RC24663 (W0810-036) October 7, 2008
Computer Science

IBM Research Report

Network Flow Based BSM Asssignment

Hua Xiang, Haoxing Ren
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

Tingdong Zhou
IBM System & Technology Group

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Network Flow Based BSM Asssignment

Hua Xiang, Haoxing Ren, Tingdong Zhou*
IBM T. J. Waston Research Center
*IBM System & Technology Group

Abstract— **in current industry practice, Bottom Surface Metals (BSMs) assignment for high frequency signals on a package device is a tedious manual job. One often needs to change the assignment multiple times in order to produce a routable solution. This paper proposes a network flow based method to assign BSMs automatically. It constructs a network flow graph based on available routing resource, honoring constraints such as wiring/BSM blockages. Then it derives the BSM assignment from the min-cost max-flow solution. The resulting assignment is optimal in terms of routability and wirelength. In practice, some high speed signals require differential pair routing. This paper proposes a two-step BSM assignment algorithm to handle these differential pair constraints. The first step constructs a bi-partite graph which solves the BSM pairing problem automatically. The second step constructs a scaled flow graph to assign BSM pairs to differential pairs. Compared to the manual approach, these methods can provide an optimized solution which also significantly reduces the turn around time of board design process from days to seconds.**

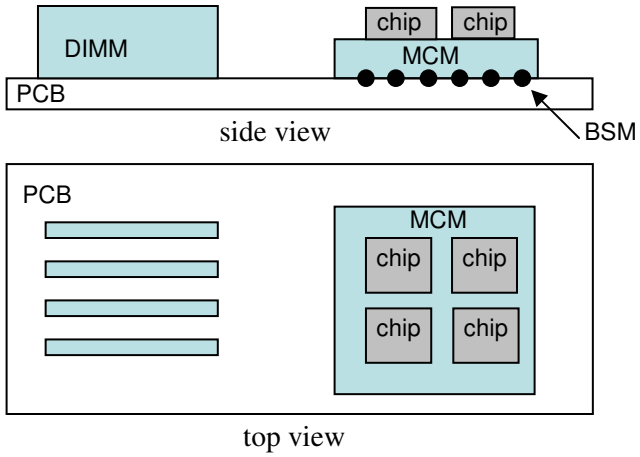

Figure 1 Package hierarchy illustration

I. INTRODUCTION

The Printed Circuit Board (PCB) of high performance server processor board is used to mechanically support and electrically connect high speed devices such as Dual In-Line Memory Modules (DIMMs) and Multi-Chip Module (MCM) which houses several processor chips and their caches. The interconnects between these devices are often high speed signals which have to be routed on one PCB layer. As VLSI systems become more and more complex, it is quite challenging to complete the routing on the PCB.

Pin locations have a huge impact on the routability and congestion of the interconnects, which further affect the chip performance and the board cost. For industry standard devices such as DIMMs, their pin locations are often predefined. But devices such as MCM are custom designed and their pins, so called Bottom Surface Metal (BSM), are assigned by package designers. Bad BSM assignments will result in scenic routes or require more PCB layers, which sometimes becomes infeasible to manufacture. Hence BSMs should be assigned with the PCB routing in mind. Figure 1 illustrates a PCB carries 4 DIMMs and one MCM which houses 4 processor chips. The BSMs are at the bottom of MCM connecting the MCM to the PCB. There are usually thousands of high speed signals between MCM and DIMMs and we have to contain all the wires in a handful of signal layers within PCB.

The current design process for BSM assignment has two steps. The first step is to divide the device BSMs into groups and assign each group to a specific group of signals, e.g. assign a BSM group for all the signals on a memory bus. The BSM groups are created such that there is less chance the wires between BSM groups can overlap. This step is often done manually in a GUI environment. The second step is then to assign the BSMs within each BSM group. This is the most time consuming step because the package designers have to manually fan out the BSMs in the group and line up them with the traces from other devices. The whole process needs several days for a processor board having 8 DIMMs. Furthermore, any future change to one BSM assignment requires reassignment of many other BSMs in the same group because the wring channels are already fully utilized. Clearly, it is highly desired to get the second step of assigning BSMs within a BSM group done automatically by a CAD tool.

