A Mask Reuse Methodology for Reducing System-on-a-Chip Cost

Subhrajit Bhattacharya, John Darringer, Daniel Ostapko, Youngsoo Shin[†]

IBM T.J. Watson Research Center, Yorktown Heights, NY
{sbhat, jad}@us.ibm.com
[†]Dept. of EE, Korea Advanced Institute of Science and Technology, Republic of Korea
youngsoo@ee.kaist.ac.kr

Abstract

Today's System-on-a-Chip (SoC) design methodology provides an efficient way to develop highly integrated systems on a single chip by utilizing pre-designed intellectual property (IP) or "cores". However, once assembled, the physical design and manufacturing process that follows does not benefit from the reuse of these cores. We propose an alternative Mask Reuse Methodology (MRM) where most cores are provided with hardened layouts, significantly reducing the number of components for chip-level processing and the associated turn-around time. In addition, each core has a pre-verified mask set, which can be re-used to significantly reduce the overall mask cost and mask manufacturing time. Since mask cost and design and verification times are rapidly becoming prohibitive for low or even medium volume ASIC parts, the proposed MRM methodology can help reduce the barrier for ASIC starts. We provide details of the methodology, as well as an assessment of its impact on design time and design cost with an example of a network processor SoC.

1. Introduction

Continued technology scaling is enabling more and more devices to be placed on a single chip. However, the design effort and manufacturing cost for such a highly integrated application-specific integrated circuit (ASIC) chip is also growing to the point where only a high volume product may be economically viable. At the 90 nm node, it is projected that the mask cost for an ASIC will be more than a million dollars, compared to three hundred thousand dollars in the 180 nm node [1], [2].

To address the need for increased design productivity, semiconductor companies have developed a System-on-a-Chip (SoC) design methodology that integrates pre-designed and pre-verified components, called intellectual property (IP) or cores, into single chip designs. Many semiconductor companies offer SoC platforms composed of a set of commonly used cores pre-assembled as a complete design to provide an even faster path to implementation [3]. The architect may use such a platform for the bulk of the design and may add custom logic to complete the RTL design. The final RTL description then enters a detailed implementation phase that consists of logic synthesis, floorplanning, and physical design [4]. Most of the cores used in a SoC are usually *soft* cores provided as synthesizable RTL description. This allows some customization during implementation, but each core must be re-synthesized, placed and wired each time it is used in a new design. Moreover, many designs are processed by the design tools as a flat netlist, ignoring the SoC hierarchy. Hence the reusability of an IP block is almost completely ignored after the RTL phase.

This SoC methodology does not benefit from core reuse in the manufacturing process either. Each new SoC product requires a complete and expensive new mask set. Overall, while today's SoC design methodology does take advantage of pre-designed platforms and cores to significantly reduce the time spent at the architectural stage; it does little to reduce the complexity and cost of the later physical implementation. This inability of exploiting reuse concept at the physical implementation stage is becoming more significant as critical dimensions continue to decrease and the resulting increase in mask costs challenges the economic viability of the ASIC paradigm.

In this paper we propose a novel technique called *Mask Reuse[5]* to lower both the costs and turnaround time (TAT) associated with mask manufacture. We describe a *Mask Reuse Methodology (MRM),* where most cores are provided with hardened layouts to simplify and speed-up the physical design process as well as reducing product cost by exploiting mask reuse during manufacturing.

The remainder of the paper is organized as follows. In the next section, we discuss the cost of ASIC manufacturing. In Section 3 we consider alternative approaches for reducing design time and costs, and in Section 4 we present a new mask reuse methodology for reducing design and mask cost with minimal impact on area and performance. To establish a base for comparison, we define a standard SoC design methodology in Section 5. Section 6 details the proposed mask reuse methodology and in Section 7 we compare the cost benefits of the two

Proceedings of the Sixth International Symposium on Quality Electronic Design (ISQED'05)
0-7695-2301-3/05 $ 20.00 IEEE

approaches on an example of a network processor implemented as an SoC. Finally conclusion follows in Section 8.

2. Cost of ASIC Design and Manufacturing

The cost of ASIC production can be divided into two components: design costs and manufacturing costs. Design costs include those incurred during high-level design, logic implementation, functional verification, and physical design. The costs in the stages of mask preparation and production, manufacturing, and test constitute manufacturing costs.

