HALO : An Efficient Global Placement Strategy for Standard Cells

(HALO : 효율적 표준셀 배치 알고리듬)

梁 榮 日*, 慶 宗 旻*

(Yeung Yil Yang and Chong Min Kyung)

要 約

이 논문에서는, 셀 순서 결정 방향을 번갈아 가며 선형 정렬 방법을 계층적으로 적용하여 회로의 2차원 배치를 얻어내는 HALO라는 효율적 셀배치 알고리듬을 제안하였다. HALO가 min-cut이나 FDR에 근거를 둔 방법들보다 좋은 결과를 얻을 수 있는 이유를 이론적으로 설명하였다. HALO를 표준셀 배치를 위한 프로그램으로 실현하였고, 셀의 수가 752와 2907개인 성능 비교 회로 *primary* 1과 *primary* 2에 대하여 반 주변 배선길이를 비교하였을 때 지금까지 얻은 가장 좋은 결과(참고 문헌 6)보다 각각 7% 와 24% 적은 배치 결과를 얻어냈다.

Abstract

This paper describes an efficient global cell (module) placement strategy called HALO (Hierarchical Alternating Linear Ordering) which generates global 2-D placement of circuit modules by hierarchical application of linear ordering in alternating direction. We tried, in principle, to explain why HALO *should* perform better than other typical, somehat successful, analytical approaches such as min-cut, force-directed relaxation (FDR) or its likes. We have implemented HALO as a program for standard cell placement. Experimental results on two benchmark circuits, *primaryl and primary 2* consisting of 752 and 2907 cells, respectively have shown a decrease of half-perimeter routing length by 7% and 24%, respectively compared to the best available results[6] obtained so far. Total CPU time including the following detailed placement was less than half of the earlier work[6]

I. Introduction

The problem of finding optimal placement of circuit modules in 2-D space with various objectives such as chip area, wire length, etc. has been tackled by many approaches. As the number of circuit modules (cells) in a circuit is drastically increased, IC designers are now in greater demand of a fast, high-performance placement packages than ever before. Simulated annealing has shown a better performance than other heuristic appro-

*正會員, 韓國科學技術院 電氣 및 電子工學科
(Dept. of Electrical Eng., KAIST)
接受日字 : 1989年 6月 19日

aches for the standard cell placement, but only with enormous amount of CPU time.[1] other heuristic algorithms such as min-cut[2,3], FDR (force-directed relaxation)[4], and its variations[5,6,7,8] have been quite successful in itself, and have helped us in understanding the interal mechanism of the 2-dimensional circuit placement process. We will briefly mention the drawbacks of these heuristic algorithms, and explain later how these are suppressed in our algorithm, HALO (hierarchical alternating linear ordering). Min-cut is basically a method for hierarchically confining the modules in each compartment until each includes only one module. The only criteria of dividing the whole modules into two groups is just number of cuts, ignoring the global connectivity among modules. Therefore, it is a pure top-down approach where the decision in the upper-level is made with very little information, or anticipation on the lower level decision. On the other hand, FDR tries to *order* the modules directly in 2-D space by using spring force relaxation analogy. The resultant placement, therefore, *should* be more global than the result of min-cut. However, most experimental results have shown the superiority of min-cut. The reason is due to the fact that FDR completely ignores the size of the modules, while min-cut considers the size information. There are some variational forms of FDR reported which have tried to take the module size into consideration while still preserving the FDR's inherent nature of 'globality'. For example, Tsay *et. al*[8] proposed performing the FDR processes in each of the partition of the whole chip with some coordination between them by using the concept of 'floating pin' which is similar to 'terminal propagation'[3] in min-cut. A closer coordination between two FDR processes in each chip partition using constrained optimization technique was reported in the name of GORDIAN by Kleinhans *et. al.* [6] The experimental results of GORDIAN[6] on the benchmark circuits are better than any other published results performed on the same circuits.

In this paper, we propose an algorithm called HALO (hierarchical alternating linear ordering) for the global placement of standard cells. (For convenience, the whole placement process is divided into global placement where relative module positions are obtained while some inter-module overlaps are allowed, and the detailed placement where such overlaps are removed) In section II, we describe the algorithm HALO and explain why HALO should outperform other heuristic algorithms such as min-cut and FDR and their variations, especially the GORDIAN.[6] We have implemented HALO in our program for standard cell placement. The remaining procedures in the standard cell placement, i.e., row assignment, feedthrough cell assignment (we chose to perform feedthrough assignment during the placement, rather than during the global routing process in other cases to minimize the effect of the feedthrough cells on the eventual widths of each row.) and the intra-row cell assignment will be explained briefly in section III. Finally experimental results are given with discussions in section IV.

