Novel Pin Assignment Algorithms for Components with Very High Pin Counts

Tilo Meister, Jens Lienig
Dresden University of Technology
Dresden, Germany
tilo@ieee.org, jens@ieee.org

Gisbert Thomke
IBM Research & Development
Boeblingen, Germany
thomke@de.ibm.com

Abstract

The wiring effort and thus, the routability of electronic designs such as printed circuit boards, multi chip modules and single chip modules largely depends on the assignment of signals to component pins. For modern components that have as many as several thousand pins, this pin assignment cannot be optimized manually. This paper presents four novel pin assignment algorithms that automatically create optimized pin assignments for wiring substrate designs with components that have very high pin counts. We also present and evaluate quality estimation metrics that enable fast assessment of the pin assignment results. The efficiency of our algorithms allows the creation of optimized pin assignments using only minutes of computation time. We show the applicability of all four algorithms, including their strengths and weaknesses, in specific design applications.

1. Introduction

Pin counts and operation frequencies of electronic components constantly increase and so the layout synthesis of wiring substrates (on which these components are to be placed) becomes ever more complex. The typical basic steps of layout synthesis for wiring substrates, such as printed circuit boards (PCBs), multi-chip modules (MCMs) and single-chip modules (SCMs), are placement of components, interconnect routing and verification.

The high I/O counts and low pitches of the components push the wiring capacities of the substrates to their limits. Also, the increasing complexity of integrated circuits (ICs) makes the delays of signals external to ICs a bottleneck for operation speeds of electronic devices. *Pin assignment*, i.e., the optimized assignment of signals to the pins of the components, has become a crucial step in the layout process in order to reduce the subsequent wiring effort and improve routability. Furthermore, electrical characteristics can be improved as well due to reduced wirelengths, signal intersections and a smaller number of routing layers.

Figure 1 (a) shows a pin assignment task for a graphics processor unit (GPU) placed on an AGP (Accelerated Graphics Port) board. The AGP specification exactly defines the pin assignment of the AGP connector, which is at the bottom edge of the board in Figures 1 (a) and (b). On the other side, the GPU design allows the flexibility to optimize the signal allocation of its pins. Since the assignment of the AGP signals to the GPU pins massively influences the necessary wiring effort between the AGP connector and the GPU, this pin assignment must be optimized. Figure 1 (b) shows one possible pin assignment illustrated by the shortest connections between the two pins of each net (*flylines*).

This paper concerns the optimization of pin assignments from the viewpoint of wiring substrates in an isolated stage, following the placement of the components. Specifically, the position of all components is fixed during the pin assignment process. Interconnect routing (wiring) takes place in a subsequent stage.

There is a significant difference to the related pin assignment algorithms for VLSI design, which typically deal with millions of components each having only a few pins. In contrast to those algorithms, the pin assignment algorithms presented in this paper handle pin assignment tasks with components that each have several thousand pins.

Producing an optimized pin assignment without creating the final wiring geometry represents a challenge

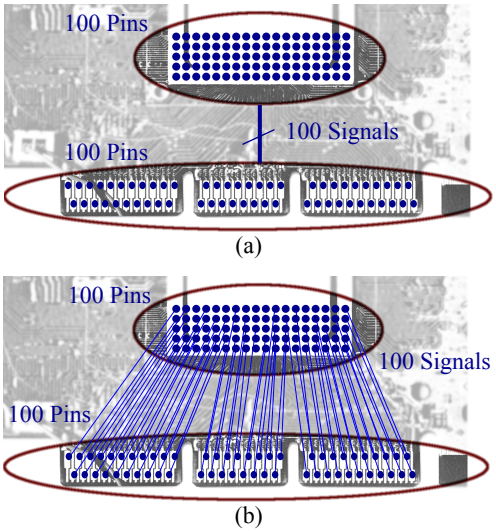

Figure 1. Illustration of a simple pin assignment for a graphics processor unit (GPU). (a) Pin assignment task for the roughly 100 signals of a GPU. (b) One possible pin assignment for the signals of the GPU, illustrated by the shortest connections ("flylines") between the two pins of each net.

