# Introducing Irregularity to Routing Architecture of Structured ASIC for Better Routability

Insup Shin, Donkyu Baek, and Youngsoo Shin

*Department of Electrical Engineering, KAIST*
*Daejeon 305-701, Korea*

***Abstract*—Imposing regularity presents a fundamental limitation to any structured ASIC, or more generally any programmable logic device. It has been recently shown that irregularity can be introduced in structured ASIC, in particular in programmable logic elements of structured ASIC, through a special photolithography process, and the degree of irregularity can be customized for each particular design by manufacturing a few extra masks. We experiment how irregularity can be introduced to routing architecture of structured ASIC. When a whole routing area is made of an array of two routing architectures, the area is reduced by 8% to 16% (compared to standard structured ASIC) due to less white space, which is made possible by improved routability; the total wirelength is reduced by 6% to 14%. The new routing architectures and routing algorithm specific to the architectures are presented.**

## I. Introduction

A structured ASIC, or more generally a programmable logic device, consists of an array of programmable logic elements [1]–[3], or simply called tiles. Since all tiles have the same architecture, redundancy is inherently involved, i.e. some part of tiles are left unused; the amount of redundancy is different for different applications. Redundancy may be reduced if tiles of more than one architecture comprise an array, and the array is customized for different applications. This is made possible through multiple exposures during lithography process and using a blocking mask [4].

This concept is explained in Fig. 1. Consider the two arrays of tiles, of type Tile 1 and Tile 2. Each tile is associated with a special blocking mask. We first take Tile 1 and its blocking mask and perform photolithography. The tiles that are aligned with white regions in the blocking mask are patterned on the wafer, while those aligned with black regions are not. We now repeat photolithography using Tile 2 and its blocking mask on the same wafer. The result is a die, shown on the bottom right corner of Fig. 1, that contains some tiles of type Tile 1 and some of type Tile 2; how they are mixed and how they are located are determined by the blocking masks. Current photolithography equipment does not allow two masks to be arranged back to back; however, the same concept can be realized through a double exposure. Manufacturing time increases in this method: 20% longer if two types of tiles are used, and 40% when three are used [4].

Irregularity has been experimented with tile architectures in

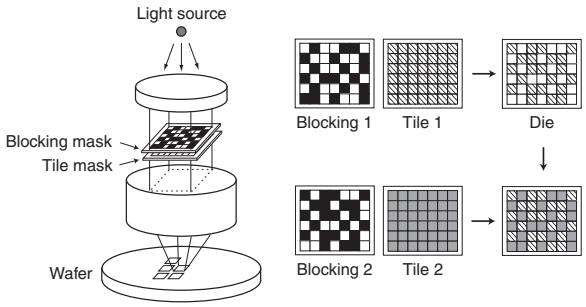


Fig. 1. Multiple exposures using blocking masks.

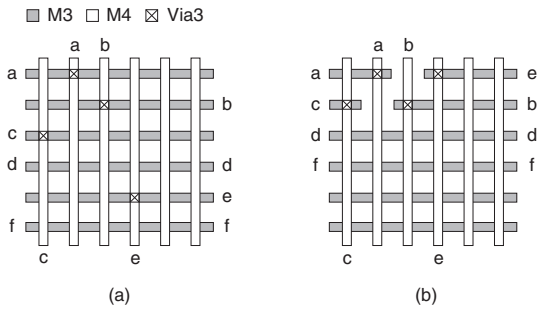


Fig. 2. (a) Standard regular routing architecture and (b) a routing architecture, in which some horizontal M3 tracks are broken to improve routability.

[4]. In this paper, we aim to introduce irregularity to routing architecture instead. Consider Fig. 2(a), which is a grid made of M3 and M4 metal layers and put on top of each tile. Each horizontal M3 track is occupied exactly by one net. If the first two tracks are broken as shown in Fig. 2(b), each of them can now accommodate two nets; two tracks are left unoccupied as a result, which improves routability of a whole design. If the first M3 track needs to be occupied by a net f, the architecture of Fig. 2(a) is a better choice because, in Fig. 2(b) architecture, connection is made with detour.

