# Area, Delay, Power, and Cost Trends for Metal-Programmable Structured ASICs (MPSAs)

Usman Ahmed, Guy G.F. Lemieux, Steven J.E. Wilton

*Department of Electical and Computer Engineering*
*University of British Columbia*
*Vancouver, BC, Canada*
{uahmed, lemieux, stevew}@ece.ubc.ca

***Abstract*—As integrated circuits are scaled to finer process geometries, the risk involved with the increased design effort and high NRE costs becomes too great for some applications. FPGAs offer one solution, but for high performance, high volume, or low power applications, FPGAs may not be suitable. For some of these applications, structured ASICs may provide a better solution. Structured ASICs share many of the same characteristics as FPGAs, but consume less power, are more dense, and can run faster. Despite these advantages, structured ASICs have not yet achieved the level of popularity some had predicted. There are several possible reasons, including unfamiliar technology, immature CAD, and claimed advantages which have not yet been concretely demonstrated. In much the same way that it has helped improve FPGA adoption, we believe that an increased public research effort can begin to address many of these issues. This paper takes a step in this direction by investigating metal-programmable structured ASICs, or MPSAs. We determine the area, delay, and power trends and quantify the cost advantages of MPSAs relative to cell based ICs (CBICs) for a wide range of possible MPSA logic architectures and layout assumptions. In particular, we quantify the impact of the number of user-defined metal mask layers on these metrics. Results suggest the number of these *programmable layers* should be as small as possible for most MPSAs, unless very large die sizes are required.**

## I. Introduction

Time-to-market is vital in the integrated circuit (IC) industry. ICs manufactured using the latest process technology suffer from extreme complexity, enormous design effort, and high non-recurring engineering (NRE) costs. This is limiting adoption of the latest process technologies in cell-based ICs (CBICs); advanced process technologies (90nm and below) still account for only 49% of TSMC's revenue [1]. Field-programmable gate arrays provide one way of addressing this problem, however, they may not be suitable for applications which require low power, high volumes or high performance [11]. In particular, applications in the emerging portable and hand-held device market often require lower power than what is available in today's FPGAs, but faster turn-around time than can be achieved using CBICs.

For some of these applications, structured ASICs may be preferred. Structured ASICs offer a faster turn-around time than CBICs, consume less power, are faster and more dense than FPGAs. A structured ASIC is a generic IC that is partially fabricated and can be "programmed" to implement any digital circuit by adding one or more metal layers and/or via layers [19]. This partial fabrication of the device improves the cost and turnaround time. Power consumption is reduced (compared to an FPGA) since programmable switches are not required; in an FPGA, these switches consume significant static and dynamic power. For these reasons, we expect that the structured ASICs are an increasingly important design methodology, especially in the platform-based designs and the hand-held/battery powered device markets. This advantage will continue to grow at finer processes such as 32nm or below.

Although structured ASICs were introduced several years ago, they have not achieved the traction that many anticipated. There are many possible reasons for this, including unfamiliar technology, immature CAD, and claimed advantages which have not yet been concretely demonstrated. We believe that, as technology continues to advance, the advantages of structured ASICs will become even more compelling, especially for low-power handheld applications. When that happens, we will need new architectures, CAD tools, and design flows. Since structured ASICs are so similar to FPGAs in their internal structure (they are based on a grid of logic cells connected using "configurable" connectors), it is the FPGA community that is most suitable to address these issues. In this paper, we take a step in this direction by investigating metal-programmable structured ASICs, or *MPSAs*.

The cost, turnaround time, performance, and power are the key advantages of structured ASICs. These factors depend on the number of metal and/or via layers that are used to customize a structured ASIC. Intuitively, we would like to minimize the number of layers that can be used for customization, since this minimizes the cost to the designer and may shorten turn-around time. On the other hand, if the device is not flexible enough, the implementation of a circuit on the structured ASIC will require more space and possibly be slower and consume more power than is required. This conflicting criteria suggests that there is an optimum number of layers that must be configurable, and this is the main focus of this paper.

The configurability question is important because the earlier structured ASIC offerings ranged from a single via customization [2] up to 6-metal and 6-via customization [3]. The academic treatment of structured ASICs has focused mainly on specific architectures [13], [10], [14], [15]. Our goal in

this paper is to look at the configurability of across a range of different architectures, focussing on structured ASICs that can be customized by both metal and via layers. Specifically, our contributions in this paper are:

- A cost model to estimate the manufacturing cost of structured ASICs as a function of area and number of configurable layers.
- A framework to evaluate the configurability of MPSAs.
- A study of the relation between the number of configurable layers and the area, delay, power and manufacturing cost across a broad range of architectures.