II. HALO (Hierarchical Alternating Linear Ordering)

The overall procedure for the standard cell placement using HALO as a global placement strategy is shown in Fig. 1. HALO receives the circuit connectivity as its input data and calculates the center-of-mass positions of the circuit modules. After some number (typically 2 or 3) of iterations of row assignment and feedthrough

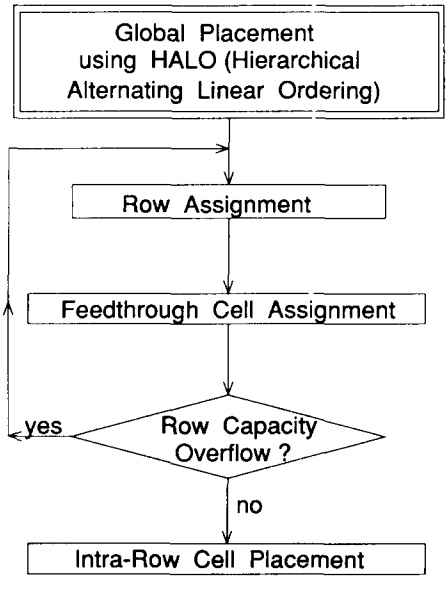
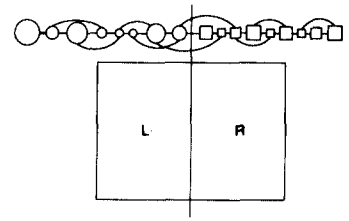
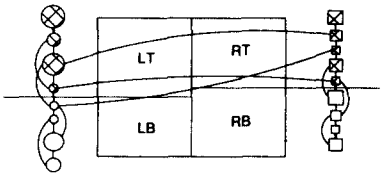


Fig.1. Overall standard cell placement flowchart using HALO for global placement.

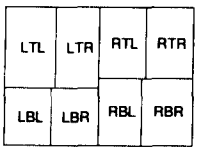
cell assignment, the whole process is finished with the intra-row cell placement. Fig. 2(a), (b), (c) and (d) show, in succession, how the cells are confined to each of the two (not necessarily two, but it can be three, depending on the aspect ratio of the partition in question) subregions using linear ordering. The criteria used in the linear ordering of cells will be explained later in this section. Let us assume that the cells are linearly ordered as shown in Fig. 2(a). We divide the chip area into two subregions such that each can accommodate each (left and right, in this case of horizontal partition) group of cells. Usually the two groups are divided such that the sum of module areas in each group are approximately the same. In Fig. 2(a), the modules represented as circles are permanently assigned to L-partition, so do those represented as boxes to R-partition. It is very important to note that in dividing the modules into two partitions as in Fig. 2(a), all the circuit connectivity information has been reflected. This needs to be contrasted to min-cut which only minimizes the number of cuts. Compared with FDR-based approaches,[6,8] the size of the modules are taken into account. In GORDIAN[6], the module sizes are introduced too late, i.e., only after the dot (of size zero) distribution using constrained optimization ignoring the module size is obtained. Moreover, the constrained relaxation of module positions in 2-D is redundant, because only the ordering in one direction is utilized in performing the partition. In HALO, module size information is taken into account *simultaneously*, not *posteriori*. Once the modules are assigned to L- and R-partition using the horizontal ordering which we call H1-ordering (the 1st ordering in horizontal direction), V1-ordering is performed as shown in Fig. 2(b), in both the L- and R-partition simultaneously. The connectivity among modules still exist, although the modules belong to different partitions. After V1-ordering, the shaded circles and shaded boxes are confined to LT- and RT-partition, respectively, while empty circles and empty boxes are confined to LB- and RB-partition, respectively. Fig 2(c) shows the result of H2-ordering in a similar way. Let us assume now that all the cells are V2-ordered, and we want to perform H3-ordering. Fig 2(d) shows a snapshot during the H3-ordering process. The cells in the shaded region are already H3-ordered and those in the right side of the multi-frontal (MF) sweep line are V2-ordered. (H3-ordered cells are already ordered in the H1, V1, H2, V2 and H3-ordering, while V2-ordered cells are ordered in the H1, V1, H2 and V2-ordering.)