The question therefore is how we use both routing architectures together. Imagine that one mask consists of an array of routing architecture in Fig. 2(a), just as Tile 1 in Fig. 1, and an array of Fig. 2(b) comprises another mask. Using two blocking masks designed for a particular application, we can mix two routing architectures, which we experiment in

**978-1-4673-2845-6/12/$31.00** 

this paper. Notice that manufacturing time should increase marginally in this method because multiple exposure is applied only to M3 layers, as opposed to multiple exposure being applied to more than one layers in [4].

The remainder of this paper is organized as follows. In Section II, routing algorithm specific to the proposed routing architecture is presented. The basic idea is extended in two ways in Section III: when more than one metal track are broken, and when metal track is broken in more than one way. Experimental results are presented in Section IV, and we summarize the paper in Section V.

## II. Routing Architecture and Algorithm

### A. Architecture

Fig. 3(a) illustrates a standard routing architecture, which we assume for conventional structured ASIC design. Each grid is made of horizontal M3 tracks and vertical M4 tracks, and is aligned with one tile placed below. To make a connection between M3 tracks in adjacent grids, an array of M2 segments is placed, and then the actual connections are made by placing vias. Similarly, an array of M3 segments, which is not shown in the figure, is used to connect M4 tracks in adjacent grids located in vertical direction. If more metal layers are needed, a similar routing grid can be made from M5 and M6 tracks, and M4 and M5 are connected using vias.

The customized routing architecture that we propose is shown in Fig. 3(b), in which we assume that only M3 can be broken in some tracks. Notice that M3 is broken in different ways in two adjacent grids, so that we exploit more flexibility. The two routing architectures are supported by two separate masks as shown in Fig. 3(c), and their mix is implemented by using blocking masks. There are many different ways to customize M3 layers, and M4 layers can also be customized; these options are explored in Section III.

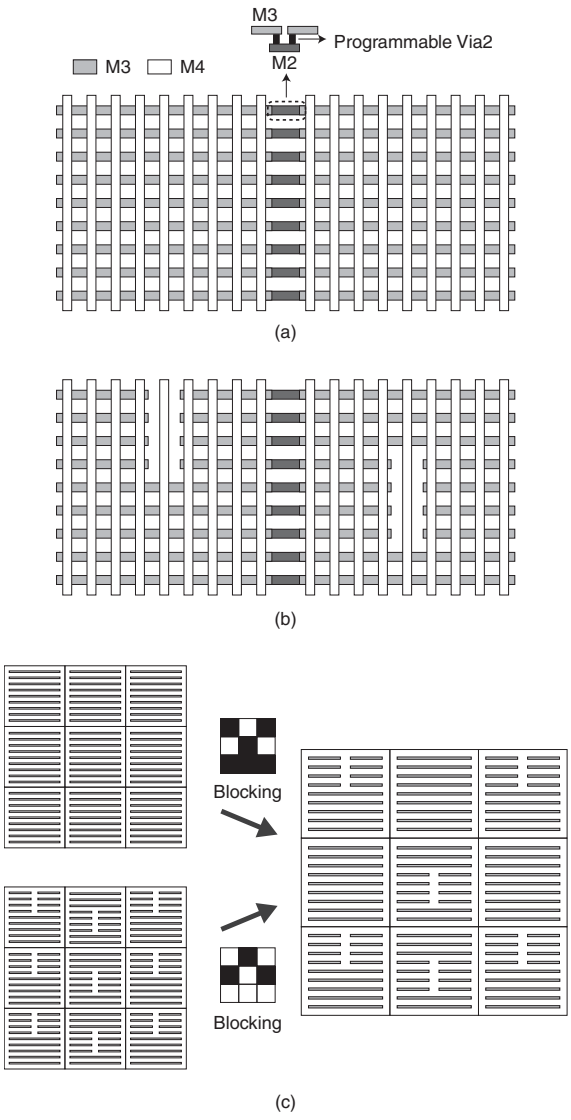


Fig. 3. (a) Standard routing architecture, (b) customized routing architecture to improve M3 utilization, and (c) mix of standard and customized routing architectures using blocking masks.

### B. Algorithm

Before we actually perform routing, we need to determine how standard and customized routing architectures, shown in Fig. 3, are mixed together, i.e. which routing architecture is assigned to each tile. This is done through trial global routing using virtual routing architecture which is shown in Fig. 4(a). It is similar to our customized routing architecture, except that broken M3 tracks can be re-connected (if it is needed) using a wire segment named virtual jumper. Note that virtual jumper, which uses M2 track, is not supported in real implementation, because a tile layout that implements logic function uses M2 as well as M1. Depending on the presence of virtual jumper, a virtual routing architecture encompasses both standard and customized routing architectures.