The design cost of developing an ASIC can vary from $10 million to $20million, the latter number being given in the International Technology Roadmap for Semiconductors (ITRS) for a low-power PDA SoC product. Without continued advances in design tools and methodologies, the design cost for such chips would continue to increase at a 39% CGR, and will be more than $50M to $100M by the year 2014.

Similarly, the number of masks required for an ASIC chip is increasing and can be considerably more than 30 for advanced process technologies such as 90nm and below. Moreover, the complexity of each mask is growing fast as feature sizes are shrinking, further increasing mask costs. High-end mask sets can cost more than $100K per mask. If more than 30 masks are required, the total mask cost will exceed $3M. The need for re-spins can double or triple the mask costs.

Our MRM reduces design cost and time especially during physical design as well as reduces mask cost and time.

3. Alternative Approaches

Since building a standard cell ASIC is extremely expensive, only a high volume product may be economically viable. However, there are alternative approaches, especially for low volume ASICs. An FPGA which requires no additional manufacturing beyond electrical customization, is a popular alternative. However, the unit cost of an FPGA is relatively high and an FPGA is orders of magnitude inferior to standard cell-based ASICs in area, power consumption and performance [6]. As a result, FPGAs are not viable for high-performance and/or medium to high volume designs.

Gate array-based solutions are also popular. They have an advantage by mixing comparable design costs with relatively low manufacturing costs, since they require custom masks only for the metal layers. While gate arrays have performance-area-power characteristics closer to standard cell-based ASICs, they are not well suited for functions typically implemented in custom microprocessors or SRAMs, since device-level optimization can not be performed.

Structured ASICs may alleviate the limitation of gate array-based approaches by combining the flexibility of gate arrays with the performance and area efficiency of custom blocks [7]. Via-Programmable Gate Array (VPGA), for example, uses only via patterns to customize a gate array [1]. The performance is shown to be very close to standard cell-based designs, but at the cost of 50% area penalty and at similar extra cost of power consumption. There are hybrid architectures that combine structured ASICs and FPGA cores on the same platform[8]. However, both structured ASICs and hybrid architectures are not economical enough for medium to high volume parts and/or low-power applications.

The mask-reuse approach to design and manufacturing has area/power/delay characteristics within a few percent of standard cell-based ASICs. Furthermore, it has a significant advantage in view of cost, which makes it ideal solution for critical or low power and low or medium volume parts. Since the trend is towards higher memory content and including one or more embedded processors in the chip, we believe MRM could become a technology of choice for future SoC designs.

4. Strategy for Mask Reuse

The strategy for mask reuse requires the following: the SoC logic is available either as a hard core or it will be implemented with gate array logic. Each hard core has its own set of masks which capture all the logic design, circuit tuning, layout optimization, and lithographic resolution enhancement processing. If a core is contained in a rectangular layout region, then the same region from each of the masks in the set can be used to expose the core. Thus the masks are reusable. If a chip can be constructed predominantly from existing cores, then the bulk of the time required to generate the mask data and to verify the masks can be eliminated. In addition, if a core is to be reused, more effort can be justified in optimizing it and characterizing its behavior and performance, which can increase the probability of correct behavior and performance for the total design.

The logic not available as hard cores is exposed from a mask that provides the patterns to generate the gate-array structure. The areas occupied by the hard cores are covered when exposing the wafer to the gate-array masks. The unique masks required for the chip would be those that describe the interconnections between macros and those required to personalize the gate array structures. Thus, all of the FEOL (Front End of Line) masks would be reused and only a small number of the BEOL (Back End of Line) masks would be required. Since these masks often have features that are less critical, the approach should significantly reduce the mask cost of producing a

unique chip. Several changes are required in the layout generation and manufacturing processes to accommodate reusable masks and they are discussed next.

4.1 Manufacturing Process Modifications for MRM

The mask exposure process involves projecting light through a reticle onto the wafer. The entire mask is projected onto the wafer in sections or fields. Typically, a single reticle is used which contains the pattern for the entire chip. The MRM employs a *partitioned mask (PM)* in which more than one reticle or parts of the same reticle are used in exposing a chip. This is accomplished with shutters or blades that limit the transmission to only a portion of the mask. Alignments also become necessary.

Since the alignment within a partition is similar to the standard method, additional misalignment would be expected only between partitions. If necessary, modifications could be made along the boundaries of the partitions to overcome these potential misalignments. Misalignment or blurring of shapes may occur as shown in Figure 1(a) and in Figure 1(b) respectively. These problems can be overcome by widening the connections as shown in Figure 1(c).