BSM assignment problem is similar to macro pin assignment in the sense that the MCM can be considered as a macro, and the BSMs considered as macro pins. Many algorithms have been proposed to perform macro pin assignment. Two conventional approaches are: 1): a two-step approach where pin assignment is followed by routing [1] [2] [3] [4], and 2): a net-by-net approach [5] [6] [7] [8] where pin assignment and routing for a single net are performed simultaneously. A newer approach [9] performs pin assignments and routing simultaneously with a min-cost max-flow formulation. This work follows a similar spirit of that approach.

Two related problems of BSM assignment are chip IO placement and escape routing. Although seems similar, there are major differences between these problems. Chip IO placement assigns available chip level IO circuits to chip internal signals under constraints such as timing closure, signal integrity and power integrity. There are several works [10] [11] addressing this problem. They formulate it as a linear programming (LP) problem to optimize the wiring cost under electrical constraints. Although this formulation can also be applied to BSM assignment problem, the lack of routability analysis of this approach makes it unattractive. In high speed designs, vias seriously degrade the signal

characteristics, therefore high frequency nets are normally routed in a planar fashion on every layer. Hence those nets are routed with escape routing algorithms [12] [13] [14] [15] [19] that route nets in one layer, sometimes minimize or completely avoid crossings. Although this work uses a routing modeling similar to these escape routing algorithms, the difference of this work to them is in that the main objective of escape routing algorithms is to find good routes assuming the BSM locations are fixed, while this work is to find good BSM assignments, which have feasible routing solution.

The main contributions of this work are:

- We propose a min-cost max-flow based algorithm which gives an optimal single-ended BSM assignment solution in terms of routability and wirelength for predefined BSM groups. To the best of our knowledge, this is the first time such a solution is given for the BSM assignment problem.
- We also propose a two-step method to extend this approach to handle differential pair constraints. The two steps are BSM pairing and pair assignment. We model the BSM pairing problem as a bi-partite problem which guarantees to return an optimal BSM pairing solution. The pair assignment is easily handled as a scaled version of the single ended BSM assignment problem.

The paper is organized as follows: Section II describes the BSM assignment problem; Section III gives the min-cost max-flow network flow based algorithm to solve the single-ended BSM assignment problem; Section IV extends this algorithm to deal with pair constraints; experiment results on three industry board designs are given in Section V followed by conclusion in Section VI.

II. Problem Formulation

As mentioned earlier in the introduction section, the BSM assignment process has two steps. The first step is to define the BSM group for each signal group. Each BSM group may contain up to hundreds of signals. The BSM groups are selected to reduce possible wiring overlaps. For example, assign a set of BSMs on the left of MCM to a signal bus coming from left. After BSM groups are defined, the second step is to assign BSMs in each group. This work deals with the automation of the second step.

For each BSM group, we are given a set of routing resource on PCB, which is defined by the BSM group assignment step. Assuming each BSM group is given completely non-overlapping routing regions, we can solve the BSM assignment problem of each group separately. Therefore, we will continue our discussion based on BSM assignment of a single group. For the BSM assignment problem on the entire MCM, it can be easily solved by iteratively apply our approach on all BSM groups.

Each BSM group is assigned to connect to a group of signals from outside devices. In the rest of the paper, we assume that those signal nets connect to a DIMM. Therefore the pin locations on one end of these nets are predefined. The other end of these nets is BSM and we need to assign each net a BSM from the given BSM group. Since vias are not allowed for high frequency net routing, each net needs to be routed completely on a single layer. To minimize the total number of layers, we need to assign a maximum number of BSMs on each layer such that all the two-pin nets between DIMM pins and BSMs on that layer have no crossing. Therefore we try to assign BSMs for a maximum number of nets on one layer, and iteratively assign for the remaining nets on other layers.

