# A Parallel Slice Maze Router.

Hesham Keshk, Shin-ichiro Mori,
Hiroshi Nakashima, and Shinji Tomita

Department of Information Science
Kyoto University, JAPAN.
Tel. +81-75-7535393 Fax +81-75-7535379
{keshk,moris,nakasima,tomita}@ kuis.kyoto-u.ac.jp

## Abstract

Parallel computers are used to overcome the long computation time and the large memory size required for automated wire routing. We introduce a parallel router consisting of two parts: "global routing," and "detailed routing." In the global routing, the whole grid is divided to partitions, and it is required to determine in which partitions each net will be routed. In the detailed routing, one processor element (PE) determines the nets which have a non overlapping global paths. The other PEs route these nets in parallel within their own grid partitions.

We introduce a new way of dividing the grid to layers, and dividing layers to slices. Each PE, in the detailed routing, has a responsibility for one or more slice. The program is implemented on AP1000 using MCM benchmarks data. The result shows that this way of division could obtain a high degree of parallelism. This algorithm is suitable for inner vias technology.

## 1 Introduction

The efficiency of parallelism depends on how to divide the grid to partitions, and how to assign these partitions to the PEs. In this paper we introduce a new way of dividing the grid to slices to decrease the communication cost between PEs. Some researches divide the grid to square partitions with equal areas, and every PE will be responsible for one partition. For example, If we have 2 layers grid with dimensions 1000 × 1000 and there are 16 PEs, then each partition will be 2 × 250 × 250 as shown in Fig1. To route a network (Fig 2a), global routing detects firstly in which partitions the nets will pass (Fig 2b). For parallel detailed routing, some researches [11] [12] locates virtual terminals on the boundaries of the partitions (Fig 2c). Then each PE can route the part of the path located inside its partition independently on the other PEs(Fig 2d). This way is fast and all PEs, which share routing one net, can work simultaneously without any dependence or communication between them. But, the main disadvantage of this method is as follows: As it is very difficult to find the ideal virtual terminals (Fig 2e), sometimes undesired paths (long with many vias and bends) will be obtained due to the improper choices of virtual terminals (Fig 2f).

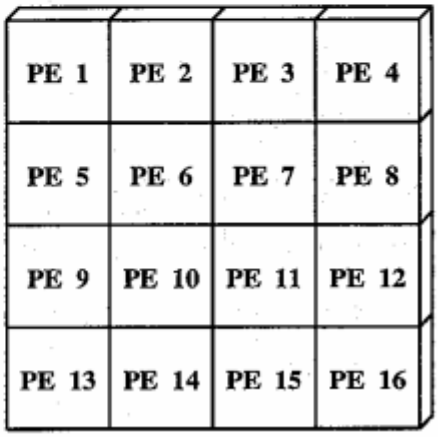


Figure 1: Divide grid to partitions.

Some other researches route the net in a sequential manner (one partition after the other). The PE, which is responsible for the partition