978-3-9810801-3-1/DATE08 © 2008 EDAA

because performing detailed routing for the sole purpose of evaluating the pin assignment quality is not feasible. Instead, we have developed metrics for the quality estimation of a pin assignment that represent the expected quality of the actual detailed routing. We have verified our quality estimation metrics by comparing them with the actual routing results.

In summary, the contributions of this paper are four novel pin assignment algorithms that automatically generate optimized pin assignments for components with very high pin counts. Previously these pin assignments had to be created manually which requires an effort of months. Our algorithms reduce the required effort to days. To the best of our knowledge, our approach allows for the first time an automatic pin assignment of components with several thousand pins each.

2. Previous works

Early work on pin assignment concerned topological, heuristic algorithms to reduce routing intersections, thus improving routability [1] [2] [3] [4]. This work focused on discrete components with small pin counts of no more than approximately 20 pins.

The evolution of ICs led to new pin assignment challenges. One typical VLSI pin assignment task is channel pin assignment [5] [6]. Another problem is to optimize the pin assignment in standard cell and macro cell VLSI designs [7]. Furthermore, pin assignment is an important stage in FPGA designs [15]. All referenced approaches on pin assignment have been able to improve the design quality, often evaluated by wirelength and/or routing congestion.

While recent work has been focusing on pin assignment in VLSI designs, little attention has been paid to pin assignment from the viewpoint of wiring substrates, such as PCBs and MCMs [8] [9] [10]. In contrast to VLSI designs, which have up to millions of cells with few pins each, designs of wiring substrates can contain hundreds of components with up to thousands of pins per component. In addition, pin assignment algorithms for VLSI commonly map pins to the outline of cells. This however, is impractical for area array components with very high pin counts placed on a wiring substrate. These significant differences make it impractical to adapt VLSI pin assignment algorithms to current designs of wiring substrates.

3. Definition of the pin assignment problem

Pins are the electric transitions of signals between different components of the design. They are mainly introduced for two reasons. First, pins serve as the well defined interfaces between different components, which allows us to cope with design complexity by applying a hierarchical design approach. Second, pins are the electric joints between components of different technologies.

In either case, the assignment of a specific net (signal) is generally not constrained to a certain pin (location). In most cases, a signal can be assigned to any pin within a certain area. Depending on which *pin location* is chosen, electrical properties for the respective signal differ. In particular, the wiring effort both within the component as well as external to the component can change significantly.

The general pin assignment problem is to assign all nets (signals) to unique, valid pin locations so that the overall design is optimized. In most cases, the optimization is judged by routability and electrical characteristics.

The algorithms we present in this paper solve a subproblem of the general pin assignment problem which is highly relevant for PCB, MCM and SCM designs. Please note that for the sake of simplicity, we refer to assigning pins to fixed pin locations (rather than signals to fixed pins) in the remainder of this paper. As we will show in the following, this allows an easier explanation of the algorithms without modifying the pin assignment problem itself. Taking this into account, the pin assignment problem can be formulated as follows:

Input data:

- Two independent sets of predefined pin locations, defined as *FROM pin location set* and *TO pin location set*. Both sets contain exactly p pin locations each.
- $n=p$ two-terminal (two-pin) nets.

Output data:

- Assignment of the $2 \cdot n$ pins of the n given two-terminal nets to the $2 \cdot p = 2 \cdot n$ given pin locations.

Objective:

- Routability of the design (as defined in Section 5).

Constraints:

- Each net must have exactly one pin at any one *FROM pin location* and one pin at any one *TO pin location*.
- There is exactly one pin at each pin location.
- All nets (signals) are of the same priority and nets do not prefer or refuse a pin location.

From the above definition it follows that the pin assignment problem is equivalent to finding exactly one *TO pin location* and one *FROM pin location* for the pins of any one net. During pin assignment, an objective function is minimized (see Section 5).

Please note that it is irrelevant which particular net is assigned to a pair of pin locations because a net does not prefer a certain pin location. In other words, the subsequent assignment of the specific nets to the pairs of pin locations has no effect on the objective and could as well be created randomly.