## II. Cost Model

In this section, we describe a first order cost model that predicts the cost per die ($C_{die}$) as a function of the number of configurable layers in a structured ASIC. The cost per die depends on more than just the die area; a larger die with fewer layers to be customised can turn out to be less expensive than a smaller die with more customizable layers. To estimate $C_{die}$, we write:

$$C_{die} = C_{base} + C_{custom} + C_{pkg} + C_{test} \tag{1}$$

where $C_{base}$ is the cost of the partially fabricated device (i.e., the cost shared across all the customers), $C_{custom}$ is the cost to customize the pre-fabricated chip to implement a particular circuit, $C_{pkg}$ is the packaging cost, and $C_{test}$ is the testing cost. In this paper, we assume that $C_{pkg}$, and $C_{test}$ are constants in our experiments, so they are not considered in our $C_{die}$ calculations; they do depend upon the user's design, but they do not depend upon the range of SA implementations we consider (i.e., area or number of configurable layers).

The base and customization costs can be further subdivided into three parts: (1) a non-recurring cost of preparing the mask sets, (2) cost of setting up the fab line, and (3) wafer processing cost. We can, therefore, write $C_{base}$ as:

$$C_{base} = \underbrace{\left(\frac{C_{sm_l}N_{fm_l} + C_{sm_u}N_{fm_u} + C_{fs_1}}{V_{tot}}\right)}_{\text{Mask costs}} + \underbrace{\left(\frac{C_{wpm}N_{fm_l} + C_{sw}}{N_{gdpw}}\right)}_{\text{Wafer costs}}$$

where $N_{fm_l}$ is the number of lower fixed masks, $N_{fm_u}$ is the number of upper fixed masks (e.g., required for power grid etc.), $C_{sm_l}$ is the cost for a single lower-level mask (e.g., poly mask, metal-1 mask etc.), $C_{sm_u}$ is the cost for a single upper-level mask (e.g., metal-4 mask), $V_{tot}$ is the expected total volume and $C_{fs_1}$ is the fab setup cost of the SA device for all customers, $C_{wpm}$ is the wafer processing cost for a single mask, $C_{sw}$ is cost of single unprocessed wafer, and $N_{gdpw}$ is the number of good-dies-per-wafer.

The customization cost ($C_{custom}$) can be calculated in a similar fashion as:

$$C_{custom} = \underbrace{N_s\left(\frac{C_{sm_u}N_{cm} + C_{fs_2}}{V_c}\right)}_{\text{Mask costs}} + \underbrace{\left(\frac{C_{wpm}(N_{cm} + N_{fm_u})}{N_{gdpw}}\right)}_{\text{Wafer costs}}$$

where $N_s$ is the number of customer silicon spins, $N_{cm}$ is the number of custom masks, and $V_c$ is the volume per customer. $N_{cm}$ can be calculated as:

$$N_{cm} = N_{rl} \times N_{mprl}$$

where $N_{rl}$ is the number of routing layers, and $N_{mprl}$ is the number of masks needed for each layer. In an MPSA, a routing layer consists of one metal layer and one via layer.

We are interested in analyzing the sensitivity of the cost function to the number of configurable routing layers ($N_{rl}$) and the die area ($A_{die}$). By substituting the values of $C_{base}$, $C_{custom}$ and $N_{cm}$ in Equation 1 and rearranging the terms, $C_{die}$ can be written as:

$$C_{die} = \frac{K_0}{N_{gdpw}} + N_{rl}\left(\frac{K_1}{N_{gdpw}} + K_2\right) + K_3 \tag{2}$$

where $K_0$, $K_1$, $K_2$, and $K_3$ are constants that depend on the volume requirements and various foundry costs, but are fixed for a given structured ASIC product. Values for these constants are:

$$K_0 = C_{wpm}(N_{fm_l} + N_{fm_u}) + C_{sw}$$
$$K_1 = N_{mprl}C_{wpm}$$
$$K_2 = \frac{N_s C_{sm_u} N_{mprl}}{V_c}$$
$$K_3 = \frac{C_{sm_l}N_{fm_l} + C_{sm_u}N_{fm_u}}{V_{tot}} + \frac{C_{fs_1}}{V_{tot}} + N_s\left(\frac{C_{fs_2}}{V_c}\right)$$

Using the parameter values shown in Table I, typical values for $K_0$, $K_1$, $K_2$, and $K_3$ are \$4400, \$440, \$1.44, and \$0.999, respectively.

### A. Yield Model for $N_{gdpw}$

The number of good-dies-per-wafer ($N_{gdpw}$) depends on number of dies per wafer ($N_{dpw}$) and die yield ($Y_{die}$), and is given as:

$$N_{gdpw} = N_{dpw} \times Y_{die}$$

The number of dies per wafer can be approximated as [9]:

$$N_{dpw} = \frac{\pi \times \left(\frac{D_{waf}}{2}\right)^2}{A_{core}+A_{io}+A_{scribe}} - \frac{\pi \times D_{waf}}{\sqrt{2\,(A_{core}+A_{io}+A_{scribe})}}$$

In this equation, $D_{waf}$ is the wafer diameter, $A_{core}$ is the core area, $A_{io}$ is the area for input and output pads, and $A_{scribe}$ is the scribe area. A scribe is a ring around the die reserved for wafer testing and die cutting; it mostly influences the small die area. These three components of area are illustrated in Figure 1. If $P_w$ and $S_w$ represent the pad width and scribe width, respectively, then $A_{io}$ and $A_{scribe}$ can be calculated as:

$$A_{io} = 4P_w\left(P_w + \sqrt{A_{core}}\right)$$

$$A_{scribe} = 4S_w\left(S_w + 2P_w + \sqrt{A_{core}}\right)$$

The die yield can be estimated as [18][6]:

$$Y_{die} = Y_0 \times \left(1 + \frac{(A_{core} + A_{io}) \times D_0}{\alpha}\right)^{-\alpha}$$