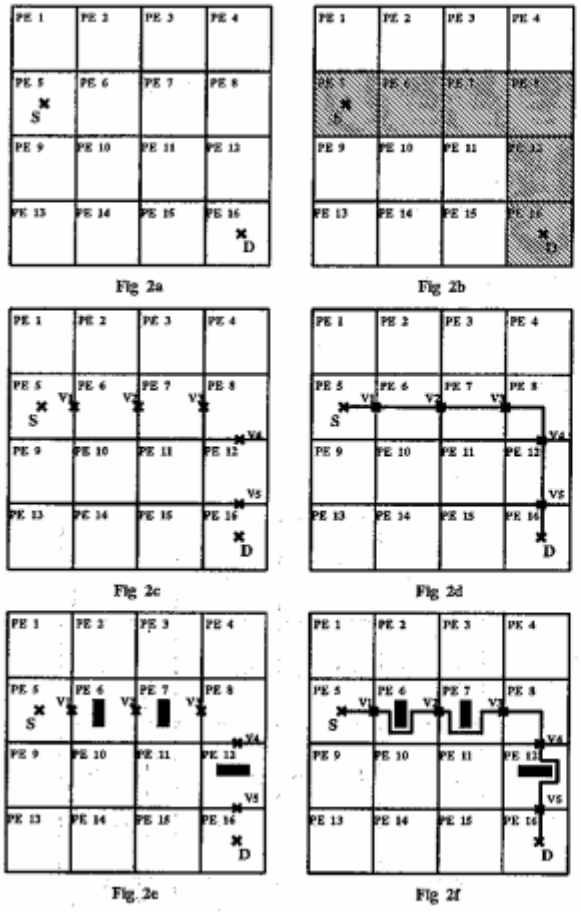


Figure 2: Virtual terminals.

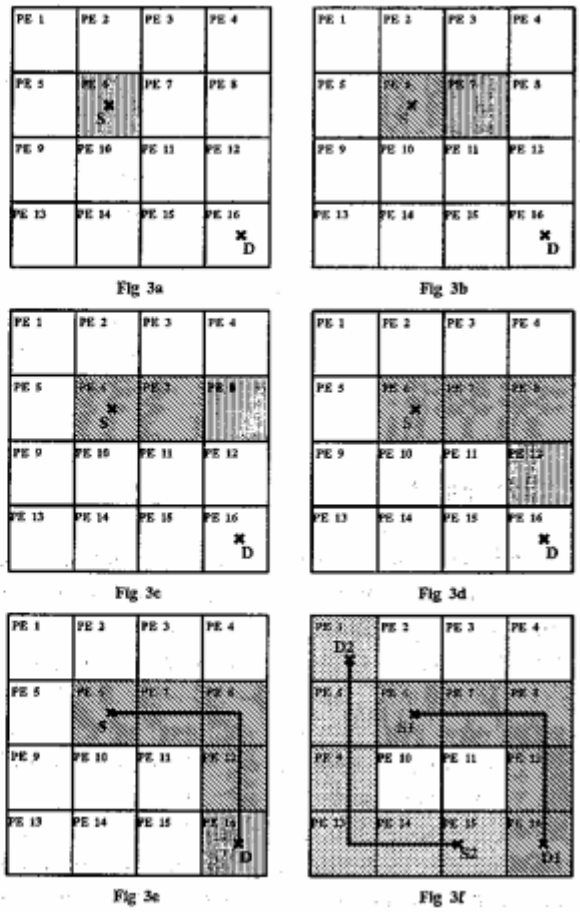


Figure 3: Sequential partitions routing.

which contains the source, starts propagation until it reaches to the boundary of its partition(Fig 3a). Then, this PE sends the cost of this boundary to its neighborhood which continues propagation until it reaches to the boundary of the next partition(Fig 3b). This step is repeated (Fig 3c,3d) until the destination point is reached (Fig 3e). This method can obtain good net path but it uses many messages per net and the parallelization is poor. The number of messages is proportional with net length. To improve the parallelism, the propagation can be started from all net terminals simultaneously. Also, more than one net can be routed at the same time (Fig 3f) if their net paths do not share in any partition (global paths has no intersection).

In this paper, we divide grid to layers, divide each layer to slices, and assign one or more slice to each detail PE. Dividing grid to layers allows many nets to be routed simultaneously, and dividing each layer to slices decreases the communication cost. The number of messages is proportional to number of vias (global bath bend) instead of net length. Our router consisting of two parts: "global routing," and "detailed routing." both global and detailed routing are done simultaneously using different PEs.

## 2 Global routing

In global routing the whole grid is divided to partitions, and it is required to determine in which partitions each net will be routed. This is done without knowing the exact path within each partition. Global routing could be used to decrease the required time for detailed routing, to get higher routing ratio, to improve the parallelization between nets, and to obtain better routing quality (shorter wire length and less number of vias).