We employ the VPR router [5] for global routing. It initially routes each net using a maze router, finding the shortest path regardless of routing capacity. At this stage, a virtual routing architecture is divided into several regions, and each region is associated with a node with number indicating routing capacity. Fig. 4 illustrates how regions are identified for M3 tracks. The edge between nodes models potential connection using a via. The regions for M4 tracks can be identified in similar way, and corresponding nodes are also shown in Fig. 4(b); the edges between M4 nodes and M3 nodes are not shown for simplicity of presentation. After maze routing, the router reroutes each net in sequence, based on a cost derived from overuse of the routing resource [5]; this process is repeated until all nets have been successfully routed.

The result of trial global routing using virtual routing architecture guides us to choose a real routing architecture (either standard or customized) for each routing grid. This is done through the computation of potential overflow. Fig. 5(a) corresponds to a result of trial global routing. If the same routing is implemented on a standard routing architecture as shown in Fig. 5(b), two nets (j and k) cannot be connected causing two overflows. Fig. 5(c) shows that a customized routing architecture yields one overflow (net d). We thus

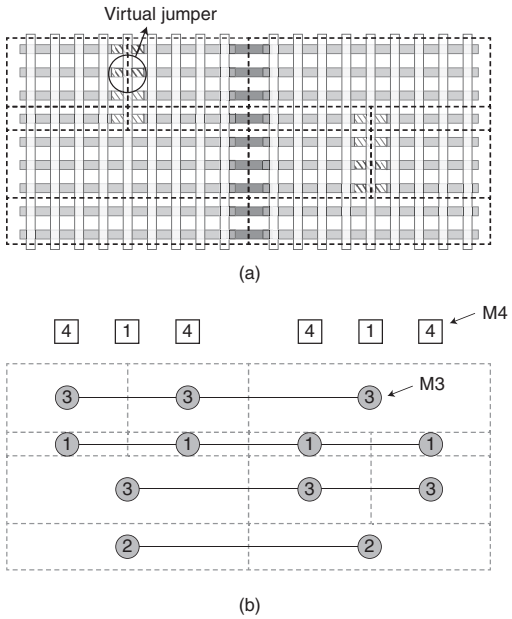


Fig. 4. (a) A virtual routing architecture, and (b) a graph modeling routing resources and their connections.

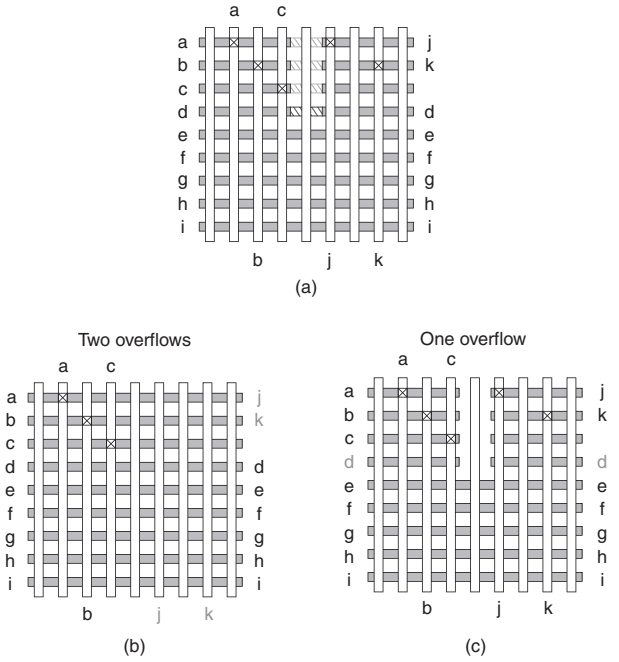


Fig. 5. Overflow computation: (a) global routing using virtual routing architecture, (b) the same routing when standard routing architecture is assumed, and (c) the same routing with customized routing architecture.

choose a customized routing architecture for this particular grid. If connection of all nets can be accomplished in both routing architectures, a customized routing architecture is assumed so that more routability is provided.