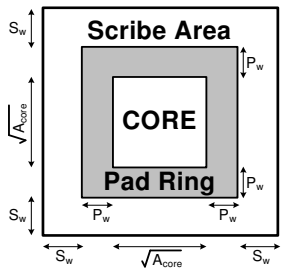



Fig. 1. Core, Pad and Scribe Area

where $Y_0$ is the multiplier to account for material and systematic yield, $D_0$ is the defect density, and $\alpha$ is the cluster factor. The yield may be affected by the number of routing layers; each additional layer may cause the yield to reduce. On the other hand, the regularity in MPSA fixed layers can help to improve the yield. It is not known which of these conflicting effects would be significant. We currently assume both of these to have negligible effect on $Y_{die}$.

Most of the parameters in the above cost model are confidential information of a foundry and are not disclosed. The cost numbers (such as, $C_{sm_l}$, $C_{sm_u}$ and $C_{wpm}$) can also vary from one foundry to another. Table I shows the parameter values we use to estimate $C_{die}$. We obtained and confirmed data from various sources, including several news articles and contacts in industry. For a range of values of $A_{core}$ and $N_{rl}$, the output of cost model is shown in Figure 2, respectively.

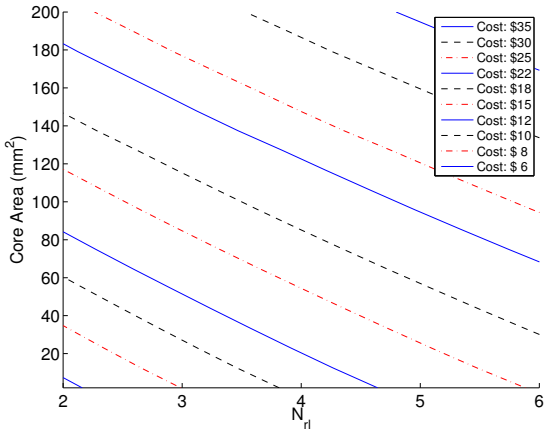



Fig. 2. MPSA Cost Model

## III. Framework

We have set up a framework to study the trade-offs associated with different configurability choices in MPSAs. The framework allows us to collect statistics such as die area, delay, power and manufacturing cost for different MPSA architectures. In this section we describe how we model an MPSA architecture, our CAD flow, and the statistics that we collect.

TABLE I
Values of Parameters used in the Cost Model

| Param. | Value | Comments |
|---|---|---|
| $N_{fm_l}$ | 18 | *Fixed masks below the configurable masks*<br>(1) A 10-metal, 90nm process requires 34 masks [12]; we assume 45nm also requires 34 masks[a]<br>(2) Device fabricated up to metal-2, and subsequent layers require single mask |
| $C_{sm_l}$ | \$107k | *Single mask cost for lower layers*<br>(1) 45nm mask set costs \$2.5M [4]<br>(2) Cost of lower level masks is 3x that of upper level masks |
| $C_{sm_u}$ | \$36k | *Single mask cost for upper layers* [12] |
| $V_{tot}$ | 2M | *Total volume* |
| $C_{wpm}$ | \$220 | *Wafer processing cost per mask*<br>(1) Cost to process a 45nm wafer: \$8000<br>(2) 34 masks total |
| $D_{waf}$ | $300mm$ | *Wafer diameter* |
| $P_w$ | $150\mu m$ | *Pad width* |
| $S_w$ | $100\mu m$ | *Scribe width* |
| $N_s$ | 2 | *Number of silicon spins*<br>One prototype plus one re-spin |
| $V_c$ | 100k | *Per customer volume* |
| $N_{fm_u}$ | 2 | *Number of fixed masks above the configurable masks (e.g., for power grid)* |
| $Y_0$ | 0.9 | *Material and systematic yield* [6] |
| $N_{mprl}$ | 2 | *Number of masks per routing layer*<br>One mask each for metal and via |
| $D_0$ | 1395 per $cm^2$ | *Defect rate* [6] |
| $\alpha$ | 2.0 | *Cluster factor* [6] |
| $\frac{C_{fs_1}}{V_{tot}}$, $\frac{C_{fs_2}}{V_c}$ | 0 | $C_{fs_1}$, $C_{fs_2}$*: Fab line setup costs*<br>Any fab setup costs can be ignored, esp. when these are divided over the volume |
| $C_{sw}$ | 0 | *Cost of a single, unprocessed wafer*<br>Assumption: Cost of an unprocessed wafer is negligible compared to the processing cost |
| $C_{pkg}$, $C_{test}$ | 0 | *Packaging cost, Testing cost*<br>These costs are not considered because these are independent of die area and $N_{rl}$ |

[a] Even with $N_{fm_l}$ as high as 36, the results are not significantly different.