Figure 1 shows an example for this pin assignment problem. The package of the GPU defines one set of pin locations. The second set of pin locations is defined by the AGP connector. The pin assignment problem is to find an assignment between the pin locations of the GPU and the pin locations of the AGP connector that optimizes the routability of the AGP signals on the AGP board.

With n being the number of two-terminal nets, the number of possible pin assignments is $n!$. Even for the small number of $n=25$ nets, this results in the vast number of $1.55 \cdot 10^{25}$ possible pin assignments to choose from. For current designs n is in the magnitude of thousands, thus making it unlikely that the pin assignment problem can be solved comprehensively.

4. Algorithms for pin assignment

We have developed four pin assignment algorithms based upon the input data as defined in Section 3. Due to the complexity of the pin assignment problem, the algorithms of Sections 4.1, 4.2, and 4.3 are heuristic. The algorithm presented in Section 4.4 uses linear optimization with optimal solutions but limits optimization to properties that can be formulated as a linear objective function.

4.1. Recursive bisectioning of pins

Both pin location sets are recursively split using a horizontal or vertical cut line. First, the entire set is cut into halves. Then the half set below/left and the half set above/right are each split with a cut line of the other orientation. If an odd number of pin locations has to be split, the subset above/left will contain one location more than the subset below/right. The alternating vertical and horizontal cuts are repeated until each subset of pin locations contains only one location. Figures 2 (a) – (d) show this recursive bisectioning of both sets.

The relative position between a pin location and each cut defines a unique binary sequence (*position number*) for each pin location. Here, the digit "0" denotes that a pin location is above/right a cut line, while the digit "1" means that a pin location is below/left. The pin assignment is created based on these position numbers, i.e., the *FROM pin location* and *TO pin location* with the same position number are assigned to each other. Figure 2 (d) shows the resulting pin assignment.

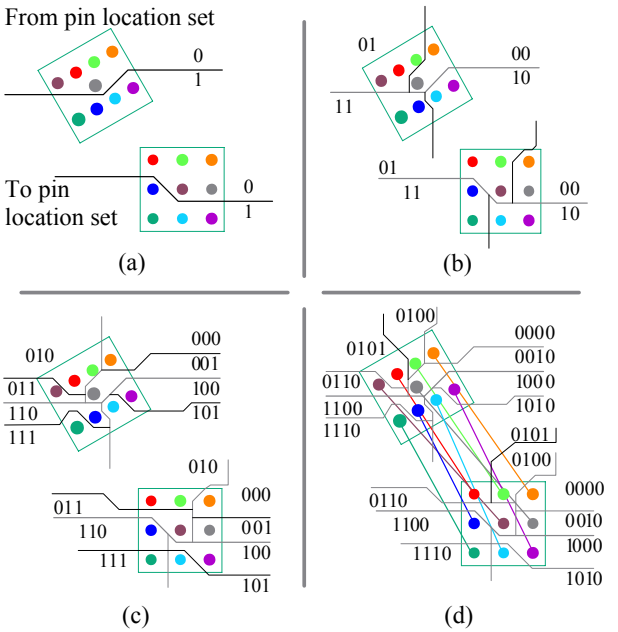

Figure 2. Pin assignment by recursive bisectioning of pin location sets. The binary numbers denote whether a subset of pin locations is above/right (0) or below/left (1) of each cut.

4.2. Projection of pins onto a line

The pin locations of both pin location sets are projected on a line. This line is oriented perpendicular to the virtual line that connects the medians of the two sets. Based on the order of the projected pin locations, the pin assignment is created. The locations of each set are numbered from 1 to p according to the order of their projected positions (Figure 3 a). The *FROM pin location* and the *TO pin location* with the same location number are assigned to each other (Figure 3 b).

The line of projection does not need to be the same for both pin location sets. If a preferred direction of fan out is known for a bus, the line of projection can be oriented to be perpendicular to this preferred direction. This creates a pin assignment with minimum intersections within the wiring of the bus. Using this methodology, routing intersections can easily be reduced even if a bus is wired around a component in order to approach it from the "far side".