We model the routing resource inside the routing region for a BSM group as a 2-D routing grid. The adjacent grid nodes are connected by edges which represent wire segments. Also the BSMs and DIMM pins are located in the centers of routing grids. Figure 2 gives an example of the routing grid. On the PCB, both the horizontal/vertical and the 45˚ diagonal routing are allowed. In Figure 2, the hexagons on the left side (i.e., $d_1...d_6$) represent the DIMM pins, and the squares on the right side (i.e., $b_1...b_6$) are the BSMs. Also both DIMM pins and BSMs can be accessed from eight directions as illustrated by the four horizontal/vertical line segments and the four 45˚ line segments. In this example, we need to assign BSM from ($b_1...b_6$) for each of the 6 nets connected to ($d_1...d_6$).

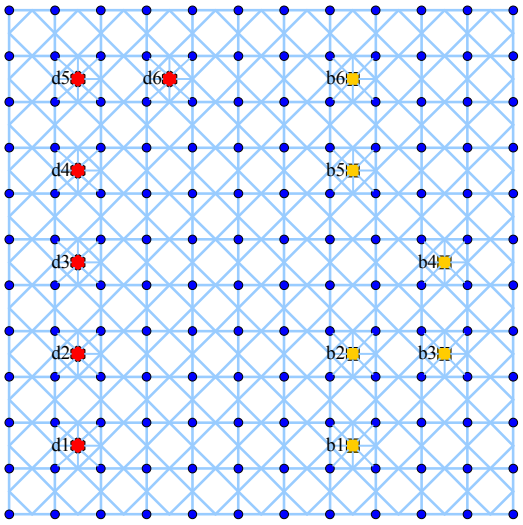

Figure 2 A BSM assignment problem

In practice, the BSM pitch is twice of the routing pitch. Furthermore, there might be pre-assigned BSMs for special nets such as power and ground. Those BSMs will be considered as blockage and ignored in the routing grid.

III. ALGORITHM

To solve the BSM problem, we first construct a network graph G = (V, E) based on the routing grid, and then apply a min-cost max-flow algorithm [17] to get a maximum flow solution. The optimal BSM assignment can be derived from the flow solution.

First, two nodes, a source node and a sink node are created. The source node is connected to the DIMM pin nodes, while all the BSM nodes are connected to the sink node.

Second, the input graph for a min-cost max-flow algorithm is required to be a directed graph. Therefore, each edge on the routing grid (except the edges connecting to DIMM pins and BSMs) is represented by two edges with opposite directions. The edges between a BSM pin node and a routing grid node are from the routing grid node to the BSM pin node since the routing ends once the pins are reached. Similarly, the edges between a DIMM pin node and a routing grid node are from the DIMM pin nodes to routing grid nodes.

In this model, each edge and each node have a capacity which specifies how many wires are allowed to go through. To prevent routing crossing, capacity of each edge and node is set to 1 since only one route is allowed. Also each edge in the routing grid is associated with a cost 1. The edge cost helps to shorten the total routing wire length. The edges between a source/sink node and a pin/BSM node are artificially created, and their cost is 0. Figure 3 shows the graph construction for the problem in Figure 2. The two numbers in each number pairs on edges are the edge capacity and cost, respectively.

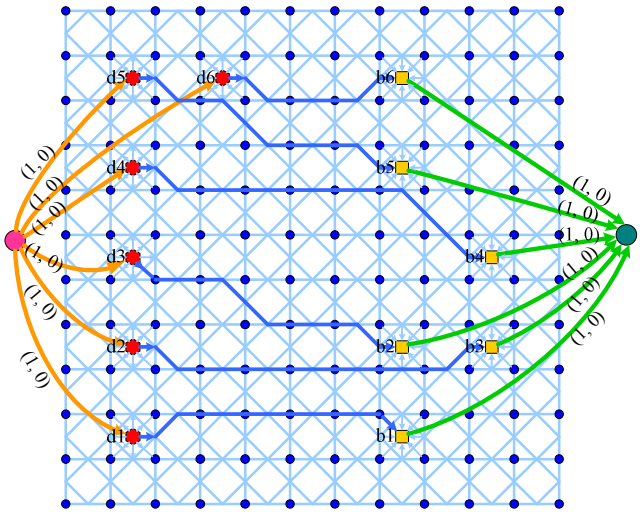

Figure 3 Network flow graph construction

As we note that the classical network flow problem only assigns capacities on flow edges. We handle node capacity by splitting the node Q into two nodes Q_{in} and Q_{out}. One edge is added between Q_{in} and Q_{out} with a capacity 1 and a cost 0. Then all the original incoming edges are pointed to Q_{in} while all the original out-coming edges are pointed out from Q_{out}. Figure 4 illustrates the idea of the node splitting.