### A. Architecture Model

When modeling the logic block architecture, we prefer to model it without worrying about the low-level, layout related details. From the perspective of the interconnect, the various logic block options differ only in their size and the number of inputs and outputs. Therefore, we abstract the logic block as a rectangular block with a certain number of pins on it. The logic block size (height and width) and the position of pins are specified in terms of routing tracks. Figure 3 illustrates the modeling process for a simple 2-input logic block.

### B. CAD Flow

Our CAD flow is shown in Figure 4. The flow starts with a technology mapped circuit. The first step is to initialize the placement by reading in the physical size (height and width) of a logic block, the location of logic block inputs and outputs, and location of I/O pads of the circuit. The placement grid is set to a minimum square (i.e., if the technology mapped circuit has $N$ blocks, then the initial grid size would be $\lceil\sqrt{N}\rceil \times \lceil\sqrt{N}\rceil$). Following this, we perform placement, route

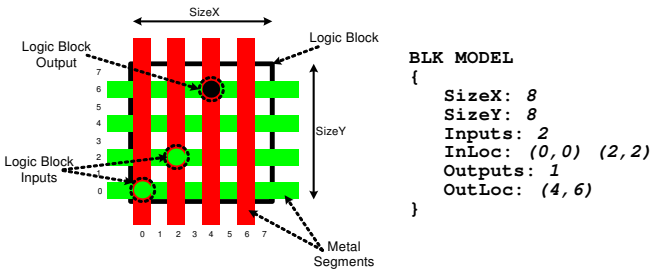



Fig. 3. Modeling an Architecture

the circuit for a given number of routing layers, and calculate routing congestion. If there is any congestion, we increase the placement grid size and repeat these steps. The following subsections describe the placement and routing stages in more detail.

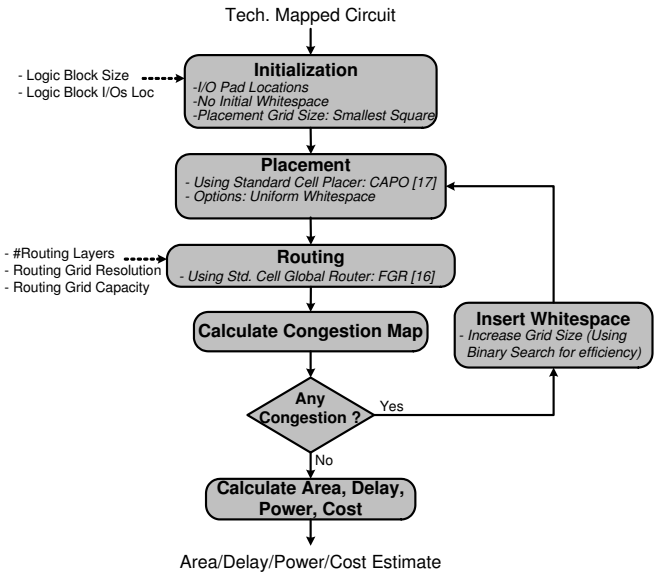



Fig. 4. CAD Flow

*1) Placement:* The placement problem for MPSA is similar to the FPGA placement problem, since all the pre-fabricated logic blocks have same size and are arranged on a grid. This implies that an FPGA placer, e.g. VPR [8], may be suitable. However, there are two problems with this approach. First, the number of blocks to be placed can be fairly large, especially in the case of fine grained logic blocks. We have found that the simulated annealing placement algorithm of VPR is slow with such large circuits. Second, VPR does not have an ability to insert whitespace[1] to remove congestion. Whitespace insertion is crucial, because with small logic blocks, and the routing being done on top of them, MPSAs are likely to experience congestion. For these reasons, we use a standard cell ASIC placer which is faster and has the ability to insert whitespace.

We are using the CAPO [17] standard cell placer. It has different options for whitespace insertion; we use the uniform whitespace distribution. To eliminate congestion, we increase the grid size, thus creating whitespace, and then re-place the circuit, resulting in a better distribution of whitespace. Some circuits require a large amount of whitespace, therefore, to speed up the flow we use binary search to find the minimum routable grid size.

*2) Routing:* After placement, the next step is to route all the nets to estimate wirelength. In our flow, we use the FGR global router [16]. In addition to the list of nets to route, the inputs to router include the number of available metal and via layers for routing, the resolution of the global routing grid (number of logic blocks encapsulated in a global routing tile), and the grid capacity (number of metal wires that can pass through the global routing tile).

The MPSA routing problem is very similar to the ASIC routing problem. Detailed routing in ASICs confines the connections to the given global routing and deals mainly with meeting the design rules [7]; in general, the quality of routing result is dictated entirely by the global route. Therefore, to simplify the flow we do not perform detailed routing. As is typical with ASICs, we assume that a successful global routing result can always be detail routed with negligible wirelength overhead.