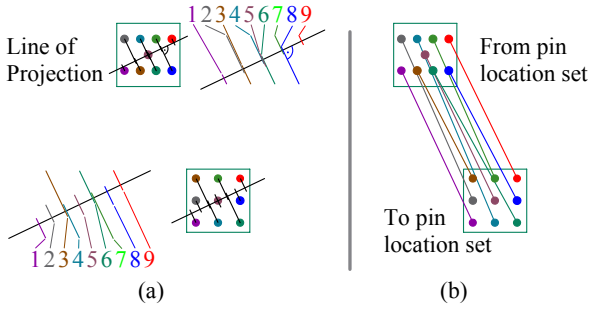

Figure 3. Pin assignment by projection. (a) Ordering of pin locations by projection. (b) Pin assignment.

4.3. Removal of signal intersections

Here, the crossings of the flylines of nets are used to model the signal intersections in the real layout that are to be minimized. First, an initial pin assignment is created. Any random pin assignment may be used for this. Based on this initial solution, crossings of two flylines are located. The pin assignment of the two intersecting nets is then swapped to remove the flyline crossing. This process is iterated until all flyline crossings are removed.

Figure 4 shows an unambiguous example. However, in general more than one pin assignment without flyline crossings may exist. As shown in Section 5, the resulting pin assignment as well as the number of iterations depend on (1) the initial pin assignment and on (2) the processing sequence of nets.

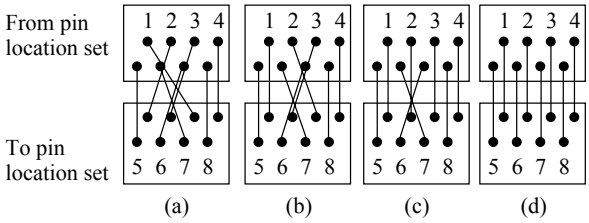

Figure 4. (a) Initial pin assignment. (b) In the upper set of pins, pins at locations 1 and 2 are swapped. (c) Pins 2 and 3 are swapped. (d) Pins 6 and 7 are swapped.

4.4. Pin assignment as linear assignment problem

In [11] and [12], the authors describe an approach to create a pin assignment by solving a linear assignment problem (LAP) on a cost matrix. The elements of the cost matrix are calculated from estimated wirelengths, logical design structure and signal timing.

In our work we use an enhanced approach to create optimized pin assignments for very high pin counts that additionally is able to minimize flyline crossings. The used cost matrix K is of size $p \times p$ and represents the cost of all possible pin assignments.

$$K = \begin{pmatrix} k_{11} & k_{12} & \cdots & k_{1p} \\ k_{21} & k_{22} & \cdots & k_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ k_{p1} & k_{p2} & \cdots & k_{pp} \end{pmatrix}$$

Element k_{ij} is the cost if the *FROM pin location i* is assigned to the *TO pin location j*. The pin assignment is given by the assignment matrix X.

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{p1} & x_{p2} & \cdots & x_{pp} \end{pmatrix} \quad \text{where} \quad x_{ij} \in \{0,1\}$$

$$\sum_{i=1}^{p} x_{ij} = 1 \quad ; \quad \sum_{j=1}^{p} x_{ij} = 1$$

for all i and j.

Element x_{ij} is 1 if *FROM pin location i* is assigned to *TO pin location j*, else $x_{ij} = 0$. By solving the linear assignment problem on the cost matrix K, the optimal pin assignment which minimizes the sum of all costs

$$Overall\ Cost = \sum_{i=1}^{p} \sum_{j=1}^{p} \left(x_{ij} \cdot k_{ij} \right)$$

can be found in $O(p^2 \cdot \log p)$ time. An overview of suitable algorithms is given in [13]. We use the Hungarian algorithm [16] which solves the linear assignment problem in $O(p^3)$ time.

