteilung kann auf existierende sowie auf zukünftige CAD-Systeme angewendet werden, um deren Leistungen zu beurteilen.

Ce travail est partiellement financé par la Commission pour l'Encouragement de la Recherche Scientifique, Crédit N° 1085.1.

Adresse des auteurs

Dr *C. Piguet, R. Zinszner, E. Dijkstra* et *G. Berweiler*, Centre Suisse d'Electronique et de Microtechnique S.A. (CSEM), Maladière 71, 2000 Neuchâtel 7.

1. Introduction

Un système automatique de Conception Assistée par Ordinateur (CAO) pour circuits intégrés VLSI est généralement composé de trois parties essentielles, à savoir une librairie de cellules, un logiciel permettant leur assemblage en accord avec les spécifications du concepteur et enfin des logiciels d'analyse, de simulations et de test.

Les qualités requises pour un système CAO *idéal* sont les suivantes:

- Produire un circuit correct au premier tour d'intégration.
- Garantir des temps et des coûts de développement très réduits.
- Permettre la réalisation d'architectures VLSI.
- Fournir des puces caractérisées par une bonne densité d'intégration.
- Etre indépendant des différentes technologies CMOS.

Les deux derniers points sont aussi importants que les autres. En effet, le rendement et le nombre de circuits par plaque décroît rapidement avec la taille de la puce. En outre, le choix d'une fonderie de silicium ne doit pas être imposé par le système CAO, qui doit donc pouvoir générer le même circuit pour différentes technologies.

Il nous paraît intéressant de proposer une répartition des systèmes CAO automatiques en quatre classes, basée sur la manière dont les cellules de la librairie sont conçues. Une cellule peut en effet être stockée en librairie sous forme d'un dessin en layout géométrique ou en layout symbolique, ce qui définit deux premières classes. Une cellule peut encore être stockée en librairie sous forme d'une procédure écrite dans un langage de programmation adéquat et capable de générer du layout géométrique ou du layout symbolique, ce qui définit les deux classes restantes. Le tableau I représente la classification proposée.

Classification proposée Tableau I

Classes	Layout géométrique	Layout symbolique
Dessins de cellules	cellules standards	layout symbolique
Procédures de génération de cellules	layout procédural géométrique	layout procédural symbolique

Chaque classe nécessite des logiciels particuliers pour générer la librairie de cellules ainsi que pour l'assemblage des cellules. Le type de librairie peut également impliquer le choix de certains simulateurs. Les avantages et inconvénients de ces logiciels pourront alors être mis en évidence.

2. Cellules standards [1]

Dans cette approche, les cellules de la librairie sont dessinées via un éditeur graphique en layout géométrique.

La figure 1 décrit les conséquences de l'utilisation d'une telle librairie de cellules. Les layouts des cellules sont entièrement définis au niveau géométrique, c'est-à-dire par le dessin de tous les rectangles et polygones de tous les masques nécessaires à la fabrication. Il en résulte que les cellules sont complètement rigides, à savoir que la forme de la cellule ainsi que les positions des entrées et sorties ne peuvent pas être modifiées.

L'utilisation de cellules rigides a pour première conséquence que leur assemblage doit être effectué automatiquement par un logiciel de placement et routage automatique. En effet, il n'est pas possible de modifier les cellules pour procéder à un assemblage par collage, notamment en modifiant les positions des entrées et sorties. La figure 2 [1] représente un layout obtenu par un tel logiciel, les cellules de même hauteur ayant leurs entrées et sorties

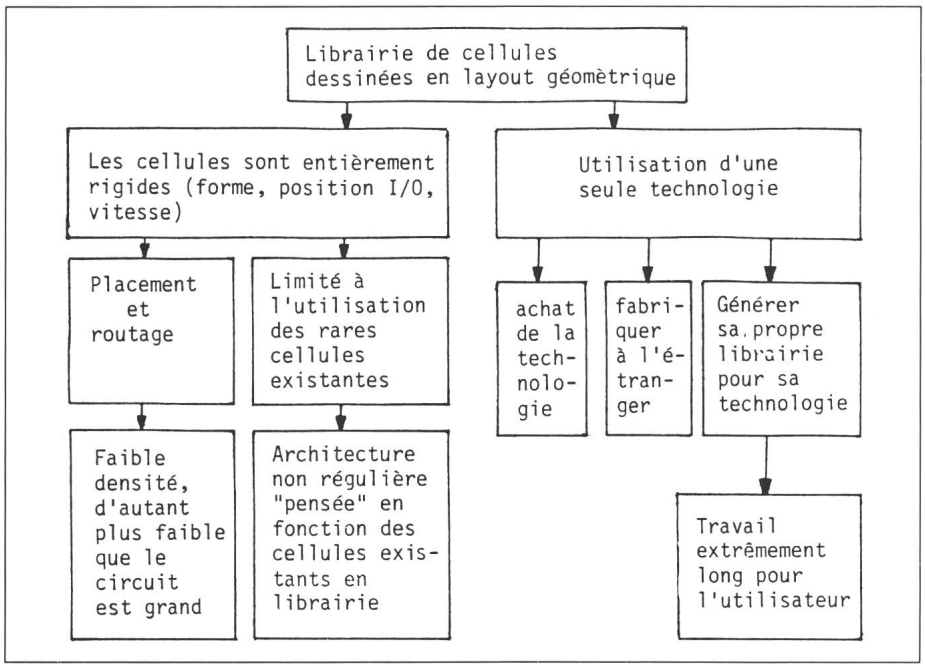


Fig. 1 Cellules standards

sur leurs bords inférieurs et supérieurs. Les cellules sont placées en de longues bandes séparées par des canaux d'interconnexions. La densité d'intégration des puces obtenue est relativement faible, bien que celle des cellules soit excellente. Elle est même très faible pour des circuits VLSI, les connexions étant en moyenne plus longues pour de tels circuits. La surface occupée par les connexions est de 3 à 5 fois la surface occupée par les cellules [1].

