High Speed Layout Synthesis for Minimum-Width CMOS Logic Cells via Boolean Satisfiability

Tetsuya Iizuka[†], Makoto Ikeda[†‡], and Kunihiro Asada[†‡]

†*Dept. of Electronic Engineering, University of Tokyo*
‡*VLSI Design and Education Center (VDEC), University of Tokyo*
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{*iizuka, ikeda, asada*}*@silicon.u-tokyo.ac.jp*

Abstract— This paper proposes a cell layout synthesis method via Boolean Satisfiability (SAT). Cell layout synthesis problems are first transformed into SAT problems by our formulations. Our method realizes the high-speed layout synthesis for CMOS logic cells and guarantees to generate the minimum width cells with routability under our layout styles. It considers complementary P-/N-MOSFETs individually during transistor placement, and can generate smaller width layout compared with pairing the complementary P-/N-MOSFETs case. To demonstrate the effectiveness of our SAT-based cell synthesis, we present experimental results which compare it with the 0-1 ILP-based transistor placement method and the commercial cell generation tool. The experimental results show that our SAT-based method can generate minimum width placements in much shorter run time than the 0-1 ILP-based transistor placement method, and can generate the cell layouts of 32 static dual CMOS logic cirsuits in 54 % run time with only 3 % area increase compared with the commercial tool.

I. Introduction

The cell-based ASIC is one of the major methodologies for designing VLSI. Automation of standard-cell layout generation improves the design time for creating new standard-cell libraries. Many papers have been published in the area of the standard-cell synthesis. The layout generation systems based on pre-defined symbolic layout are commonly used, and can generate the layouts with comparable quality to the manually designed layouts in reasonable run time. However, they only generate the layout which have been defined in advance, and can not be used for on-demand layout generations. Guruswamy, et al.[5] proposed an automatic P&R method for transistor-level layout generation using Simulated Annealing (SA). This method automatically synthesized better quality layouts than manually designed layouts. However, the heuristic method does not guarantee the optimal solution and SA is heavily time-consuming.

Several exact cell synthesis methods have also been proposed. Gupta and Hayes[3][4] proposed the CMOS cell width minimization via Integer Linear Programming (ILP). This method solves the width minimization of two dimensional transistor placement exactly. However, this method treats complementary P-/N-MOSFETs pair and aligns only these MOSFETs vertically. There are some cases that the cell width becomes smaller when complementary MOSFETs are not aligned vertically. Maziasz and Hayes[7] proposed the exact algorithm for width and height minimization of CMOS cells. This method minimizes both width and height considering intra-cell routability. However, this method also treats complementary MOSFETs pair and the layout styles of this method has some difference from the practical cell layout styles. For example, it does not use horizontal polysilicon for routing.

In this paper, we propose a high speed layout synthesis method for the minimum-width CMOS logic cells via Boolean Satisfiability (SAT). Cell layout synthesis problems, i.e. transistor placement and intra-cell routing problems, are first transformed into SAT problems by our formulations. Devadas[2] has transformed various layout problems such as channel routing and partitioning into SAT problems. However, the transistor placement and intra-cell routing problems could not be solved. Our method can generate the minimum width transistor placement with routability under our layout styles via SAT. It considers complementary P-/N-MOSFETs individually during transistor placement, and can generate smaller width layout than the previous exact methods explained above which treats the complementary P-/N-MOSFETs pair. Furthermore, we defined more practical layout styles than the previous exact cell synthesis method, so that we can handle the multiple sized transistors and the horizontal polysilicon. Our method does not minimize the cell height because in this paper we focused on the standard-cell layout synthesis whose height is usually fixed. Our SAT-based cell synthesis method can generate the minimum placements in much shorter run time than the 0-1 ILP-based transistor placement method, and can generate cell layouts in much shorter run time than the commercial cell generation tool. Therefore, we think that it can significantly shorten time-to-market of a cell library, and can also be used for many other applications such as on-demand cell synthesis, because it generates a layout quickly from a netlist, not from a pre-defined symbolic layout.