### C. Metrics

The metrics we use to compare different configurability choices include core area, delay, power and manufacturing cost. The core area is calculated by multiplying the logic block area with the size of the placement grid. We use the Elmore delay model to estimate the delay of each net and then use the average net delay as our delay metric. We use the average net delay, as opposed to critical path delay, for two reasons. First, the number of routing layers only affects the interconnect and this effect is captured in the average net delay. Second, it allows us to compare different configurability choices without knowing the internal details of the logic blocks. For the power metric, we are concerned with the dynamic power dissipated in the interconnect since this is the only component of power that would change as we vary the number of routing layers. We use the total interconnect (metal and via) capacitance as a first order estimate for power. Finally, we use the cost model described in section II to estimate the manufacturing cost of the die.

## IV. Results

In this section, we investigate the impact of the number of programmable layers on the cost, area, speed, and power of the device. We also present the cost advantages of an MPSA to a CBIC for several technologies.

To obtain this data, we employed an experimental methodology. We used nineteen largest MCNC benchmark circuits[2] that have commonly been used in FPGA research [8].

[1]By whitespace we mean an entire empty logic block.

[2]One of the circuits, s38584.1, contains a net with more than 3000 pins which was too large for the router. We chose to exclude the benchmark rather than modify it.

The flow described in the previous section assumes a technology-mapped circuit, however, the technology mapping depends on the internal structure of each physical cell on the MPSA. In order to focus our attention on the interconnect architecture, we abstract the contents of the cell by representing only its input and output pins and cell area. This means that an exact technology mapping is impossible. Instead, we perform a clustering step to produce an interconnect netlist that approximates a real technology-mapped netlist. Our benchmark circuits are written in terms of 2-input gates; we cluster these basic gates such that each cluster has a specific number of inputs and outputs that matches the number of inputs and outputs of a particular logic block architecture. Such a clustered netlist has many of the properties (such as fan-in and fan-out distributions, Rent parameter etc.) of a real technology mapped circuit. We use T-VPack ([8]), an FPGA clustering algorithm, for this purpose.

Because we avoid real technology mapping, we much be careful to never compare the results of two different logic blocks (I/O counts) directly. Hence, we do not draw any conclusions about which logic block is better. Instead, we average results across logic blocks and only compare layout density.

Our experimental methodology also requires an estimate of the layout area (height and width) and pin locations of each physical cell on the MPSA. The layout area for a particular logic block depends upon the contents (number of gates) and the effort of the layout artist; both of which are hard to estimate precisely. Instead, we sweep the area across a range of values. The minimum cell area represents a very dense layout that is determined by the number of logic block pins. If the pins of a logic block can fit in an area of $x \times x$, then we assume the minimum area is $2x \times 2x$. For maximum area, we find an area number for an "average" gate by averaging the areas of different basic standard cells such as NAND, NOR, MUX, etc. We then scale this area number by the number of outputs of a logic block and use it as the maximum area. Within this range, we sweep through five equally spaced points. The pin locations are randomly generated around each cell. The different logic block *types* (i/o counts) and the corresponding layout area values used in our experiments are shown in Table II.

TABLE II
LOGIC BLOCKS USED IN EXPERIMENTS

| Type | | Block Layout Area(Width×Height) | | | | |
|---|---|---|---|---|---|---|
| | | High Density | ··· | | | Low Density |
| *IN* | *OUT* | *Min.* | *Small* | *Medium* | *Large* | *Max.* |
| 2 | 1 | 8x8 | 12x12 | 15x15 | 19x19 | 22x22 |
| 4 | 2 | 12x12 | 17x17 | 22x22 | 27x27 | 32x32 |
| 6 | 3 | 12x12 | 19x19 | 26x26 | 33x33 | 39x39 |
| 8 | 4 | 16x16 | 23x23 | 30x30 | 37x37 | 44x44 |
| 10 | 5 | 16x16 | 25x25 | 33x33 | 42x42 | 50x50 |
| 12 | 6 | 20x20 | 29x29 | 37x37 | 46x46 | 54x54 |
| 14 | 7 | 20x20 | 30x30 | 40x40 | 50x50 | 59x59 |
| 16 | 8 | 20x20 | 31x31 | 42x42 | 53x53 | 63x63 |

The trends for area, delay and power as a function of the number of routing layers, averaged over all the MCNC circuits, are shown in Figure 5. The plots show averaged (geometric mean) data of all the different logic block types for all the circuits. The plots are normalized to the values of minimum block layout area with two routing layers. We define the *nominal area* to be the core area of the minimum area block with two routing layers. The nominal area in these plots is $0.008mm^2$. There are four important observations. First, for larger block layouts, the area, delay, and power does not change as we increase the number of routing layers. This is because, the blocks are so large that even with two layers