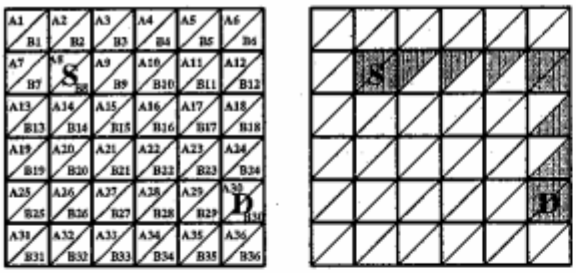


fig 4a fig 4b

Figure 4: Global routing for a net

Our global router has a restricted x and y direction (only x direction is allowed for odd layers and y direction for even layers). The global routing has to detect in which layers and in which partitions within these layers each net will be routed. We assume that the number of layers is already known before routing. An example of two layers grid is shown in fig 4a. Each layer is divided to 36 partitions. The output of the global router for the net (S,D) is as shown in fig 4b. Two layers are assigned to the partitions which contain net terminals (A8,B8,A30,B30) or path bends (A12,B12). While one layer only is assigned to the other partitions (A9,A10,A11,B18,B24).

We assign two layers to the partitions which contain net terminals to be able to route nets like the example shown in fig 5. In this example, it is required to route both nets A and B (Fig 5a). If there is only one layer available and we start by route net A, then net B can not be routed (Fig 5b). And if we start by net B, then net A can not be routed (Fig 5c). But if there are two layers available at the partitions which contain net terminals, both nets A and B could be routed (Fig 5d).

We assign only one layer for the other partitions to increase parallelization between nets. In the detailed routing, nets can be routed simultaneously if their global paths has no common partitions. Fig 6 shows an example of 6 nets which can be routed simultaneously in the detailed routing.

The global routing is done using more than one PE in parallel. The nets are divided to the global PEs. For example, if the total number of

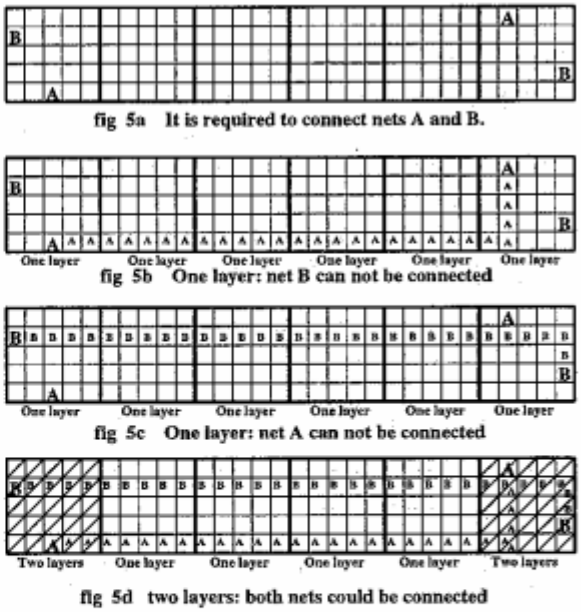


fig 5a It is required to connect nets A and B.

fig 5b One layer: net B can not be connected

fig 5c One layer: net A can not be connected

fig 5d two layers: both nets could be connected

Figure 5: Conflicting nets.

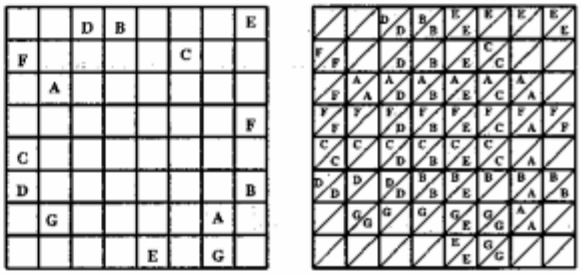

Figure 6: Route more than one net simultaneously

nets is 1000 and number of global PEs is 5, then each PE will be responsible for finding a path for 200 nets. Note that the nets are already sorted in ascending order according to their lengths. We assign nets 1,6,11,....,996 to $PE_1$. This assignment gives better load balancing than to assign nets 1,2,3,.....,200 to the same PE. This is because finding global paths for short nets takes less time than long nets.

To find net paths, each PE routes its nets sequentially (one after the other) using maze running algorithm[1] with different costs. The cost of each partition depends mainly on the number of available tracks inside this partition.