Once the network flow graph is constructed, the min-cost max-flow algorithm can be applied to find an optimal flow solution. We can then derive the BSM assignment from the flow solution. Each flow from the source node to the sink node should pass one DIMM pin node and one BSM node. This BSM is then assigned for the net connected to the DIMM pin. The thick lines in Figure 3 show the flow solution to the example in Figure 2. And the pin assignment solution is (d_1, b_1) (d_2, b_3) (d_3, b_2) (d_4, b_4) (d_5, b_5) (d_6, b_6).

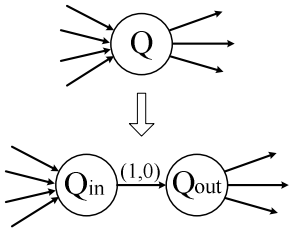

Figure 4 Node splitting for node capacity

The optimality of the min-cost max-flow algorithm guarantees that the maximum number of feasible nets can get connected. Therefore, if the problem has a feasible solution, the algorithm guarantees to return one. Furthermore, the edge cost helps to identify a solution which has the minimum total wire length so that more routing resource can be saved for future usage.

After applying network flow algorithm on one layer, some BSMs may not be assigned due to the routing resource limitation. In this case, we can construct the flow graph for the unassigned BSMs on another layer and apply the algorithm again. By repeating this process, finally all BSMs can get assigned. It is easy to see that the number of routing layers for the given BSM problem is also minimized.

We now summarize the BSMA (BSM Assignment) algorithm as follows. NETS is the set of nets to get connected between the DIMM pins and the BSM pins.

BSMA **Algorithm** (NETS)
1. *layerID* = 0;
2. *netset* = NETS;
3. while (|*netset*| > 0)
4. Construct the network graph for *LayerID*
5. Assign capacity and cost
6. Apply the min-cost max-flow algorithm
7. Record the BSM assignment to *RoutedNets*
8. *netset* = *netset* – *RoutedNets*
9. *LayerID* ++
10. if (*LayerID* > *Available_Layers*)
11. return “No Feasible Solution”
12. endwhile

Theorem 1. The BSMA algorithm can exactly solve the BSM problems. The algorithm guarantees to find the minimum number of routing layers with a feasible BSM assignment solution as long as one solution exists. Furthermore, the returned solution has the minimum total wire length.

Finding a min-cost max-flow solution in a flow network is a classical problem, and several polynomial algorithms are available [17] [18] [12]. Also the maximum number of routing layers is fixed for a given design. Therefore, if the double scaling algorithm in [17] is used, the time complexity of the BSMA algorithm can be bounded by O(VElogV) where V is the number of nodes in G and E is the number of

edges. It is easy to see that both V and E are linearly bounded by the nodes in the routing grid.

IV. PAIRING CONSTRAINTS

Advanced server designs often use differential pair as signaling technique for high frequency signals. The routing paths of differential pairs are required to be close enough, in other words, those paired nets should be routed together and their BSMs should be adjacent. For convenience, we call the BSM problem with the pairing constraints as PBSM (Paired BSM) problem.

Usually the differential pairs are assigned in two steps:

1) *Pair up two adjacent BSMs.*
2) *Assign the BSM pairs to differential pairs.*

However, both steps are done MANUALLY in industry.

In this section, we extended the BSMA algorithm to handle the pairing constraints. We still take the conventional two-step approach but both steps are done automatically.