L'utilisation d'une telle librairie a des répercussions sur la méthodologie de conception. Celle-ci est finalement analogue à la conception de circuits imprimés, les cellules remplaçant les boîtiers de circuits intégrés. Le concepteur doit impérativement se satisfaire du nombre relativement faible (50 à 100) de cellules à disposition, celles-ci étant plutôt simples, comme des portes et des bascules. Il n'a finalement pas d'autre choix que de générer un circuit du type logique câblée, résultant de l'assemblage des portes et des bascules à disposition. Des architectures régulières de type processeur ou en logique programmée doivent être exclues, alors que dans bon nombre de cas leur temps de conception serait nettement réduit par rapport à un circuit en logique câblée.

Enfin, une librairie de cellules dessinées en layout géométrique implique une dépendance totale envers une seule technologie. Néanmoins, celle-ci sera choisie de manière à être disponible dans plusieurs fonderies de silicium. D'autre part, l'utilisateur d'un tel système achète en général plusieurs librairies de cellules dans des technologies différentes.

Ces outils sont disponibles commercialement et largement utilisés. En effet, ils ne sont pas hors de prix; ils sont d'un emploi aisé puisque la méthode de conception est analogue à celle des cartes; en outre, les cellules sont très bien caractérisées électriquement. Cette approche est donc la meilleure économiquement pour des séries limitées de circuits de moyenne complexité.

3. Layout procédural géometrique [2]

3.1 Principe

Dans un tel système, les cellules de la librairie ne sont plus des layouts géométriques, mais sont constituées par de petits programmes ou procédures capables de générer le layout géométrique de la cellule désirée. Certains systèmes CAO utilisent des langages spécialisés [2], mais il est parfaitement possible d'utiliser des langages de haut niveau, comme Pascal.

L'avantage fondamental d'une telle approche est la possibilité de paramétriser les procédures. Celles-ci sont alors capables de générer des variantes de la même cellule selon les besoins du concepteur. Les types de paramètres généralement disponibles pour de telles procédures sont les suivants [3]:

- paramètres liés au type de cellule, comme le nombre d'entrées d'une porte logique, le nombre de bits d'une unité arithmétique et logique ou le nombre de bits d'une mémoire;
- paramètres liés aux performances électriques de la cellule, comme la largeur des transistors;
- paramètres liés à la topologie de la cellule, comme la possibilité d'étirer la cellule pour lui donner une hauteur normalisée ou pour déplacer, dans une certaine plage, une connexion d'entrée ou de sortie (stretch).

L'ensemble de ces procédures est souvent appelé un compilateur de cellules [4]. La librairie de cellules comporte bien évidemment des portes et des bascules, mais également des cellules plus complexes comme des mémoires mortes ou vives ainsi que des compteurs ou des unités arithmétiques. L'utilisateur a donc à sa disposition l'équivalent d'une librairie de cellules en layout géométrique comportant un très grand nombre de cellules plus ou moins complexes. Cela grâce au fait qu'une seule procédure permet de générer un très grand nombre de variantes en layout géométrique.

3.2 Procédure générant une porte NOR

L'exemple suivant décrit une procédure générant une porte NOR. Cette procédure NOR (E, Wp, Wn, Hn, H) est paramétrée par cinq paramètres, à savoir:

- E, le nombre d'entrées,
- W_p et W_n, les largeurs respectivement des transistors p et n en microns,
- H_n et H, les hauteurs respectivement du caisson et de la cellule.

Cette procédure génère une porte NOR dans une seule technologie CMOS 4 µm du CSEM. Il existe donc des valeurs minimales pour les paramètres W_p, W_n, H_n et H, qui sont res-

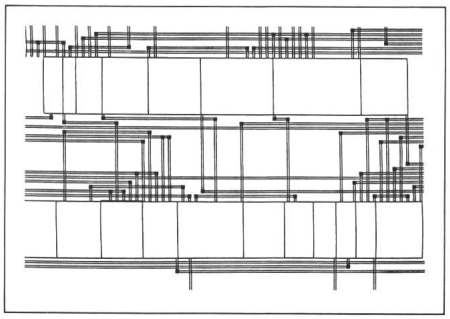


Fig. 2 Assemblage par placement et routage automatique

pectivement de 7 µ pour les largeurs de transistors, 32 µ pour le caisson et 56 µ pour la hauteur de la cellule. D'autres paramètres comme P, N, O_{fn} et O_f sont calculés, indiquant les différences entre les valeurs minimales de W_p, W_n, H_n et H et les valeurs désirées par le concepteur. Enfin, les paramètres hauteur et largeur, qui sont également calculés en fonction des paramètres précédents, indiquent les dimensions de la cellule.

La cellule est ensuite décrite par un ensemble de rectangles dont les coordonnées en microns dépendent des paramètres calculés plus haut. La syntaxe utilisée pour définir un rectangle est extrêmement simple:

box x_1, y_1, x_2, y_2, masque

Les deux points x_1, y_1 et x_2, y_2 sont les deux coordonnées extrêmes du rectangle. Le point 0,0 est situé en bas à gauche de la cellule.

Les cinq premiers rectangles sont respectivement le cadre de la cellule, le masque de dopage et celui du caisson, ainsi que les deux diffusions p^+ et n^+. La figure 3a représente les 5 rectangles générés, le caisson étant en pointillés et les diffusions en noir.

```
box 0, 0, largeur, hauteur, cadre
box 0, 0, largeur, 32+N+Ofn, dopage
box 0, 0, largeur, 27+N+Ofn, caisson
box 12, 39+N+Ofn, 19+largeur-entrée,
    46+N+P+Ofn, diff-p
box 12, 10, 19+largeur-entrée, 17+N,
    diff-n
```

Pour chaque entrée E, une boucle de la procédure génère un conducteur en silicium polycristallin vertical. Comme la technologie choisie comporte des poly bidopés, il est nécessaire de court-circuiter les diodes poly créées lors du passage du poly sur le masque de dopage. Ceci est réalisé par un contact et un rectangle de métal. Les six rectangles décrits plus loin sont le premier un poly vertical pour connecter la sortie de la porte aux côtés inférieur et supérieur de la cellule, avec les deux rectangles pour court-circuiter la diode poly, ainsi que trois rectangles métal pour l'alimentation GND, pour la sortie de la porte et pour l'alimentation VDD. La figure 3b représente les rectangles ainsi générés.

```
FOR w = 0 TO E-1
  box 20+w*13, 0, 24+w*13, hauteur,
      poly
  box 20+w*13, 28+N+Ofn, 24+w*13,
      36+N+Ofn, métal
  box 20.5+w*13, 29+N+Ofn, 23.5+w*13,
      35+N+Ofn, contact
NEXT w
box largeur-9, 0, largeur-5, hauteur, poly
box largeur-9, 28+N+Ofn, largeur-5,
    36+N+Ofn, métal
box largeur-8.5, 29+N+Ofn, largeur-5.5,
    35+N+Ofn, contact
box 0, 2, largeur, 6, métal
box 0, 50+N+P+Ofn+Of, largeur,
    54+N+P+Ofn+Of, métal
box 13.5, 20+N, largeur-4, 24+N, métal
```