Each PE, after finding a global path for one net, broadcasts this global path to the other PEs to update their grid cost according to this path. As every grid partition has a capacity, there is no problem if more than one PE use the same grid partition for different nets at the same time.

Assume that the PEs for global routing are $PE_1, PE_2, ....., PE_n$. The global routing algorithm for processor $PE_m$ is as follow:

1. Sort nets in ascending order according to their lengths.
2. Take k equal 0.
3. If there is any messages from other PEs, then update grid cost.
4. Find the least cost path for net which has an order $(k \times n + m)$.
5. update the grid cost according to this path.
6. Broadcast this path to the other global PEs.
7. Increase k by 1.
8. If $(k \times n + m)$ greater than total nets number, then stop.
   otherwise repeat 3.

# 3 Detailed routing

In the detailed routing, it is required to find the exact path for each net within the partitions which are previously assigned to this net during global routing. Net paths must not intersect with each other. In our algorithm, both global and detailed routing are done in parallel using different PEs. When any global PE finishes routing certain number of nets (one group), it sends the global paths of these nets to the assigning PE, and starts to route another group of nets.

## 3.1 Maze running algorithm.

Maze running algorithm [5] guarantees to find the shortest path between two points if one exists. On the other hand, maze running algorithm is time and memory consuming. Many researches [6] [1] modified the original maze to be faster. Some other researches [7] improve the speed using hardware implementation. There

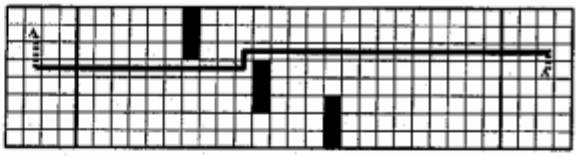

Figure 7: Unrestricted detailed direction.

are also researches which aimed to decrease the required memory [8]. The main idea to speed up maze running algorithm is to limit the propagation area. Of course, using global routing as a preliminary step decreases the propagation area. Although maze running algorithm has a serious disadvantage that previous routed net paths may prevent the other nets to be routed later, maze still gives higher connection ratio than many other algorithms. Rip-up and reroute is already introduced to overcome this disadvantage, but it needs long time to be implemented. It is difficult also to decide which nets have to be rerouted.

In this paper, although global routing has a restricted X and Y direction (only X direction is allowed for odd layers and Y direction for even layers), detailed routing allows both X and Y directions on all layers. Without this policy some nets could not be connected. As global routing assigns only one layer at the middle part, so two directions must be available on all layers to be able to route nets like which shown in fig 7. We use maze running algorithm with different cost for detailed routing. The cost mainly depend on the direction. On odd (even) layers, The cost of X (Y) direction is less than the cost of Y (X) direction. The cost of via is much higher.

## 3.2 Space division (PEs allocation).

In this paper, detailed PEs are divided firstly to the layers. If we take the example of having 2 layers grid with dimensions $1000 \times 1000$ and 16 PEs for detailed routing, then 8 PEs are assigned to each layer. In the second step, every layer is divided to slices instead of square areas. The length of each slice is equal to the grid dimension (1000 in this example), while slice width is equal to grid dimension divided by number of PEs for this layer($1000/8 = 125$). Odd layers are divided to slices in X direction, while slices