A global routing is performed one more time, which is then followed by a detailed routing. Each row of tiles is taken one by one and detailed routing is performed in horizontal direction; the process repeats for each column of tiles for

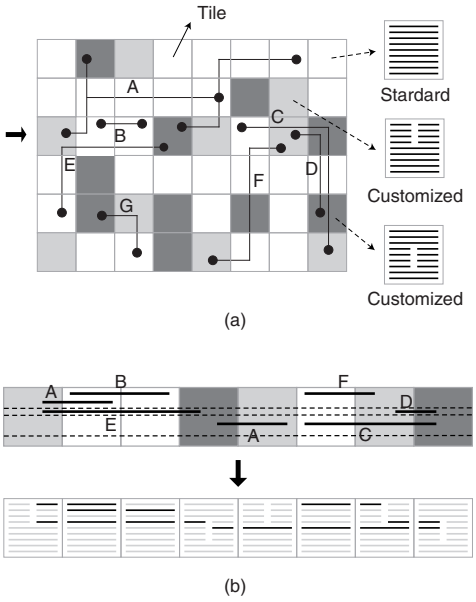


Fig. 6. (a) Global routing result and (b) LEA-based detailed routing.

vertical connections. As shown in Fig. 6(b), each net is represented by an interval within a row (or column) of tiles. The detailed routing problem can then be solved by the optimal left-edge algorithm (LEA) [6].

## III. Extensions

### A. Multiple Metal Layers

We assumed that only M3 is broken in some routing tracks in the previous section. The idea can be extended to more than one metal layer, e.g. both M3 and M4 are broken. The routing algorithm remains almost the same as in Section II-B.

The calculation of overflow and subsequent choice of real routing architecture (standard or customized) are done on M3 and M4 layers independently.

*1) Sharing Blocking Masks:* We need four blocking masks if both M3 and M4 are broken for customized routing architecture: two to select either standard or customized in M3 layer; another two in M4 layer. Blocking masks may be shared between M3 and M4 layers to reduce mask cost, but this comes at a cost of extra constraint on the selection of routing architecture.

Consider Fig. 7. A trial global routing is first performed using virtual routing architecture. At each grid, the amount of overflow is calculated for each layer, and appropriate routing architecture is chosen following the same method in Section II-B. Assume that the result is given in Fig. 7(b), each grid is associated with one routing architecture (A or B) for M3 and another (C or D) for M4. Since each of two blocking masks is shared by M3 and M4 layers, if one blocking mask is used to choose A for M3 and C for M4, for example, another blocking mask must be used to choose B for M3 and D for M4. Therefore, there are two combinations on how blocking masks are used: $\{$(A, C), (B, D)$\}$ and $\{$(A, D), (B, C)$\}$. The

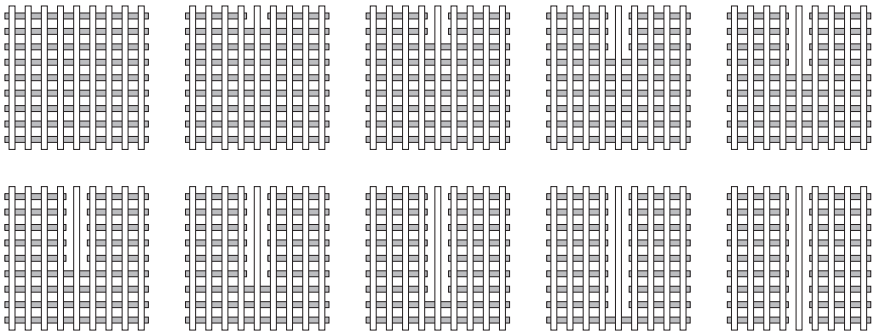

Fig. 8. Various customized routing architectures in addition to a standard routing architecture.

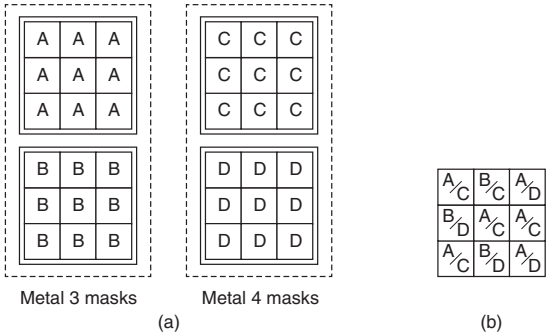


Fig. 7. (a) Metal masks: A and C contain standard routing architecture, and B and D correspond to customized routing architecture, and (b) a partial result of routing algorithm.