Les trois rectangles suivants sont des conducteurs métal liant l'alimentation VDD à la diffusion p^+, cette même diffusion à la sortie et le métal autour du contact sur cette diffusion. La boucle qui suit génère les métaux verticaux entre la diffusion n^+ et la sortie, et la deuxième boucle les métaux verticaux entre cette même diffusion et GND. Enfin, les autres rectangles sont les contacts. La figure 3c représente ces rectangles.

```
box 13.5, 40+N+Ofn, 17.5,
    50+N+P+Ofn+Of, métal
box largeur-16, 24+N, largeur-12,
    45+N+Ofn, métal
box largeur-18, 40+N+Ofn, largeur-12,
    45+N+Ofn, métal
FOR y = 0 TO E STEP 2
  box 13.5+y*13, 11, 17.5+y*13, 20+N,
      métal
NEXT y
FOR x = 1 TO E STEP 2
  box 13.5+x*13, 6, 17.5+x*13, 16, métal
NEXT x
box largeur-10.5, 20+N, largeur-3.5, 24+N,
    poly
box largeur-8.5, 20.5+N, largeur-5.5,
    23.5+N, contact
box 14, 12, 17, 15, contact
box 14, 41+N+Ofn, 17, 44+N+Ofn,
    contact
FOR v = 1 TO E
  box 14+v*13, 12, 17+v*13, 15, contact
NEXT v
box 14+largeur-entrée, 41+N+Ofn,
    17+largeur-entrée, 44+N+Ofn,
    contact
```

La figure 4 représente deux portes NOR avec différents paramètres, à savoir:
- NOR (3, 20, 7, 32, 56) pour la figure 4a,
- NOR (6, 18, 10, 40, 72) pour la figure 4b.

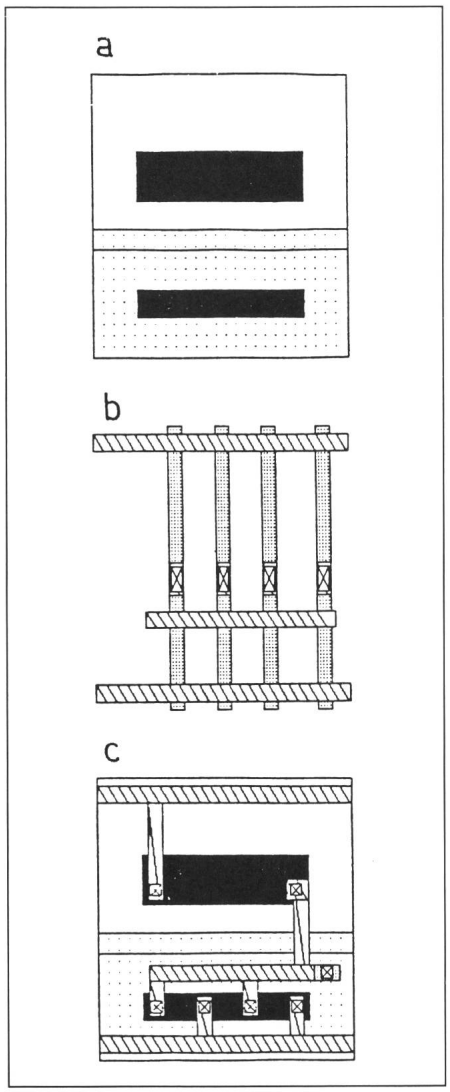


Fig. 3 Génération de rectangles pour une porte NOR

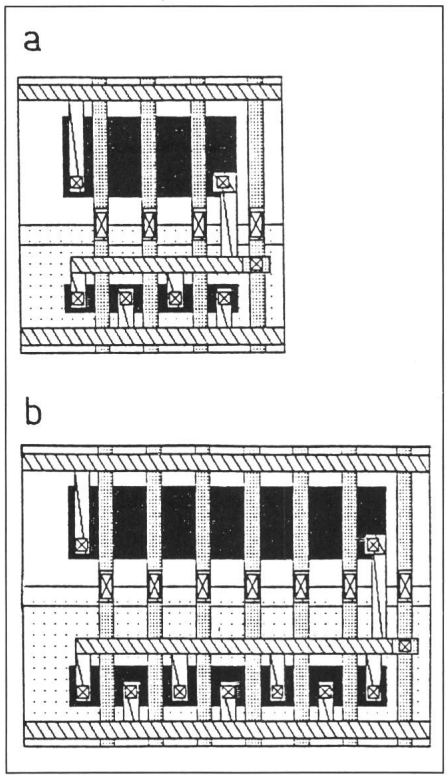


Fig. 4 Variantes d'une porte NOR

On constate que l'écriture d'une telle procédure est relativement aisée pour des cellules régulières. Dans l'exemple précédent, la génération des rectangles pour une entrée supplémentaire de la porte NOR s'effectue au travers des boucles de la procédure. Il en irait de même pour des structures très régulières comme les mémoires.

3.3 Utilisation du layout procédural géométrique

La figure 5 représente les conséquences liées à l'utilisation d'une librairie de cellules géométriques générées procéduralement.

La topologie d'une cellule est entièrement rigide si elle est générée procéduralement au niveau géométrique. Il n'est pas possible de permuter des transistors ou des points de connexion d'entrée ou de sortie. La forme et la surface des cellules ne peuvent pas être choisies par le concepteur. Elles dépendent en grande partie des valeurs données aux paramètres fixant, par exemple, le nombre d'entrées et la largeur des transistors d'une porte NOR.