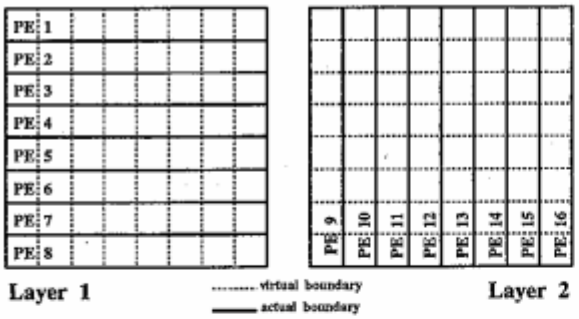


Figure 8: Dividing the grid to slices

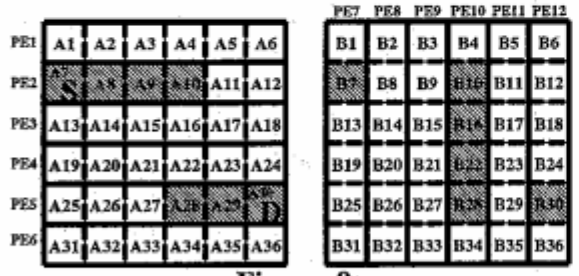


Figure 9a

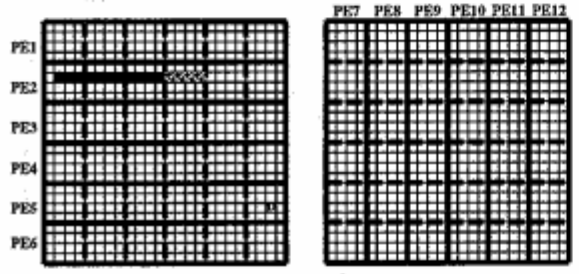


Figure 9b

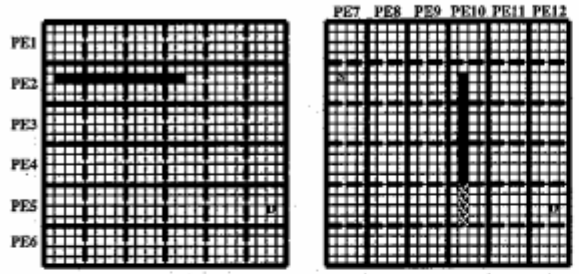


Figure 9c

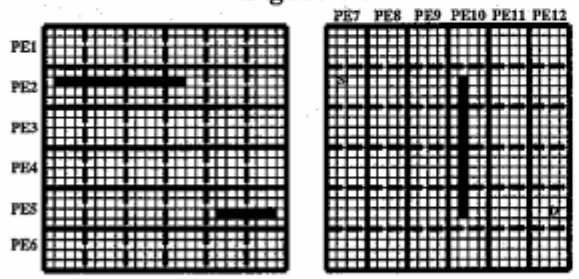


Figure 9d

Figure 9: Detailed routing.

for even layers are in Y direction. In the third step, each slice is divided by virtual boundaries to partitions. The virtual boundaries are used to divide the memory of each PE to different parts. Virtual boundaries for odd/even layers are in the same positions as actual boundaries of even/odd layers (Fig 8). By using virtual boundaries, we could have wave propagation for more than one net in the same slice at the same time if their global bath does not overlapping in any partition.

In the global routing, the number of partitions for one layer must be equal to $(K \times C)^2$ , where K is the number of detailed PEs assigned to one layer and C is the number of slices for one detailed PE. When C increased, better load balance for detail routing could be obtained, but the global routing become more difficult (number of partitions become larger) and the succeeding connection ratio may also decreased (this is happen when slice become very narrow).

### 3.3 Algorithm

Fig 9 is an example to show how the detailed routing is done for a net. If the global output of this net is as shown in fig 9a, the detailed routing is done as the following steps.

1. The master (assignment) PE checks all partitions on the global path for this net (partitions B7, A7, A8, A9, A10, B10, B16, B22, B28, A28, A29, A30, B30). If all these partitions are marked free, then master PE marks these partitions as busy, sends this net identification to $PE_7$ which is the processor responsible for the first partition on the global path (B7).

2. $PE_7$ sends the data of partition B7 to $PE_2$. This data informs $PE_2$ about occupied grid points and to which net they are occupied. $PE_7$ now could not use B7 but it could use any other partitions within its slice to route other nets.

3. $PE_2$ finds the least cost path between the source and the end of net global path within its slice (virtual boundary between A10 and A11). This path consists of two parts, settled in partitions (B7, A7, A8, A9), and temporary in partition A10 (fig 9b).