In contrast to previous works using linear assignment, we include flyline crossings in the cost of the pin assignment with the following cost function:

$$k_{ij} = TargetLength \cdot \alpha \cdot Diff_{ij} + l_{ij}$$

Here, l_{ij} is the net length which is minimized in parallel with the estimated number of flyline crossings *Diff* (see below). *TargetLength* is the distance between the medians of both pin location sets. Finally, α (≥ 0) is the parameter to adjust the weight between length minimization and minimization of flyline crossings.

TargetLength and l_{ij} are either calculated as half-perimeter wirelength (HPWL) or in Euclidean geometry. The coordinates of the pin locations are (x_i, y_i) and (x_j, y_j). The medians of the two pin location sets are $\left(\overline{x_{From}}, \overline{y_{From}}\right)$ and $\left(\overline{x_{To}}, \overline{y_{To}}\right)$. We define $dx = \left| x_i - x_j \right|$, $dy = \left| y_i - y_j \right|$, $\overline{dx} = \left| \overline{x_{To}} - \overline{x_{From}} \right|$ and $\overline{dy} = \left| \overline{y_{To}} - \overline{y_{From}} \right|$. Then *TargetLength* and l_{ij} are either one of the following:

1. Euclidean net length:

$$TargetLength = \sqrt{\overline{dx}^2 + \overline{dy}^2} \quad ; \quad l_{ij} = \sqrt{dx^2 + dy^2}$$

2. HPWL net length:

$$TargetLength = \overline{dx} + \overline{dy} \; ; \qquad l_{ij} = dx + dy$$

The estimated number of flyline crossings $Diff_{ij}$ is calculated based on a *target pin assignment* without these crossings. This target pin assignment is created once. First, our linear assignment methodology with $\alpha = 0$ is used to create an initial pin assignment. The initial pin assignment is optimal with respect to the sum of the lengths of all nets. Since α is set to zero, *Diff* is not needed to calculate the cost matrix K. Second, this initial pin assignment is fed to the algorithm of Section 4.3 to create the target pin assignment without flyline crossings.

For the target pin assignment, either a circle around the center of a pin location set or a straight line intersecting all flylines are determined. All nets and consequently all *FROM* and *TO pin locations* are numbered from 1 through p according to the order they are intersected by the line/circle. Let F_i be the number assigned to the *FROM pin location i*. Accordingly, let T_j be the number assigned to the *TO pin location j*. Using the above definitions, $Diff_{ij}$ is calculated as follows:

$$Diff_{ij} = |F_i - T_j|$$

Since the target pin assignment is free from flyline crossings, $Diff_{ij}$ and thus the cost function increases as the number of flyline crossings increases. To be more general, $Diff_{ij}$ not only allows including flyline crossings in the cost function but actually includes the configuration of the target pin assignment as an objective of linear assignment.

If neither a circle nor a line to intersect all flylines can be found for the target pin assignment, $Diff_{ij}$ cannot be computed and the respective part of the objective function is set to 0.

5. Experimental results

The pin assignment algorithms presented in Section 4 have been evaluated with numerous industrial designs from IBM. Due to the lack of suitable benchmarks, PCB, MCM and SCM designs have been used. We present the results for a MCM design with 2930 signals and 2112 power/ground pins (labeled MCM1). Figure 5 shows part of MCM1, which carries dies that are flip chip mounted on top of a chip carrier. The bottom side of the chip carrier is covered with a regular array of pins which are either signal pins or power/ground pins. Within the chip carrier, wiring connects each chip pin (pad) to a pin on the bottom of the chip carrier. The pin assignment algorithms are used to create assignments between all signal pins of the dies and all signal pins on the bottom side of the chip carrier. The signal nets considered are two-terminal nets.

The created pin assignments are evaluated by means of fast quality estimation metrics. These are net lengths in half-perimeter wirelength (HPWL) and in Euclidean geometry, and the number of flyline crossings. The applicability of these metrics is shown by relating them to the real routing results as discussed later.