La possibilité d'étirer la cellule, notamment en fixant sa hauteur comme pour la porte NOR de l'exemple ci-dessus, ou en déplaçant des points d'entrée ou de sortie, permet de modifier quelque peu la cellule pour l'adapter à son environnement. Cela s'effectue au prix d'une moins bonne densité d'intégration de la cellule, mais permet, dans certains cas, de procéder à un assemblage par collage des cellules. Généralement, cette possibilité n'est utilisée que pour un petit nombre de cellules de la librairie. Les gros modules, comme les mémoires, ne comportent pas cette possibilité.

Le logiciel d'assemblage des cellules est finalement basé sur le même principe que celui utilisé par les cellules standards, à savoir un placement et routage automatique. Ce logiciel est néanmoins plus sophistiqué, puisqu'il doit interconnecter des cellules de forme quelconque, et le placement doit pouvoir être effectué ou modifié interactivement. En général, l'existence de gros modules comme des mémoires ou des unités arithmétiques et logiques implique l'établissement préalable d'un plan directeur de la puce de manière à obtenir un bon placement initial visant à faciliter le routage. La densité d'intégration peut être qualifiée de moyenne pour autant que des modules très denses comme des mémoires soient utilisés et que le placement initial soit bon.

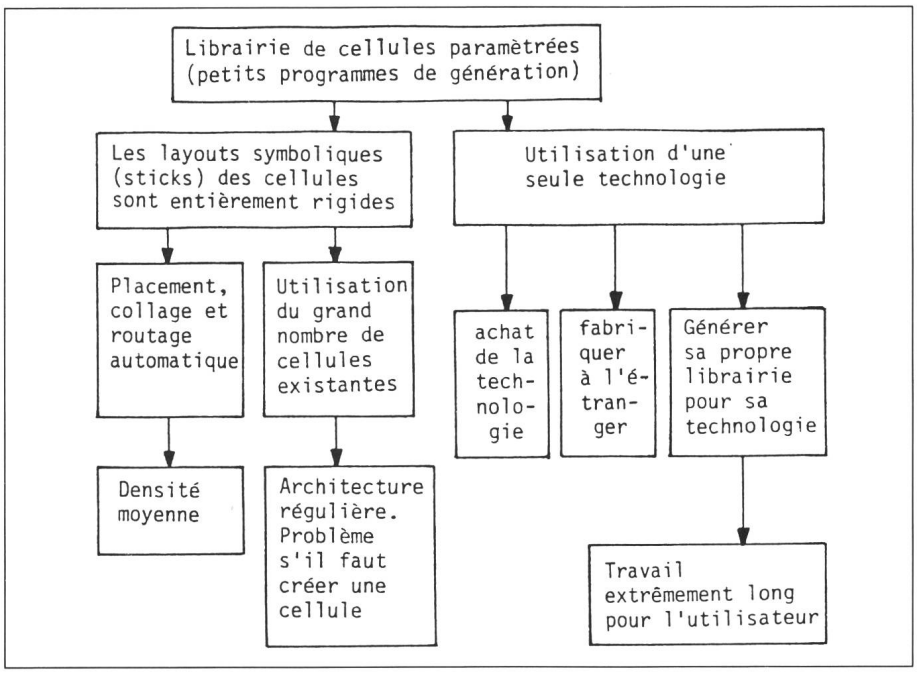


Fig. 5 Layout procédural géométrique

L'équivalent de la librairie de cellules comporte, du point de vue de l'utilisateur, un grand nombre de cellules, y compris des mémoires, des PLA, des registres, des compteurs et des unités arithmétiques. Il est donc possible de recourir à des architectures de type processeurs, et le choix d'une architecture de circuit n'est en général pas limité.

La limite fondamentale d'un tel système CAO est la dépendance envers une seule technologie. Les procédures de génération ne manipulent que des rectangles et des polygones. Leur paramétrisation en fonction des règles de layout d'une technologie rendrait les procédures trop complexes, à tel point qu'il est plus rentable d'écrire une procédure par technologie pour une cellule. Les technologies industrielles comportent en général sur le plan des règles de layout un certain nombre d'exceptions et d'originalités dont il est difficile de tenir compte dans une procédure. Par exemple, la procédure du § 3.2 génère des contacts court-circuitant les diodes poly, le poly étant bidopé. D'autres technologies comportent un poly monodopé, ce qui impliquerait la modification de cette procédure pour supprimer la génération de ces rectangles.

3.4 Systèmes CAO existants

Une telle approche est utilisée dans une partie du système de VLSI Technology Inc. (VTI) appelée compilateur de cellules [4; 5]. Ce compilateur comporte aussi bien des portes et des bascules que des mémoires, compteurs et unités arithmétiques. Les cellules peuvent être paramétrées par le nombre d'entrées ou le nombre de bits ainsi que par les performances en vitesse. L'assemblage ne s'effectue que par routage, aucune adaptation de la cellule à son environnement n'étant possible. Cette librairie de cellules est totalement dépendante des technologies de VTI et non adaptable, aujourd'hui, à une autre technologie par l'utilisateur.

Le compilateur de silicium Genesil [6; 7] est également basé sur la même approche. La librairie de cellules est composée de modules permettant de générer des architectures de processeurs très puissants (comme la puce Microvax I) dans les technologies actuellement prévues dans Genesil.

4. Layout symbolique

La librairie est composée de cellules dessinées via un éditeur graphique en layout symbolique. Un traducteur permet de générer un layout géométrique à partir du layout symbolique pour n'importe quelle technologie [8; 9]. Ce logiciel assure également la compac-

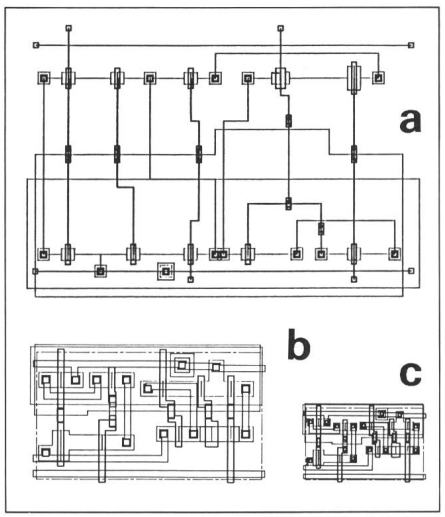


Fig. 6 Cellule compactée par CABBAGE
a Layout symbolique avant compaction
b Layout compacté; technologie 6 µ, 221×128 µm
c Layout compacté; technologie 4 µ, 147×77 µm

tion du layout géométrique. La figure 6 [9] représente une même cellule en layout symbolique traduite en deux technologies différentes. L'utilisateur peut facilement décrire un fichier technologique pour ajouter au traducteur une technologie de son choix.