TABLE I
BENCHMARK CIRCUITS

| Group | Circuit | # Tiles | # Nets |
|---|---|---|---|
| ITC | b15 | 2824 | 3723 |
| | b17 | 8635 | 11629 |
| | b18 | 10616 | 14421 |
| OpenCores | aes | 3843 | 4513 |
| | aes_core | 6931 | 10472 |
| | aquarius | 7974 | 10260 |
| | oc54 | 4422 | 6580 |
| | pci_bridge | 11952 | 11854 |
| | tv80 | 2589 | 3596 |
| | ucore | 6245 | 7684 |
| | usb_funct | 7405 | 8714 |
| | warp | 10486 | 14545 |

number of times all the elements of each combination is used is counted. In Fig. 7(b), there are 4 instances of (A, C), 2 instances of (B, D), 2 of (A, D), and 1 of (B, C); the first combination is used 6 times in total while the second is used in 3; we thus decide that blocking masks are shared following the first combination. The grids with (A, D) and (B, C) are now replaced by one of (A, C) and (B, D) that comes with less overflow.

### *B. Multiple Customized Routing Architectures*

There are many different ways to break M3 or M4 tracks, other than the method of Fig. 3(b). Fig. 8 shows, in addition to standard routing architecture, 9 different customized routing architectures. The problem is to select 2 routing architectures out of 10 (together with how they are mixed) in a way that routing is best performed.

This can be done by an algorithm similar to Section II-B. A trial global routing is again performed using a virtual routing architecture. We assume each routing architecture one by one, and check each grid for overflow. The two routing architectures are then chosen such that the number of grids, which cause overflow in both routing architectures, is minimized.

## IV. EXPERIMENTS

Test circuits were prepared using ITC benchmarks and open cores [7]. They are listed in Table I. Each circuit was implemented using structured ASIC proposed in [4], which is based on 45-nm CMOS technology [8]. The width of one tile corresponds to 12 M4 tracks; the height is 11 M3 tracks. Routing architecture was designed accordingly.

Routability can always be improved by allocating white space. An appropriate amount of white space can be determined by a few iterations of placement (of tiles and white space together) and subsequent routing. The resulting layout area and total wirelength are used to assess various routing architectures.

### *A. Result and Analysis*

The area and wirelength of each test circuit when a standard routing architecture (see Fig. 3(a)) is employed are shown in columns 2 and 3 of Table II. The next two columns denote the corresponding figures when both standard and customized routing architectures (Fig. 3) are employed. The area is reduced by 8% on average due to less white space, which in turn is made possible by improved routability; the total wirelength is reduced by 6% on average.

The result when two routing architectures are chosen following the method of Section III-B is presented in columns 6 and 7. Routability should improve since the best two architectures are chosen, but it is somewhat canceled out due to smaller area, which causes more congestion and some detour of connections. The last two columns present the area and wirelength when virtual routing architecture (see Fig. 4) is assumed; they serve as references to assess the results of columns 4–7.

TABLE II
COMPARISON OF VARIOUS ROUTING ARCHITECTURES. IN CUSTOMIZED ARCHITECTURE, ONLY M3 IS CONSIDERED

| Circuit | Standard | | Standard + Customized | | Choice of two routing architectures | | Virtual | |
|---|---|---|---|---|---|---|---|---|
| | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) |
| b15 | 20364 | 121 | 18803 | 115 | 18803 | 113 | 17465 | 108 |
| b17 | 67706 | 430 | 62074 | 413 | 62074 | 410 | 62074 | 407 |
| b18 | 76284 | 493 | 76284 | 476 | 76284 | 472 | 76284 | 466 |
| aes | 36881 | 208 | 36881 | 194 | 33203 | 197 | 33203 | 190 |
| aes_core | 149252 | 539 | 99561 | 477 | 85362 | 440 | 85362 | 431 |
| aquarius | 137385 | 552 | 114519 | 532 | 98183 | 488 | 85931 | 468 |
| oc54 | 47711 | 260 | 42422 | 247 | 42422 | 243 | 38190 | 236 |
| pci_bridge | 68721 | 395 | 68721 | 379 | 68721 | 382 | 68721 | 373 |
| tv80 | 20367 | 122 | 20367 | 117 | 20367 | 116 | 20367 | 114 |
| ucore | 59863 | 298 | 53888 | 295 | 53888 | 291 | 53888 | 279 |
| usb_funct | 45659 | 285 | 45659 | 271 | 42561 | 271 | 42561 | 264 |
| warp | 75372 | 394 | 64606 | 333 | 60308 | 346 | 53227 | 328 |
| Average | 1.00 | 1.00 | 0.92 | 0.94 | 0.88 | 0.94 | 0.85 | 0.90 |