Given that (x_{ai}, y_{ai}) and (x_{bi}, y_{bi}) are the coordinates of the two pins of net i, p is the number of nets in the pin

assignment task, $dx_i = |x_{ai} - x_{bi}|$, $dy_i = |y_{ai} - y_{bi}|$, we define the following metrics:

- Sum of all HPWL net lengths (*SHPWL*):

$$SHPWL = \sum_i^p dx_i + dy_i$$

- Matched wirelengths (*HPWL MATCH*):

$$HPWLMATCH = p \cdot \max(dx_1 + dy_1, \ldots, dx_p + dy_p) - SHPWL$$

HPWL MATCH indicates the routing effort necessary to match the lengths of all wires of a bus.

- Average net lengths in Euclidean geometry (*AVG Flylines*):

$$AVG\ Flylines = \frac{1}{p} \sum_i^p \sqrt{dx_i^2 + dy_i^2}$$

- The standard deviation of the net lengths in Euclidean geometry (*STD Dev*):

$$STD\ Dev = \frac{1}{p-1} \sqrt{\sum_i^p \left(AVG\ \ Flylines\ \ - \sqrt{dx_i^2 + dy_i^2} \right)^2}$$

STD Dev (similarly to *HPWL MATCH*) indicates the routing effort necessary for matching wirelengths.

- The number of flyline crossings of all nets.

The pin assignment algorithms (*PAA*, see Table 2) of Section 4 are implemented in ANSI C and run on a 2.0 GHz PC. The routing geometry is created with the commercial SPECCTRA autorouter on the same machine. The results for design MCM1 are shown in Table 1.

We compare the results of our pin assignment algorithms amongst each other with a special emphasis on the results of the pin assignment algorithm of Section 4.4. This is reasonable because the algorithm of Section 4.4 finds the optimal pin assignment with respect to a linear objective. That is, when optimizing *SHPWL* and *AVG Flylines*, the generated pin assignment is the global optimum with respect to either *SHPWL* or *AVG Flylines*. Therefore, we use those results to evaluate the other automatic pin assignments. Table 1 depicts the difference between the individual results and the global optima as a percent of the global optimum for *SHPWL* and *AVG Flylines*. The results of the pin assignment algorithm of Section 4.1 are the reference for the remaining quality metrics (*HPWL MATCH* and *STD Dev*).

It is important to note that the manual creation of an optimized pin assignment for the above test designs requires 1–2 months, with results similar to those of the algorithms presented in Sections 4.1 and 4.3 [17]. The algorithms presented in this paper reduce this time to days.

The experimental results prove that, except for algorithm 4.2, the results are close (differences ≤ 7.0 %) to the optimum for *SHPWL* and *AVG Flylines*.

Considering *HPWL MATCH* and *STD Dev*, algorithm 4.1 gives the best results. Minimum flyline crossings are obtained by algorithm 4.3 which always removes *all* crossings or by algorithm 4.4 which optimizes net length and flyline crossings in parallel. Algorithm 4.2 is not appropriate for design MCM1 because the nets are arranged radial and there does not exist a preferred direction for the so-called escape routing. Reasonable