La figure 7 représente les conséquences de l'utilisation d'une telle librairie. Les layouts symboliques des cellules en librairie sont par définition dits rigides, puisque le principe même de librairie interdit à l'utilisateur de modifier les cellules pour éviter l'introduction d'erreurs. Il est par contre possible de donner à certains traducteurs des contraintes sur la hauteur ou la position d'entrées ou de sorties. Celui-ci en tient compte en étirant la cellule, ce qui est peu favorable à une bonne densité d'intégration. Néanmoins, un assemblage par collage est possible dans certains cas, mais le plus souvent celui-ci devra s'effectuer par routage.

Les conséquences sur la méthodologie de conception et le type d'architecture sont analogues à celles concernant les cellules standards. Il n'est en effet pas possible de décrire en layout symbolique des modules comme des mémoires. En effet, toutes les mémoires requises ont en général une taille différente, et les approches procédurales (chap. 3 et 5) s'imposent de manière évidente, la taille de la mémoire étant un des paramètres.

L'avantage de cette approche est l'indépendance des technologies.

Il n'existe pas de systèmes CAO qui proposent des librairies de cellules en layout symbolique, d'une part en raison de la faible densité d'intégration et, d'autre part, en raison du fait que la caractérisation électrique de chaque cellule doit être refaite pour chaque utilisation, après traduction une technologie connue. Cette approche est utilisée en complément d'une librairie en layout géométrique pour générer, au besoin, une cellule manquante. C'est le cas du système CAO de VTI [5].

L'intérêt de cette approche en tant que librairie est certainement une librairie de circuits analogiques où la densité est de peu d'importance, mais où le dimensionnement des transistors ainsi que l'adjonction de capacités sont essentiels.

5. Layout procédural symbolique

5.1 Principe

La librairie est constituée de procédures capables de générer des cellules en layout symbolique. Un logiciel, capable de traduire le layout symbolique en layout géométrique, est également nécessaire.

L'objectif est de supporter une méthodologie de conception fondamentalement différente de celle qui consiste à utiliser un logiciel de placement et de routage automatique. Pour obtenir une excellente densité d'intégration, le concepteur doit élaborer un plan directeur de la puce en estimant les surfaces des modules. Il génère alors les différents modules en faisant appel à la librairie, en spécifiant les contraintes de forme des modules ainsi que les positions des entrées et des sorties pour satisfaire au mieux le plan directeur adopté. L'assemblage s'effectue ensuite essentiellement par collage, bien que le routage soit utilisé pour l'interconnexion des gros modules. Il en résulte que les procédures de génération de cellules seront essentiellement paramétrées en fonction des contraintes liées à l'environnement de la cellule (forme, positions des entrées et sorties).

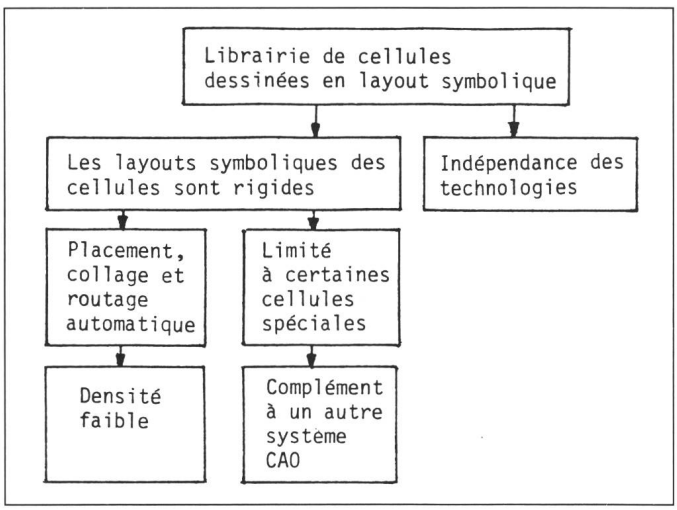
L'obtention d'une bonne densité d'intégration des cellules de la librairie requiert le choix d'un layout symbolique ordré ou orienté. Si les transistors d'une cellule sont ordrés sur une grille virtuelle avant la traduction en layout géométrique par un logiciel de compaction (chap. 4), la densité d'intégration de la cellule est meilleure que celle obtenue à partir d'un layout symbolique non ordré [10; 16]. Dans un layout dit orienté [11; 12; 17], certains conducteurs ne sont implantés que dans une seule direction, ce qui rend le layout symbolique fortement ordré. Il en résulte que le concepteur de cellules ne peut pas concevoir autre chose qu'un layout ordré et, par conséquent, que la densité d'intégration d'une cellule sera toujours au moins acceptable.

Un autre avantage d'un layout symbolique orienté est la possibilité d'écrire des procédures pas trop complexes pour la génération automatique des cellules d'une librairie, ces procédures étant paramétrées par la forme et les positions des entrées et sorties des cellules.

5.2 Procédure générant une porte XOR

L'exemple suivant décrit une procédure générant la porte XOR de la figure 8 en layout symbolique orienté. Les variables seront matérialisées par des pistes verticales, tandis que les branches du circuit seront placées horizon-

Fig. 7
Layout symbolique



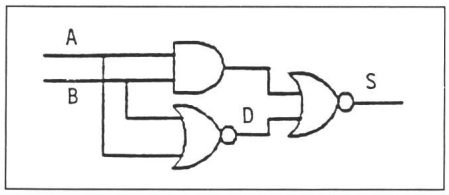


Fig. 8 Porte XOR

talement. Les branches représentent des transistors MOS connectés en série.

La librairie comporte trois informations concernant la porte XOR:

- la liste des branches, une branche comportant des transistors en série branchés entre deux pistes,
- l'ensemble des graphes de précédence [13], chaque graphe décrivant un placement possible des pistes verticales,
- l'ensemble des topologies de la cellule associées aux graphes de précédence, chaque topologie indiquant le placement des branches du circuit.