II. Layout Styles

Our layout styles are described in Table I and illustrated in Fig. 1. By using these styles, the cell synthesis problems are efficiently transformed into SAT problems and the search for a solution is speeded up. The general styles for one dimensional width minimization of static dual CMOS logic cells were proposed by Uehara and vanCleemput[10]. Our layout style No.1 through 5 are for the transistor placement based on Uehara's

TABLE I
OUR LAYOUT STYLES

1.	Static dual CMOS logic circuits.
2.	Transistors are drawn up in two horizontal rows, the upper row for P-MOSFETs and the bottom row for N-MOSFETs.
3.	Two transistors which have the same GATE nodes are vertically aligned.
4.	Two transistors which have the same SOURCE/DRAIN terminals are placed in the neighboring column to share their diffusions.
5.	The bottom of the P-MOSFETs and the top of the N-MOSFETs are aligned to G-Region.
6.	The intra-cell routing uses polysilicon and first metal layers.
7.	All nets to connect SOURCE/DRAIN terminals of P-MOSFETs are in P-region.
8.	All nets to connect SOURCE/DRAIN terminals of N-MOSFETs are in N-region.
9.	All nets to connect GATE terminals are in G- or Top- or Bottom-regions.
10.	GATE terminals are connected by the polysilicon layer in Top- or Bottom-region, and by the first metal layer in G-region.
11.	Same nodes in P-region and N-region are connected by the vertical first metal through G-region at the top of N-region and the bottom of P-region.
12.	Vertical first metals and the GATE terminals are connected by the horizontal first metals in G-region.
13.	VDD are connected from the top of P-diffusion through Top-region by the vertical first metal.
14.	GND are connected from the bottom of N-diffusion through Bottom-region by the vertical first metal.
15.	Single contact from metal to diffusion or polysilicon.
16.	No dogleg is used.

styles. However, our layout styles have some difference from theirs. The first difference is the style No.3. The complementary MOSFETs are aligned vertically in Uehara's style. In contrast, MOSFETs which have the same GATE nodes can be aligned vertically in our layout style. As explained in section I, there are some cases that the generated layout has smaller width using our layout style. The second difference is that our method can take multiple sized transistors as an input, so that it can deal with more practical problems, while almost all previous optimal cell synthesis methods assumed the uniform sized transistors. As described by No.5 in Table I and illustrated in Fig. 1, the bottom of the P-MOSFETs and the top of the N-MOSFETs are aligned to G-region.

The layout style No.6 through 16 in Table I are for intra-cell routing. These styles are based on Maziasz's[7] styles. Five routing regions are defined in the cell area as are defined in Maziasz's styles. The P- and N-regions are over the each diffusion, the G-region is between the P- and N-regions, the Top-region is above the P-region and the Bottom-region is below the N-region, as shown in Fig. 1. Our method can deal with outer channel polysilicon routing, i.e. the connection which runs above P diffusions and below N diffusions as described by No. 9 and 10 in Table I. By using outer channel routing, we can avoid using second metal layers for intra-cell routing in many cases. Therefore, our routing styles can be widely

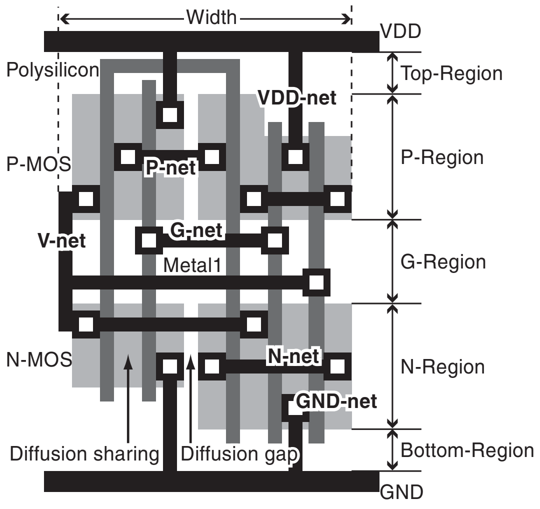

Fig. 1. An illustration of our layout styles

accepted by other layout methods.

III. Transistor Placement

In this section, we explain the SAT constraints for transistor placement. Given N P- and N N-MOSFETs, we have to place these $2N$ transistors in the minimum area. This problem can be transformed into the problem that places all transistors using minimum number of columns as illustrated in Fig. 2. The P-MOSFETs are aligned in the upper row and the N-MOSFETs are in the bottom (No.2 in Table I). The P- and N-MOSFET which are placed on the same column must have the same GATE nodes (No.3 in Table I). The neighboring MOSFETs must face the same SOURCE/DRAIN nodes each other to share their diffusions (No.4 in Table I). The empty columns result in the diffusion gaps in the final layouts. We transform these constraints into boolean constraints.