Allowing porosity for through connections may require additional space to allow for misalignment as shown in Figure 2. These modifications can be handled with a combination of rules and post-processing.

Since the cores are being reused, extra effort will be necessary to assure that they will work in different environments. The form factor should be rectangular to be consistent with the reticle masking capability. Accommodations for power and through metal porosity for signals must be defined as well as the location of ports.

Although there are many factors in the cost equation, the percentage increase in manufacturing cost (as opposed to mask production cost) should be relatively small.

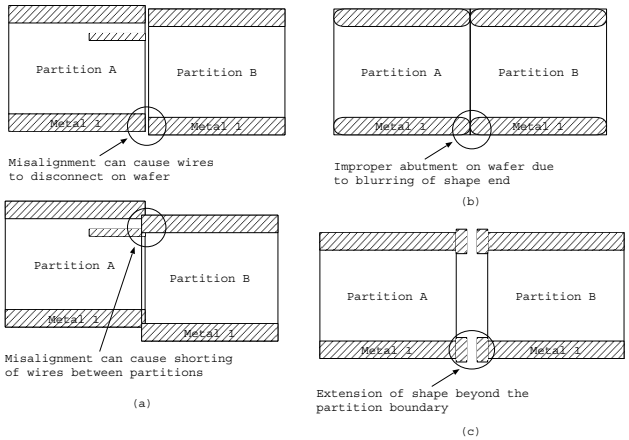

Figure 1: (a) Misalignment, (b) blurring, and (c) extension.

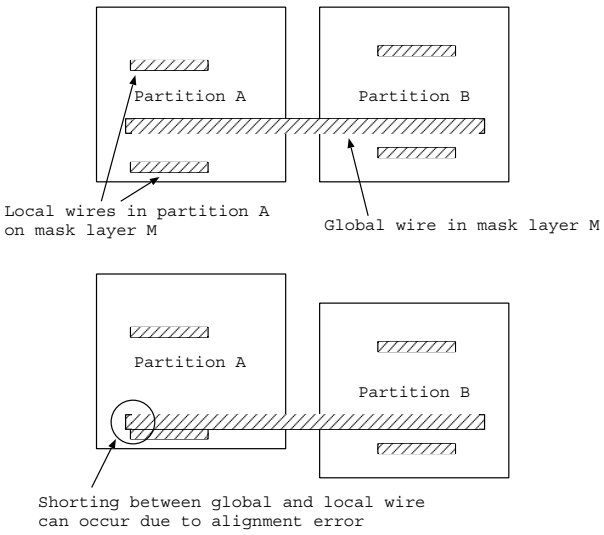

Figure 2: Global wiring.

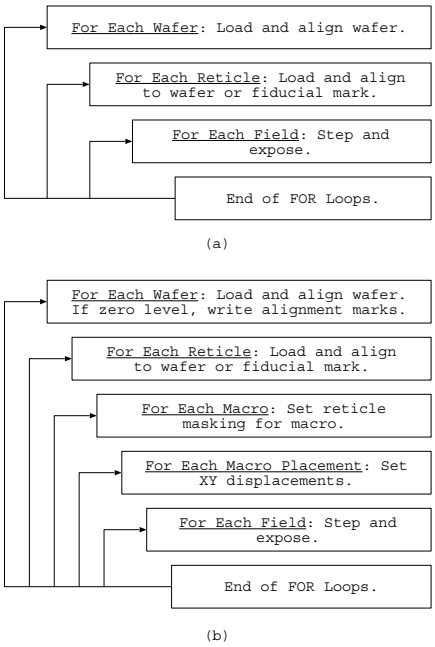

Figure 3: Exposure steps, (a) normal and (b) modified.

The flowchart for a typical exposure process and a modified flowchart for PM are shown in Figure 3 while costs are discussed in Section 7.4.

5. Standard SoC Design Methodology

A SoC is an ASIC, but is largely based on re-usable IP components or cores. Examples of cores are processors, memory components such as SRAMs and ROMs, and peripherals such as a UART. The cores communicate through standard bus interfaces that are controlled by bus arbiters, which are also available as IP [9]. After selecting

the IP components for the design, RTL generation is initiated. RTL design mostly involves connecting the IP to the bus [1][3]. Additional custom logic may be required if the core library can not provide the required functionality or performance. After RTL design, the RTL netlist goes through synthesis and physical design. The RTL netlist is hierarchical and each core is in a separate node of the hierarchy. The flow described below is a representative standard post-RTL physical design flow.