We still use the example in Figure 2 to illustrate our idea. Please note that in practice, the BSM pitch is twice of the routing pitch. To solve the PBSM problem, we start from constructing a scaled routing grid as outlined by the dark blue grid in Figure 6. Every two adjacent horizontal/vertical/45˚ edges are represented by one dark grid edge. For example, the first two vertical lines are represented by the first dark vertical line, and the 3rd and the 4th vertical lines correspond to the 2nd dark vertical line. The constructed dark grid is used for both BSM pairing and PBSM assignment.

BSM Pairing The first step of our PBSM algorithm is to pair up two adjacent BSMs. A BSM can be paired with a BSM on its up, down, left and right side, but not diagonal side. Since any two horizontal or vertical adjacent BSMs can be assigned as a pair, there are many BSM pair configurations. If we arbitrarily assign two adjacent BSMs as a pair, it is very likely that some BSMs cannot get paired. For example, for the 6 BSMs in Figure 2, if we make b_2 and b_3 as a pair, then b_1 and b_4 cannot be paired. In this section, we convert the BSM pairing problem into a bi-partite graph so that an optimal pairing configuration can be identified.

Based on the scaled grid, we color the whole grid as a chess board. Figure 5 (a) shows the grid piece that covers all of the BSMs. In this way, we divide the BSMs into two groups: BSMs in dark grid tiles and BSMs in white grid tiles. For convenience, we call the two types of nodes as dark BSMs and white BSMs, respectively. For any dark BSM, if there are adjacent white BSMs, then edges are created between the dark BSM and the white BSMs. In Figure 5, the white BSM b_3 and b_1 are the neighbors of the black BSM b_2. So two edges are created which point from b_2 to b_1 and b_3, respectively. It also means that b_2 can be paired with either b_1 or b_3.

This graph is a bi-partite graph because there is no edge between dark nodes or between light nodes. Also the edges represent all the pairing possibility between two BSMs.

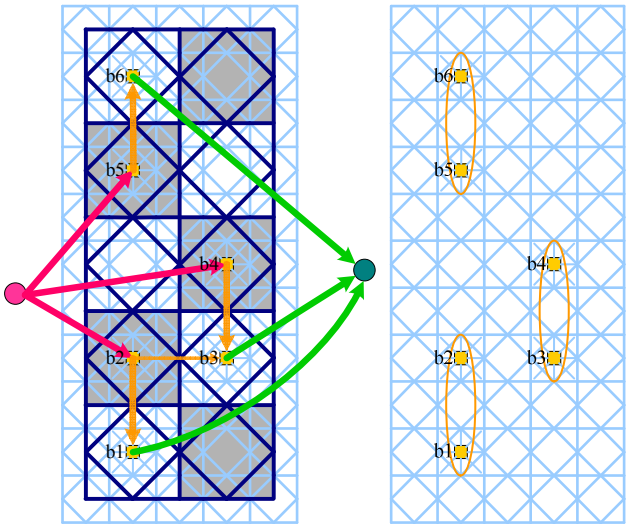

Figure 5 (a) The corresponding flow network for the bi-partite graph to do BSM pairing (b) A BSM pairing solution

Solving a bi-partite problem is simple. We can construct a direct network flow graph using dark BSM nodes as source nodes, white BSM nodes as sink nodes, and flow edges from dark BSMs to white BSMs with a capacity of 1 as shown in Figure 5 (a). The maximum flow solution is a maximum pair assignment solution. The nodes connected by an edge with flow are a pair as indicated with ellipses in Figure 5 (b). The BSM_Pairing algorithm is summarized as follows.

***BSM_ Pairing* Algorithm**

1. Setup the scaled grid
2. Color the scaled grid as a chess board
3. Construct the bi-partite flow network
4. Apply the max-flow algorithm
5. Derive the BSM pairing solution

Theorem 2. The BSM_Pairing algorithm can exactly solved the BSM pairing problems with the guarantee that the maximum number of BSM pairs can be identified.

PBSM Assignment After BSM pair assignment, the second step is to assign differential pairs to BSM pairs. We still draw on the network flow to derive the assignment solution.