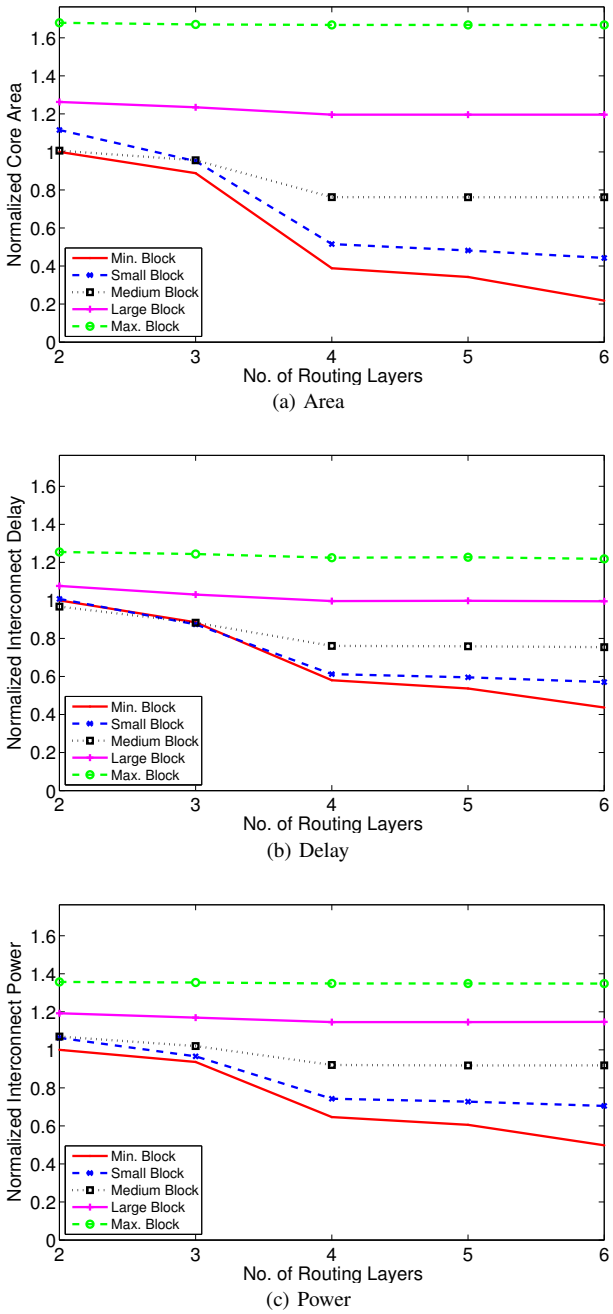
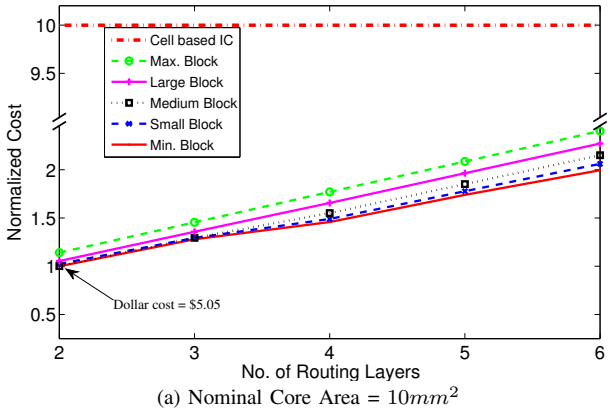
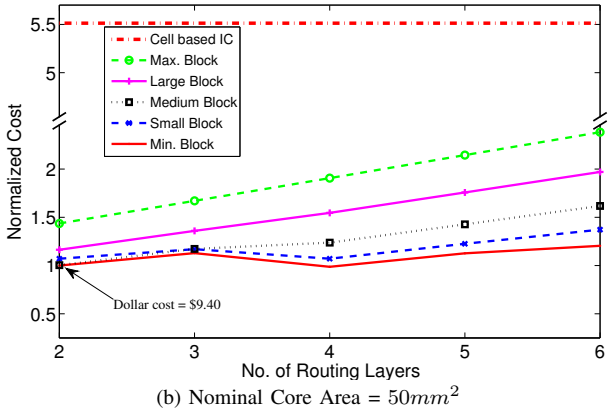


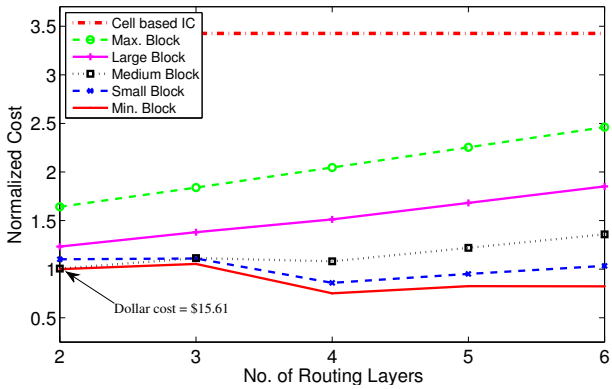

Fig. 5. MCNC Circuits: Area, Delay and Power Trends (Nominal Core Area at 45nm = $0.008mm^2$)