TABLE III
COMPARISON OF VARIOUS ROUTING ARCHITECTURES. BOTH M3 AND M4 ARE CONSIDERED FOR CUSTOMIZED ARCHITECTURE

| Circuit | Standard + Customized | | Standard + Customized (shared blocking masks) | | Choice of two routing architectures | | Virtual | |
|---|---|---|---|---|---|---|---|---|
| | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) | Area ($\mu m^2$) | Wirelength (mm) |
| b15 | 18803 | 110 | 18803 | 116 | 17465 | 98 | 17465 | 96 |
| b17 | 62074 | 408 | 62074 | 413 | 57309 | 367 | 57309 | 344 |
| b18 | 76284 | 472 | 76284 | 477 | 76284 | 436 | 70427 | 410 |
| aes | 33203 | 191 | 33203 | 193 | 33203 | 188 | 30192 | 170 |
| aes_core | 99561 | 474 | 99561 | 477 | 85362 | 426 | 85362 | 389 |
| aquarius | 98183 | 484 | 98183 | 498 | 98183 | 460 | 85931 | 425 |
| oc54 | 38190 | 250 | 42422 | 239 | 38190 | 225 | 38190 | 212 |
| pci_bridge | 64435 | 362 | 64435 | 379 | 64435 | 341 | 64435 | 322 |
| tv80 | 20367 | 117 | 20367 | 115 | 18676 | 108 | 18676 | 100 |
| ucore | 53888 | 280 | 53888 | 283 | 53888 | 268 | 49000 | 243 |
| usb_funct | 42561 | 269 | 42561 | 271 | 42561 | 242 | 42561 | 237 |
| warp | 56546 | 329 | 60308 | 341 | 50278 | 321 | 47638 | 285 |
| Average | 0.87 | 0.92 | 0.88 | 0.94 | 0.84 | 0.86 | 0.81 | 0.80 |

Table III presents the results when both M3 and M4 are involved in customized routing architecture. The comparison of using standard and customized routing architectures in Table II and Table III reveals the benefit in the latter both in area and wirelength. Columns 2–5 indicate that sharing blocking masks, i.e. using 2 blocking masks (columns 4–5) over 4 (columns 2–3), is not a bad option given little difference of area and wirelength. Finally, choosing the best two routing architectures, both in M3 and M4, gives substantial benefit of area and wirelength, as indicated by columns 6–7.

## V. CONCLUSION

We have shown that introducing irregularity to routing architecture of structured ASIC greatly improves routability, which helps reduce both circuit area and wirelength. The increase of manufacturing time is expected to be marginal since multiple exposures are performed only on M3 and M4.

Several customized routing architectures have been experimented with; the search for the best architecture remains a quest.

## ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2010-0013439).

## REFERENCES

[1] K. Y. Tong *et al.*, "Regular logic fabrics for a via patterned gate array (VPGA)," in *Proc. CICC*, Sept. 2003, pp. 53–56.
[2] N. Shenoy, J. Kawa, and R. Camposano, "Design automation for mask programmable fabrics," in *Proc. DAC*, June 2004, pp. 192–197.
[3] Y. Ran and M. Marek-Sadowska, "Designing via-configurable logic blocks for regular fabric," *IEEE Trans. on VLSI Systems*, vol. 14, no. 1, pp. 1–14, Jan. 2006.
[4] D. Baek, I. Shin, S. Paik, and Y. Shin, "Selectively patterned masks: structured ASIC with asymptotically ASIC performance," in *Proc. ASPDAC*, Jan. 2011, pp. 376–381.
[5] V. Betz and J. Rose, "VPR: a new packing, placement and routing tool for FPGA research," in *Proc. FPL*, Sept. 1997, pp. 213–222.
[6] A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures," in *Proc. DAC*, 1971, pp. 155–169.
[7] "Opencores," Available http://www.opencores.org/.
[8] "Nangate 45nm open cell library," Available: http://www.nangate.com/.