Each transistor has P variables to identify its placement where $P = \lceil \log_2 W \rceil$ and W is the number of the columns, and one variable to identify whether the SOURCE/DRAIN terminals are flipped or not. $\lceil X \rceil$ indicates a minimum integer which is equal to or larger than X. Thus, the total number of variables needed for this formulation is $2N \times (\lceil \log_2 W \rceil + 1)$. Here, we describe the boolean constraints:

MOS overlap constraints: N-MOSFETs must not overlap in the same column, which is expressed by the following logic equation

$$n_{i1} \oplus n_{j1} \vee n_{i2} \oplus n_{j2} \vee \cdots \vee n_{iP} \oplus n_{jP} = 1 \qquad (1)$$

where $n_{i1}, n_{i2}, \ldots, n_{iP}$ are the variables for placement of N-MOS i, and $\oplus$ is the exclusive-or boolean operator. The same logic equation must hold for P-MOSFETs.

Unused column constraints: If $W < 2^P$, the columns with the top $2^P - W$ distinct bit vectors of length P, correspond to unused columns as illustrated in Fig. 2. No N-MOSFETs can be placed on these unused columns. The following logic equation expresses this constraint for N-MOSFETs.

$$n_{i1} \oplus u_{k1} \vee n_{i2} \oplus u_{k2} \vee \cdots \vee n_{iP} \oplus u_{kP} = 1 \qquad (2)$$

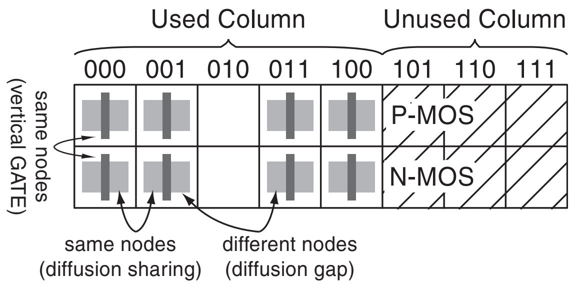
Used Column
Unused Column
000 001 010 011 100 101 110 111
P-MOS
N-MOS
same nodes (vertical GATE)
same nodes (diffusion sharing)
different nodes (diffusion gap)

Fig. 2. A formulation of the transistor placement

where $u_{k1}, u_{k2}, \ldots, u_{kP}$ are constant bit vectors which indicate the unused column k. Same logic equation must hold for P-MOSFETs.

Vertical GATE constraints: The P- and N-MOSFET which are placed on the same column must have the same GATE nodes. Assume G_{pi} is the group of P-MOSFETs which have the same GATE nodes as that of N-MOS i, this constraint is expressed as follows.

$$\bigvee_{j \in G_{pi}} \overline{(n_{i1} \oplus p_{j1} \vee n_{i2} \oplus p_{j2} \vee \cdots \vee n_{iP} \oplus p_{jP})} = 1 \tag{3}$$

Neighboring MOS constraints: The MOSFETs which face the different nodes each other can not be on the neighboring columns. Assume $C_n(i,k)$ implies that N-MOS i is on the column k, the following logic equation expresses this constraint for N-MOSFETs.

$$GAP_n(i,j) \wedge \Bigl(\bigvee_{k=0}^{W-2} C_n(i,k) \wedge C_n(j,k+1)\Bigr) = 0 \tag{4}$$

Here, $GAP_n(i,j)$ takes the value of 1 if NMOS i can not share diffusion with NMOS j placed to its immediate right, otherwise 0. Same logic equation must holds for P-MOSFETs.

These boolean constraints express the all possible placement in W columns under our layout styles. These constraints are expressed in Conjunctive Normal Form (CNF) to be solved by the CNF-SAT solver. If there is no satisfiable assignment using W columns, it is guaranteed that there is no possible placement of width W. Therefore, we can find the minimum width placement using the procedure described below.

1. Given a transistor netlist and enumerate the number of MOSFETs. The initial number of columns W is set to the same number of N-MOSFETs (=#P-MOSFETs).
2. Try to find a satisfiable assignment for the boolean constraints constructed for W columns. If a satisfiable assignment is found, these MOSFETs can be placed on the W columns and this procedure terminates. Otherwise, go to step 3.
3. $W = W + 1$. Go to step 2 again.