Table 1: MCM benchmark

| | chips no. | nets no. | pins no. | grid size |
|---|---|---|---|---|
| mcc1 | 6 | 802 | 2495 | 599 × 599 |
| mcc2 | 37 | 7118 | 14659 | 2032 × 2032 |
| mcc3 | 37 | 7118 | 14659 | 3386 × 3386 |

4. $PE_2$ sends data of B7 back to $PE_7$, so $PE_7$ could now use this partition also in routing other nets. $PE_2$ sends message to the master PE to free partitions (B7, A7, A8, A9), sends the data of partition A10 to $PE_{10}$.

5. $PE_{10}$ finds the least cost path between the received temporary path in A10 and the end of net global path within its slice (the virtual boundary between B28 and B34). The settled part of the path is in partitions (A10, B10, B16, B22) and the temporary part is in B28 (fig 9c).

6. $PE_{10}$ sends data of A10 back to $PE_2$, sends message to the master PE to free partitions (A10, B10, B16, B22), sends data of partition B28 to $PE_5$, sends message to $PE_{12}$ to send data of B30 to $PE_5$.

7. When $PE_5$ received data of B28 and B30, it finds the least cost path between the destination and the received temporary path. All this path is settled (fig 9d). $PE_5$ sends data of B28 back to $PE_{10}$ and data of B30 back to $PE_{12}$. $PE_5$ sends a message to the master PE to free partitions (B28, A28, A29, A30, B30).

The main advantages of this technique are:

1. Decrease of the communication cost since the number of messages per net is proportional with number of vias instead of net length.

2. Improve resulting routing paths by using less number of vias.

3. Improve parallelism by allowing large number of nets to be routed simultaneously.

4. obtain good net path (no virtual terminals).

Table 2: Parallelization result

| | no. of PES | | | | time | time |
|---|---|---|---|---|---|---|
| | G | M | D | T | sec | × PEs |
| mcc1 | 1 | 1 | 4 | 6 | 62 | 372 |
| | 2 | 1 | 8 | 11 | 32 | 352 |
| | 2 | 1 | 12 | 15 | 25 | 375 |
| | 3 | 1 | 20 | 24 | 16 | 384 |
| | 4 | 1 | 24 | 29 | 14 | 406 |
| | 5 | 1 | 40 | 46 | 11 | 506 |
| | 7 | 1 | 60 | 68 | 9 | 612 |
| mcc2 | 3 | 1 | 6 | 10 | 592 | 5920 |
| | 6 | 1 | 12 | 19 | 316 | 6004 |
| | 12 | 1 | 24 | 37 | 172 | 6364 |
| | 15 | 1 | 30 | 46 | 150 | 6900 |
| | 24 | 1 | 48 | 73 | 95 | 6935 |
| | 29 | 1 | 60 | 90 | 85 | 7650 |
| mcc3 | 2 | 1 | 4 | 7 | 1410 | 9870 |
| | 3 | 1 | 8 | 12 | 830 | 9960 |
| | 4 | 1 | 12 | 17 | 628 | 10676 |
| | 5 | 1 | 16 | 22 | 474 | 10428 |
| | 6 | 1 | 20 | 27 | 383 | 10341 |
| | 8 | 1 | 24 | 33 | 324 | 10692 |
| | 10 | 1 | 32 | 43 | 250 | 10750 |
| | 12 | 1 | 40 | 53 | 210 | 11130 |
| | 14 | 1 | 48 | 63 | 179 | 11277 |
| | 18 | 1 | 60 | 79 | 145 | 11455 |

G:global , M:master , D:detail , T:total

## 4 Result

We test this algorithm with 3 industrial MCM (multi-chip module) benchmarks [13] shown in table 1. The total number of vias is almost or less than half of the number of vias obtained in[13] for the same data. We obtained connection ratio of 97.5%, 98%, and 99% for mcc1, mcc2, mcc3 respectively. As mentioned in [13] the sequential 3D maze required 59 min to route mcc1 on sun sparc station II.

In our program, we use at least one PE for each layer plus master and global PEs. So, we could not calculate the speed up comparing with one PE. Also, as we use maze algorithm, the total required memory for the whole grid is proportional with $M \times N \times L$ where M,N are the grid dimensions, and L is the number of layers. So the required memory for mcc2 or mcc3 is much more than the available memory per one PE.