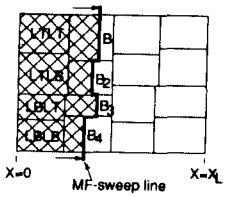
(a) H1-ordering



(b) V1-ordering



(c) H2-ordering



(d) V2-ordering is finished. H3 ordering is in progress using multi-frontal (MF) sweep line.

Fig.2. HALO procedure.

As shown in Fig. 3, the cells are classified into three groups during the cell-ordering, ORDERED, ACTIVE and UNORDERED. Initially all the cells are in UNORDERED and the ordered cells are moved, one by one, into ORDERED. The cells which are not ordered yet but have common nets with the cells in ORDERED are moved into

ACTIVE. Only the cells in ACTIVE are the candidates to be selected as the next cell in the linear order. Nets also can be classified into three groups during the cell-ordering. i) *New net* is connected to none of the cells in ORDERED. ii) *Terminating net* is the net having connection between only one cell in ACTIVE and one or more cells in ORDERED. iii) *Continuing net* is the net having connection among one or more cells in ORDERED and at least two cells in ACTIVE.

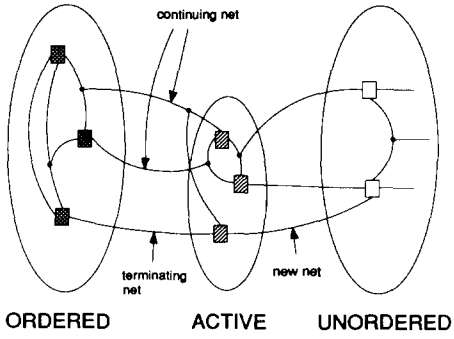


Fig.3. The status of the cells during the cell-ordering.

Algorithm 1 is a heuristic called Finding-Cell-Order for finding the orders of the cells. The inputs to this procedure are a set of regions and the cell-ordering direction. Each cell is assigned to one of the regions. The area of the regions is equal to the area summation of the cells asssigned to those regions. During the cell-ordering, the regions are classified into three groups. All the cells in ORDERED REGION are ordered. If none of the cells in a region are ordered, the region in UNORDERED REGION. The region some of which are ordered is ACTIVE REGION. In Fig. 2(d), the number of regions is 16 and the cell-ordering direction is horizontal. We define the ith point B_i as the x-value of the vertical interface line which is the boundary between the ordered cells and unordered cells in the ith region as shown in Fig. 2(d). Initially all the B_i s are set to the leftmost x-value (lowermost y-value) of the regions, if the cell-ordering direction is horizontal (vertical). A cell assigned to the region having the mimimum B_i is selected, one by one, for the ordering according to the cell selection criteria and B_i is updated. Followings are the cell selection criteria in the order of the priority.

1) Select a cell c_i such that the number of nets connecting the ORDERED and the other groups is minimized after the movement of the cell c_i. (The difference between *terminating nets* and *new nets* of the cell c_i is maximum.)
2) Select a cell which has the most number of *terminating nets.*
3) Select a cell which has the most number of *continuing nets.*

Algorithm: Finding-Cell-Order
Inputs: a circuit, a set of regions, a cell-ordering direction
Outputs : Linear order of the cells
Step 1) Set ORDERED, ACTIVE, and ACTIVE-REGION empty. Insert all the cells into UNORDERED and calculate B_i for the regions. Insert regions r_i's into ACTIVE-REGION, which have the minimum B_i.
Step 2) Select the regions r_i's from ACTIVE-REGION, whose length is minimum (B_4 in Fig. 2(d)) and then select a cell c_i among the cells assigned to the region r_i. If the region r_i has no cells in ACTIVE, then select the cell c_i which has the least connections with other cells, else select a cell c_i in ACTIVE according to the cell selection criteria mentioned above. Move a cell c_i into ORDERED denoting that the cell c_i is ordered. Move all the cells c_j's from UNORDERED to ACTIVE, such that c_j is connected to c_i. The length of the region r_i, B_{ri}, is increased by the amount of the area occupied by the cell c_i divided by the height (width) of the r_i if the cell-ordering direction is horizontal (vertical). If there are unordered cells in the region r_i, go to step 2.
Step 3) Remove the region r_i from ACTIVE-REGION and insert the regions r_j's to ACTIVE-REGION, such that the region r_j is adjacent to the region r_i through the right-side (upper-side) of the region r_i when the cell-ordering direction is horizontal (vertical). If ACTIVE-REGION is empty, then STOP, else go to step 2.