IV. Intra-cell Routing

We next explain the SAT constraints of the intra-cell routing in this section. Our styles of the intra-cell routing are listed by No.6 through 16 in Table I. We defined five types of net listed below as illustrated in Fig. 1.

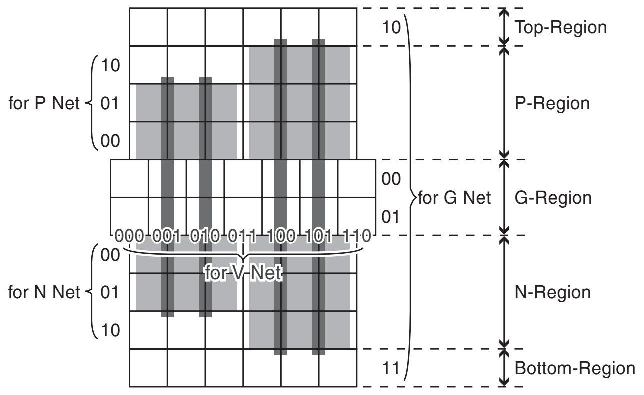

Fig. 3. A formulation of the intra-cell routing

N-net The nets which connect the N-MOS SOURCE/DRAIN.

P-net The nets which connect the P-MOS SOURCE/DRAIN.

G-net The nets which connect the GATE terminals.

V-net The nets which connect the SOURCE/DRAIN terminals of P- and N-MOSFETs and the GATE terminals.

VDD/GND net The net which connect the VDD/GND nodes to VDD/GND rail which runs top/bottom of cell area.

We also defined the region which each type of net can run as described by No. 7 through 14 in Table I. The bit vectors which correspond to the row or column numbers are assigned to each region as illustrated in Fig. 3. The routing grids are defined as illustrated in Fig. 3. We use the columns which are moved half column in G-region. The number of rows of the P-region/N-region is determined by the width of P-MOSFETs/N-MOSFETs and the design rules because the grid size is determined by the minimum width and spacing rules of metal or polysilicon and some other design rules. The number of rows of Top- and Bottom-regions are fixed to 1. Therefore, the number of rows of G-region W_g is given by the following equation

$$W_g = H_{cell} - W_p - W_n - 2 \tag{5}$$

where H_{cell} is the total number of rows of each cell, which is fixed for all cells so that the height of all cells are uniform.

Assume that P_n, P_p, P_g, and P_v are the number of the boolean variables for each type of net, they are expressed as $P_n = \lceil \log_2 W_n \rceil$, $P_p = \lceil \log_2 W_p \rceil$, $P_g = \lceil \log_2 (W_g + 2) \rceil$, $P_v = \lceil \log_2 W_v \rceil$ where W_n, W_p, and W_g are the number of the rows of each region, and W_v is the number of the columns of G-region. The G-net can be placed on $W_g + 2$ rows because they can be on the Top- and Bottom-region besides the G-region. The variables of the nets which belong to more than two groups consist of the combination of each group's variables. Therefore the total number of variables used for the SAT formulation of intra-cell routing is expressed as

$$\sum_{i \in net} \Bigl(a_i P_n + b_i P_p + c_i P_g + d_i P_v\Bigr) \tag{6}$$

where a_i, b_i, c_i, and d_i are the variables which take the value of 1 if the net i belongs to each group, otherwise 0. Here, we construct the boolean constraints:

Net overlap constraints: We define the interval of a N-net i as $I_n(i) = [l_{ni}, r_{ni}]$, where l_{ni} is the position of the leftmost terminal in the N-region which the net has to be connected to, and r_{ni} is the position of the rightmost one. The nets with intersecting intervals can not be placed on the same row. For each net pair i, j, the logic equation describing this constraint is as follows

$$n_{i1} \oplus n_{j1} \vee n_{i2} \oplus n_{j2} \vee \ldots \vee n_{iP_n} \oplus n_{jP_n} = 1$$
$$i \neq j, \;\; I_n(i) \cap I_n(j) \neq \phi \qquad (7)$$

where $n_{i1}, n_{i2}, \cdots, n_{iP_n}$ are the variables which correspond to the placement of N-net i. The same logic equation must hold for P-net and G-net. For V-net, all pairs of nets must satisfy the above logic equation. Therefore, the logic equation for V-net is expressed as follows

$$v_{i1} \oplus v_{j1} \vee v_{i2} \oplus v_{j2} \vee \ldots \vee v_{iP_v} \oplus v_{jP_v} = 1, \;\; i \neq j \qquad (8)$$

where, $v_{i1}, v_{i2}, \cdots, v_{iP_v}$ are the variables which correspond to the placement of V-net i.