To satisfy the pairing constraints, we treat each pair of nets as one supper net. In this case, one supper net routing actually requires a routing resource for two nets. Therefore, we perform the net routing on the scaled graph as the dark grid in Figure 6. For DIMM pins, they are already paired, and the pairing information is a part of the input. For paired DIMM pins and BSMs, we let the middle point of a pair of DIMMs/BSMs as the pseudo DIMM/BSM pin location (i.e., the center of the ellipses in Figure 6). Then the source node

is connected to all the pseudo DIMM pins, and all the pseudo BSM pins are connected to the sink node. Of course, the number of flows pushed on the scaled graph is the number of net pairs (i.e., half of the total nets). For the scaled graph, the edge capacity and node capacity are still 1 since it means a supper net (a pair of two nets). Using the scaled flow network graph, we can find a routing solution for all pairs as the thick lines (in orange, pink and green colors) in Figure 6.

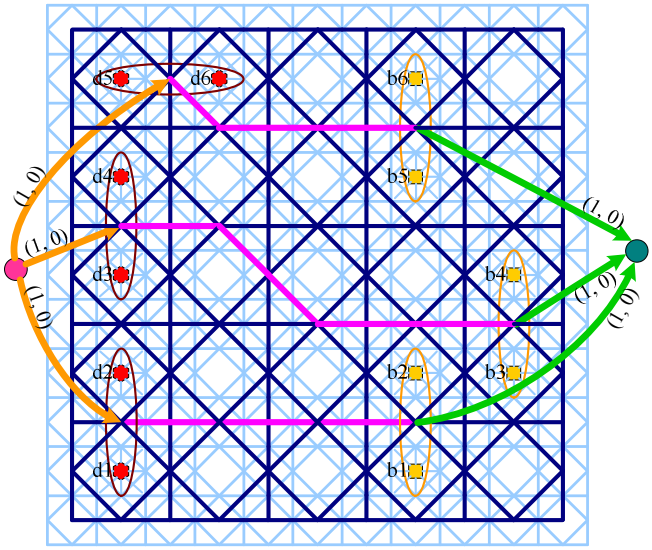

Figure 6 A scaled network flow solution

Similar to the single-ended BSM problem, we can derive the assignment of BSM pairs to differential pairs from the min-cost max-flow solution of the scaled flow graph. Once we get the pair assignment, one more step is required to split one path into two paths in the original routing grid as illustrated in Figure 7.

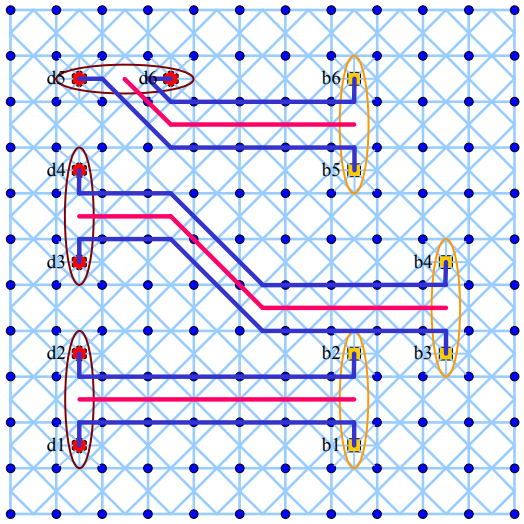

Figure 7 A PBSM solution based on the path splitting

The PBSMA (PBSM Assignment) algorithm is summarized as follows.

PBSMA **Algorithm** (NETS)

1. *layerID* = 0;
2. *netset* = NETS;
3. while (|*netset*| > 0)
4. Construct the scaled network graph for *LayerID*
5. Assign capacity and cost
6. Apply the min-cost max-flow algorithm
7. Split each path into two paths on the original grid
8. Record the pin assignment solution to *RoutedNets*
9. *netset* = *netset* – *RoutedNets*
10. *LayerID* ++
11. if (*LayerID* > *Available_Layers*)
12. return "No Feasible Solution"
13. endwhile

V. EXPERIMENTAL RESULTS

We implement the proposed algorithm in C on an AIX workstation (1.6GHz). The test cases are derived from industry designs. Table 1 summarizes the basic information of the three test cases.