(a) Nominal Core Area = $10mm^2$

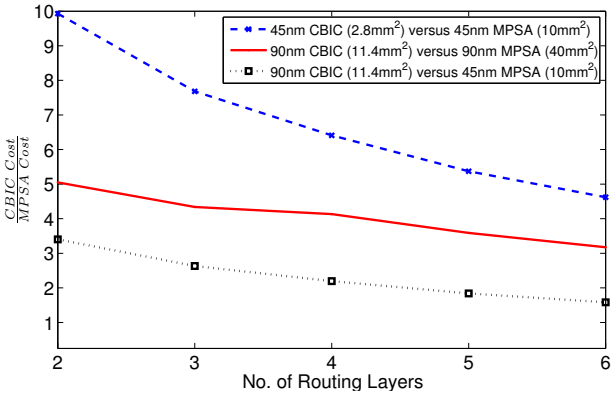



(b) Nominal Core Area = $50mm^2$

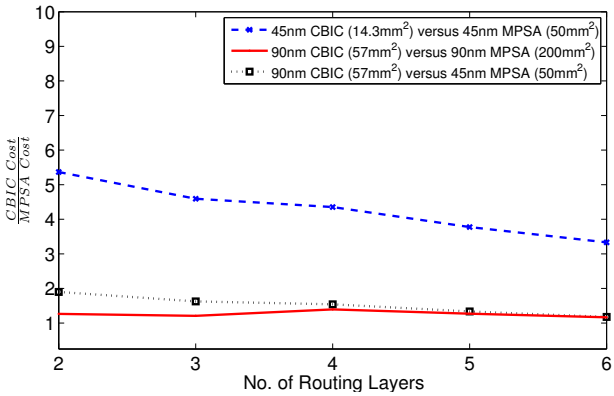



(c) Nominal Core Area = $100mm^2$

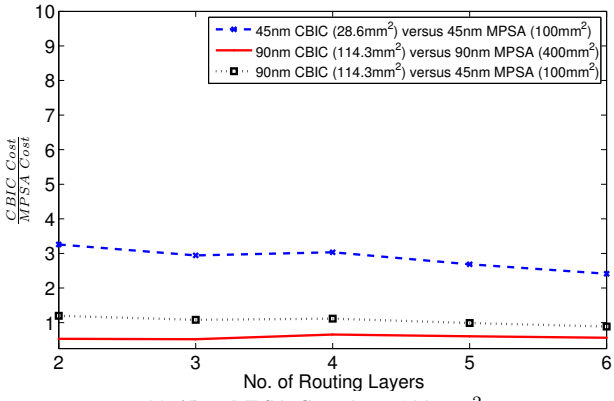
Fig. 6. MCNC Circuits: Trends for Die Cost at 45nm




(a) 45nm MPSA Core Area=$10mm^2$
90nm MPSA Core Area=$40mm^2$




(b) 45nm MPSA Core Area=$50mm^2$
90nm MPSA Core Area=$200mm^2$




(c) 45nm MPSA Core Area=$100mm^2$
90nm MPSA Core Area=$400mm^2$

Fig. 7. Cost advantage of MPSAs over CBIC at 90nm and 45nm (higher value means MPSA is lower cost)

there is no congestion and therefore, there is no effect of adding subsequent routing layers. Second, for smaller block layouts, the improvements in area, delay and power are quite small after four routing layers. Third, in some cases, given the same number of routing layers, the core area with larger blocks can be smaller than the core area with small logic blocks (e.g., core areas for "Medium" and "Small" blocks with two routing layers in Figure 5(a)). This is primarily because of the uniform whitespace distribution scheme used during placement. The total whitespace required for small blocks is more than the whitespace inserted for larger blocks which increases the core area. The use of an intelligent whitespace insertion algorithm (one that inserts whitespace only at the congested areas) could alleviate this problem. Finally, area is the most sensitive to the addition of extra routing layers, while power is the least sensitive. These trends are similar when the averaged data shown in Figure 5 is examined for individual logic block types (i/o counts), but these data are not shown due to space constraints.

Next, we estimated the dollar cost by applying the cost model described in Section II.[3] However, the circuits we used are quite small. This is impractical, and artificially increases ($N_{gdpw}$) significantly, reducing Equation 2 to $N_{rl}K_2 + K_3$. Because of this, we scaled the core area to a realistic value before applying the cost model. The cost trends for core areas of $10mm^2$, $50mm^2$, and $100mm^2$ are shown in Figure 6. It can also be seen from Figure 6 that, for small die sizes, the cost of adding an extra layer is almost always greater than any cost savings due to area reduction. For large die sizes, additional routing layers reduce cost modestly for only the most dense block layouts.

We also show the estimated CBIC cost in Figure 6, produced using the core area of a "Min" block layout area, six routing layers, and all custom masks. It can be seen that, despite the small area of CBICs, there is a huge gap between the cost of MPSA and CBIC for smaller dies. This difference, however, diminishes as the die sizes grow, suggesting that it is still cost effective to use CBICs for extremely large designs.