La porte XOR comporte 7 branches, ainsi que 7 graphes de précédence associés à 7 topologies différentes. Les graphes de précédence comportent de 6 à 10 pistes, permettant ainsi de décrire des topologies avec plus ou moins d'associations de branches deux à deux dans la même ligne du layout orienté. Chaque association possible réduit le nombre total de lignes de la cellule.

Le concepteur, lors de l'appel de la porte XOR en librairie, indique le nombre de pistes désiré pour la cellule, ainsi que les pistes sur lesquelles seront placées les entrées A et B et la sortie S. Le logiciel de génération de la cellule, sur la base de la contrainte donnée par le concepteur, cherche si le premier graphe de précédence satisfait à la contrainte, auquel cas la solution optimale est trouvée. En effet, les graphes de précédence sont classés en librairie de manière à choisir en premier les topologies optimales de la cellule. En cas d'échec, le logiciel tentera la comparaison avec les graphes de précédence suivants.

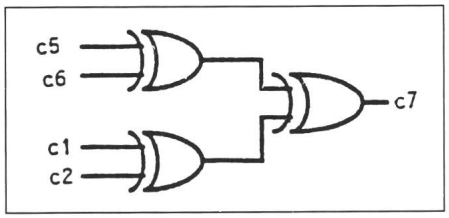


Fig. 9 Fonction logique

5.3 Exemple d'application

Le concepteur désire implémenter la fonction c7 = (c5 XOR c6)XOR(c1 XOR c2). La figure 9 représente cette fonction logique qui comporte donc 3 portes XOR. Les contraintes sur les entrées et la sortie sont les suivantes:

- c5, c6, c1 et c2 sont placées sur le bord supérieur aux pistes 3, 4, 5 et 6;
- la sortie c7 est placée sur le bord inférieur à la piste 3;
- le nombre de pistes total est de 12;
- les alimentations sont placées aux pistes 1 et 2, de même qu'aux pistes 11 et 12.

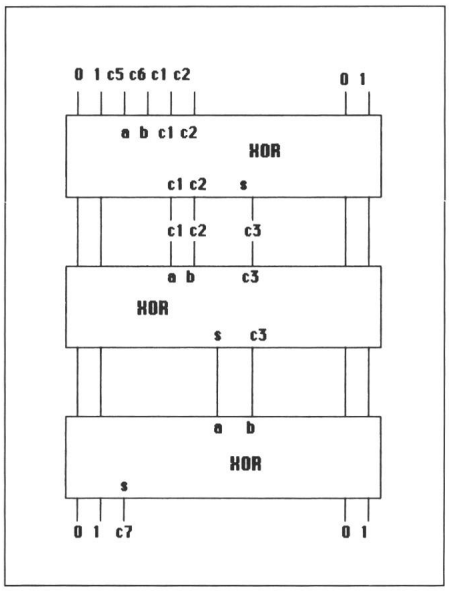


Fig. 10 Plan directeur

La figure 10 représente le plan directeur de ces trois cellules XOR.

Le concepteur peut alors utiliser le programme décrit pour générer les trois portes XOR en tenant compte de leurs interconnexions. Il connaît les contraintes sur les entrées de la première cellule et sur les sorties de la dernière. Il paraît donc logique de commencer par la première cellule et de reporter ensuite les contraintes sur la 2e, etc.

La première porte XOR est générée avec la contrainte:

?XOR([0, 1, a, b, c1, c2, U, P, R, T, 0, 1])

La contrainte sur l'ordre des pistes est donnée ici en énumérant dans l'ordre les 12 pistes requises, les pistes 0 et 1 étant les alimentations, a et b les entrées, c1 et c2 des pistes traversant la cellule. Les majuscules représentent

Première cellule Tableau II

```
TRANCHE ESSAI;
  TERM(-,+);
    CELLULE ESSAI;
    ORDRE(-,+,A,B,C1,C2,D,S,D,B,-,+);
      TOP(-,+);
        ZONEN;
        DIFF    -(A,B)S/S(D)-;
        DIFF    -(A)D/D(B)-;
        ZONEP;
        POLY    D=D;
        POLY    B=B;
        DIFF    +(D,A)S/S(B,D)+;
        DIFF    +(A,B)D;
    BOTTOM(+,-);
  TERM(-,+);
END.
```

des pistes non encore attribuées à une variable de la cellule, cette attribution étant faite par le logiciel cherchant la topologie de la cellule en fonction de la contrainte donnée.

Le tableau II représente la cellule obtenue, décrite dans le langage de description présenté dans [13]. En particulier, ORDRE(...) énumère dans l'ordre les pistes, la sortie S étant placée à la 8e piste. L'expression

DIFF - (A,B)S/S(D)

signifie que deux branches ont été associées dans la même ligne, la première branche comportant deux transistors A et B connectés entre - et S.

Pour la deuxième cellule, les entrées a et b sont placées aux pistes 5 et 6, là où étaient placées les connexions c1 et c2 de la première porte. En outre, il faut prévoir que la sortie S de la première porte traverse la deuxième cellule par la connexion c3 à la piste 8, là où était placée la sortie S de la première cellule. La contrainte est donc:

?XOR([0, 1, U, P, a, b, R, c3, T, X, 0, 1])

La structure obtenue place la sortie S à la neuvième piste (tab. III).

La troisième porte admet comme entrées les sorties des deux précédentes qui sont logées aux pistes 8 et 9. Ce seront donc les entrées a et b. En outre, le cahier des charges impose la sortie S à la troisième piste. La contrainte est donc:

?XOR([0, 1, s, U, P, R, T, a, b, X, 0, 1])

La structure obtenue est représentée au tableau IV. La variable S a dû être répétée.