Unused row/column constraints: If $W_v < 2^{P_v}$ for V-net, the columns, with $2^{P_v} - W_v$ distinct bit vectors, correspond to unused columns. No net can be placed on these unused columns. The same case emerges when $W_g + 2 < 2^{P_g}$ for G-net. Whereas for N (P) nets, if $min_{ni} < 2^{P_n}$ ($min_{pi} < 2^{P_p}$) where min_{ni} (min_{pi}) is the minimum grid number of N (P) diffusion whose node has to be connected to N (P) net i, the N (P) net i can not be placed on the columns denoted by $2^{P_n} - min_{ni}$ ($2^{P_p} - min_{pi}$) distinct bit vectors. This constraint is expressed by the following logic equation.

$$n_{i1} \oplus u_{k1} \vee n_{i2} \oplus u_{k2} \vee \cdots \vee n_{iP_n} \oplus u_{kP_n} = 1 \qquad (9)$$

The constant bit vector $u_{k1}, u_{k2}, \ldots, u_{kP_n}$ indicates the unused row/column. The same logic equation must hold for P-net, G-net, and V-net.

VDD/GND net constraints: P-net must not overlap the VDD-nets. We assume V is the group of the column numbers which has to be connected to VDD. If $x \in V$ and $x \in I_p(i)$, P-net i can not be placed on the top row of the minimum width diffusion whose column belongs to $I_p(i)$ and V. For N-net, it must not overlap the GND-net. We define the group G whose members are the column numbers which has to be connected to GND. If $x \in G$ and $x \in I_n(i)$, N-net i can not be placed on the bottom row of the minimum width diffusion whose column belongs to $I_n(i)$ and G. These constraint is expressed as Eq. (9) where $u_{k1}, u_{k2}, \ldots, u_{kP_n}(u_{kP_p})$ indicates the row the N-nets (P-nets) can not be placed on.

Pass of V-net constraints: If a V-net is placed on the column c, there must be no horizontal net over the column c in the G-region so that the V-net can go through the G-region. Assume H_g is the group of horizontal nets in the G-region and $I_{hg}(i) = [l_i, r_i]$ is the interval of the horizontal net where l_i (r_i) is the leftmost (rightmost) terminal or net in the G-region, this constraint is described as follows.

$$|X| = 0, \;\; X = \{x \mid x \in H_g, \; c \in I_{hg}(x)\} \qquad (10)$$

V-net to GATE connection constraints: If V-nets must be connected to the GATE terminals, these V-nets have to be placed on the column which they can be connected to in G-region by horizontal first metal layers, i.e. there is at least one empty column in G-region between such V-nets and terminals. This constraint is described as follows using H_g and I_{hg} defined before.

$$|X| < W_g, \;\; X = \{x \mid x \in H_g, \; I_{hg}(x) \cap I_{hg}(i) \neq \phi\} \qquad (11)$$

V-net to SOURCE/DRAIN connection constraints: If V-nets must be connected to the SOURCE/DRAIN terminals of N-MOSFETs, these V-nets have to be placed on the column which they can be connected to on the top of N-region by horizontal first metal layers, i.e. there is no net placed on the top of N-region between such V-nets and terminals. Assume H_n is the group of horizontal nets which are placed on the top of N-region and $I_{hn}(i) = [l_i, r_i]$ is the interval of the horizontal net where l_i (r_i) is the leftmost (rightmost) terminal or net in the N-region, this constraint is described as follows.

$$|X| = 0, \;\; X = \{x \mid x \in H_n, \; I_{hn}(x) \cap I_{hn}(i) \neq \phi\} \qquad (12)$$

In case of P-MOSFETs, these V-nets and terminals must be connected to on the bottom of P-region and the same logic equation must hold for P-MOSFETs.