Area planning and floorplanning. Area planning is the task of estimating the area of each core. This process begins by adding up the area of each component in the core to produce the total area for the core. To account for wiring resources, this total area is typically multiplied by an empirical factor. The shape of the core is initially a square, but this can be changed in the floorplanning step.

The floorplan takes into account the connectivity of the blocks. The aspect ratio of the soft macros can be tuned during floorplanning to produce a better fit or allow better connectivity. An example floorplan is shown in Figure 4.

Flattening hierarchy with movebounds. After floorplanning, the hierarchy is typically removed since most placement tools operate best on a totally flat design. However, during the flattening step the floorplan information is kept in the form of "movebounds". A movebound defines a region on the chip image where all the logic inside the core should be placed. The movebound is defined by the shape and position of the core in the floorplan. The placement tool usually will not place cells belonging to the core outside its movebounds. Thus netlist flattening with movebounds preserves the floorplan information to a large extent.

Detailed placement and optimization. In this phase leaf level cells are placed on the chip image and in-place optimization such as re-synthesis, resizing and buffer insertion may be done. We employ a placement driven synthesis (PDS) tool [4] for this step. PDS relies on Steiner estimates to compute wire-lengths during the placement optimization process. PDS iterates over placement and optimization until it achieves the target cycle time. Detailed routing follows detailed placement.

6. Mask Reuse Methodology for SoCs

The primary difference between the MRM approach and the standard SoC flow is that in the MRM flow, the SoC is implemented as an interconnected set of hard cores and the logic required to integrate the cores or to provide custom functions are implemented using gate-array cells. This allows reuse of the pre-verified masks for each hard core during manufacturing.

Floorplanning. The floorplanning step is similar to that described in Section 5. However, in the standard methodology, most cores are soft and their aspect ratios can be changed, while in MRM most cores are hardened,

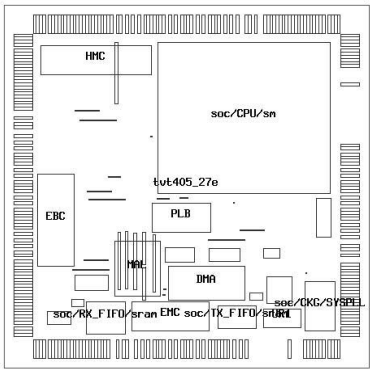

Figure 4: Final floorplan for 405 PBD.

and their aspect ratio can not be changed during floorplanning.

Mapping additional logic to gate-arrays. Logic which is not available as hard cores is mapped to gate-array cells instead of standard cells.

Top-level placement and optimization. PDS is used for placement, routing and timing optimization. PDS operates on only the logic mapped to gate-arrays and not on hard cores, placing the gate-array cells wherever there is empty space between the hard cores.

7. Case Study: IBM 405PBD SoC

We performed several experiments on an SoC design based on the 405PBD platform [3] offered by IBM Microelectronics for creating derivative SoC designs with reduced turn-around-time. Figure 5 shows a block diagram of our example network processor based on the 405PBD. The goal of the experiments are to compare the quality of a design manufactured using MRM (Section 6) with a design manufactured using the traditional SoC flow (Section 5) and custom masks. Design and mask costs are evaluated in Section 7.3 and Section 7.4.

The 405PBD is available for building SoC chips in IBM's SA27E (0.18um) technology. At the center of the 405PBD is the PowerPC 405, a versatile low power

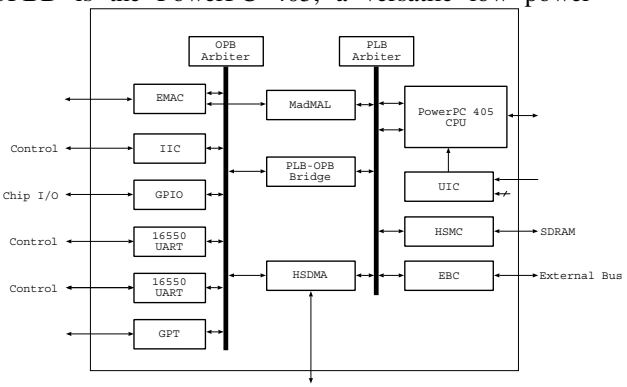

Figure 5: Block diagram of IBM 405PBD.

embedded processor core used in a wide variety of embedded applications. The 405PBD has two buses for on-chip communications: a high-speed Processor Local Bus (PLB) and a slow bus for peripherals called the On-chip Peripheral Bus (OPB) [9]. The 405PBD has an Ethernet subsystem which consists of the receiver and transmitter for physical connection to an Ethernet, an EMAC IP for Ethernet packet assembly and disassembly, and the MADMAL IP which provides direct-memory access for the EMAC over the PLB bus. The memory access is through the high speed memory controller (HSMC). The 405PBD has several peripherals for off-chip communication on the OPB bus such as the UART and I^2C. The 405PBD also has a shared DMA and interrupt controller (UIC) blocks.