Finally, we compare the cost of implementing a design in MPSA and CBIC. We look at two different process technologies – 90nm and 45nm. The area of the 90nm implementation is 4x the area of 45nm implementation. For MPSA costs we assumed a "Medium" block layout area whereas for CBIC we assumed "Min" block layout area. With these assumptions, the CBIC implementation of a design takes 3.5x less area than the MPSA implementation in the same process technology. The ratio of CBIC costs to MPSA costs are then shown in Figure 7. It can be seen that, for smaller dies, the MPSAs are more cost effective than CBICs despite a 3.5x area penalty. The cost effectiveness improves as we scale to finer process geometries. The figure also shows the comparison of a 90nm CBIC implementation versus a 45nm MPSA implementation. Again, MPSAs are much cheaper than CBIC implementation, especially for small die sizes. This suggests that the MPSAs can make modern technologies much more affordable.

## V. Conclusions

This paper has presented area, delay, power and cost trends for MPSAs. Area is most sensitive, whereas the power is least sensitive to the number of customizable layers. The sensitivity also varies with logic block layout density; high-density layouts have greater sensitivity than low-density layouts. In most cases, to achieve lowest cost, the number of customizable layers must be as small as possible; the area saving does not translate into a cost saving.

We also have demonstrated that a design implemented in a 2-layer 45nm MPSA, with a core area of $10mm^2$, is 10x cheaper than the same design implemented in CBIC; the advantage increases for smaller die. Compared to using an older technology, the same $10mm^2$ design is still 3.5x less costly than a 90nm CBIC implementation at $11.4mm^2$. The cost advantage of MPSAs makes modern technologies much more accessible, giving products access to further benefits (higher clock speed, lower power) of a modern process that is not available in older-technology CBICs.

There are a number of limitations in this work. First, in our delay and power estimates, we did not consider delay and power dissipation of the logic blocks or precise critical paths. Second, the whitespace insertion scheme used in the CAD flow distributes the whitespace uniformly across the whole die rather than inserting it only at congested locations. This can cause the core area to escalate, especially with small block sizes and with fewer routing layers. Finally, we assume that there are dedicated power and clock networks for the logic blocks and we do not consider their area overhead. However, despite these limitations we believe our results are still important and the trends we show are representative of MPSAs.

In the future, we plan to extend this work by addressing some of these limitations. In this regard, we plan to use large circuits with embedded macro blocks (e.g. circuits from [5]). We also plan to extend this work to via-programmable structured ASICs (VPSAs).

[3] We explored a range of values for the parameters in Table I; although the final costs changed, the trends remain similar.

## References


[1] http://www.eetimes.com/showArticle.jhtml?articleID=217200925.

[2] Nextreme Structured ASIC, eASIC Corp. http://www.easic.com/pdf/asic/nextreme_asic_structured_asic.pdf.

[3] Lightspeed Logic http://web.archive.org/web/20080111112237/http://www.lightspeed.com/products.html.

[4] The Advent of Next Generation Lithography Technologies in Advanced Semiconductor Processing, *Frost & Sullivan Press Release*, Aug. 27, 2007.

[5] eASIC Placement Contest. http://www.easic.com/index.php?p=university.

[6] *The International Technology Roadmap for Semiconductors (ITRS)*, chapter Yield Enhancement, pages 7–10. 2007.

[7] S. Batterywala, N. Shenoy, W. Nicholls, and H. Zhou. Track assignment: A desirable intermediate step between global routing and detailed routing. In *ICCAD*, pages 59–66, Nov. 2002.

[8] V. Betz, J. Rose, and A. Marquardt. *Architecture and CAD for Deep-Submicron FPGAs*. Kluwer Academic Publishers, 1999.

[9] J. L. Hennessy and D. A. Patterson. *Computer Architecture: A Quantitative Approach*. Morgan Kaufmann Inc., 2006.

[10] A. Koorapaty, V. Kheterpal, P. Gopalakrishnan, M. Fu, and L. Pileggi. Exploring logic block granularity for regular fabrics. In *DATE*, 2004.

[11] I. Kuon and J. Rose. Measuring the gap between FPGAs and ASICs. *IEEE Trans. on CAD*, 26(2):203–215, 2007.

[12] Z. Or-Bach. Paradigm shift in ASIC technology: In-standard metal, out-standard cell. *eASIC White Paper*, September 2005.

[13] L. Pileggi, 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. In *DAC*, 2003.

[14] Y. Ran and M. Marek-Sadowska. Via-configurable routing architectures and fast design mappability estimation for regular fabrics. In *ICCAD*, pages 25–32, 2005.

[15] Y. Ran and M. Marek-Sadowska. Via-configurable routing architectures and fast design mappability estimation for regular fabrics. *IEEE Trans. on VLSI*, 14(9):998–1009, Sept. 2006.

[16] J. A. Roy and I. L. Markov. High-performance routing at the nanometer scale. In *ICCAD*, pages 496–502, 2007.

[17] J. A. Roy, D. A. Papa, S. N. Adya, H. H. Chan, A. N. Ng, J. F. Lu, and I. L. Markov. CAPO: Robust and scalable open-source min-cut floorplacer. In *ISPD*, pages 224–226, 2005.

[18] C. Strapper, F. Armstrong, and K. Saji. Integrated circuit yield statistics. *Proc. of the IEEE*, 71(4):453–470, April 1983.

[19] B. Zahiri. Structured ASICs: Opportunities and challenges. In *ICCD*, page 404, Oct. 2003.