Algorithm 1. An algorithm to find the orders of the cells

III. Detailed Placement

Eaxh cell is located in the approximate positions where it ought to be as a result of HALO procedure. To obtain the final placement of standard cells, the cells should be assigned to their rows and positioned within the row. Besides, feedthrough cells should be added if necessary. In this section, the procedure for obtaining the final standard cell layout from the result of HALO, the global placement is briefly explained.

Feedthrough cells in the standard cell layout are added to complete the connections for a net. Fig. 4 shows the center-of-mass positions of the cells with the row assignment where the shaded cells have a common net n_i. The ith (kth) row is the nearest row to the jth row among the rows above (below) the jth row, which has cells connected to a net n_i. A feedthrough cell f_i is introduced in the jth row for the complete connection of the net n_i, and positioned in the jth row according to the distribution pattern of the cells connected to the net n_i in ith row and kth row. Algorithm 2 is a heuristic called Row-Assignment for assigning the cells to the rows. In the procedure, the iterations from step 1 to step 3 are necessary to make the widths of the rows uniform, because it is hard to predict the number of feedthrough cells required after the row assignment. In each iteration, the widths of the rows are calculated as the summation of the widths of the cells assigned in the current iteration and the widths of feedthrough cells required in the row assignment of the previous iteration.

Algorithm: Row-Assignment

Input: center-of-mass positions of the cells, the result from HALO

Step 1) Sort the cells in y-value non-increasing order and put this sorted list into CELL-LIST. For each row r, set n_r (the number of the feedthrough cells assigned to the row r) and fl_r (the flag to determine whether the row r is full or not) to zero and EMPTY, respectively.

Step 2) For a front cell c_i of CELL-LIST, find the row r such that r is the nearest row to the cell c_i and fl_r is EMPTY. If the summation of the widths of the cells assigned to the row r and the width of a feedthrough cell multiplied by n_r is enough to fill the row r, set fl_r is FULL. Delete the cell c_i from CELL-LIST. If there are cells in CELL-LIST, go to step 2.

Step 3) For each row r, set the number of the feedthrough cells required in the row r to n_r. Calculate the width of the row which is the summation of the width of the cells and the feedthrough cells. If the difference between the minimum width of the row and maximum width of the row is greater than the specified value, go to step 1.

Step 4) For each feedthrough cell f_i, determine the position of the feedthrough cell f_i. STOP.

Algorithm 2. An algorithm for the row assignment and feedthrough cell assignment

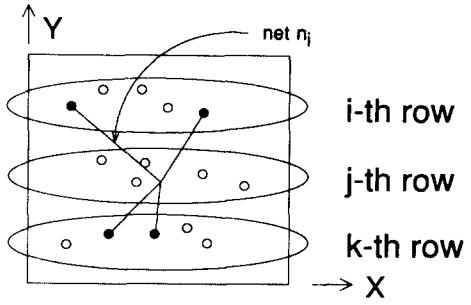


Fig.4. The center-of-mass positions of the cells (the result from HALO) with row assignment, where shaded circles are the cells connected to a net n_i.

The cells are assigned to their rows upon completing the row assignment. In order to determine the positions of the cells within the row, the heuristic which is modified from the cell-ordering described in the previous section. To obtain the final standard cell layout, the global routing algorithm proposed in Ref. 11 and greedy channel router[12] are used.

The computation time of the proposed algorithm, HALO, is bounded by a polynomial time, $O(n(\log n)^2)$, where n denotes the number

of the cells of a circuit. Fig. 5 shows the computation time of the proposed algorithm as a function of the number of the cells. The experiments show that the proposed algorithm has the O(n log n)-time behavior.