The SAT formulation which consists of these constraints allows us to determine whether the placement is routable or not. To find a routable placement, we iterate the generation of placements and satisfiability checks. If a placement is proved to be unroutable, we add the new clause which suppresses the previous placement to CNF to generate a new placement. The algorithm of our SAT-based cell generation is described in Fig. 4.

```
Given: a spice netlist
Procedure cell_gen
  W = #N-MOS ( = #P-MOS)
  Generate a CNF P for placement at W.
  while P is UNSAT
    W++
    Generate a new CNF P for placement at W.
  end while
  Generate a CNF R for intra-cell routing.
  while R is UNSAT
    if there is no possible placement at W
      W++
    end if
    Generate a new placement at W.
    Generate a new CNF R for intra-cell routing.
  end while
end Procedure
```

Fig. 4. Our cell generation algorithm

V. Experimental Results

The formulations explained in section III and IV and the algorithm described in Fig. 4 enables us to generate a minimum-width routable cell layout from a spice netlist via Boolean Satisfiability. To test the effectiveness of our SAT-based cell synthesis method, we compare it with the 0-1 ILP-based transistor placement and the commercial cell generation tool.

TABLE II

The comparison results of the transistor placement

Circuit			Problem Size				Run Time (sec.)				
			ILP		SAT		ILP		SAT		
name	#tr.	#col.	#var.	#ineq.	#var.	#cla.	OPBDP	CPLEX	OPBDP	CPLEX	Chaff
ao222	14	8	1054	1926	56	1624	0.59	134.17	1.30	14.17	**0.15**
ao44	20	11	2767	5450	100	5360	1551.45	>3600	20.36	>3600	**0.66**
aoi211	8	4	241	394	24	244	0.07	0.04	0.03	0.06	**0.01**
fad1	28	15	7011	14082	140	47404	>3600	>3600	>3600	>3600	**201.05**
gen3	16	10	1509	2836	80	3342	9.95	2872.81	11.6	2150.69	**1.67**
had1	14	9	1054	1892	70	2762	2.78	456.63	10.17	687.35	**1.33**
mux2	12	8	699	1240	48	1260	0.39	37.74	0.92	20.12	**0.23**
nand2	4	2	39	56	8	18	0.03	~**0**	0.05	0.02	0.03
nand3	6	3	114	178	18	94	0.04	0.02	0.03	0.03	**0.01**
xnor2	10	6	432	716	40	790	0.17	1.56	0.30	1.91	**0.08**
Total Ratio	—	—	—	—	—	—	>33.04	>71.18	>17.72	>68.21	1.00

A. Comparison with the 0-1 ILP Formulation

We conducted the run-time comparison with 0-1 ILP formulation in transistor placement stage. To conduct the experiment, we also transformed the transistor placement problems into the 0-1 ILP problems. Our formulation of the ILP is based on the Gupta and Hayes' formulation[4]. The differences are that we assume only one dimensional placement and P-/N-MOSFETs with the same gate nodes can be aligned vertically, while they assumed two dimensional placement and only complementary P-/N- MOSFETs can be aligned vertically. The details of the ILP formulation are abbreviated in this paper due to limitations of space.

We used the CNF-SAT solver Chaff[8], the 0-1 ILP solver OPBDP[1], and the generic ILP solver CPLEX 8.1[6] for the experiments. The Chaff and OPBDP experiments were conducted on a 750 MHz UltraSPARC-III workstation with 2 GB of RAM. The CPLEX experiments were done on a Pentium-III 1 GHz with 2 GB of RAM. We used default settings for all of them. A time-out limit of 3,600 seconds was used for each run. We experimented the ILP problems and the ILP problems simply transformed from CNF for the SAT solver as the input files for the two ILP solvers. Table II lists the results of solving the transistor placement problems by SAT and ILP. Although Table II contains only 10 circuits, we tested 30 static dual CMOS logic circuits in a standard-cell library. The data in the row of Total Ratio means the ratio of the total run times for the 30 circuits. For each circuit the table indicates the number of transistors, the resultant number of columns, the problem size for the ILP as well as for the SAT formulations (number of ILP variables and inequalities are expressed as #var. and #ineq., CNF variables and clauses are as #var. and #cla.), the OPBDP and CPLEX run times using the ILP formulation, and OPBDP, CPLEX, and Chaff run times using the SAT formulation as the input. The problem size of SAT formulation is the size of the problem which have a satisfiable assignment first.