Table 1. Test Case Property

Test	Routing Region	#BSMs	#nets	Pairing constraints
Test1	300x600	108	108	N
Test2	300x600	108	108	N
Test3	300x600	113	108	Y

As shown in Figure 8, Test1 has two connector modules (on the left) connecting to a MCM (on the right). Like DIMM pins, the connector pins are predefined. We only need to assign a critical group of BSMs allocated to connect to those pins. The red lines show the connections between connector pin and its assigned BSM using BSMA algorithm.

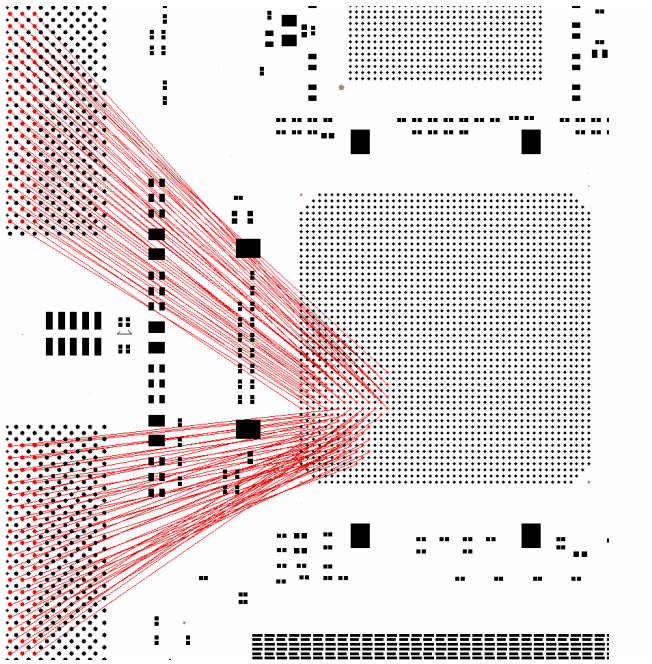

Figure 8 Rat nets of the BSM assignment for Test1

Table 2 shows the results of the two algorithms. For all three testcases, the BSM assignments are completed using only one layer and the runtimes are less than one minute. (When the BSMA algorithm is applied, the pairing constraints are ignored.) The running time for PBSMA algorithm is even much faster because the underlying flow network is smaller. In practice, for this kind of test cases, designers have to do the pin pairing/assignment manually, and it takes several days to complete one design.

Table 2. Test Results

Test	Algorithm	Layers	Assigned BSMs	Total Wirelength	Run Time (s)
Test1	BSMA	1	108	29380.72	48.7s
Test2	BSMA	1	108	26552.80	46.2s
Test3	BSMA	1	108	26384.04	46.7s
Test3	PBSMA	1	108	27043.01	7.77s

Test3 is a test case with the pairing constraints. With the BSMA algorithm, the pairing constraints are not honored. One the other hand, the PBSMA algorithm generates a solution so that the paired nets can get similar routing path. Figure 9 shows a piece of the routing and pin assignment of Test3. The purple thin wires are the routing guide generated by the scaled network flow, and the blue thick wires show the routing path of each net. Comparing to the optimal BSMA solution, the total wirelength of Test3 with PBSMA algorithm only increases 2.5%. This also validates the efficiency of the PBSMA algorithm.

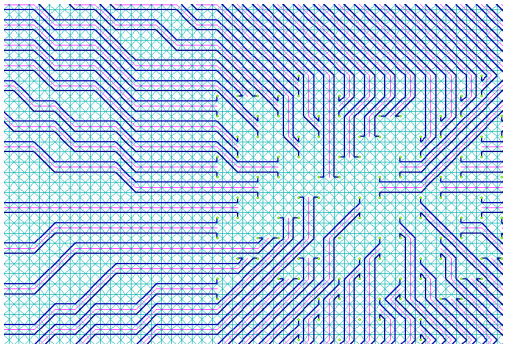

Figure 9 A piece of Test3 BSM assignment solution with pairing constraints.