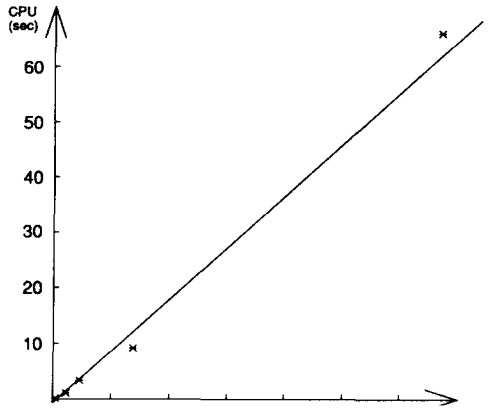


Fig.5. The computation time versus the circuit size (the number of the cells), where the number of the cells are scaled by n log n. The numbers of the cells of the circuits used in the experiments are 67, 144, 270, 752, and 2907.

IV. Experimental Results and Discussions

The proposed algorithm, HALO (hierarchical alternating linear ordering) is implemented as a global placement package for standard cells using C language in SUN4/UNIX.

Table 1 shows the characteristics of the example circuit. The first two columns represent the number of the cells and nets. The last column represents the number of the circuit's pads, i.e., the number of inputs plus the number of outputs. Table 2 shows the comparison of the placement results among four different methodologies in terms of the number of routing tracks and the number of feedthrough cells. In Table 2, the first two represents the classical placement procedure based on the pairwise interchange (PI) after the linear ordering and folding (LOF) and the generalized force directed relaxation (GFDR) after the linear ordering and folding. The third methodology represents the placement procedure based on constrained multi-stage graph model (CMSG)[13] and the last is the proposed algorithm, HALO. From Table 2, we can see that the proposed algorithm requires the smaller number of routing tracks and feedthrough cells compared to the first two methodologies and the smaller number of routing tracks compared to the third methodology.

Table 1. Statistical data for the test circuits.

Citcuit		Number of cells	Number of nets	Number of input/output
1	ALU	67	81	22
2	COM	144	166	25
3	CAL	270	301	22

Table 2. Performance comparison for the circuits in Table 1.

circuit		LOF+PI	LOF+GFDR	CMSG	HALO
ALU (4rows)	Total number of channels	34	33	32	28
	Number of feedthroughs	7	5	0	2
COM (5rows)	Total number of channels	46	45	49	45
	Number of feedthroughs	5	5	0	2
CAL (6rows)	Total number of channels	82	72	70	65
	Number of feedthroughs	50	19	0	9

Table 3 shows the characteristics of the benchmark circuits, *primary1 and primary2*, in Ref. 10. Table 4 shows the comparison of the half-perimeter routing length of the benchmark circuits. The cells of the circuits, *primary1 and primary2*, were placed in 17 and 26 rows, with estimated channel widths of 220*um* and 270*um*, respectively The data of the first three methods, i) min-cut with terminal propagation,[3] ii) the relative/transportation method[14], and iii) GORDIAN[6]

are obtained from Ref. 6. The half-perimeter routing length of the proposed algorithm is calculated with (or without) inserting feedthrough cells. The value in the parenthesis is the half-perimeter routing length without the insertion of feedthrough cells. As shown in Table 4, the half-perimeter routing length of the proposed algorithm for the circuits,[10] *primary1 and primary2*, is less than that of GORDIAN[6] about 7% and 24%, respectively. HALO still has resulted in 4% and 17% improvement compared to Ref. 6, even after including the feedthrough cells. Table 5 shows the placement results for the circuits in Table 3. Fig. 6 shows the final layout for the placement of the circuit called primary1 in 16 rows. In the placement of standard cells of the circuits, *primary1 and primary2*, feedthrough pins within a cell were not used.

Table 3. Statistical data for the benchmark circuits.

circuit	Number of cells	Number of nets
primary 1	752	904
primary 2	2907	3029

Table 4. Half-perimeter routing length comparison for the circuits in Table 3, where the value in the parenthesis is the half-perimeter routing length when feedthrough cells are not inserted.

algorithm	half-perimeter routing length[m]	
	primary 1	primary 2
Min-Cut	1.739	9.823
RT	2.177	8.685
GORDIAN	1.503	8.142
HALO	1.439 (1.397)	6.73 (6.169)

Table 5. Placement results for the circuits in Table 3.