Deuxième cellule Tableau III

```
TRANCHE ESSAI;
  TERM(-,+);
    CELLULE ESSAI;
    ORDRE(-,+,B,*,A,B,D,C3,S,A,-,+);
      TOP(-,+);
        ZONEN;
        DIFF    -(D)S;
        DIFF    -(A,B)S;
        DIFF    -(B)D/D(A)-;
        ZONEP;
        POLY    B=B;
        POLY    A=A;
        DIFF    +(A,B)D;
        DIFF    +(B,D)S;
        DIFF    +(D,A)S;
    BOTTOM(+,-);
  TERM(-,+);
END.
```

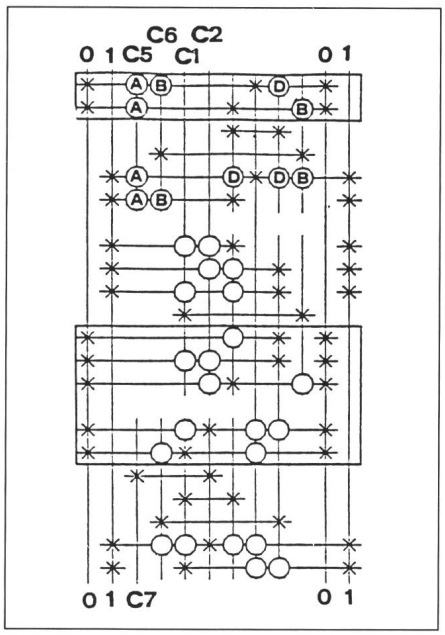


Fig. 11 Layout symbolique des trois portes XOR

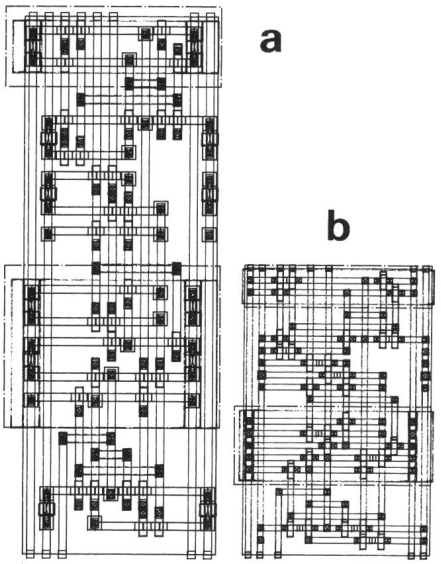


Fig. 12 Layouts géométriques des trois portes XOR
a CSEM 4 µm orienté métal
b SACMOS 3 µm orienté poly

La figure 11 représente le layout symbolique obtenu pour les trois cellules après assemblage par collage. Un traducteur basé sur des microlibrairies d'une quinzaine de microcellules géométriques chacune [12] permet de générer les layouts géométriques de la figure 12 en technologie CSEM 4 µm [18] et en technologie SACMOS 3 µm.

On constate que ces trois cellules identiques au point de vue fonction sont toutes trois différentes au niveau topologique, et ceci déjà au niveau du layout symbolique. On peut également se convaincre que le travail effectué par le concepteur est assez simple, et de ce fait très rapide. Les trois cellules générées comportent ensemble 33 MOS. En technologie CSEM 4 µm, la densité obtenue est d'environ 1300 MOS/mm², alors qu'elle est d'environ 2500 MOS/mm² en technologie SACMOS 3 µm.

Dans cet exemple, la détermination des contraintes pour chacune des cellules ainsi que leur assemblage est effectué manuellement. Une automatisation de ce travail apparaît comme parfaitement réalisable.

5.4 Utilisation de layout procédural symbolique

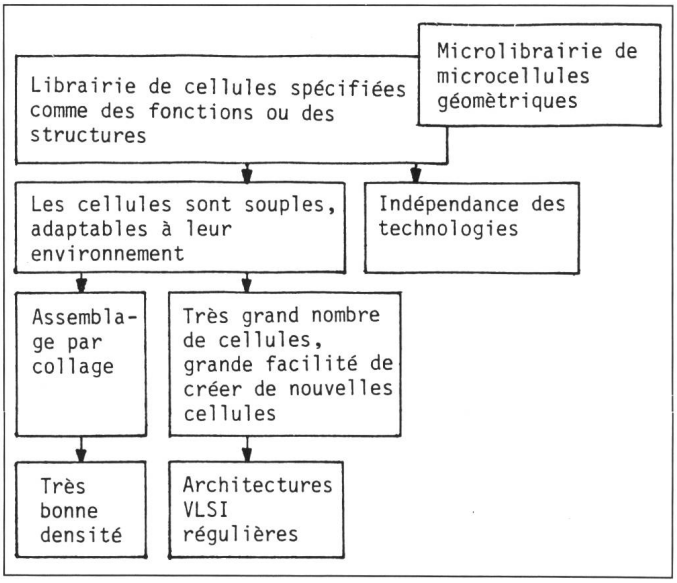
L'exemple du § 5.3 démontre clairement que la méthode de conception est essentiellement basée sur un plan directeur préalable et un assemblage par collage. Comme pour toute méthode de conception, il est possible de l'appliquer manuellement comme dans l'exemple du § 5.3. Une automatisation est également possible, nécessitant un éditeur graphique pour dresser le plan directeur et spécifier les interconnexions. Ce dernier logiciel est donc très différent d'un éditeur de schémas utilisé pour les méthodes de placement et de routage automatique. Le logiciel d'assemblage doit également pouvoir coller des cellules en plus de la possibilité de routage.

La figure 13 représente les conséquences de l'utilisation d'une telle librairie. L'assemblage par collage garantit une très bonne densité d'intégration. La librairie peut contenir un grand nombre de cellules et de modules, permettant la conception d'architectures VLSI. La création aisée de cellules de librairie par l'utilisateur

Troisième cellule Tableau IV

```
TRANCHE ESSAI;
  TERM(-,+);
    CELLULE ESSAI;
    ORDRE(-,+,S,B,D,S,D,A,B,*,-,+);
      TOP(-,+);
        ZONEN;
        DIFF    -(D)S/S(A,B)-;
        DIFF    -(B)D/D(A)-;
        ZONEP;
        POLY    S=S;
        POLY    D=D;
        POLY    B=B;
        DIFF    +(B,D)S/S(D,A)+;
        DIFF    D(A,B)+;
    BOTTOM(+,-);
  TERM(-,+);
END.
```

Fig. 13 Layout procédural symbolique



nécessitera l'écriture de logiciel générant automatiquement les graphes de précédence et les topologies associées d'une cellule à partir des équations logiques d'une cellule. La référence [13] traite en partie ce problème. Enfin, l'indépendance des technologies est garantie, l'adjonction de nouvelles technologies pouvant être rapidement effectuée.