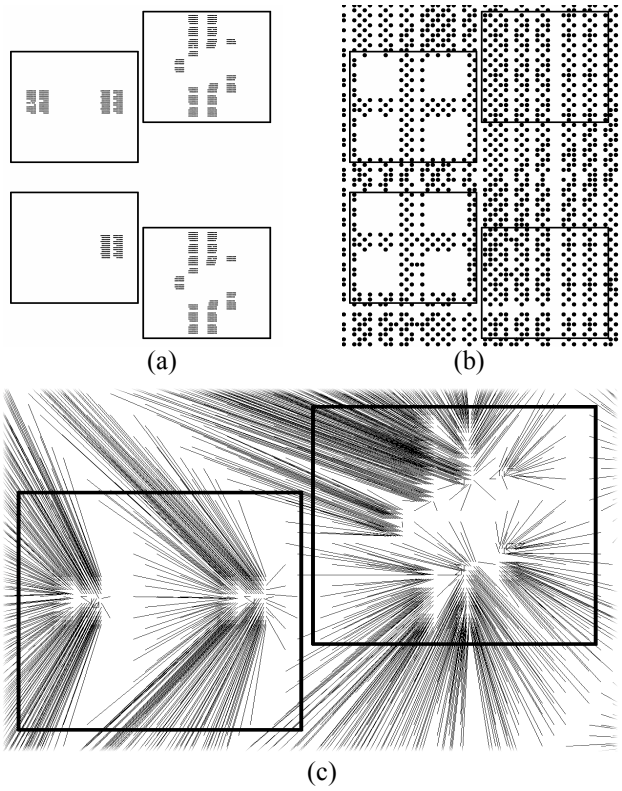

Figure 5. Details of multi chip module design MCM1. (a) Footprint of signal pins of the chips. (b) Footprint of the signal pins of the chip carrier. (c) Sample pin assignment of the two upper chips illustrated by flylines.

results were efficiently obtained with algorithm 4.2 for complex designs with a topology comparable to the one in Figure 1.

The results also show that our metrics *SHPWL* and *AVG Flylines* are well related to the actual routing length and the number of vias (Figure 6). For example, the pin assignment with the least *SHPWL* results in the shortest actual routing length as well. The same holds for the correlation of the standard deviation of the lengths of the routed nets (*STD Dev Routed*), *STD Dev* and the standard deviation of the Manhattan lengths of the nets. A correlation between the flyline crossings and the via count is not observable for design MCM1 because each net is routed on a plane pair with pre-assigned preferred routing directions.

The presented pin assignment algorithms create pin assignments having different qualities. None of the presented algorithms can be identified as superior to the other algorithms in all aspects. Therefore, the best choice of pin assignment algorithm for a certain design depends on the individual requirements of the respective layout. For designs that above all require shortest wirelengths, algorithm 4.4 gives best results. For the least signal intersections and thus minimum routing layers, algorithm 4.3 is preferable. The best matching of wirelengths is achieved by algorithm 4.1. Finally, algorithm 4.2 creates pin assignments that respect a preferred direction of signal fan-out.

Table 1. Experimental results for test design MCM1.

PAA #	*SHPWL*	*HPWL MATCH*	*AVG Flylines*	*STD Dev*	Flyline Crossings	PAA Time (in m:s)	Routed Length	Vias	STD Dev Routed	Routing Time (h:m:s)
1	*+4.1%*	**94553**	+4%	**6.96**	**18272**	**0:01**	**51495**	**5259**	**9.7**	**0:16:11**
2	+132%	+228%	+123%	+180%	**86847**	**0:01**	+141%	+72%	+221%	**2:49:06**
3	+3.2%	+38%	+2.1%	+16%	**0**	**0:17**	-1.2%	-2.8%	+17%	**0:11:59**
4	+62%	+263%	+56%	+122%	**0**	**0:13**	+56%	+16%	+131%	**0:30:31**
5	+0.8%	+48%	**12.38**	+18%	**240**	**4:14**	-3.3%	-4.7%	+19%	**0:15:38**
6	**47734**	+63%	+7.0%	+36%	**63500**	**2:14**	-2.8%	-5.2%	+28%	**0:15:40**
7	+0.0%	+46%	+0.9%	+16%	**0**	**2:52**	-3.8%	-4.2%	+16%	**0:12:03**

Table 2. Pin assignment algorithms (PAA).

PAA #	Pin Assignment Algorithm
1	Section 4.1
2	Section 4.2
3	Section 4.3 (Initial pin assignment created with Section 4.1)
4	Section 4.3 (Initial pin assignment created with Section 4.2)
5	Section 4.4 $l_{ij} = \sqrt{dx^2 + dy^2}$ / $\alpha = 0$
6	Section 4.4 $l_{ij} = dx + dy$ / $\alpha = 0$
7	Section 4.4 $l_{ij} = dx + dy$ / $\alpha = 0.1$

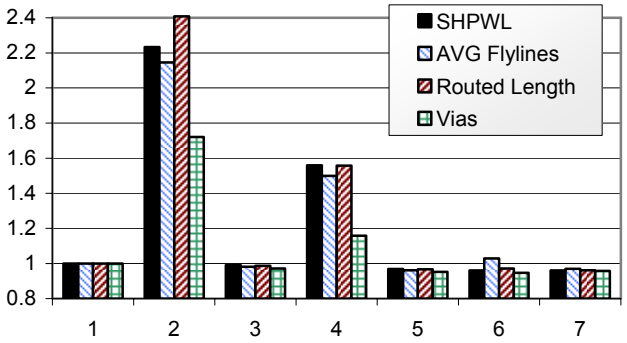

Figure 6. Correlation of two of our estimation metrics (*SHPWL*, *AVG Flylines*) and actual routing results (*Routed Length*, *Vias*) of algorithms (PAA #) 1–7. Values are normalized to the respective results of PAA# 1.