VI. CONCLUSION

In this paper, we address the BSM (Bottom Surface Metals) assignment problem. In the current industry practice, this kind of problems is handled manually, which takes long time (measured by days) and big design efforts. We propose a network flow based algorithm which can optimally solve the general BSM problems within a few seconds. For the BSM problem with differential pairing constraints, we develop a two-step BSM assignment algorithm. The first step finds the optimal solution for BSM pairing, and the second step assigns BSMs based on a scaled flow network so that the pairing constraints are honored. The experimental results demonstrate the effective and efficiency of our algorithms.

REFERENCES

[1] Koren, N.L. "Pin Assignment in Automated Printed Circuit Board Design". 9th Design Automation Workshop, 1972

[2] Mory-Rauch, L. "Pin Assignment on a Printed Circuit Board ". 15th IEEE Design Automation Conference, 1978.

[3] H. N. Brady, "An approach to topological pin assignment," IEEE Trans.Computer-Aided Design, vol. CAD-3, pp. 250–255, 1984.

[4] X. Yao, M. Yamada, and C. L. Liu, "A new approach to the pin assignment problem," in Proc. ACM/IEEE Design Automation Conf., 1988, pp. 566–572.

[5] J. Cong, "Pin assignment with global routing for general cell designs," IEEE Trans. Computer-Aided Design, vol. 10, pp. 1401–1412, Nov. 1991.

[6] S. G. Choi and C. M. Kyung, "Three-step pin assignment algorithm for building block layout," Electron. Lett., vol. 28, no. 20, pp. 1882–1884, 1992.

[7] T. Koide, S. Wakabayashi, and N. Yoshida, "An integrated approach to pin assignment and global routing for VLSI building-block layout," in Proc. Eur. Conf. Design AutomationWith Eur. Event ASIC Design, 1993,pp. 24–28.

[8] L. E. Liu and C. Sechen, "Multilayer pin assignment for macro cell circuits," IEEE Trans. Computer-Aided Design, vol. 18, pp. 1452–1461,Oct. 1999.

[9] H. Xiang, X. Tang, and M. D. F. Wong, "Min-cost flow-based algorithm for simultaneous pin assignment and routing," IEEE Trans. Computer-Aided Design, vol. 22, pp. 870-878, July. 2003.

[10] W.-K. Mak, "I/O placement for FPGAs with multiple I/O standards," IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, pp. 315−320, February 2004.

[11] J. Xiong, Y.-C. Wong, E. Sarto, and L. He, "Constraint driven I/O planning and placement for chip-package co-design", ASPDAC, p24-p27, 2006

[12] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, Introduction to Algorithms. Cambridge, MA: MIT Press, 1992.

[13] J. Hershberger and S. Suri, "Efficient breakout routing in printed circuit boards," in Proc. 13th Annu. Symp. Comput. Geom., 1997, pp. 460–462.

[14] M. Yu andW.W. Dai, "Single-layer fanout routing and routability analysis for ball grid arrays," in Proc. ICCAD, 1995, pp. 581–586.

[15] M.M Ozdal.; M. D. F. Wong, and P. S. Honsinger, "Simultaneous Escape-Routing Algorithms for Via Minimization of High-Speed Boards", IEEE Trans. Computer-Aided Design, vol. 27, pp. 84–95, Jan. 2008.

[16] A. Titus, B. Jaiswal, T. Dishongh, and A. N. Cartwright, "Innovative circuit board level routing designs for BGA packages," IEEE Trans. Adv. Packag., vol. 27, no. 4, pp. 630–639, Nov. 2004.

[17] R. K. Ahuja, A. V. Goldberg, J. B. Orlin, and R. E. Tarjan. Finding minimum-cost flows by double scaling, Mathematical Programming 53, pp.243-266, 1992.

[18] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. Network Flows, Prentice Hall, 1993.

[19] M. M. Ozdal, Escape routing for dense pin clusters in integrated circuits. DAC 2007: 49-54.