7.1 Approximating the MRM

The cores used in the example SoC are available in IBM's Core Connect Library but mostly as soft cores, not hard as they would be in a true MRM. Since the process of creating a hard core is a relatively difficult process, we converted the soft cores to *pseudo-hard cores* to approximate hard cores for the case study by applying the following steps:

1. We determine the core's size and form factor using the *area planner* described in Section 5.1. We perform placement of the core's logic inside the planned area followed by correction of electrical violations such as capacitance and slew violations. When the pseudo-hard core is used in a SoC design, the placed and optimized logic inside the planned area is not touched.
2. Signal and power grid routing is not created for the pseudo-hard core unlike for a real hard core. But wiring layers are reserved above the pseudo-hard core. This prevents routing in the reserved areas when the pseudo-hard core is used in an SoC design

7.2 Experimental Results

The results of the standard flow and the MRM flow for the 405PBD are summarized in Table 1. Since the chip area is one of the most important factors in determining mask or packaging cost, all other factors being the same, we discuss area results in detail. The area of a design is the sum of the areas of the hard cores plus the area of pseudo-hard cores (without any multiplying factor) and the area of the standard cells or gate-array cells with a multiplying factor as discussed in Section 5 (Area Planning). The area of the pseudo-hard cores is not multiplied by a factor since the scaling was already done during the area planning phase of the core creation. The resulting area of the design produced by the MRM flow is 3.4% larger than the area of the design produced using a standard flow.

Table 1: Comparison of standard and MRM flows

Item	Standard Flow	MRM Flow
Objects	125,000	10,492
Area (mm2)	21.93	22.68
Delay (ns)	9.899	9.920
Runtime (min)	204	81

One reason for the larger area is that the pseudo-hard cores in the MRM flow have an inflexible shape and can not be customized to fit in their environment. Secondly, in the MRM flow all top-level logic is implemented with gate-array cells which are larger than standard cells. If the chip contains customer logic, assuming that the logic is implemented using gate-array cells instead of standard cells, the area penalty would be higher. Table 2 illustrates the effect of adding custom logic. We assume that the custom logic is similar to a HSMC high-speed memory controller IP, which is a fairly large core, consisting of 19,775 gates. The second column gives the area of the HSMC block if it were to be implemented using standard cells. The gate-array implementation area reported in the third column is about 12% larger than the standard cell implementation. The second row of Table 2 reports the numbers for the 405PBD with an additional HSMC core. In the SoC methodology, both HSMC cores are implemented using standard cells while in the MRM methodology, one is implemented as a pseudo-hard core while the second one is implemented using gate arrays. The area for the standard flow design is 22.90 mm^2 and is 23.77 mm^2 for the MRM design. Thus having customer logic implemented with gate-array cells increases area by 3.7% compared to a standard cell design. Without the additional customer core, the overhead is only 3.4% as reported in Table 1.

MRM designs can be expected to have slightly reduced performance since gate-array implementations are slightly slower than standard cell designs. Using hard cores compared with soft cores also reduce flexibility in achieving timing closure. However, as we can see in Table 1, the cycle time for the MRM design is 9.920 ns compared to 9.899 ns for the SoC design for a slow down of only 0.28%.

Table 2 Impact of custom logic.

Design	Objects	Std. Area	MRM Area
HSMC	19,755	0.977	1.094
405PBD+HSMC	144,755	22.90	23.77

7.3 Design Cost/TAT Analysis

The major reduction in design time is due to the significantly reduced number of objects processed in the MRM methodology. For the 405PBD, the number of objects reduced from 125000 to 10492, a reduction of a

factor of ten. As can be seen from Table 1, PDS runtime is reduced by a factor of 2.5X from 204 to 81 minutes.

Another bottleneck in creating large designs is the time required for verification. Though an MRM flow does not reduce RTL simulation time, it does reduce the time required for checking the equivalence of the final netlist and the initial RTL since each hard core is pre-verified and untouched in the MRM flow. Though estimates are difficult to provide, it is clear that MRM will simplify the design process and reduce design time and costs.