Circuit	Number of rows	Number of feedthroughs	Total number of channels
primary 1	16	576	300
primary 1	17	613	319
primary 2	26	3720	993

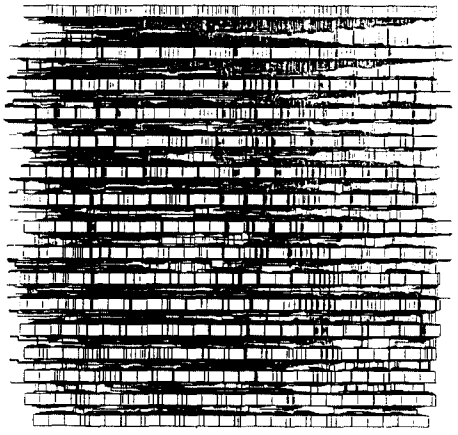


Fig.6. A final layout of the standard cell placement of the circuit called primary1 (first circuit in Table 3).

In conclusion, we proposed a fast heuristic algorithm, HALO for the placement of standard cells, which should outperform in principle and have outperformed previous heuristic algorithms in many experiments. The CPU time for the *primary1* and *primary2*, run on SUN4 (7MIPS machine) were 9.43 sec and 65.50 sec, respectively, which shows that the proposed HALO approach has merits in the sense of CPU time over the previous approaches.

Referecnes

[1] C. Sechen and A. Sangiovanni-Vincentelli, "Timber Wolf 3.2:A new standard cell placement and global routing package," *Proc. 23rd Design Automation Conference*, pp. 423-439, June 1986.

[2] M. Breuer, "Min cut placement," *J. Design and Fault Tolerant Computing*, 1, (4), pp. 343-363, Oct. 1977.

[3] A.E. Dunlop and B.W. Kernighan, "A procedure for placement of standard-cell VLSI circuits," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 4, no. 1, pp. 92-98, Jan. 1985.

[4] N.R. Quinn and M.A. Breuer, "A force-directed component placement procedure for PCB's," *IEEE Trans. on Circuits and Systems*, vol. 26, no. 16, pp. 663-670, June 1979.

[5] G.J. Wippler, M. Wiesel, and D.A. Mlynsky, "A combined force and cut placement for hierarchical VLSI layout," *Proc. ACM/IEEE 19th Design Automation Conference*, pp. 617-677, 1982.

[6] J.M. Kleinhans, G. Sigl, and F.M. Johannes, "GORDIAN: A new global optimization/rectangle dissection method for cell placement," *IEEE Int'l Conference on Computer-Aided Design, ICCAD-88*, pp. 506-509, 1988.

[7] C.K. Cheng and E.S. Kuh, "Module placement based on resistive network optimization," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. CAD-3, no. 3, pp. 218-225, July 1984.

[8] R.S. Tsay, E.S. Kuh and C.P. Hsu, "Module placement for large chips based on sparse linear equation," *Int'l J. of Circuit Theory and Applications*, vol. 16, pp. 416-423, 1988.

[9] Sungho Kang, "Linear ordering and application to placement," *Proc. ACM/IEEE 20th Design Automation Conference*, pp. 457-463, 1983.

[10] B. Preas and K. Roberts, *Physical Design Workshop*, Hilton Head, South Carolina, 1987.

[11] Jinsheng Cong and Bryan Preas, "A new algorithm for standard cell global routing," *IEEE Int'l Conference on Computer-Aided Design, ICCAD-88*, pp. 176-179, 1988.

[12] R.L. Rivest and C.M. Fiduccia, "A greedy channel router," *Proc. ACM/IEEE 19th Design Automation Conference*, pp. 418-424, 1982.

[13] H.G. Cho and C.M. Kyung, " A heuristic standard cell placement algorithm using constrained multi-stage graph model," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 7, no. 11, pp. 1205-1214, Nov. 1988.

[14] K.M. Just, J.M. Kleinhans, and F.M. Johannes, "Relative Placement and the Transportation Problem for standard-cell layout," *Proc. ACM/IEEE 23rd Design Automation Conference*, pp. 308-313, 1986.

著 者 紹 介

梁 榮 日 (正會員)

1960年 3月 3日生. 1983年 2月 경북대학교 전자공학과 학사학위 취득. 1985年 2月 한국과학기술원 전기 및 전자공학과 석사학위 취득. 1988年 8月 한국과학기술원 전기 및 전자공학과 박사학위 취득. 현재 한국과학기술원 연수 연구원. 주 관심분야는 CAD, VLSI architecture 및 설계등임.

慶 宗 旻 (正會員) 第25券 第10號 參照

현재 한국과학기술원 전기 및 전자공학과 부교수