Compared with the two ILP solvers, the SAT solver Chaff has shorter run times in most cases. Moreover, the ILP solvers can not solve some large circuits in one hour whereas Chaff can in minutes. The total run times of the ILP solvers are about 17 to 70 times longer than the SAT solver. These results clearly shows that the SAT formulation and the SAT solver is more suitable for solving the transistor placement.

The layouts generated by our method are equal to or smaller than the one dimensional layout generated by Gupta and Hayes' method[3] which treats complementary P-/N-MOSFETs pair. For example the width of the full adder fad1 in Table II is 15 using our method whereas the minimum width of the one dimensional layout of the full adder described in [3] is 16.

B. Comparison with the Commercial Cell Generation Tool

We also compared our cell generation method with the commercial cell generation tool ProGenesis 4.2[9]. The experiments were conducted on a 750 MHz UltraSPARC-III workstation with 2 GB of RAM. Table III lists the results of generating the 32 static dual CMOS logic circuits in a standard-cell library by our method and ProGenesis. The height of all circuits is fixed to 10 rows in this experiment. The table shows the number of transistors, the SAT problem size of the intra-cell routing, the resultant width of cells generated by the ProGenesis and our method, and the CPU times spent on generating a routable cell layout from a netlist. In the column of resultant width, we assumed the 0.35 μm process technology. The layouts generated by ProGenesis and our method are both design rule correct. The asterisk in the column width for ProGenesis means that it used second metal layers to route these circuits, and the column #col. lists the number of columns needed for each circuit to complete the intra-cell routing using our layout style. One column is added to the minimum width placement of ao33, ao44, oa33, and oa44 to complete the intra-cell routing of each circuit as shown in Table II and III. This results means that the minimum width layouts without the routability are not the minimum width layouts when some styles of the intra-cell routings are defined.

ProGenesis generates smaller width layouts for 18 circuits, because it can use the bending gate in G-region and the smaller width SOURCE/DRAIN diffusion if it has no contact on it. However, it uses the second metal layers to complete the intra-cell routing for fad1, gen2, gen3 and the width of fad1, gen3, had1, maj3, nand22, nor4 are larger than our method. The run times of our method are smaller than ProGenesis for almost all cases. Our method can generate all these 32 circuits in total 54 % run times with only 3 % area increase compared with ProGenesis. The snapshot of fad1 layout is illustrated in Fig 5 for example.

TABLE III

THE COMPARISON RESULTS OF THE CELL SYNTHESIS

Circuit		SAT		Resultant Width			Run Time (sec.)	
		Problem Size		ProGenesis	Proposed		ProGenesis	Proposed
name	#tr.	#var.	#cla.	width (μm)	width (μm)	#col.		
ao222	14	14	337	**11.85**	13.20	8	144.96	**1.06**
ao33	16	20	1102	**13.95**	15.90	*10*	**185.07**	392.26
ao44	20	20	1420	**15.50**	18.90	*12*	**291.79**	637.26
aoi21	6	5	14	5.40	5.40	3	28.51	**0.04**
aoi211	8	7	16	**6.45**	6.90	4	52.06	**0.02**
buf1	4	4	49	**3.90**	4.20	2	18.67	**0.06**
eno	10	8	144	8.40	8.40	5	77.67	**0.04**
eor	10	8	144	8.40	8.40	5	63.54	**0.06**
fad1	28	36	6169	26.05*	**24.00**	15	509.27	**305.76**
gen2	12	14	295	**11.00***	11.70	7	141.17	**0.16**
gen3	16	20	520	21.40*	**16.20**	10	386.69	**2.20**
had1	14	25	2184	17.90	**15.00**	9	236.1	**18.35**
inv	2	2	2	2.40	2.40	1	10.35	**0.05**
maj3	12	14	215	10.80	**10.20**	6	87.93	**0.20**
mux2	12	22	900	**10.60**	13.20	8	77.28	**6.42**
nand2	4	4	4	**3.70**	3.90	2	16.56	**0.04**
nand22	6	6	81	5.80	**5.70**	3	27.02	**0.04**
nand3	6	4	7	5.40	5.40	3	25.91	**0.04**
nand4	8	4	7	6.90	6.90	4	33.84	**0.04**
nand44	10	8	150	**8.30**	8.70	5	76.76	**0.06**
nor2	4	4	4	3.90	3.90	2	16.73	**0.05**
nor22	6	6	81	**5.40**	5.70	3	30.77	**0.05**
nor3	6	5	8	**5.10**	5.40	3	26.53	**0.06**
nor4	8	5	8	8.10	**6.90**	4	40.98	**0.07**
nor44	10	8	150	**8.10**	8.70	5	62.03	**0.07**
oa222	14	14	335	**12.00**	13.20	8	167.45	**1.07**
oa33	16	17	1095	**13.25**	15.90	*10*	**204.5**	443.94
oa44	20	20	1506	**15.70**	18.90	*12*	295.87	**116.41**
oai21	6	7	18	5.40	5.40	3	30.03	**0.04**
oai211	8	7	18	**6.55**	6.90	4	54.60	**0.05**
xnor2	10	14	229	**8.80**	10.20	6	55.43	**0.09**
xor2	10	14	219	**8.65**	10.20	6	59.00	**0.12**
Total	—	—	—	305.5 (1.00)	315.9 (1.03)	—	3535.07 (1.00)	1926.18 (0.54)