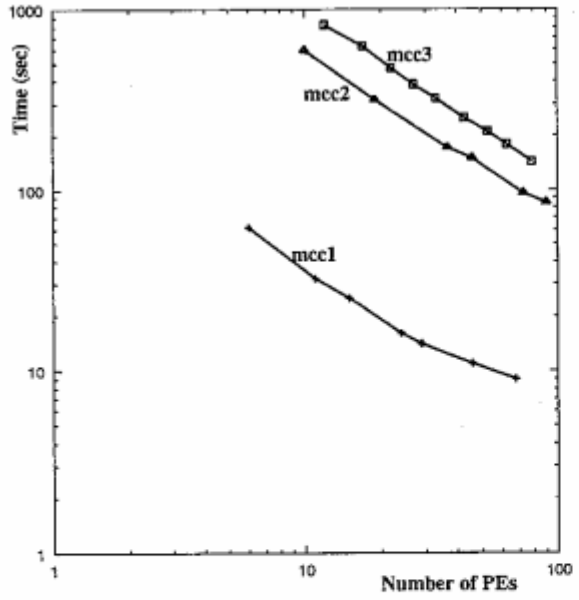


Figure 10: Time against number of PEs for mcc1.

Table 2 shows total routing time (global + detail) using different number of PEs. The number of slices per layer are 30, 80 and 120 for mcc1, mcc2 and mcc3 respectively. Figure 10 shows the time against number of processors for mcc1, mcc2, and mcc3 in a logarithmic scale.

## 5 Conclusion

A parallel global and detailed slice maze router is introduced. This router has a new technique to divide the grid to the PEs. This technique decreases the communication cost, improves routing quality by decreasing number of vias and improves parallelism between nets. Both the global and detailed routing are done in parallel. We divide each layer to slices, divide the slices to partitions. Each PE from the detailed routing PEs will be responsible for one or more slice.

## 6 Acknowledgment

We want to thank Fujitsu Laboratories Ltd for offering us the parallel computer AP1000 to implement our parallel algorithms. We also thank all our laboratory members for their great help.

## References

[1] J. Soukup "Fast maze router", Proc. Design Automation Conf. 78, pp 100-102, 1978.

[2] M. Burstein and R. Pelavin "Hierarchical wire routing", IEEE Trans. on CAD 84, pp 223-337, 1984.

[3] M. Marek-Sadowska "Global router for gate array", IEEE proc. Int. Conf. on Computer Design 84, pp 332-337, 1984.

[4] E. S. Kuh and M. Marek-Sadowska "Global Routing", Layout design and verification 1986, pp 169-198, 1986.

[5] C. Y. Lee "An algorithm for path connections and its applications", IRE Trans. on Electronic Computers, vol. EC-10, pp 346-365, 1961.

[6] J. H. Hoel "Some Variations of Lee's algorithm", IEEE Trans. on computer vol. C-25, pp 19-24, 1976.

[7] K. Suzuki "A gridless router : software and hardware implementation", VLSI 87, 1987.

[8] J. Soukup "Maze router without a grid map", ICCAD 92, pp 382-385, 1992.

[9] Keshk, Mori, Nakashima and Tomita "A new technique to improve parallel automated single layer wire routing", Proc. Performance evaluation of parallel systems 1993, PEPS'93, pp 134-141, 1993.

[10] Y. Takahashi "Parallel maze running and line search algorithms for LSI CAD on binary tree multiprocessors", Word conferenece on information/comm. Seoul 89, pp 128-136, 1989.

[11] M. Goda, N. Toyama and T. Watanabe "A parallel detailed router TRED" proc. of Transputer/Occam Japan 1994.

[12] Shimamoto, Hane, Shirakawa, Tsukiyama, Shinoda, Yui and Nishiguchi "A distributed system for multilayer SOG", Proc. of EURO-DAC 92, pp 298-303, 1992.

[13] Kei-Yong and J. Cong, "An efficient multilayer MCM router based on four via routing", DA 30, pp 590-595, 1993.