Aucun système CAO commercial de ce type n'existe. Certains travaux universitaires [14; 15] ont été effectués dans ce domaine pour une technologie n-MOS. Le CSEM travaille à un tel système CAO [12; 13].

6. Conclusion

La classification proposée au tableau I permet de faire ressortir certaines caractéristiques qui ont été mises en évidence. En particulier, les tableaux V, VI et VII résument les caractéristiques des quatre approches proposées sur les plans du type de circuit qu'il est possible de concevoir, la densité d'intégration et l'indépendance des technologies. Cette comparaison n'est bien entendu valable que pour des systèmes CAO automatiques. La sûreté de conception ainsi que les temps de développement sont donc équivalents pour les quatre approches.

Il est également possible de caractériser les systèmes CAO commerciaux en utilisant la classification proposée. Les systèmes les plus performants utilisent en général plusieurs des approches décrites selon le type de modules à concevoir. Les tableaux VIII, IX et X représentent un système CAO de placement et routage classique, le système VTI [4; 5] et le système GENESIL [6; 7].

Le système CAO GENESIL permet de traduire directement les spécifications d'un circuit, écrites en langage de haut niveau, en layout géométrique. Actuellement, aucune intervention au niveau graphique n'est nécessaire ni même possible.

Le système CAO en cours de construction au CSEM est représenté au tableau XI. Les quatre approches sont utilisées, mais pour des cellules de type différent.

Les travaux au CSEM ont jusqu'ici exclusivement porté sur les filières de génération de cellules et de modules. La méthode de conception proposée, basée sur le plan directeur et l'assemblage par collage, a été utilisée (et testée) manuellement pour des circuits industriels.

Architectures Tableau V

Architectures	Layout géométrique	Layout symbolique
Dessins de cellules	Logique câblée	Logique câblée
Procédures de génération de cellules	Architectures régulières VLSI	Architectures régulières VLSI

Densité d'intégration du circuit complet Tableau VI

Densité	Layout géométrique	Layout symbolique
Dessins de cellules	faible	faible
Procédures de génération de cellules	moyenne	très bonne

Dépendance ou indépendance des technologies Tableau VII

Technologie	Layout géométrique	Layout symbolique
Dessins de cellules	dépendant	indépendant
Procédures de génération de cellules	dépendant	indépendant

Placement et routage classique Tableau VIII

Placement et routage autom.	Layout géométrique	Layout symbolique
Dessins de cellules		
Procédures de génération de cellules		

Système CAO de VTI Tableau IX

VTI	Layout géométrique	Layout symbolique
Dessins de cellules		
Procédures de génération de cellules	compilateur de cellules	

Système CAO GENESIL Tableau X

GENESIL	Layout géométrique	Layout symbolique
Dessins de cellules		
Procédures de génération de cellules		

Système CAO en cours de construction au CSEM Tableau XI

CSEM	Layout géométrique	Layout symbolique
Dessins de cellules	spécialité	circuits analogiques
Procédures de génération de cellules	mémoires ROM	circuits digitaux

Bibliographie

[1] *A.J. Kessler* and *A. Ganesan:* Standard cell VLSI design: A tutorial. IEEE Circuits and Devices Magazine 1(1985)1, p. 17...34.

[2] *R.F. Ayers:* VLSI silicon compilation and the art of automatic microchip design. Englewood Cliffs, N.J., Prentice Hall, 1983.

[3] *S. Trimberger:* Automated performance optimization of custom integrated circuits. Thesis of the California Institut of Technology, Pasadena, 1981.

[4] *S. Nance* a.o.: Cell-layout compilers simplify custom-IC design. EDN 28(1983)19, p. 147...158.

[5] *R. Beresford:* Advances in customization free VLSI system designers, Electronics 56(1983)3, p. 134...145.

[6] *S. Baker:* Silicon compiler puts systems house in VLSI business. Electronic Engineering Times 13(1983) October 8.

[7] *S.C. Johnson:* Silicon compiler lets system makers design their own VLSI chips. Electronic Design 32(1984)20, p. 167...181.

[8] *M.Y. Hsueh:* Symbolic layout and compaction for integrated circuits. Memo UCB M 79/80. Berkely, University of California, 1980.

[9] *R. Zinszner, H. DeMan* and *K. Croes:* Technology independent symbolic layout tools. IEEE International Conference on Computer-Aided Design 1983 (ICCAD-83), p. 12...13.

[10] *K.H. Schmidt* and *K.D. Müller-Glaser:* NMOS dense gate matrix VLSC design. IEEE J. Solid-State Circuits 18(1983)2, p. 157...159.

[11] *A.D. Lopez* and *H.-F.S. Law:* A dense gate matrix layout method for MOS VLSI. IEEE J. Solid-State Circuits 15(1980)4, p. 736...740.

[12] *C. Piguet* a.o.: A metal-oriented layout structure for CMOS logic. IEEE J. Solid-State Circuits 19(1984)3, p. 425...436.

[13] *C. Piguet, E. Dijkstra* and *G. Berweiler:* Automatic generation of CMOS layout cells from a hardware description language. Euromicro 85, Brussels, September 3...6, 1985.

[14] *J. Luhukay* and *W.J. Kubitz:* A layout synthesis system for NMOS gate-cells. 19th ACM/IEEE Design Automation Conference 1982, p. 307...314.

[15] *W.J. Kubitz, J.F.P. Luhukay* and *F.L. Wong:* An automatic NMOS cell synthesis system. IEEE International Conference on Computer-Aides Design 1983 (ICCAD-83), p. 190...191.

[16] *J.C. Martin* e.a.: Synthèse automatique de circuits CMOS. Bull. ASE/UCS 74(1983)5, p. 221...223.

[17] *C. Piguet:* Méthodes de conception de circuits intégrés complexes. Bull. ASE/UCS 75(1984)1, p. 12...17.

[18] *C. Piguet* et *M. Bertarionne:* Génération automatique de layout orienté métal pour circuits CMOS. Bull. ASE/UCS 75(1984)1, p. 35...39.