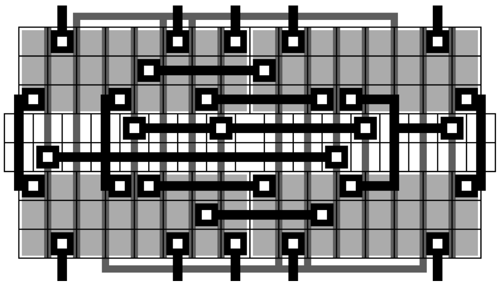

Fig. 5. The resultant cell layout of "fad1"

VI. CONCLUSIONS

We proposed a high speed layout synthesis method for the minimum-width CMOS logic cells via Boolean Satisfiability. In this paper, cell synthesis problems are first transformed into SAT problems by our formulations. We showed that the SAT formulation is more suitable for the transistor placement by comparing the run time of the SAT and the ILP formulations of it. We also showed that the width of placements generated by our method are smaller than that of previous method using our layout styles. Our method generates the cell layouts of 32 static dual CMOS logic circuits in 54 % run times with only 3 % area increase compared with the commercial cell generation tool ProGenesis. We think that our method can significantly shorten time-to-market of a cell library and can also be useful for many other applications such as on-demand cell synthesis.

ACKNOWLEDGEMENTS

This work is supported by VLSI Design and Education Center(VDEC), University of Tokyo.

REFERENCES

[1] P. Barth, "OPBDP: A Davis-Putnam Based Enumeration Algorithm for Linear Pseudo-Boolean Optimization," *http://www.mpi-sb.mpg.de/units/ag2/software/opbdp/.*

[2] S. Devadas, "Optimal Layout Via Boolean Satisfiability," *in Proc. IEEE/ACM Int. Conf. on Computer Aided Design*, pp. 294–297, 1989.

[3] A. Gupta and J. P. Hayes, "Width Minimization of Two-Dimensional CMOS Cells Using Integer Programming," *in Proc. IEEE/ACM Int. Conf. on Computer Aided Design*, pp. 660–667, 1996.

[4] A. Gupta and J. P. Hayes, "CLIP: An Optimizing Layout Generator for Two-Dimensional CMOS Cells," *in Proc. ACM/IEEE 34th Design Automation Conference*, pp. 452–455, 1997.

[5] M. Guruswamy, et al., "CELLERITY: A Fully Automatic Layout Synthesis System for Standard Cell Libraries," *in Proc. ACM/IEEE 34th Design Automation Conference*, pp. 327–332, 1997.

[6] ILOG, CPLEX, *http://www.ilog.com/products/cplex/.*

[7] R. L. Maziasz and J. P. Hayes, "Exact Width and Height Minimization of CMOS Cells," *in Proc. ACM/IEEE 28th Design Automation Conference*, pp. 487–493, 1991.

[8] M. W. Moskewicz, et al., "Chaff: Engineering an Efficient SAT Solver," *in Proc. ACM/IEEE 38th Design Automation Conference*, pp. 530–535, 2001.

[9] prolific, ProGenesis, *http://www.prolificinc.com/progenesis.html.*

[10] T. Uehara and W. M. vanCleemput, "Optimal Layout of CMOS Functional Arrays," *IEEE Trans. on Computers*, vol. C-30, pp. 305–312, May 1981.