6. Summary

In this paper we have presented four pin assignment algorithms that have proven to give good results for PCB, MCM and SCM pin assignment tasks. To the best of our knowledge, our approaches allow for the first time an automatic pin assignment of components with several thousand pins each.

The presented algorithms are currently in use in the design flow for industrial designs at IBM where they have shown their robustness and quality, combined with an impressive improvement in time efficiency.

Only a few objective metrics for pin assignment quality have been known so far. Therefore, in addition to the well established HPWL, we have introduced flyline crossings, matched wirelengths, the length of the flylines and the standard deviation of the length of flylines as quality estimation metrics for pin assignment. In the future, generally accepted benchmarks and additional, time-efficient objective metrics are desirable to better evaluate the quality of pin assignments.

References

[1] N. L. Koren. Pin assignment in automated printed circuit board design. *Proc. of 9th Workshop on Design Automation*, 72–79, 1972.

[2] L. Mory-Rauch. Pin assignment on a printed circuit board in Design Automation. *Proc. of the 15th Conference on Design Automation*, 70–73, June 1978.

[3] T. D. Am, M. Tanaka, Y. Nakagiri. An approach to pin assignment in printed circuit board design. *ACM SIGDA Newsletter*, 10(2):21–33, 1980.

[4] H. Brady. An approach to topological pin assignment. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 3(3):250–255, July 1984.

[5] Y. Cai, D. Wong. Optimal channel pin assignment. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 10(11):1413–1424, Nov. 1991.

[6] T. Koide, S. Wakabayashi, N. Yoshida. An optimal channel pin assignment with multiple intervals for building block layout. *EURO-DAC '92, EURO-VHDL '92*, 348–353, Sept. 1992.

[7] J. Westra, P. Groeneveld. Towards integration of quadratic placement and pin assignment. *Proc. IEEE Computer Society Annual Symposium on VLSI, 2005*, 284–286, May 2005.

[8] S.Chen, W. Tseng, J. Yan, S. Chen, Printed circuit board routing and package layout codesign. *APCCAS 2002* 1:155–158, 2002.

[9] Y. Kubo, A, Takahashi. A global routing method for 2-layer ball grid array packages. *Proc. of the 2005 ISPD*, 36–43, 2005.

[10] M. Yu, W.W.-M. Dai. Pin assignment and routing on a single-layer pin grid array. *Proc. of the ASP-DAC '95/CHDL '95/VLSI '95*. 203–208, 1995.

[11] M. Pedram, M. Marek-Sadowska, E. Kuh. Floorplanning with pin assignment. *ICCAD-90, Digest of Technical Papers, 1990*, 98–101, Nov. 1990.

[12] M. Pedram, K. Chaudhary, E. Kuh. I/O pad assignment based on the circuit structure. *Proc. ICCD '91*, 314–318, Oct. 1991.

[13] Z. Galil. Efficient algorithms for finding maximum matching in graphs. *ACM Comput. Surv.*, 18(1):23–38, 1986.

[14] H. N. Gabow. An efficient implementation of Edmonds algorithm for maximum matching on graphs. *Journal of the ACM*, 23(2):221-234, 1976.

[15] S. Hauck, G. Borriello. Pin assignment for multi-FPGA systems. *IEEE Transactions on CAD of Integrated Cirtcuits & Systems*, 16(9):956-964, Sept. 1997.

[16] Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2:83-97, 1955.

[17] IBM Research & Development, Boeblingen, Germany, Internal Documents.