7.4 Mask and Manufacturing Costs

In our study, we do not implement changes to the manufacturing process required by MRM, hence the numbers used in this section are estimates from available manufacturing data. Mask manufacturing takes a significant amount of time since it involves generating and verifying huge amounts of shapes data. Since in the MRM flow most mask data already exist and is pre-verified, only the metal layer masks need to be generated and verified. We assume that only the top four metal layers require customization, implying that eight custom masks (two per metal layer) have to be created. If we assume the standard SoC methodology requires at least 30 custom masks, the result is a 73% reduction in the number of custom masks required. Thus assuming mask costs of $3 million (Section 2) MRM could produce possible savings of $2.19M as well as a 73% reduction in turn around time.

However, the MRM approach does require some special processing for each wafer. We assume the following: 45 seconds to expose a wafer per layer, less than 1 second to reposition the reticle to expose a different hard core, and 20 seconds to swap in a new reticle. We assume that the 12 hard cores in our design fit on a single reticle. The time for litho of an MRM layer would be 57s (45 + 12). Assuming 22 MRM layers and 8 custom layers, total time for litho would be (57×22 + 45×8) = 1614 s. The time for litho in an all custom flow with 30 custom layers would be (30×45) = 1350s. Thus the litho overhead for an MRM flow would be about 20%. If litho cost is about 35% of the total wafer processing cost [10], total wafer processing costs being $5000 to $10000 [1] each wafer would cost 35% × 20% = 7% more – a possible additional cost of $350 to $700 per wafer.

So assuming above costs, the MRM approach would be best for low or medium volume designs requiring less than 5000 wafers.

8. Conclusions and Further Work

Today's SoC design methodology provides an efficient way to develop highly integrated systems on a single chip by utilizing pre-designed cores. However, once assembled, the physical design and manufacturing steps still follow the traditional ASIC process and do not benefit from reuse of these cores. We propose an alternative Mask Reuse Methodology where most cores are provided with hardened layouts, significantly reducing the number of components for chip-level processing and the associated turn-around time. In addition, each core has a pre-verified mask set, which can be re-used to significantly reduce mask costs and mask manufacturing time. We show that in addition, our approach also reduces design time with minimal degradation in area and performance. The MRM approach looks increasingly attractive for future designs as mask costs spiral upwards and reuse content grows in designs.

Acknowledgement: We are grateful to Jung Yoon of IBM for help with manufacturing related data.

9. References

[1] L.Pillegi, H.Schmit, A.J. Strojwas, P.Gopalakrishnan, V.Kheterpal, A.Koorapaty, C.Patel, V.Rovner, and K.Y. Tong, "Exploring regular fabrics to optimize the performance-cost trade-off", DAC, June 2003.

[2] R.Morse, "Data management: Understanding chip-finishing, tapeout, and data", Microlithography Short Course Notes, Feb. 2004.

[3] R.A. Bergamaschi, S.Bhattacharya, R.Wagner, C.Fellenz, M.Muhlada, F.White, J.-M. Daveau, and W.R. Lee, "Automating the design of SoCs using cores", IEEE Design and Test of Computers, vol. 18, pp. 32--45, 2001.

[4] W.Donath, P.Kudva, L.Stok, P.Villarrubia, L.Reddy, A.Sullivan, K.Chakraborty, "Transformational placement and synthesis", Proc. Design Automation and Test in Europe Conference and Exhibition, Mar. 2000.

[5] G.S. Ditlow, F.-L. Heng, M.A. Lavin, D.L. Ostapko, and J.H. Yoon, "Partitioned mask layout", United States Patent 6,383,847, May 2002.

[6] P.S. Zuchowski, C.B. Reynolds, R.J. Grupp, S.G. Davis, B.Cremen, B.Troxel, "A hybrid ASIC and FPGA architecture", ICCAD, Nov. 2002.

[7] B.Zahiri, "Structured ASICs: Opportunities and challenges", ICCD, Oct. 2003.

[8] eASIC, The Configurable Logic Company™ http://www.easic.com.

[9] IBM Corp., IBM CoreConnect bus architecture. http://www.ibm.com/chips/products/coreconnect/index.html.

[10] MEDEA+: Helping Europe's Extreme UV technology to win the battle for the Next Generation Lithography Solution. http://www.medeaplus.org/webpublic/publ_april2002.html.